Integration of energy management and production planning

Application to steelmaking industry

RACHID LABRIK

Master of Science Thesis
Stockholm, Sweden 2014
Integration of energy management and production planning

Application to steelmaking industry

R A C H I D L A B R I K

Master’s Thesis in Optimization and Systems Theory (30 ECTS credits)
Master Programme in Computer Science and Engineering (300 credits)
Royal Institute of Technology year 2014
Supervisor at KTH was Xiaoming Hu
Examiner was Xiaoming Hu

TRITA-MAT-E 2014:06
ISRN-KTH/MAT/E--14/06--SE

Royal Institute of Technology
School of Engineering Sciences

KTH SCI
SE-100 44 Stockholm, Sweden
URL: www.kth.se/sci
Abstract

Steelmaking industry, one of the most electricity-intensive industrial processes, is seeking for new approaches to improve its competitiveness in terms of energy savings by taking advantage of the volatile electricity prices. This fluctuation in the price is mainly caused by the increasing share of renewable energy sources, the liberalization of energy markets and the growing demand of the energy. Therefore, making the production scheduling of steelmaking processes with knowledge about the cost of the energy may lead to significant cost savings in the electricity bills.

With this aim in mind, different models are developed in this project in order to improve the existing monolithic models (continuous-time based scheduling) to find an efficient formulation of accounting for electricity consumption and also to expand them with more detailed scheduling of Electric Arc Furnace stage in the production process.

The optimization of the energy cost with multiple electricity sources and contracts and the production planning are usually done as stand-alone optimizers due to their complexity, therefore as a new approach in addition to the monolithic model an iterative framework is developed in this work. The idea to integrate the two models in an iterative manner has potential to be useful in the industry due to low effort for reformulation of existing models.

The implemented framework uses multiparametric programming together with bilevel programming in order to direct the schedule to find a compromise between the production constraints and goals, and the energy cost. To ensure applicability heuristic approaches are also examined whenever full sized models are not meeting computational performance requirements.

The results show that the monolithic model implemented has a considerable advantage in terms of computational time compared to the models in the literature and in some cases, the solution can be obtained in a few minutes instead of hours. In the contrary, the iterative framework shows a bad performance in terms of computational time when dealing with real world instances. For that matter a heuristic approach, which is easy to implement, is investigated based on coordination theory and the results show that it has a potential since it provides solutions close to the optimal solutions in a reasonable amount of time.

Multiparametric programming is the main core of the iterative framework developed in this internship and it is not able to give the solutions for real world instances due to computational time limitations. This computational problem is related to the nature of the algorithm behind mixed integer multiparametric programming and its ability to handle the binary variables. Therefore, further work to this project is to develop new approaches to approximate multiparametric technique or develop some heuristics to approximate the mp-MILP solutions.

Keywords: Scheduling, Linearization methods, Mixed integer linear programming, multiparametric programming, Bilevel programming, Sensitivity Analysis;
# Table of contents

List of Figures .................................................................................................................. 6  
List of Tables .................................................................................................................... 7  

## 1 Introduction .................................................................................................................. 8  
1.1 Master thesis context ................................................................................................. 8  
1.2 Motivation ................................................................................................................ 8  
1.3 Methodology ............................................................................................................ 11  

## 2 Problem description, goal and scope ......................................................................... 13  
2.1 Problem description .................................................................................................. 13  
2.1.1 Melt shop steelmaking ....................................................................................... 13  
2.1.2 Production planning ........................................................................................... 14  
2.1.3 Energy management ......................................................................................... 14  
2.2 Goal and scope ........................................................................................................ 15  

## 3 Literature Review ....................................................................................................... 17  
3.1 Scheduling problems ............................................................................................... 17  
3.2 Scheduling with respect to electricity cost ................................................................ 18  
3.3 Bilevel programming .............................................................................................. 19  
3.4 Multiparametric programming ............................................................................... 20  

## 4 Detailed Scheduling of Electric Arc Furnace ............................................................ 21  
4.1 Motivation ................................................................................................................ 21  
4.2 Formulation ............................................................................................................. 22  
4.3 Integration of the energy awareness to EAF stage ................................................ 24  
4.3.1 Exact cost calculation ....................................................................................... 24  
4.3.2 Heuristic approaches ....................................................................................... 26  
4.3.3 Heuristic validation ......................................................................................... 27  

## 5 Monolithic model ........................................................................................................ 29  
5.1 “Slot neighborhood” model ................................................................................... 29  
5.1.1 Description ....................................................................................................... 29  
5.1.2 Mathematical formulation ............................................................................... 29  
5.1.3 Improvements .................................................................................................. 32  
5.2 “Variable duration” model .................................................................................... 33  
5.2.1 Mathematical formulation ............................................................................... 33  
5.2.2 Improvements .................................................................................................. 35  

## 6 Iterative framework .................................................................................................... 38  
6.1 Motivation and description ..................................................................................... 38  
6.2 Algorithm using multiparametric programming ................................................... 39  
6.2.1 Multiparametric programming ........................................................................ 39  
6.2.2 Solving the energy management model ............................................................. 41  
6.2.3 Algorithm description ...................................................................................... 42  
6.3 Algorithm using bilevel programming .................................................................... 45  
6.3.1 Bilevel programming ....................................................................................... 45  
6.3.2 Algorithm description ...................................................................................... 48  
6.4 Heuristic approaches .............................................................................................. 48  

## 7 Numerical case study results ...................................................................................... 51  
7.1 Detailed Scheduling of electric arc furnace ............................................................ 51
7.2 Monolithic model .................................................................................................................. 52
  7.2.1 “Duration variable” model’s performance ........................................................................ 52
  7.2.2 Improvement’s effect on the model ................................................................................ 54
7.3 Iterative framework .............................................................................................................. 54
  7.3.1 Multiparametric programming results ............................................................................. 54
  7.3.2 Algorithm’s results ........................................................................................................ 56
  7.3.3 Heuristic Framework ..................................................................................................... 58
8 Summary and conclusions ...................................................................................................... 60
9 Further Work ........................................................................................................................ 61
  9.1 Improving multiparametric programming ......................................................................... 61
  9.2 Approximation of multiparametric programming technique .............................................. 61
  9.3 Improvement of the heuristic approach in the framework ..................................................... 62
References ..................................................................................................................................... 63
Appendix A : Production planning model .................................................................................. 66
Appendix B : Energy management model .................................................................................. 67
List of Figures

Figure 1: World Electricity Consumption by region [3] ................................................................. 9
Figure 2: Day-ahead prices for 08/11/2013 on EPEX SPOT [4] .............................................. 10
Figure 3: Scheduling under energy constraints [5] ....................................................................... 10
Figure 4: World energy consumption by sector [8] and energy use by industrial sectors [9] .... 11
Figure 5: Melt shop's stages [5] .................................................................................................. 13
Figure 6: Energy purchase structure ........................................................................................... 14
Figure 7: Integration of energy management and production planning ...................................... 16
Figure 8: Detailed Electric Arc Furnace scheduling ..................................................................... 21
Figure 9: Electricity price time slots with task-slot assignment variables in EAF stage .......... 25
Figure 10: Heuristic assignment variables .................................................................................. 26
Figure 11: Exact slot consumption ............................................................................................... 27
Figure 12: Different cases to calculate electricity consumption in each time slot for “Slot neighborhood” model .......................................................................................... 31
Figure 13: Different cases to calculate electricity consumption in each time slot for “Variable duration” model ................................................................................. 33
Figure 14: Elimination of redundant binary variables ................................................................. 37
Figure 15: An iterative framework to solve the problem of integration of energy management and production planning .............................................................. 43
Figure 16: Integration of energy management and scheduling as a bilevel program .............. 45
Figure 17: Algorithm using bilevel programming ........................................................................ 48
Figure 18: Heuristic approach as an iterative framework to solve the problem of integration 49
Figure 19: Production schedules for the model with detailed EAF (bottom) and without it (Top) ..................................................................................................................... 52
Figure 20: Example of mp-LP results for 2 hours ........................................................................ 56
Figure 21: Onsite generation's decision binaries in the first slot (left) and in the second slot (right) ......................................................................................................................... 56
Figure 22: Heuristic behavior for 8 products and 3 hours ......................................................... 58
Figure 23: The marginal cost and its applied intervals for a particular example ..................... 62
List of Tables

Table 1: “Slot neighborhood” model notations.................................................................30
Table 2: The numerical value of the parameters ..............................................................51
Table 3: Performance of detailed scheduling of EAF model ...........................................52
Table 4: Optimization results of detailed scheduling of EAF stage ................................52
Table 5: Performance comparison between monolithic models .....................................53
Table 6: Elimination of binary variable's effects on the monolithic model ......................54
Table 7: The computational time of mp-MILP ................................................................55
Table 8: Iterative framework's results ............................................................................57
Table 9: Heuristic's results for non-monolithic integration of EM and PP ......................58
Table 10: The notation of Production planning model [7] .............................................66
Table 11: Energy management model's notation [12] ....................................................67
1 Introduction

This project focuses on the integration of the energy management and production planning with application to steelmaking industry. First, the project context and the problem description will be presented. Then, the models that are developed to solve the problem will be explained and the results achieved will be discussed. Finally, the references that are used, are listed.

1.1 Master thesis context

The topic of this project is a part of an ongoing research project in ABB within the industrial Demand-Side Response (iDSR) topic. The overall objective is to optimally shift energy intensive production of goods to lower price times in order to reduce energy cost while satisfying complex production process constraints.

ABB is one of the largest engineering companies and one of the largest conglomerates in the world. It helps its customers in energy supply, manufacturing and retail to use electrical power efficiently, to reduce environmental impact in a sustainable way and to increase business performance and industrial productivity. It is the leader in power and automation technologies. [1]

ABB has seven Corporate Research Centers (CRC) around the world in order to ensure a success and technological innovation in electrical engineering, power electronics, and industrial automation. The main CRC’s customers are ABB business units, which transform the result of research and development into commercial and innovative products. The ABB corporate research in Ladenburg (Germany) focuses in plant automation, factory automation, building automation, service solutions and power device mechatronics. These areas are organized in 8 research groups in order to maintain high level of operational excellence. [1]

This project is carried out within Process and Production Optimization group, which deals with many disciplines such as, production planning and scheduling, quality optimization and decision support for industrial practitioners and plant managers.

In this thesis, energy management will be integrated to production planning and many algorithms and models are built in order to achieve an optimal cost saving of the energy within a reasonable computational time. Throughout the whole thesis’s report, motivations are given behind any decision made for a better understanding of the development process.

1.2 Motivation

It is expected that future’s energy markets will be exposed to highly volatile electricity availability because governments are choosing to invest in the use of renewable energies for electricity generation. This is mainly because of increasing fossil fuel costs, the need to increase energy security and to decrease the emissions. This kind of green energies is associated with a high level of uncertainty and unpredictability.

In Germany, it is projected to have 30% of electricity from renewable energy sources in the year 2020 as an official target whereas it is currently about 20% (2013). Some optimistic scenarios have high estimations as they expect a share of 50% of electricity from renewable
energy in 2030 and over 80% in 2050 [2]. It is challenging to achieve the ambitious targets, therefore a number of different supporting technologies shall be employed to assure success and stable grid operation.

Nowadays electricity demand is in total increase, it is doubled from 1990 to 2011 and it is projected to grow by 81% from 2011 to 2035 according to The World Energy Outlook 2013 and International Energy Agency (IEA) [3] as show in Figure 1.

![World Electricity Consumption by Region](image)

**Figure 1: World Electricity Consumption by region [3]**

Another important feature of the problem is the liberation of the energy markets which began in early 1990’s all over the world. Many markets have been created in order to deal with trade and supply of the electricity and to express the real price of energy. This changed the business model of the companies and the energy purchasing policies. Generally, different types of contracts can be made, one can mention for example long term contract, which is given with a constant price varying from a week up to a year, time of use contract (TOU, on and off peak) and day-ahead market, which provides hourly varying price.

Due to this uncertainty and unpredictability of the energy sources, the liberalization of energy markets and the growing demand of the energy, the price of the electricity has gone through many fluctuations as shown in Figure 2. This figure shows the hourly price of the electricity at the European spot market EPEX SPOT [4] for November 8, 2013. One can see easily the considerable difference between prices during the night and during the day.
Adjusting the electricity use in order to take advantage of the off peak hours can lead to a better control of the electricity bills. For that matter, the production planning should be done with minimization of energy cost to ensure profitability, as described in Figure 3.

One way to deal with the fluctuation of the energy, which can create problems to the stability of electricity grids, is to use the energy storage systems. These systems are generally expensive or not available. Industrial demand side management (iDSM) technique provides a better solution in order to solve this problem by improved matching of the supply to the demand. It has two components, one of them is the energy efficiency, which is a part of technological improvements, and it is more related to the design or the use of particular equipment, i.e. the same result is achieved using less energy input. The other component is demand response and it can be achieved by intelligent scheduling of the production,
considering energy availability information. This technique has the advantage of low upfront cost because it does not need any hardware investments for industry.

Industrial Demand-Side Response strategy can be used to take into consideration both production planning and the changes in the electricity prices over time. This new approach has two features [6]:

- Price-based demand response: Customers can respond to the price structure to take advantage of lower priced period and avoid the higher priced periods.

- Incentive-based demand response: A load-reduction or load-increase incentives are given to customers, this happens when either there is a grid reliability problem or high electricity prices or in case of excess in power supply. Another set up of this feature is when to penalize customers if they fail to respond to their contractual commitments.

The solutions developed in this project are validated on a real-life industrial use case. In terms of energy consumption, the industrial sector uses 51.7% of energy, as shown in Figure 4. The steelmaking industry is chosen as an energy intensive batch and semi-continuous production process. It has a share of 38% of the consumption of the energy compared to the other industries such as chemicals and aluminum industry. Optimizing the production planning regarding to electricity price for this application may bring benefits for the plant, as already shown in the literature [7].

![Figure 4: World energy consumption by sector and energy use by industrial sectors](image)

1.3 Methodology

Methods used within the scope of the project rely on mathematical programming. Some optimization methods are explored to solve scheduling problems. Continuous time scheduling models are used to model the production planning and discrete time models for energy management.

The scheduling problem is modeled using global precedence binary variables that answer the question which product follows which, and also using assignment binary variables which assign a given product to be processed on a specific machine. For the energy management, there are binaries employed for modeling of on/off decisions such as for example the use of onsite generation as a possible source of electricity.
Using Mixed-Integer Programming approaches for modeling of both problems implies using some specific modeling and reformulation techniques such as the McCormick envelopes [10], or exact linearization methods in order to prevent creating a hard to solve non-linear program.

Two models are available to validate the results. One simulates the production planning model whereas the other one simulates the energy management model. Developed solution algorithms employ the use of bilevel programming and multiparametric programming.
2 Problem description, goal and scope

2.1 Problem description

2.1.1 Melt shop steelmaking

The steelmaking process studied in this project consists of four stages as shown in Figure 5. The steel is gone through every stage starting from EAF and finishing at CC. In the first stage there are two electric arc furnaces (EAF), where scrap is melted into heats using high volumes of electricity. This stage compared to other stages consumes around 85% of the total electricity consumption used to produce one product. The batch (which is called a heat) is taken to argon oxygen decarburization (AOD) where carbon content is reduced from the steel melt by oxidation which is done by blowing oxygen and argon into molten steel. The duration of this stage can vary depending on the steel grade desired. In the third stage, which is called ladle metallurgy facility (LF) units, the steel is refined by adding some materials in order to preserve the target steel grade. Besides, a temperature adjustment is done in this stage in order to meet entry requirements of the next stage. Finally, the melt steel is solidified in a continuous caster (CC) and it is cut into steel slabs.

The overall objective of the project is to determine the production schedule for the above process while minimizing the total cost of the electricity and the amount of time needed to schedule the whole production (makespan). For this matter, there are two separate models that optimize each one of these components (energy cost and makespan), but they must be combined in order to ensure finding true optimal solution for the problem. The first one is the production planning (PP) optimizer which provides an optimized production planning for melt shop in steel plant by minimizing the overall makespan. This model gives an optimal schedule with starting and finishing time of each task without taking into consideration the electricity price whereas the second model is energy management (EM). It is an electricity purchase optimizer and it has the full knowledge of the sources of the energy. It assumes predefined production schedules in a form of a load that shall be satisfied at all times and optimizes the energy purchasing strategy using an economic cost flow network.
2.1.2 Production planning

Steelmaking scheduling is a large scale combinatorial problem with many requirements and critical production constraints that should be satisfied [11]: The equipment should process only one product (from many to be processed) at a time, and in multistage production, one previous stage of a job need to be completed before processing the new one. In addition, each job has its due time. More complex constraints need to be ensured at the last stage of the process, where continuous casting of group of products called heat groups shall be maintained.

This model has been implemented in the literature and it is modeled following the equations (1 to 17) from Hader & Harjunkoski [7]. The equations and the variables used are given in the Appendix A.

2.1.3 Energy management

Energy management model has full knowledge of all different sources of electricity and its objective is to minimize the energy consumption cost with a given load distribution. These sources of electricity include a long term contract with a constant price and volume, an hourly day ahead markets with volatile price over the day, a time-of-use contract with two price levels and an internal source as an onsite generation that produce electricity. The plant has also the possibility to sell surplus of electricity to the grid. The important assumption made is that the demand for the production process shall always be met. The optimization is done using the property of the balancing node, where all flows from the sources should be equal to the flow into sinks. Each arc between two defined nodes has a defined cost. The goal of the program is to minimize the total flow cost.

Figure 6 shows different sources of electricity used and it is modeled as an economic flow network.

![Figure 6: Energy purchase structure](image-url)
This model is described following the equations (12-22) from Hadera et al. [12] and it is fully presented in the Appendix B with its different variables and equations.

Besides the electricity cost, the penalties paid due to deviation from a pre-agreed load curve are also included in the electricity bills. These penalties are paid when exceeding a predefined buffer, which is fixed by the provider. This buffer can be different for under and over consumption.

These penalties are implemented by Hadera & Harjunkoski [7] and integrated to the monolithic model. This additional cost can affect the optimization only if it is placed in the production planning model because it depends only on the load, which is just a parameter in the energy management model. These penalties are used in all the models implemented in this project. They are included in the total energy cost implicitly (if there is no explicit distinction between deviation penalties and energy cost).

2.2 Goal and scope

In this project, we are dealing with two complex systems: the production scheduling which focuses on minimizing the makespan (the amount of time required to complete a set of activities) and the energy management which minimizes the total cost of a given schedule by trying to optimize the purchasing strategies of electricity. The goal of this project can be divided into two different sub-goals, which are:

- Improvements of the energy-aware scheduling models by
  - Finding more efficient formulation of accounting for electricity consumption in continuous time based scheduling models
  - Expanding the continuous time formulation with more detailed scheduling of the EAF stage in the production process
- The integration of energy and scheduling without transformation into monolithic model

Coordination of the production scheduling and electricity purchase optimization has potential to achieve cost savings compared to situation where the two problems are treated separately. The scheduling model used is a continuous time scheduling problem in contrast to a discrete time scheduling model due to its accuracy and ability to accommodate complex production rules. For the continuous time problems it is a challenge to expand the models with awareness of electricity consumption by production tasks ([7] and [12]).

Energy awareness can be integrated monolithically to the production process in two different ways. One way could be to add just the hourly price of the electricity (one single price) to the scheduling model in order to steer the planning to take into consideration the costly slots. Another way is to include different purchase options of the energy (i.e. full knowledge of all different sources of electricity), which can be achieved by putting all possible contracts of the electricity as well as the on-site generation. The first approach with single price curve is simpler and potentially faster compared to the second approach, however it does not capture the entire properties of the problem.

Integration of energy management and scheduling is a challenging problem due to its many variables and constraints that cannot be solved within reasonable amount of time without
using some heuristic approaches and some calculations before starting the optimization. Therefore, a monolithic model could be one possibility to integrate these two problems if some improvements are done. Another way of treating this problem is as an iterative framework using some signals between these two components in order to direct the schedule to find a compromise between the makespan and the energy cost, Figure 7.

In this project, many improvements of the energy aware scheduling model are investigated. First, a detailed electric arc furnace scheduling model is introduced as a way to capture higher benefits of the cost performance. Second, new monolithic models are implemented in order to come up with a new model with a better computational performance. Besides, some improvements, which are related to the elimination of some binary variables, will be explored. Finally, some composition methods will be explored as well in order to convert the problem to an iterative framework.

For the solution to the above stated problems limitations, advantages and potential issues should be investigated. To ensure applicability heuristic approaches should be examined whenever full sized models are not meeting computational performance requirements.

The models should be general as much as possible in order to allow flexibility in order to change the parameters of the problem easily to deal with the real cases in the industry. Therefore, the algorithms implemented should allow for adaption to other production processes.

Figure 7: Integration of energy management and production planning
### 3 Literature Review

#### 3.1 Scheduling problems

“Scheduling is concerned with allocation of resources over time so as to execute the processing tasks required to manufacture a given set of products.” [13]. For a given amount of time and resources, and for a predefined objective function, finding a feasible solution or an optimal solution could be trivial or complex. The main challenge of a scheduling problem “is how to provide a perfect match, or near perfect match, of machines to jobs and subsequently determine the processing sequence of the jobs on each machine in order to achieve the goal.” [14]

Scheduling problems have gained a considerable attention for the past forty years and have been the subject of a significant research with many techniques ranging from unrefined dispatching rules to sophisticated parallel branch and bounds algorithms and bottleneck heuristics [15]. Job-shop is one of the most popular models in scheduling theory and it has been studied very well as it is considered to have a general representation of scheduling problems and has earned reputation for being difficult to solve. It is treated as comparative test-bench case for different techniques [15].

The single machine-sequencing problem is one of the simplest scheduling problems because it does not involve any assignment variables. If there is more than one equipment or machine, the model has to include beside sequence variable, the assignment variables, which tell to which machine, a product is assigned to. The machines can be arranged in different ways depending on the type of the problem. They can be used in parallel, in series or in a complex way. When dealing with multistage optimization problem, many constraints should be ensured such as the precedence between stages, which means that a given order is processed in a particular stage after it has gone through the previous one [16].

Production scheduling in the steel industry has been recognized as one of the most difficult industrial scheduling problems [11]. Méndez et al. [17] present a classification for scheduling problems of batch processes and features that characterize their optimization models. It involves many on-off decisions (binary variables) and many critical constraints, which makes problems too large and complicated to solve. A large number of complicated chemistry, geometrical and scheduling rules is involved in those models which make the complexity very big and therefore an excessive computational time is needed.

Heuristic methods are introduced to find good solutions, or to find simply feasible solutions for the really difficult problems. Most research now consists of designing better heuristics for specific instances of scheduling problems [18]. But, those heuristic models are limited to a specific set of constraints and problem formulation.

A scheduling model is described mathematically by defining the sets, the parameters, the variables, the constraints and the objective function. The sets are the specific characteristics of the problem such as machines, stages, tasks…etc. The parameters represent the data inputs used in the model, which are fixed and defined before the optimization like resource usage. The variables are the unknown decisions and values that are determined after the optimization. The constraints describe the problem limitations and restrictions whereas the
objective function is the goal behind the optimization such us minimizing the cost of the energy and the finishing time of the tasks.

For the continuous-time based steel scheduling a new approach is developed by Harjunkoski and Grossmann [11] using decomposition strategy and it is proved to be efficient up to almost 100 products. The solution provided is not optimal but still the method produces solutions within 1-3% of the global optimum. This model served later as an extension to energy-aware scheduling models developed by Hadera & Harjunkoski [7], Hadera et al. [12].

3.2 Scheduling with respect to electricity cost

Taking into consideration the energy cost when scheduling the production becomes increasingly important because of the volatile price of the electricity. Being an energy intensive production process, scheduling with respect to electricity cost in steelmaking industry has gained attention in the literature recently.

In the study by Ashok [19], a mixed integer linear programming formulation is proposed using discrete time model that includes time of use electricity tariffs and peak load management for scheduling mini steel in India using electric-arc furnaces for steel manufacturing. In the context of increasing electricity prices and the introduction of time varying electricity rates by utilities, mini steel plants can reschedule their operations to reduce their electricity bills. The model is coupled with an optimization formulation utilizing integer programming for minimizing the total electricity-cost satisfying production, process flow and storage constraints for different tariff structures. The methodology proposed can be used for determining the optimal response for any industry under time varying tariffs. The case study of a steel plant shows that significant reductions in peak-period demand (about 50%) and electricity cost (about 5.7%) are possible with optimal load schedules. The utility can also get significant reduction in the peak coincident demand if large industries optimally reschedule their productions in response to time-of-use (TOU) tariff.

Another problem for scheduling the production with regards to energy cost in steelmaking industry is described in [36], but the model is limited to the single stage of EAFs. This model takes into account the time-dependent power prices and it is assumed that the price of the total cost of the power consumption is a linear function of the load. Beside the regular constraints of steel production, the constraints of the capacity supply are considered as well. The authors propose a Lagrangian relaxation algorithm based on a discrete-time MILP. After relaxing the limitation constraints of machine capacity and the power supply, the original problem is decomposed into easy subproblems, which are solved to optimality using polynomial algorithm.

Scheduling for a foundry with electricity costs is also discussed in the literature [20]. The model considers penalties for electricity overconsumption when predefined electricity level is violated. The overall goal is to minimize the electricity bill accounting for energy and human resource constraints. This leads to better solutions in terms of energy cost and overall energy consumption. A hybrid heuristic is proposed based on a two-step mathematical programming approach that improves significantly the computation time, compared to the full MILP model.

Scheduling techniques are also applied to air separation plants and cement plants. Recently, Mitra et al. [21] developed a discrete-time MILP model to address the production planning in air separation plants under time-sensitive electricity prices for continuous power-intensive
processes whereas Castro et al. [22] developed a new continuous-time scheduling formulation for continuous plants under electricity cost in cement industry. Karwan and Keblis [23] also discussed air separation plant processes with a focus on different electricity tariff designs and plant utilizations.

### 3.3 Bilevel programming

Modeling scheduling problems under energy cost as an iterative framework have not received enough attention in the literature. The classical methods that are used in practice are monolithic models together with some decomposition approach such as Benders decomposition in order to speed up the computational time.

Iterative framework’s techniques have been in the literature when dealing with game theory problems and their applications due to their nature and structure, especially bilevel programming.

A bi-level programming program is a hierarchical mathematical program that contains an optimization problem in the constraints [24]. It can be seen as a sequential game, which is originated from the Stackelberg game theory. In a Stackelberg game, the upper level decision maker which is the dominant player, called the leader, acts first by choosing its optimal position and then the lower level decision maker, called the follower, optimize its objective function given the leader’s position determined [25].

Bi-level programming problems (BLPP) are NP-hard problems, which basically means that there is no polynomial time algorithm to solve them unless P=NP [26]. Therefore finding a suitable method to solve them could be difficult due to the size of the problems we are dealing with. Besides, there are many approximation mythologies to solve BLPP by replacing the lower level problem and convert the whole problem into a single monolithic model. Most of these methods require assumptions of smoothness, linearity or convexity.

Integration of the production planning and energy management can be seen as a bi-level programming problem or a Stackelberg game where each player tries to optimize its own goal. There is a wide research done in this field during last decades [24] and the challenge will be to adapt those methods to our case. Other similar systems but simpler has been treated and solved, for example a hierarchical framework using a two-level game to model interaction between a utility (leader) and its many individual consumers (followers) [27].

To solve linear and convex BLPP, there are many algorithms: Branch-and-bound algorithms, Cutting plane algorithms, Barrier and penalty functions algorithms, SQP algorithms, heuristics algorithms, multi-parametric programming. There is also a number of evolutionary algorithms to solve complex bilevel programs which does not satisfy the classical assumptions imposed by the usual methods, but those algorithms are computationally intensive ( [28], [24] and [29]).

The conventional solution approach is using Karush-Kuhn-Tucker approach by transforming the problem into a single problem. This method is based on gradient information by converting the inner problem into constraints using KKT optimality conditions, so it is not applicable for mixed integer bilevel program. Therefore, a reformulation of the mixed integer inner problem as a continuous should be done in order to apply KKT conditions [28].
3.4 Multiparametric programming

Multiparametric programming is a technique to solve optimization problems that involves parameters that vary between a lower and upper bound. The goal behind this method is to obtain the objective function value and the decision variables as an explicit function of the parameters as well as the regions in the space of the parameters where those functions can be applied [30]. Multiparametric programming gives a complete map of all possible optimal solutions, therefore there is no need to reoptimize when looking for an optimal solution for a particular setup of the parameters, we have just to substitute the values of the parameters in the obtained functions (those functions are given as a value functions of the parameters) to get the optimal decisions. This method provides an interesting tool to analyze the effect of the variations and changes in a mathematical program as well as identifies the sensitivity of the objective function to the parameters. One can make the optimization more efficient by investigating the regions to make better decisions.

Multiparametric programming has been used in many applications especially when dealing with variations in the data. These variations can originate from the uncertainty of the parameters, which are associated with many systems and processes such as fluctuations in the price, demand and supply. These kinds of problems require running the optimization problems many times due to the change of conditions and data values to get the optimal action accordingly. The advantage of this method is to get the optimal map of the space of the parameters without enumerating exhaustively the entire space.

Multiparametric programming is based on sensitivity analysis theory [31], which provides another way to deal with uncertainty and variation of the parameters. This technique predicts the outcome when a change in the neighborhood of the varying parameters occurs. This includes analyzing the changes in the objective function coefficients and the right hand side value of the constraints. Multiparametric programming generalize sensitivity analysis, it provides multiple regions depending on the parameters where the solution is optimal.
4 Detailed Scheduling of Electric Arc Furnace

4.1 Motivation

The use of electric arc furnace (EAF) in stainless steel production is important due to its flexibility and adaptability to accommodate fluctuations in the demand and also its ability to alloy steel from scrap of different types [32]. EAF steelmaking is contributing significantly to the production of steel in the world which can be estimated with a share of about 40% to 45% of the total steel production [32]. The process is done by melting the scrap in a furnace using electrodes and electrical energy. The electrical power of normal EAFs is within the range 50-120MW, depending on the size of the furnace. The temperature of melting is within the range of 1500-1550°C, depending upon the composition of the steel scrap [33].

Thus EAF consumes a lot of electric energy, it is an intensive batch production process. Therefore, the cost and availability of electrical power are two factors that should be taken into consideration when scheduling the products in this stage. An optimal schedule will allow planning electricity-intensive operations in a more exact way than found in the literature in order to take full advantage of the possibilities of demand side management.

In this chapter, we are going to study in detail the EAF process steps for more detailed scheduling. Typically there are 4 scrap-smelting steps of variable duration and 3 scrap loading operations of given duration constituting one EAF task. Steel plant scheduling models available in literature do not distinguish the smelting and loading operations [7], [34], [35]. The whole production task allows to get one heat as shown in Figure 8. The loading operations shown in Figure 8 in a blue color represent the breaks within which the loading of new scrap occurs and power-off times are accounted. Therefore, shifting those loading operations to on-peak will decrease the cost of the energy. In similar manner, shifting the power-on operations of smelting to times of reduces energy cost will reduce the electricity bill as well.

![Figure 8: Detailed Electric Arc Furnace scheduling](image)

This Additional flexibility in the EAF process is given by the fact that in order to create a certain type of heat some constant, predictable amount of energy needs to be delivered to the process. Thus, each smelting step requires a given amount of energy that should sum up to the total energy requirement of the designed heat (product).
We assume further that the power is a variable, which means that more power to electrodes will shorten the smelting time, thus the starting and finishing time of the EAF stage can be flexible, since a heat requires a certain amount of energy to be delivered, regardless of within how long time interval the delivery occurs (when losses are neglected). This model will give us the possibility to adapt our power, for example by using the maximum power if we want to finish early.

4.2 Formulation

The model is kept as general as possible in order to integrate it to the whole melt shop process without changing our variables and equations. Besides, the same notations are used as in the paper [7].

We define the sets, variables and parameters of our model:

Sets
- $I$: Set of smelting tasks at EAF stage
- $M$: Set of machines
- EAF: Subset of EAF machines
- $P$: Set of products
- ST: Production stages
- $S$: Set of time slots
- SM: Set of machines belonging to a given stage

The set M defines the equipment, and in our case it represents two electric arc furnaces in parallel, EAF1 and EAF2. The products are the heats we get from EAFs after melting the scrap. The starting and finishing time of loading operations can be easily calculated using only the set of smelting tasks.

Variables
- $h_{p,m,i}$: Duration of a given smelting step [min]
- $\rho_{p,m,i}$: Power level for a given smelting step operation [MW]
- $t_{p,m,i}^s$, $t_{p,m,i}^f$: Finish and start time of a given smelting step [min]
- $t_{st,p}^s$, $t_{st,p}^f$: Finish and start time of each stage [min]
- $x_{m,p}$: Binary variable, equal to 1 if the product $p$ is assigned to machine $m$ otherwise it is zero.

Parameters
- $\rho_{m}^{\text{max}}$: Physical upper limit of power consumption (e.g. due to infrastructure) [MW]
- $\rho_{m}^{\text{min}}$: Physical lower limit of power consumption (it is used in McCormick envelopes to eliminate the non-linearity, can be zero) [MW]
- $y_p$: Total energy needed to smelt a given heat [MWmin]
- $\beta_{p,i,m}$: Energy to be delivered in one smelting step [MWmin]
- $T^{\text{load}}$: The loading operation time [min]
- $M$: Big M value, the maximum of the starting and finishing time.
- $h_{p,m,i}^{\text{min}}$, $h_{p,m,i}^{\text{max}}$: Total min and max duration of time a given heat can be processed at EAF stage [min]
- $h_{p,m,i}^{\text{min}}$, $h_{p,m,i}^{\text{max}}$: Total min and max duration of time a given smelting step of a heat can be processed at EAF stage [min]
- $p_s$: The price of the electricity in each slot
The total energy amount delivered to form a complete heat (product) that can be further processed in AOD is known and shall be maintained:
\[ \sum_{i \in I, m \in M} \beta_{p,i,m} = \gamma_p \quad \forall p \in P \]

**Model**

The equation (1) states that the power supplied to EAF must not exceed a physical limit of the electrical infrastructure, thus an upper and lower bound must be set. It also forces the power to be zero when the product is not assigned to the machine. Each heat (product) assigned to a machine requires a certain amount of energy to be delivered by the electrodes during a smelting step, between two subsequent scrap loading operations, in order to allow the heat to melt the scrap metal, as captured in equation (2). Constraint (3) ensures that the total time duration of processing at EAF stage is restricted with specific lower and upper bounds, this is done using the assignment variable because the starting and finishing time of the stage do not introduce a machine as a parameter. Each smelting step have a starting and ending times restrictions, therefore two bounds are introduced as stated in equation (4). Besides, it enforces the duration to be zero for an unassigned machine. Equation (5) defines the finishing time \( t_{p,m,i}^f \) as the starting time \( t_{p,m,i}^s \) plus the duration of the smelting task \( h_{p,m,i} \). Unassigned machine gets a zero starting time, equation (6). The stage starting and finishing times \( t_{st,p}^s, t_{st,p}^f \) are connected with the starting and finishing times of the last smelting task as showed in equations (7-8). Finishing time of any scrap loading operation, which is the starting time of a smelting task, is equal to the finishing time of the previous smelting task plus the loading duration, which is a constant, equation (9).

\[
\begin{align*}
\rho_{m,i}^{\text{min}} \cdot X_{m,p} \leq & \rho_{p,m,i} \leq \rho_{m,i}^{\text{max}} \cdot X_{m,p} & \forall p \in P, \ m \in EAF, \ i \in I \\
h_{p,m,i} \cdot \rho_{p,m,i} = & \beta_{p,i,m} \cdot X_{m,p} & \forall p \in P, \ m \in EAF, \ i \in I \\
\sum_{m \in \text{SM}_{st,m}} h_{p,m,i}^{\text{min}} \cdot X_{m,p} \leq & t_{st,p}^f - t_{st,p}^s \leq \sum_{m \in \text{SM}_{st,m}} h_{p,m,i}^{\text{max}} \cdot X_{m,p} & \forall p \in P, \ st = 1 \\
h_{p,m,i}^{\text{min}} \cdot X_{m,p} \leq & h_{p,m,i} \leq h_{p,m,i}^{\text{max}} \cdot X_{m,p} & \forall p \in P, \ m \in EAF, \ st = 1 \ i \in I \\
t_{p,m,i}^f = t_{p,m,i}^s + h_{p,m,i} & \forall p \in P, \ m \in EAF, \ i \in I \\
t_{p,m,i}^s \leq M \cdot X_{m,p} & \forall p \in P, \ m \in EAF, \ i \in I \\
\sum_{m \in \text{SM}_{st,m}} t_{p,m,i}^s = & t_{st,p}^s & \forall p \in P, \ m \in M, \ i \in I, \ st \in ST, \ i = 1, \ st = 1 \\
\sum_{m \in \text{SM}_{st,m}} t_{p,m,i}^f = & t_{st,p}^f & \forall p \in P, \ m \in M, \ i \in I, \ st \in ST, \ i = |I|, \ st = 1 \\
t_{p,m,i}^s = & t_{p,m,i-1}^f + T_{\text{load}} \cdot X_{m,p} & \forall p \in P, \ m \in M, \ i \in I, \ st \in ST, \ \{st,m\} \in SM, i \neq 1, \ st = 1
\end{align*}
\]

Linear programs are relatively easy to solve and the global solution can be obtained numerically in a reasonable time. The equation (2) presents a non-linearity due to the bilinear term involved. It is desired to transform nonlinear terms into a standard linear equations since the MINLP are much harder to solve than MILP models. This non-linearity can be handled using piecewise McCormick envelopes [36]. Therefore the equation (2) is replaced by the following constraints:
This linearization method gives a lower bound of the mixed linear program we are dealing with. It is based on the upper and lower bound of the duration and the power variables \((h_{p,m,i}^{\text{min}}, h_{p,m,i}^{\text{max}})\) and the better the tightening range of those variables the better the approximation.

With the equations stated before, the model is fully defined as a scheduling model of the first stage in the melt shop steelmaking and it can be integrated easily to the whole melt shop process.

In order to take into consideration the cost of the energy, this model should be extended with energy awareness. There are many ways of introducing energy awareness to this model and in the next section we will explore those methods.

### 4.3 Integration of the energy awareness to EAF stage

#### 4.3.1 Exact cost calculation

In this section, by using some binary variables and expansion with a discrete time formulation, we calculate the exact energy consumption of each smelting step. Let \(y_s\) be the electricity price boundary of time slot \(s\). We assume that the time horizon is discretized by 60 min that means \(y_s = 60 \cdot s\), which corresponds to the pricing period given by the day ahead market tariffs.

We define some binary variables \(A_{i,p,s,st}, B_{i,p,s,st}, C_{i,p,s,st}, E_{i,p,s,st}\) and \(F_{i,p,s,st}\) each one of them represents a particular position of a smelting task as showed in the Figure 9. In this figure three different level of prices are taking, namely \(P_1, P_2\) and \(P_3\).

This discretization method using these kind of binaries was done by Nolde & Morari [34] and enhanced by Hadera & Harjunkoski [7]. The case D is omitted because the duration of one smelting task is never so long (i.e. less than 60min) to last over the entire time slot. The expansion is applied to each product in order to calculate the electricity consumption of the whole melt shop and we apply it to the EAF stage for each smelting step.
This can be modeled mathematically by comparing the starting and finishing time of each smelting task with time slots boundaries. The binary $A_{i,p,s,st}$ is true if the task $I$ starts and finishes in the same slots $S$, $B_{i,p,s,st}$ is true if the task $I$ starts in slot $S - 1$ and finishes in slot $S$, $C_{i,p,s,st}$ is true if the task starts in slot $S$ and finishes in slot $S + 1$ and the other binaries $E_{i,p,s,st}$ and $F_{i,p,s,st}$ define the case where the smelting task is processed before or after the slot $S$, as stated in equations (14-21). This model is based on the assumption that a time step is 60 min which corresponds to the time interval duration within which the electricity price may change.

Besides, we ensure that one of those binary variables is true for each slot, as stated in equation (23). Equation (24), calculates the overall electricity consumption of the EAF stage of a slot $q^E_{EAF}$, it is the sum of the duration spent by a smelting task in slot $s$ multiplied by the electricity consumption $\rho_{p,m,i}$. We divide by 60 in order to obtain MWh because the time unit used is min.

\begin{align*}
    t^{s}_{p,m,i} &\geq y_{s-1} - M \cdot (1 - A_{i,p,s,st}) \\
    t^{f}_{p,m,i} &\leq y_{s} + M \cdot (1 - A_{i,p,s,st}) \\
    t^{s}_{p,m,i} &\leq y_{s-1} + M \cdot (1 - B_{i,p,s,st}) \\
    y_{s-1} - M \cdot (1 - B_{i,p,s,st}) &\leq t^{f}_{p,m,i} \leq y_{s} + M \cdot (1 - B_{i,p,s,st}) \\
    t^{f}_{p,m,i} &\geq y_{s-1} + M \cdot (1 - C_{i,p,s,st}) \\
    t^{s}_{p,m,i} &\leq y_{s} + M \cdot (1 - C_{i,p,s,st}) \\
    y_{s-1} - M \cdot (1 - C_{i,p,s,st}) &\leq t^{f}_{p,m,i} \\
    t^{f}_{p,m,i} &\leq y_{s-1} + M \cdot (1 - E_{i,p,s,st}) \\
    t^{s}_{p,m,i} &\geq y_{s} + M \cdot (1 - F_{i,p,s,st}) \\
    A_{i,p,s,st} + B_{i,p,s,st} + C_{i,p,s,st} + E_{i,p,s,st} + F_{i,p,s,st} &= 1 \\
    \forall i \in I, \forall p \in P, s \in S, &\forall t \in ST, m \in M, \{st, m\} \in SM, st = 1
\end{align*}
The total electricity consumption, as stated above, has many non-linearities, this can be avoided using some linearization methods, like McCormick envelopes as used before. But, doing this will make the problem larger with many variables and more constraints. Therefore, it will be hard to solve it. For this reason, we will explore another way of calculating this energy consumption using some heuristics. And then we will evaluate their performance.

### 4.3.2 Heuristic approaches

To eliminate the non-linearity showed before and to reduce the execution time of the program we used a heuristic. In this heuristic, a smelting step is considered to be processed in a slot if a big part of it (more than 50%) is processed within the slot. The duration of a smelting step is usually between 20 to 40 minutes and the slots are uniformly distributed by 60 min. So, there are two cases to consider, as shown in the Figure 10.

![Figure 10: Heuristic assignment variables](image)

We formulate this mathematically by introducing two binary variables $Z_{p,s,i,m}$ and $T_{p,s,i,m}$. $Z_{p,s,i,m}$ is true if the smelting task I starts in slot $s$ and more than 50% of its duration is within the slot $s$ whereas $T_{p,s,i,m}$ is true if the smelting task finish in slot $s$ and more than 50% of its duration is within the slot $s$. An auxiliary variable $a_{p,s,i,m}$ is defined which is true if one of those variables $Z_{p,s,i,m}$ or $T_{p,s,i,m}$ is true, as stated in equations (25-29).

\[
q^{EAF} = \sum_{p \in P, i \in I, s \in S} \sum_{m \in EAF} r_{p,m,i} (t_{p,m,i}^f - t_{p,m,i}^s) + B_{i,p,s} (t_{p,m,i}^f - y_{s-1}) + c_{i,p,s} (y_s - t_{p,m,i}^s)) \frac{1}{60} \tag{24}
\forall s \in S
\]

\[
y_{s-1} - M (1 - Z_{p,s,i,m}) \leq t_{p,m,i}^s + \frac{1}{2} \cdot h_{p,m,i} \leq y_s + M (1 - Z_{p,s,i,m}), \tag{25}
\]

\[
y_{s-1} - M (1 - T_{p,s,i,m}) \leq t_{p,m,i}^f - \frac{1}{2} \cdot h_{p,m,i} \leq y_s + M (1 - T_{p,s,i,m}), \tag{26}
\]

\[
a_{p,s,i,m} \geq Z_{p,s,i,m} \tag{27}
\]

\[
a_{p,s,i,m} \geq T_{p,s,i,m} \tag{28}
\]

\[
0 \leq a_{p,s,i,m} \leq 1 \tag{29}
\]

\[
\forall p \in P, m \in EAF, i \in I, s \in S
\]

Equation (30) states that $a_{p,s,i,m}$ is zero if both $S_{p,s,i,m}$ and $T_{p,s,i,m}$ are false and the equation (31) forces $a_{p,s,i,m}$ to be zero for unsigned machine. Each smelting task should be assigned to one of two cases as stated in equation (32).
The total consumption in the EAF stage is given by equation (33)
\[
q_s^{EAF} = \frac{1}{60} \sum_{p \in P, m \in M, i \in I} a_{p,s,i,m} \cdot \beta_{p,i,m} \quad \forall s \in S
\]  

The price of the electricity is given by the slot consumption times the different time levels of the price \( P_s \) as stated the equation (34). The objective function of the program is the electricity cost, equation (35).
\[
cost = \sum_s q_s^{EAF} \cdot P_s
\]  
\[
\text{min}_{\text{all variables}} \text{cost}
\]  

### 4.3.3 Heuristic validation

In this section, we will evaluate the error of this heuristic by calculating the exact consumption of the electricity and comparing it with the total consumption given by the heuristic. Basically, the idea is that we run the optimization program using this heuristic as a result we get the production planning with starting and finishing time of each product. This allows us to determine the starting slot and finishing slot of each task and then calculate the exact consumption of the electricity using a small algorithm as shown in the figure 10. This algorithm is based on two GAMS functions \( \text{ceil} \) and \( \text{floor} \) and they can be defined as following:

- \( \text{floor}(x) = [x] \) is the largest integer not greater than \( x \).
- \( \text{ceil}(x) = [x] \) is the smallest integer not less than \( x \).

#### Algorithm 1 Calculation of the exact slot consumption

Set \( S^s_{p,s,i,m} = \begin{cases} 
1 & \text{if } X_{m,p} = 1 \text{ and } t^s_{p,m,i} = 0 \\
\lfloor \frac{t^s_{p,m,i}}{60} \rfloor & \text{otherwise}
\end{cases} \)  

\( \triangleright \) the starting slot of a smelting task

Set \( S^f_{p,s,i,m} = \lfloor \frac{t^f_{p,m,i}}{60} \rfloor \)  

\( \triangleright \) the finishing slot of a smelting task

Initialize \( q_s^{EAF} \) to zero

for \( p \in P, m \in M, i \in I, s \in S \), do

while \( X_{m,p} = 1 \) and \( s = S^s_{p,s,i,m} \) do

if \( S^s_{p,s,i,m} = S^f_{p,s,i,m} \) then  

\( \triangleright \) the smelting task is processed within one slot

Set \( q_s^{EAF} = q_s^{EAF} + \rho_{p,m,i} \cdot \frac{t^f_{p,m,i} - t^s_{p,m,i}}{60} \)

else  

\( \triangleright \) the smelting task starts in slot \( s \) and finishes in slot \( s + 1 \)

Set \( q_s^{EAF} = q_s^{EAF} + \rho_{p,m,i} \cdot \frac{y_s - t^f_{p,m,i}}{60} \)

Set \( q_{s+1}^{EAF} = q_{s+1}^{EAF} + \rho_{p,m,i} \cdot \frac{t^f_{p,m,i} - y_s}{60} \)

end if

end while

end for

Figure 11: Exact slot consumption
In this algorithm, we assume that the optimization is already done and that we have the values of the variables. We calculate the electricity consumption of each slot dependent on the situation we are dealing with: The smelting task starts and finishes in the same slot or it starts in a slot and finishes in the next slot. Note that the situation where the task starts in the slot before and finishes in the slot after is not possible because the maximum duration of a smelting task is less than 60min.

The heuristic error can be calculated by comparing the cost given by heuristic and the cost calculated using the exact slot consumption as stated in equation (35) where cost is given by equation (34) and $q^{EAF}_s$ is the slot consumption using the above algorithm.

$$error = \sum_s q^{EAF}_s \cdot P_s - cost$$  \hspace{1cm} (36)
5 Monolithic model

Real world optimization problems are often large and complex to solve with thousands of variables and equations. Therefore, the amount of memory and the computational time needed to solve this kind of problems is big and grow significantly with the number of variables and equations especially when dealing with mixed integer programming problems.

There are many traditional ways to deal with such problems. One approach could be to make all the decisions simultaneously by solving a monolithic optimization problem which makes it sometimes impossible to solve when it comes to the size of the real-world instances. Another approach could be to use multistage optimization algorithms, such as Benders decomposition [37], which partitions the optimization problem into small optimization problems, and then iteratively solve each small problem. This strategy can be more efficient than just solving one monolithic model. Algorithmic and iterative framework give another way of solving such problems. This technique will be explored in detail in the section 6.

In this section, the focus will be the monolithic model. For continuous-time scheduling based models dealing with integration of energy management and production planning applied to steelmaking industry, there exists limited number of developments available in literature, including [7], [12], [34]. The goal of this work is to come up with a new different formulation of the problem with less number of variables and equations in order to speed the computational time.

5.1 “Slot neighborhood” model

5.1.1 Description

The goal when integrating energy management and production planning is to calculate the overall electricity consumption in each time slot which gives the total energy cost by multiplying the price over the load. The slot neighborhood model is based on a continuous time scheduling model which uses assignment and precedence binaries in order to schedule the production as described in [7] (equations 1-17). Based on an auxiliary variable which defines the slots where a product is processed and two binary variables that define the starting and finishing slot of each task, we will be able to calculate the overall electricity consumption for each time slot.

5.1.2 Mathematical formulation

To integrate the energy awareness with the continuous time scheduling model we introduce a discretization of the time horizon uniformly (time slots) and we associate a price with each time slot. First we define the sets, parameters and variables used following the notation in Table 1. Then, we present different equations that define the model.
Table 1: “Slot neighborhood” model notations

Sets: $P$—heat (product); $HG$—heat groups; $M$—equipment (machine); $S$—time slots; $E$—electricity prices; $ST$—production stages; $SM(ST, M)$—production stages $ST$ mapped to corresponding equipment $m$;

Variables: $t^s_{m,p}, t^f_{m,p}, t^s_{p, st}, t^f_{p, st}$—starting/finishing time of heat $p$ on equipment $m$ or at stage $ST$; $q_s$—overall electricity consumption in a given time slot $s$; $Y^s_{p, s, st}, Y^f_{p, s, st}$—binaries, true when the product $p$ starts/finishes in slot $s$; $A_{p, s, st}$ an auxiliary pseudo-continuous variable, equal to 1 if the product $p$ is begin processed within the slot $s$; $X^e_{m, p}$—binary, true when heat $p$ is processed on equipment $m$; $V_{st, p, p'}$—binary, true if heat $p'$ processed after $p$ on stage $st$.

Parameters: $\tau_{p, m}$—processing duration of heat $p$ on equipment $m$; $\tau^e_{m}$—setup times for machine $m$; $y_s$—electricity price boundary of time slot $s$; $e_s$—electricity price of time slot $s$; $h_{p, m}$—specific power consumption of heat $p$ on equipment $m$.

To know within which time slot a product starts and finishes, we define the binaries $Y^s_{p, s, st}$ and $Y^f_{p, s, st}$ using big-M formulation (Equations 37-38). The strict inequality constraints in (37) and (38) are used in order to deal with the case when the starting and finishing slot are equal to the boundaries (If $t^s_{p, st} = y_s$ then the product $p$ is started in slot $s + 1$ and if $t^f_{p, st} = y_{s-1}$ then the product $p$ is finished in slot $s - 1$). These strict inequalities can be handled adding one small positive scalar $\varepsilon$ (e.g. $\varepsilon = 0.01$) as stated in equations (39-40). Equation (41) states that each product should be assigned to one machine (for each product, there is only one starting and finishing slot). Equation (42) ensures that each product is processed between slots where $Y^s_{p, s, st}$ and $Y^f_{p, s, st}$ are true: if one product start in slot $s$ and ends in slot $s'$ then this product is processed between $s$ and $s'$. If a product starts in a slot $s$ and finishes in a slot $s'$ then this product is processed outside the interval $[s, s']$ as stated in equation (43). Equation (44) define the maximum of $A_{p, s, st}$.

\[
\begin{align*}
    y_{s-1} \cdot Y^s_{p, s, st} &\leq t^s_{p, st} < y_s + M(1 - Y^s_{p, s, st}) & \forall p \in P, \ st \in ST, s \in S \\
    y_{s-1} \cdot Y^f_{p, s, st} &< t^f_{p, st} \leq y_s + M(1 - Y^f_{p, s, st}) & \forall p \in P, \ st \in ST, s \in S \\
    (y_{s-1} - \varepsilon) \cdot Y^f_{p, s, st} &\leq t^f_{p, st} \leq y_s + M(1 - Y^f_{p, s, st}) & \forall p \in P, \ st \in ST, s \in S \\
    \sum_{s \in S} Y^s_{p, s, st} &\leq 1 & \forall p \in P, \ st \in ST \\
    s' \cdot Y^f_{p, s', st} - s \cdot Y^s_{p, s, st} + 1 &\leq \sum_{s' = s}^{s} A_{p, s, st} + |S| \cdot (2 - Y^s_{p, s, st} - Y^f_{p, s, st}) & \forall s \leq s' \\
    \sum_{i < s} A_{p, i, st} + \sum_{i > s} A_{p, i, st} &\leq |S| \cdot (2 - Y^s_{p, s, st} - Y^f_{p, s', st}) & \forall s \leq s' \\
    0 &\leq A_{p, s, st} \leq 1 &
\end{align*}
\]

With equations (39-44) the variables $A_{p, s, st}$ are fully defined. These variable can only take two values $\{0, 1\}$ and this is ensured by equations (42), (43) and (44).
The goal is to use those variables in order to calculate the time duration of each product in a given slot. This can be done, by comparing the finishing and starting time variables with the discrete time grid boundaries. There are five different cases to deal with for a given time slot $s$ according to the values of $A_{p,s,st}$, $A_{p,s-1,1st}$ and $A_{p,s+1,1st}$ as shown in Figure 12. In all this cases, the time duration of a product in a slot $s$ is given by: $A_{p,s,1st} (t^f_{p,1st} - t^s_{p,1st}) + A_{p,s-1,1st} (t^s_{p,1st} - y_{s-1}) + A_{p,s+1,1st} (y_s - t^f_{p,1st})$. Considering the input data concerning production task duration it is possible to determine upfront if there is a need of the model extension to account for the cases when a task overlaps for more than 3 time slots. For our test cases considered in the later chapters it is assumed that the longest production task does not exceed 180 minutes.

<table>
<thead>
<tr>
<th>Time duration in slots</th>
<th>Case 0</th>
<th>Case 1</th>
<th>Case 2</th>
<th>Case 3</th>
<th>Case 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>P is not processed in $s$ $A_{p,s,1st} = 0$</td>
<td>$0$</td>
<td>$t^f_{p,1st} - t^s_{p,1st}$</td>
<td>$t^f_{p,1st} - y_{s-1}$</td>
<td>$y_s - t^s_{p,1st}$</td>
<td>$y_s - y_{s-1}$</td>
</tr>
<tr>
<td>P is processed only in slot $s$ $A_{p,s,1st} = 1, A_{p,s-1,1st} = 0, A_{p,s+1,1st} = 0$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P is processed in (“s” and “s-1”) $A_{p,s,1st} = 1, A_{p,s-1,1st} = 1, A_{p,s+1,1st} = 0$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P is processed in (“s” and “s+1”) $A_{p,s,1st} = 1, A_{p,s-1,1st} = 0, A_{p,s+1,1st} = 1$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 12: Different cases to calculate electricity consumption in each time slot for “Slot neighborhood” model

Therefore to get the overall electricity consumption in a slot $s$, we multiply the specific power consumption with the time duration of each product as shown in equation (45).

$$q_s = \sum_{p, m \in S_{m,t,m}} h_{p, m} \cdot A_{p,s,1st} \cdot X_{p,m} \cdot (t^f_{p,1st} + A_{p,s-1,1st} (t^s_{p,1st} - y_{s-1}) + A_{p,s+1,1st} (y_s - t^f_{p,1st}) \frac{1}{60}$$ (45)

To eliminate the non-linearity, we add three new variables $a_{s,p,s,t,m}, b_{s,p,s,t,m}$ and $c_{s,p,s,t,m}$ as defined in equations (46-48) and we use the big-M formulation to calculate them without using non-linear equalities as shown in equations (49-55).

$$a_{s,p,s,t,m} = A_{p,s,1st} \cdot X_{p,m}$$ (46)

$$b_{s,p,s,t,m} = A_{p,s-1,1st} \cdot A_{p,s,1st} \cdot X_{p,m} (t^s_{p,s,t,m} - y_{s-1})$$ (47)
The expression (as_{p,s,stm} \cdot \tau_{p,m} + bs_{p,s,stm} + cs_{p,s,stm}) express the total time duration spent in a machine \( m \) for a particular product \( p \). The overall electricity consumption in a slot is shown in equation (56).

\[
q_s = \sum_{p,s,stm} h_{p,m} (as_{p,s,stm} \cdot \tau_{p,m} + bs_{p,s,stm} + cs_{p,s,stm}) \frac{1}{60}
\]

### 5.1.3 Improvements

The computational improvement for this model is achieved by adding some tightening constraints that will speed the computational time. We present the constraints that seem to have a considerable effect in practice. Equation (57) states that the number of slots where the product is processed can be bounded using the time duration (e.g: if the duration of a product \( p \) is 45 then \( p \) is processed in at least 1 slot and at most 2 slots and if the duration of product \( p \) is 100 then \( p \) is processed in at least 2 slots and at most 3 slots). The time duration is bounded by the time step (60 min in our case) and the production time \( \tau_{p,m} \), equation (58). This equation ensures also that Equations (59-61) link the variables \( A_{p,s,st}, Y_{p,i,st}^s \) and \( Y_{p,i,st}^f \) to each other. For example equation (59) states that if a product is processed in slot “s” then it started within the slots before \( s \) (s included). Thus, one of the binaries \( Y_{p,s,stm}^s \) is true in slots before \( s \) and all of the other variables are 0.

\[
\sum_{m} X_{p,m} \cdot ceil(\frac{\tau_{p,m}}{60}) \leq \sum_{s} A_{p,s,st} \leq \sum_{m} X_{p,m} \cdot ceil(\frac{\tau_{p,m}}{60}) + 1
\]

\[
as_{p,s,stm} \cdot \tau_{p,m} + bs_{p,s,stm} + cs_{p,s,stm} \leq \min\{\tau_{p,m},60\} \cdot as_{p,s,stm}
\]

\[
A_{p,s,st} \leq \sum_{l<s} Y_{p,i,st}^s ; 1 - A_{p,s,st} \geq \sum_{l>s} Y_{p,i,st}^s
\]

\[
A_{p,s,st} \leq \sum_{l<s} Y_{p,i,st}^f ; 1 - A_{p,s,st} \geq \sum_{l<s} Y_{p,i,st}^f
\]

\[
Y_{p,s,stm}^f \leq \sum_{l<s} Y_{p,i,st}^s \cdot Y_{p,s,stm}^s \leq \sum_{l<s} Y_{p,i,st}^f
\]
particular slot, the variable \( \text{dur}_{p,m,s} \) becomes zero. This leads to another model that seems to be much simpler. We call it ‘variable duration model’ and it will be described in the next section.

5.2 “Variable duration” model

5.2.1 Mathematical formulation

For capturing of how much electricity it has been consumed in a given time slot, in this model, we introduce the variable \( \text{dur}_{p,m,s} \) which is the processing duration spent by a product in a given time slot. It is calculated using the binaries \( Y^s_{p,s,st} \) and \( Y^f_{p,s,st} \) which define the starting and finishing slot of each product. The goal of this model is to eliminate as much as possible of the auxiliary variables used in previous model and to reduce the number of equations by using just this variable to calculate the time duration and thus the load distribution.

The binaries \( Y^s_{p,s,st} \) and \( Y^f_{p,s,st} \) are defined similarly like previous model (equations 39-40). There are 5 cases we have to take into consideration when calculating this duration, as shown in Figure 13. Each one of these cases tracks the position of the product in the time horizon.

![Figure 13: Different cases to calculate electricity consumption in each time slot for “Variable duration” model](image)

**Case 1**

The product is processed within one slot (\( Y^s_{p,s,st} = 1 \) and \( Y^f_{p,s,st} = 1 \)). The time duration is the production time multiplied by the assignment variable. This means that if the product \( p \) is assigned to machine \( m \) then the variable \( \text{dur}_{p,m,s} \) is equal to production time otherwise it is zero. This can be assured with one constraint (equation (62)) Where M1 and M2 are big values.

\[
\tau_{p,m} \cdot X_{p,m} - M1(2 - Y^s_{p,s,st} - Y^f_{p,s,st}) \leq \text{dur}_{p,m,s} \leq \tau_{p,m} \cdot X_{p,m} + M2(2 - Y^s_{p,s,st} - Y^f_{p,s,st})
\]

(62) \[ \forall p \in P, m \in M, s \in S, st \in ST, \{st, m\} \in SM \]

This constraint can be further simplified by choosing carefully the big M constant (M1 and M2). We know that the duration in a particular slot is always less than the production time.
\[(\text{dur}_{p,m,s} \leq \tau_{p,m})\]. So, we can choose M2 to be zero. We also know that the duration in a particular slot is less than the time step (in our case it is 60 min). So, \[\text{dur}_{p,m,s} \leq \min \{\tau_{p,m}, \text{timeStep}\} \cdot X_{p,m}.\] The other constant M1 can be chosen carefully, the best value for M1 would be the production time \(\tau_{p,m}\) as in equation (63).

\[
\tau_{p,m}(Y_{p,s,st}^s + Y_{p,s,st}^f + X_{p,m} - 2) \leq \text{dur}_{p,m,s} \leq \min \{\tau_{p,m}, \text{timeStep}\} \cdot X_{p,m} \tag{63}
\]

\[\forall p \in P, m \in M, s \in S, st \in ST, \{st, m\} \in SM\]

**Case 2 & 3**

In case 2, the product finished in slot s and started before s, \((Y_{p,s,st}^s = 0\) and \(Y_{p,s,st}^f = 1\)). The time duration is the finishing time minus the previous time slot boundary \((t_{p,m} - Y_{s-1}X_{p,m})\). This is also formulated using big-M as stated in equation (64). In case 3, the product is started in slot s and it is finished afterwards, \((Y_{p,t,st}^s = 1\) and \(Y_{p,t,st}^f = 0\)). The time duration is the next time boundary minus the starting time of the product \((Y_s \cdot X_{p,m} - t_{s,m}^p\)), equation (65). In both cases, if the product is not assigned to the machine the assignment binary is zero \((X_{p,m} = \text{false})\) and the finishing time and the starting time of the task are zero as well \((t_{s,m}^p = 0\) and \(t_{p,m}^f = 0\)), therefore the duration becomes zero.

\[
t_{p,m}^f - Y_{s-1}X_{p,m} - M(1 - Y_{p,s,st}^f + Y_{p,s,st}^s) \leq \text{dur}_{p,m,s} \leq t_{p,m} - Y_{s-1}X_{p,m} + M(1 - Y_{p,s,st}^f + Y_{p,s,st}^s) \tag{64}
\]

\[\forall p \in P, m \in M, s \in S, st \in ST, \{st, m\} \in SM\]

\[
y_sX_{p,m} - t_{s,m}^p - M(1 - Y_{p,s,st}^f + Y_{p,s,st}^s) \leq \text{dur}_{p,m,s} \leq y_sX_{p,m} - t_{p,m}^s + M(1 - Y_{p,s,st}^f + Y_{p,s,st}^s) \tag{65}
\]

\[\forall p \in P, m \in M, s \in S, st \in ST, \{st, m\} \in SM\]

Where M can be chosen as the last bound interval \(y_{|S|}\).

**Case 4**

In the case 4, the product started before the slot s and finished after slot s. The time duration is the time step \((\text{timeStep} \cdot X_{p,m})\). This constraint can be ensured using the fact that the upper bound of \(\text{dur}_{p,m,s}\) is \(\text{timeStep}\) by pushing it to take its upper bound in the slots in between. If we assume that the product p starts in slot s and finishes in slot s’ with \(s < s'\) then in all the slots strictly between s and s’ the time duration is equal to \(\text{timeStep}\). The number of this slots is \(s'-s-1\).

Hence,

\[
\sum_{l=s+1}^{s'+1} \text{dur}_{p,m,i} = (\text{ord}(s') \cdot Y_{p,s,1}^f - \text{ord}(s) \cdot Y_{p,s,1}^s - 1) \cdot \text{timeStep} \cdot X_{p,m} \tag{64}
\]

\[\forall p \in P, m \in M, s \in S, s' \in S, st \in ST, \{st, m\} \in SM, s < s'\]

In this equation, there is a non-linearity due to the assignment variable and the term in brackets. This non-linearity can be eliminated easily by adding another summation over m.

\[
\sum_{m \in \mathcal{M}} \sum_{l=s+1}^{s'+1} \text{dur}_{p,m,i} = (\text{ord}(s') \cdot Y_{p,s,1}^f - \text{ord}(s) \cdot Y_{p,s,1}^s - 1) \cdot \text{timeStep} \tag{65}
\]

\[\forall p \in P, s, s' \in S, st \in ST, s < s'\]
Forcing \( \text{dur}_{p,m,i} \) to be zero when \( X_{p,m} \) is false is already done before in case 1. The constraint that describes the case 4 is stated in equation (66).

\[
(\text{ord}(s') \cdot Y^f_{p,s',st} - \text{ord}(s) \cdot Y^s_{p,s,st} - 1) \cdot \text{TIMEstep} \leq \sum_{m \in M_{st,m}} \sum_{i=s+1}^{s'} \text{dur}_{p,m,i} + \text{card}(s) \cdot \text{TIMEstep} \cdot (2 - Y^s_{p,s,st} - Y^f_{p,s',st}) \quad (66)
\]

\( \forall p \in P, s, s' \in S, st \in ST, s < s' \)

**Case 5**

In the case 5, the product is not processed within the slot \( s \), the time duration should be zero. This can be ensured with one tight equality constraint. The production time of a product assigned to a particular machine is equal to the total duration over all slots, equation (67). This equation states also that the duration of an unassigned product is zero and together with constraint (66), it ensures that \( \text{dur}_{p,m,s} \) is zero when the product is not processed in a slot \( s \).

\[
\sum_{s \in S} \text{dur}_{p,m,s} = \tau_{p,m} \cdot X_{p,m} \quad \forall p \in P, m \in M \quad (67)
\]

One can think about another way of modeling this fifth case. Let’s assume that a product \( p \) starts in slot \( s \) and finishes in slot \( s' \) where \( s < s' \). Thus, \( \text{dur}_{p,m,s} \) is zero in all slots before \( s \) and all the slots after \( s' \). This can be written mathematically using a single equation (68).

\[
\sum_{m \in M_{st,m}} \sum_{i<s} \text{dur}_{p,m,i} + \sum_{m \in M_{st,m}} \sum_{i'>s} \text{dur}_{p,m,i'} \leq \text{card}(s) \cdot \text{TIMEstep} \cdot (2 - Y^s_{p,s,st} - Y^f_{p,s',st}) \quad (68)
\]

\( \forall p \in P, s,s', i \in S, st \in ST, i < s < s' < i' \)

With the constraints (63) to (68) and the equations that define the starting and finishing slot of a product we are able to calculate the time duration of a product in each slot. The variable duration model will be fully defined by adding another constraint to calculate the slot consumption, equation (69).

\[
a_s = \sum_{p \in P, m \in M} \frac{h_{p,m} \cdot \text{dur}_{p,m,s}}{60} \quad \forall s \in S \quad (69)
\]

**5.2.2 Improvements**

**5.2.2.1 Tightening constraints**

Tightening constraints are very useful when dealing with large-scale optimization problems due to their ability to speed up the computational time considerably. In this section, we will explore some of them that seem to be useful in practice. We start with linking the event binaries with the assignment binaries, since we know if a product is assigned to a machine then its starting and finishing time has to occur, as stated in equations (70) and (71).

\[
\sum_{s \in S} Y^s_{p,s,st} = \sum_{m \in M_{st,m}} X_{p,m} \quad \forall p \in P, st \in ST \quad (70)
\]

\[
\sum_{s \in S} Y^f_{p,s,st} = \sum_{m \in M_{st,m}} X_{p,m} \quad \forall p \in P, st \in ST \quad (71)
\]

The equations presented in the model formulation used \( Y^s_{p,s,st} \) and \( Y^f_{p,s,st} \) to calculate \( \text{dur}_{p,m,s} \). There is a different way to link \( Y^s_{p,s,st} \), \( Y^f_{p,s,st} \) and \( \text{dur}_{p,m,s} \) using the fact that the processing duration in a time slot is always less than the time step (\( \sum_{m \in M_{st,m}} \text{dur}_{p,m,s} \leq \text{TIMEstep} \)).
This idea will allow us to use the variable \( \text{dur}_{p,m,s} \) in order to put some restrictions on \( Y_{p,s,st}^s \) and \( Y_{p,s,st}^f \).

If a product is processed in slot \( s \) then the product started in a slot before \( s \) or within \( s \). Thus, one of the binaries \( Y_{p,s,st}^s \) is true in a slot before \( s \) or within \( s \) and all of the others (after the slot \( s \)) are 0, equations (72) and (73). Similarly, if a product is processed in the slot \( s \) then its finishing time is within slots after \( s \) (including \( s \)). Thus, one of the binaries \( Y_{p,s,st}^f \) is true in slots after \( s \) and all of the others are 0 as shown in equations (74) and (75).

\[
\sum_{m \in S} \sum_{t \in \text{TIMEstep}} \text{dur}_{p,m,s} \leq \sum_{s \leq t} Y_{p,s,st}^s \tag{72}
\]

\[
1 - \sum_{m \in S} \sum_{t \in \text{TIMEstep}} \text{dur}_{p,m,s} \geq \sum_{s < t} Y_{p,s,st}^s \tag{73}
\]

\[
\sum_{m \in S} \sum_{t \in \text{TIMEstep}} \text{dur}_{p,m,s} \leq \sum_{s \leq t} Y_{p,i,st}^f \tag{74}
\]

\[
1 - \sum_{m \in S} \sum_{t \in \text{TIMEstep}} \text{dur}_{p,m,s} \geq \sum_{s > t} Y_{p,i,st}^f \tag{75}
\]

Equation (76) shows that the total electricity consumption is equal the summation over each time slot of the production time multiplied by the power of each product and machine.

\[
\sum s q_s = \sum_{p,m} \frac{h_{p,m} t_{p,m} X_{p,m}}{60} \tag{76}
\]

### 5.2.2.2 Eliminating redundant binary variables

In this section, we will explore a different way to improve variable duration model. The improvement consists of eliminating some binary variables and defines the upper and lower bound of some continuous variables. This will lead to a tighter model with smaller number of variables in total.

The idea is to fix some of the variables \( Y_{p,st,s}^s \) and \( Y_{p,st,s}^f \) which we know cannot take value of 1 and then define the bounds of \( t_{p,st}^s \) and \( t_{p,st}^f \). With this aim in mind, we will run some small optimization problems upfront to the optimization to determine the upper and lower bound of the starting and finishing time of each product in the last stage. After that, we calculate the bounds in the other stages using this information.

We use the fact that in the last stage, the products are combined into groups and casted sequentially in a predefined sequence without any setup times. Thus, finding the upper and lower bound of the first product in each heat group will allow us to determine the bounds for the other products. In order to find the bounds in the previous stages, we use the bound on the actual stage, minimum transportation times, setup times between heats and maximum hold up times between the stages.

The algorithm used is presented in Figure 14. First, \( t_{p,st}^{s,\text{min}} \) and \( t_{p,st}^{s,\text{max}} \) are calculated for all products and stages using the restrictions in the last stage. And then, all the slots that are before \( t_{p,st}^{s,\text{min}} \) and the one that are after \( t_{p,st}^{s,\text{max}} \) are fixed to zero. Similarly, the variables \( Y_{p,st,s}^f \) are fixed using \( t_{p,st}^{f,\text{min}} \) and \( t_{p,st}^{f,\text{max}} \).
Algorithm 2 Eliminating some binary variables

Initialize \( t_{p,st}^{s,min} = t_{p,st}^{s,max} = 0 \) \( > \) The lower and upper bound of \( t_{p,st}^s \)

for \( st = st_1, h_g \in HG, p \in P_{h_g}^f \) \( > \) \( P_{h_g}^f \) is the first product in the heat group \( h_g \)

Set \( t_{p,st}^{s,min} = \min_{s,t} t_{p,st}^s \)

\[ \text{Scheduler constraints} \]

Set \( t_{p,st}^{s,max} = \max_{s,t} t_{p,st}^s \)

\[ \text{Scheduler constraints} \]

end for

The bounds of the other products in the last stage

for \( st = st_3, h_g \in HG \) do

for \( p \in P_{h_g}^f + 1, \ldots, P_{h_g}^l \) do \( > \) \( P_{h_g}^l \) is the last product in heat group \( h_g \)

\( t_{p,st}^{s,min} = t_{p-1,st}^{s,min} + \min_{m \in SM_{st,m,\tau_{p-1,m}}} \tau_{p,m} \) 

\( t_{p,st}^{s,max} = t_{p-1,st}^{s,max} + \max_{m \in SM_{st,m,\tau_{p-1,m}}} \tau_{p,m} \)

end for

end for

\( t_{p,st}^{s,max} = t_{p,st}^{s,max}, \forall st \) \( > \) An upper bound in the last stage is also an upper bound in the other stages

The upper bound in stage 1 to 3

for \( st \in st_1, \ldots, st_2, p \in P \) do

for \( st' = st - 1 \) do

\( t_{p,st'}^{s,min} = t_{p,st}^{s,min} - \min_{m \in SM_{st,m,\tau_{p,m}}} \tau_{p,m} - \min_{m \in SM_{st,m,\tau_{p,m}} \& m' \in SM_{st',m',\tau_{p,m'}}} \tau_{p,m'} \) \( > t_{m,m'} \) is the minimum transportation time between machines \( m \) and \( m' \)

end for

end for

The lower bound in stages 1 to 3

for \( st \in st_1, \ldots, st_3, p \in P \) do

for \( st' = st + 1 \) do

\( t_{p,st'}^{s,min} = t_{p,st}^{s,min} + \min_{m \in SM_{st,m,\tau_{p,m}}} \tau_{p,m} + \min_{m \in SM_{st,m,\tau_{p,m}} \& m' \in SM_{st',m',\tau_{p,m'}}} \tau_{p,m'} \)

end for

end for

Set \( t_{p,st}^{f,max} = t_{p,st}^{s,max} + \max_{m \in SM_{st,m,\tau_{p,m}}} \tau_{p,m} \)

Set \( t_{p,st}^{f,min} = t_{p,st}^{s,min} + \min_{m \in SM_{st,m,\tau_{p,m}}} \tau_{p,m} \)

for \( p \in P, st \in ST, s \in S \) do

\( > \) All slots that are after the upper bound of the starting time are fixed to zero

If \( s > \lceil \frac{f_{p,st}^{max}}{60} \rceil \) then \( Y_{p,s,st}^f = 0 \) end if

\( > \) All slots that are after the upper bound of the finishing time are fixed to zero

If \( s > \lceil \frac{f_{p,st}^{max}}{60} \rceil \) then \( Y_{p,s,st}^f = 0 \) end if

\( > \) All slots that are before the lower bound of the starting time are fixed to zero

If \( s < \lceil \frac{f_{p,st}^{min}}{60} \rceil \) then \( Y_{p,s,st}^s = 0 \) end if

\( > \) All slots that are before the lower bound of the finishing time are fixed to zero

if \( s < \lceil \frac{f_{p,st}^{min}}{60} \rceil \) then \( Y_{p,s,st}^f = 0 \) end if

end for

Figure 14: Elimination of redundant binary variables
6 Iterative framework

6.1 Motivation and description

In contrast to the decomposition approach and the monolithic model seen in the previous section, in this section we are going to develop a composition strategy as a different way to look at our optimization problem, which consists of the integration of energy cost to scheduling. This approach will be implemented as an iterative framework that involves many components.

Due to the nature of our problem and its importance in the industry, the goal is to make the framework as general as possible in order to be able to further apply it to other problems with similar inputs and outputs schemes, and problem characteristics (such as for example availability of marginal cost).

Many models have already been implemented in the literature in order to model either the energy management or the production planning within different area of application. For instance, energy-aware scheduling with optimization of a single price curve or some simple tariffs has been done by Mitra et al. [21], Ashok [19], Hait & Artigues [20], Nolde & Morari [34], Castro et al. [22]. However, the common lacking extension in all these works is consideration of more complicated tariffs. The reason is that optimization of purchase strategy for complicated cost structures is usually done as stand-alone optimizers, since these problems might be as complicated as the scheduling itself. Therefore, implementing a non-monolithic model as framework to integrate these two components will be very useful in the industry because they will be no need to reformulate the whole models. A few changes will be enough to have a model that takes into consideration both the cost and the production process constraints.

Having an iterative framework between energy management and production planning will allow flexibility on those models. They can be seen as two ‘black boxes’ that interact with each other and the overall goal is to minimize the summation (in our cases) of their objective functions. Thus, they can be replaced with other models in representing different problem classes without changing the whole framework algorithm.

Two algorithms are developed to implement the iterative framework. The first one, which will be presented in the next section, is based on multiparametric programming together with some coordination theories. The second one, which is a simplified version of first one using bilevel programming, is presented in section 6.3. Finally, a heuristic approach is presented in section 6.4 in order to further simplify this algorithm and to obtain the solutions for larger problem instances than those limited by computation performance of the rigorous solution approaches shown in sections 6.2 and 6.3.
6.2 Algorithm using multiparametric programming

6.2.1 Multiparametric programming

Multiparametric programming techniques for solving Mixed Integer Linear Programs (mp-MILP) can be used to solve EM model. It can be applied to any mixed integer programming problem with parameters in the right hand side of the constraints written in the standard form (SF).

\[
\begin{align*}
\min_{x,y} & \quad c^T \cdot x + d^T \cdot y \\
\text{s.t} & \quad A \cdot x + E \cdot y \leq b + F \cdot \theta \\
& \quad x \in X, y \in \{0,1\} \\
& \quad \theta_{\text{min}} \leq \theta \leq \theta_{\text{max}}
\end{align*}
\]

where \(c, d, A, E, b, F, \theta_{\text{min}}, \theta_{\text{max}}\) are constant vectors and matrices. \(x\) and \(y\) are respectively vectors of continuous and binary variables and \(\theta\) is the varying parameter.

EM is formulated as an economic flow network using a mixed integer linear program, and the variables and equations that describe it are presented in the appendix 1.1. The varying parameters, which are the load vector, are present in the right hand side of the constraints. The algorithm to solve such problems is explained in the literature [24],[26]. It consists of decomposing the mp-MILP into two subproblems. The first subproblem is a multiparametric linear program obtained by fixing the binary variables and the second subproblem is a mixed integer problem by relaxing the parameters to be unknown variables. The solution of mp-MILP is obtained after iteratively running these two subproblems. First, the initial space of the parameters is characterized by multiple regions with a fixed integer variables using mp-LP. In the second step, new values of integer variables are identified to get a better parametric solution in current regions (i.e. small value of the objective function). Moreover, the solutions of two different integers are compared to keep the better one.

In order to apply multiparametric programming technique to EM model we rewrite the model in its standard form (SF) using matrix representation. The idea is to regroup all the continuous variables and the binary variables in separate vectors and then transform the equality constraint to inequalities as they are the only constraints present in the standard form. Let \(m\) be the number of slots (i.e. \(m = |s|\)), to rewrite our program in the standard form, we define the variables \(x \in \mathbb{R}^{7m}\) (the flow in 4 sources of electricity: long term contract, TOU, day-ahead market, onsite generation; the flow in 2 sinks: demand and sale; onsite generation auxiliary variables) and \(y \in \{0,1\}^m\), the vector of parameters \(\theta\) that contains the load distribution, the cost vector \(C\) (we put a minus for the cost of the sale node because it is a profit) and the upper and lower bound \(b^u, b^l\) of \(x\) as following:

- \(x = (f_{1,1,15}, \ldots, f_{m,1,15}, f_{1,2,15}, \ldots, f_{m,2,15}, \ldots, f_{1,15,15}, f_{1,15,16}, \ldots, f_{m,15,16}, \ldots, f_{m,15,17}, g_{1,15,4}, \ldots, g_{m,15,4})^T\)
- \(y = (G_{1,15,4}, \ldots, G_{m,15,4})^T\)
- \(d = (0, \ldots, 0)^T\) with \(m\) components
- \(\theta = (\text{load}(s_1), \ldots, \text{load}(s_m))^T\)
- \(C = (c_{1,1,15}, \ldots, c_{m,1,15}, c_{1,2,15}, \ldots, c_{m,2,15}, \ldots, c_{m,15,15}, 0, \ldots, 0, -c_{1,15,17}, \ldots, -c_{m,15,17}, c_{1,15,17}^{\text{gen}}, \ldots, c_{m,15,17}^{\text{gen}})^T\)
- \(b^l = (f_{1,1,15}^{\text{min}}, \ldots, f_{1,1,15}^{\text{min}}, f_{m,1,15}^{\text{min}}, \ldots, f_{m,2,15}^{\text{min}}, \ldots, f_{m,15,15}^{\text{min}}, f_{1,15,16}^{\text{min}}, \ldots, f_{m,15,16}^{\text{min}}, \ldots, f_{m,15,17}^{\text{min}}, 1, \ldots, 1)^T\)
For more clarity we define some small matrices with dimension $m \times m$ that will used in the standard form of the problem: the identity matrix (ones in the diagonal) $I_m$, the zero matrix $O_m$, a matrix with ones under the diagonal $U_m$, a matrix with the flow max of the onsite generation in the diagonal $L$.

$$I_m = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & 1 \end{pmatrix}, \quad O_m = \begin{pmatrix} 0 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & 0 \end{pmatrix}, \quad U_m = \begin{pmatrix} 0 & 0 & 0 \\ 1 & \ddots & 0 \\ 0 & 0 & 1 \end{pmatrix}, \quad L = \begin{pmatrix} b^u_{i,4,1} & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & b^u_{i,4,|S|} \end{pmatrix}$$

We also define two matrices in order to model the constraints related to minimum run and down time of onsite generation. The first one $R^r$ has a number of $\text{min} r$ ‘ones’ in each row starting from the diagonal whereas the other one $R^d$, has a number of $\text{min} d$ ‘ones’.

$$R^r = \begin{pmatrix} 1 & \cdots & 1 & 0 & \cdots & \cdots & \cdots & \cdots & \cdots & 0 \\ 0 & 1 & \cdots & 1 & 0 & \cdots & \cdots & \cdots & \cdots & \cdots \\ \vdots & \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots \\ \vdots & \vdots & \vdots & \vdots & 0 & 1 & \cdots & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & 0 & 1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & 0 \end{pmatrix}, \quad R^d = \begin{pmatrix} 1 & \cdots & 1 & 0 & \cdots & \cdots & \cdots & \cdots & \cdots & 0 \\ 0 & 1 & \cdots & 1 & 0 & \cdots & \cdots & \cdots & \cdots & \cdots \\ \vdots & \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots \\ \vdots & \vdots & \vdots & \vdots & 0 & 1 & \cdots & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & 0 & 1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & 0 \end{pmatrix}$$

With the components and matrices defined above, we are able to write the standard form (SF) of energy management model. The matrices $A, E, b$ and $F$ respectively with the dimension $13m \times 7m$, $13m \times m$, $13m \times 1$ and $13m \times m$ are fully defined as stated below.

$$A = \begin{pmatrix} I_m & I_m & I_m & I_m & -I_m & -I_m & O_m \\ -I_m & -I_m & -I_m & -I_m & I_m & I_m & O_m \\ O_m & O_m & O_m & O_m & I_m & O_m & O_m \\ O_m & O_m & O_m & O_m & -I_m & O_m & O_m \\ O_m & O_m & O_m & O_m & I_m & O_m & O_m \\ O_m & O_m & O_m & O_m & -I_m & O_m & O_m \\ O_m & O_m & O_m & O_m & I_m & O_m & O_m \\ O_m & O_m & O_m & O_m & -I_m & O_m & O_m \\ O_m & O_m & O_m & O_m & I_m & O_m & O_m \\ O_m & O_m & O_m & O_m & -I_m & O_m & O_m \\ O_m & O_m & O_m & O_m & I_m & O_m & O_m \\ O_m & O_m & O_m & O_m & -I_m & O_m & O_m \\ O_m & O_m & O_m & O_m & I_m & O_m & O_m \end{pmatrix}, \quad E = \begin{pmatrix} O_m \\ O_m \\ O_m \\ O_m \\ -bgM \cdot I_m \\ I_m \\ I_m - U_m \\ -I_m \\ U_m \\ -L \end{pmatrix},$$

$$b^u = (f_{1,1,5}^\text{max}, \ldots, f_{m,1,5}^\text{max}, f_{1,2,5}^\text{max}, \ldots, f_{m,2,5}^\text{max}, f_{1,3,5}^\text{max}, \ldots, f_{m,3,5}^\text{max}, f_{1,4,5}^\text{max}, \ldots, f_{m,4,5}^\text{max}, f_{1,5,5}^\text{max}, \ldots, f_{m,5,5}^\text{max}, 0, \ldots, 0)^T$$
Where \( \mathbf{0}_m \) and \( \mathbf{1}_m \) are vectors of \( m \) components of ‘zeros’ and ‘ones’ respectively and \( k \) is the percentage reduction of onsite generated power due to startup.

### 6.2.2 Solving the energy management model

In this section, we will describe the solution of EM when solving it using linear programming and define some concepts that will be used in the algorithms of the framework. The idea is to fix the binary variables to have an LP problem that can be solved using any classical method such as simplex method. This can be done by fixing the binaries and solving the resulting LP problem many times (enumeration of the best possible binary solutions using multiparametric programming for a particular set up of the load demand, as shown in section 6.2.3).

EM is formulated as an economic flow network (or minimum cost flow network). The sources and sinks of electricity are represented with nodes; the cost in the arcs between the nodes corresponds to the price of the electricity of the source. Thus, the capacity of each arc represents the maximum amount of electricity that could be bought from this particular source.

As an optimization problem, EM gives an optimal solution with a particular structure. We assume that the prices of the electricity are different from one another (otherwise two sources can be grouped to form one source with one price in case the same flow restrictions apply to both sources). In the optimal solution, the active sources of the electricity should be all saturated (the flow is equal to the maximum or the minimum capacity of the source) except of one source, which will be flexible (the flow is strictly less than the maximum capacity of the arc). This happens basically because if two different sources are active, then the problem can be further optimized by reducing the electricity taking from the expensive source and increasing the one taking from the cheapest source. Therefore, there is only one flexible source in the optimal solution. This reasoning applies only to a situation with fixed binary decisions and corresponding Linear Programming problem.

Increasing the flow in the active source by one unit will cause an increase in the objective function by the price of this source. This is called marginal cost and in mathematics, it is defined for equations as well as for variables. It is the amount by which the objective function changes if the right hand side is increased by 1. In our case, it can be seen as a shadow price because it corresponds to the price of the electricity in a particular set up of the load demand.
An optimal solution defines one contract (cost) structure as a set of saturated sources with one flexible source of electricity. The flow within the flexible source can be increased until it reaches the maximum capacity of the arc without changing the cost structure and the marginal cost. Therefore, a cost structure is defined by an optimal basis, which defines the basic variables and non-basic variables [38] (for example using simplex method to solve the problem).

To sum up, a cost structure (contract structure) is defined by an optimal basis and within this contract structure the marginal cost stays the same if the optimal basis stays the same. The latter can be seen as a shadow price and it can be used as an indicator of the price of the electricity for a specific cost structure.

6.2.3 Algorithm description

The algorithm, as shown in Figure 15, starts with solving the energy management model using mp-MILP and taking the load vector as the investigated parameter. mp-MILP solution gives different optimal regions (i.e. upper and lower bound of the load parameter in each region). Each region is associated with an optimal basis as a known property of multiparametric programming solution, shown in literature [39]. Therefore, each region defines one particular cost structure (optimal for given load parameter values) that will allow evaluating the energy cost of any load distribution coming from the scheduler model. In those regions, the cost structures are known (fixed) and the optimal values of the binaries as well. mp-MILP provides then a distribution in type of a map of the optimal cost structure depending on the load parameter.

Mathematically speaking, EM will be transformed into a set of regions (equation 79) and affine functions (equations 77, 78) when applying the multiparametric programming technique. This is done following the algorithm of Dua and Pistikopolous [39] that leads to a K group of solutions as shown below:

\[
\begin{align*}
    x^{(k)} &= m^{(k)} + n^{(k)} \cdot \theta \\
    y^{(k)} &= \bar{y}^{(k)} \\
    H^{(k)} \cdot \theta &\leq h^{(k)}
\end{align*}
\]  

(77)  
(78)  
(79)

where \([m^{(k)}]_{7m \times 1}, [h^{(k)}], [n^{(k)}]_{7m \times m}\) and \([H^{(k)}]\) are matrices defined in the real space whereas \(\bar{y}^{(k)} \in \{0,1\}^m\) is the best value of \(y\) binaries found by mp-MILP in region \((k)\).

The value of the binary variable \(y\) which defines for our test cases the time slots where the onsite generation is running or not is fixed within each region to a particular value \(\bar{y}^{(k)}\). Similarly, the continuous variables \(g^{x}_{1,i5,i4}, \ldots, g^{x}_{m,i5,i4}\) that define the starting slot of onsite generation have a fixed value in each region (i.e. \((g^{x}_{1,i5,i4}, \ldots, g^{x}_{m,i5,i4}) = m^{(k)}(6m + 1: 7m)\), the last \(m\) components of \(m^{(k)}\), since they are defined by the \(y\).
In order to get the upper and lower bound of the load parameter in each region, which will be used in the second step, two simple linear optimization problems are solved. The first one provides the lower bound $\theta_{min}^{(k)}$ of load vector $\theta$ in the region $k$ whereas the second one provides the upper bound $\theta_{max}^{(k)}$.

$$\begin{align*}
\min_\theta \sum_s \theta_s & \quad \text{s.t. } H^{(k)} \cdot \theta \leq h^{(k)} \\
\max_\theta \sum_s \theta_s & \quad \text{s.t. } H^{(k)} \cdot \theta \leq h^{(k)}
\end{align*}$$

This is generally not true when dealing with dependent constraints but in this case it can be applied because the constraints $H^{(k)} \cdot \theta \leq h^{(k)}$ can be written as $\theta_{min}^{(k)} \leq \theta \leq \theta_{max}^{(k)}$ according to mp-MILP solution, which are independent constraints.
Thus,
\[
\text{argmin} \left\{ \sum_s \theta_s \mid H^{(k)} \cdot \theta \leq h^{(k)} \right\} = \text{argmin} \left\{ \sum_s \theta_s \mid \theta^{(k)}_{\min} \leq \theta \leq \theta^{(k)}_{\max} \right\} = \theta^{(k)}_{\min}
\]
And the same goes for the maximum value:
\[
\text{argmax} \left\{ \sum_s \theta_s \mid H^{(k)} \cdot \theta \leq h^{(k)} \right\} = \text{argmax} \left\{ \sum_s \theta_s \mid \theta^{(k)}_{\min} \leq \theta \leq \theta^{(k)}_{\max} \right\} = \theta^{(k)}_{\max}
\]

Therefore, the first step (red box) implemented in Matlab-YALMIP [40] in the algorithm provides for each region the best cost structure, the upper and lower bound of the load curve and a feasible point in the region (for example the middle point). This information will be used in the GAMS model in order to steer the production and get the optimal solution that takes into account both energy cost and makespan. With this aim in mind, there is a need for an hourly price in each region that characterizes the cost structure. This price is taken to be the marginal cost of the load. These prices will be retrieved in the next step of the algorithm.

The second step of the algorithm (green box), which is implemented in GAMS, starts with a loop over all regions. For each region, the marginal costs are calculated by running EM and fixing the load curve to one feasible point inside the region and also fixing the binaries to the values found in mp-MILP solution. This provides the production planning model with a single price level.

The production planning in each iteration will be done with respect to the cost of the energy by adding another component in the objective function of PP (The marginal costs multiplied by the load, \( \sum_s MC^{(k)}_s \cdot \theta_s \)) and one constraint that restrict the load to be inside the region \( \theta^{(k)}_{\min} \leq \theta \leq \theta^{(k)}_{\max} \). Therefore, PP gives the optimal load distribution within one particular cost structure.

The solutions provided by PP can be in some cases infeasible because the bound of the regions can be very tight restricting badly the load. Those infeasible regions can be skipped. For the feasible regions, PP gives the best load distribution within one cost structure as well as the optimal makespan.

There is at least one feasible region if the ranges of the parameter giving to mp-MILP are feasible for the production planning because the mp-MILP provides a complete map of varying parameters space. In order to get the real total cost including the makespan in each region (i.e. energy cost calculated using the full knowledge of EM, not just marginal cost), the energy management model is run taking the optimal load distribution coming from PP. Therefore, the best region (best cost structure) corresponds to the region with the minimum total cost and it is obtained together with the other variables of the problem using a direct comparison between different objective functions.

This algorithm has many different components and it involves running energy management model multiple times. There is another alternative to implement this framework using bilevel programming together with multiparametric programming without running EM in the second step and it will be described in the next section.
6.3 Algorithm using bilevel programming

6.3.1 Bilevel programming

6.3.1.1 Mathematical formulation

Bilevel programming problems involve a hierarchy of two optimization problems with two different decisions making. The outer optimization problem is called the upper level (or leader) and the inner optimization problem is called as the lower level (or follower). It has the following general form.

\[
\min_{x_u, x_l} F(x_u, x_l)
\]

such that

\[
x_l \in \arg\min_{x_l} \{ f(x_u, x_l): g_j(x_u, x_l) \leq 0, j = 1, \ldots, J \}
\]

\[
G_k(x_u, x_l) \leq 0, k = 1, \ldots, K
\]

\[
x_u \in X_u, x_l \in X_l
\]

where \(x_u\) and \(x_l\) represents the upper and lower level decision vector; \(F\) and \(f\) are the objective functions of the upper and lower level problems; \(G_k\) and \(g_j\) are the inequality constraints at the upper and lower levels; \(X_u\) and \(X_l\) are the bound constraints for \(x_u\) and \(x_l\). The upper objective function \(F\) is the objective function of the bilevel program whereas the lower objective is considered as a constraint.

6.3.1.2 Application to the case study problem

Many real world applications can be modeled using the general form described above. The structure of our problem can definitely be seen as a bilevel programming problem where the leader is the production planning (PP) model and the follower is energy management (EM) model, Figure 16. The scheduling model takes the first decision by optimizing its own objective function which takes into account both the amount of time needed to finish the production and some energy cost (including the deviation penalties). It provides then a load parameter to the energy management model which reacts rationally to leader decision and moves in a way that is personally optimal (minimize the cost of the energy).

![Figure 16: Integration of energy management and scheduling as a bilevel program](image-url)
This can be formulated mathematically using the same form described above:

\[
\min_{PP,v,EM,v} \text{makespan}(PP,v,EM,v) + dev(PP,v,EM,v) + EM_{\text{cost}}(PP,v,EM,v)
\]

such that

\[
EM,v \in \arg\min_{EM,v} \{EM_{\text{cost}}(PP,v,EM,v): EM_{\text{constraints}}(PP,v,EM,v)\}
\]

\[
PP_{\text{constraints}}(PP,v,EM,v)
\]

Where \(PP,v\) represent the production planning variables including the load variable, they are the upper level decision variables; \(EM,v\) represent the energy management variables and they can be seen as the lower level decision variables; \(PP_{\text{constraints}}\) and \(EM_{\text{constraints}}\) represent respectively the constraints of the production planning model and the constraints of the energy management model. \(dev(PP,v,EM,v)\) represents the deviation penalties paid when not committed to the agreed consumption of electricity (load).

The production planning constraints (\(PP_{\text{constraints}}\)) do not depend on the variables of energy management directly but they depend on \(PP,v\) which contains the load that affect indirectly the EM variables.

### 6.3.1.3 Solution approach

EM is parameterized by the variables of PP and precisely with the load curve. Therefore, EM can be seen as a multiparametric program with load parameters being also variables when looking at the bilevel program. One way of solving this problem is to insert the obtained set of regions (load restrictions and EM variables as a function of the load parameter) in the upper level problem and then get a set of mixed integer linear programming problems, which can be solved independently to optimality. The solution of the whole problem is then obtained by a direct comparison of the solution of these multiple MILPs [30].

This is done mathematically by introducing the expressions that are obtained from mp-MILP in the bilevel program formulation. To distinguish between the load vector \(\theta\) and the other variables of the production planning model \(w\) we put \(PP,v = (w,\theta)\) and for the energy management the variables are already defined \((x,y)\). Therefore, we obtain K independent MILPs as following:

\[
\min_{w,\theta,x^{(k)},y^{(k)}} \text{makespan}(w,\theta,x^{(k)},y^{(k)}) + dev(w,\theta,x^{(k)},y^{(k)}) + EM_{\text{cost}}(w,\theta,x^{(k)},y^{(k)})
\]

such that

\[
x^{(k)} = m^{(k)} + n^{(k)} \cdot \theta
\]

\[
y^{(k)} = \bar{y}^{(k)}
\]

\[
H^{(k)} \cdot \theta \leq h^{(k)}
\]

\[
PP_{\text{constraints}}(w,\theta,x^{(k)},y^{(k)})
\]

This program can be further simplified by replacing the variable \(x^{(k)}\) and \(y^{(k)}\) with their expressions and by removing them from the functions that are not related to them (makespan, deviation penalties and \(PP_{\text{constraints}}\)).
We also remove the variable \( w \) from the function \( EM\_cost \) because it has no effect on it. Thus, the \( K \) MILP programs become:

\[
\min_{w, \theta} \text{makespan}(w, \theta) + dev(w, \theta) + EM\_cost(\theta, m^{(k)} + n^{(k)} \cdot \theta, \tilde{y}^{(k)})
\]

such that

\[
\begin{align*}
\text{PP\_constraints}(w, \theta) \\
H^{(k)} \cdot \theta & \leq h^{(k)}
\end{align*}
\]

The energy management cost can be written explicitly using the cost vector (i.e. \( EM\_cost(\theta, x, y) = c^T \cdot x \)), therefore by substituting this expression in the program, a new formulation is obtained.

\[
c^T \cdot m^{(k)} + \min_{w, \theta} \text{makespan}(w, \theta) + dev(w, \theta) + c^T \cdot n^{(k)} \cdot \theta
\]

such that

\[
\begin{align*}
\text{PP\_constraints}(w, \theta) \\
H^{(k)} \cdot \theta & \leq h^{(k)}
\end{align*}
\]

The marginal cost of the load is now analytically defined and it can be calculated using the matrix \( n^{(k)} \) as following:

\[
MC^{(k)} = \frac{\partial EM\_cost(\theta, x, y)}{\partial \theta} = \frac{\partial (c^T \cdot (m^{(k)} + n^{(k)} \cdot \theta))}{\partial \theta} = (c^T \cdot n^{(k)})^T
\]

In order to get the same form of the production planning program solved in the previous algorithm, we replace the expression \( H^{(k)} \cdot \theta \leq h^{(k)} \) with the marginal cost and the constraints \( H^{(k)} \cdot \theta \leq h^{(k)} \) with \( H^{(k)} \cdot \theta \leq h^{(k)} \). Therefore, the program becomes:

\[
c^T \cdot m^{(k)} + \min_{w, \theta} \text{makespan}(w, \theta) + dev(w, \theta) + MC^{(k)}^T \cdot \theta
\]

such that

\[
\begin{align*}
\text{PP\_constraints}(w, \theta) \\
\theta_{\min}^{(k)} \leq \theta \leq \theta_{\max}^{(k)}
\end{align*}
\]

Thus having the matrices \( m^{(k)} \) and \( n^{(k)} \) in each region will enable to calculate the marginal costs without running EM model (i.e. \( MC^{(k)} = (c^T \cdot n^{(k)})^T \)). Together with this information and the bounds of the load, one can optimize the production planning within one cost structure and then just add to the value of objective function, the component \( c^T \cdot m^{(k)} \) to get the real total cost. This is done for each region. Besides, the region with the optimal cost structure corresponds to the region with minimum total real cost.

The bilevel formulation of the problem of integration of energy to scheduling proves that the marginal costs in each region and the bounds that define each region are enough information to obtain the optimal solution using an iterative framework. This formulation proves also that the first algorithm developed using just multiparametric programming is giving the optimal solution at the end. This is basically because the first algorithm is also using marginal cost as a cost signal in the production planning model, but the only difference is that those marginal costs are calculated using energy management model instead of multiparametric programming. Therefore, the calculations that are done using energy management model can be skipped and replaced by just the matrices \( m^{(k)} \) and \( n^{(k)} \) given by multiparametric programming.
6.3.2 Algorithm description

The first step of the algorithm that is done with Matlab for solving the EM problem using multiparametric programming stays the same as the previous one, but we also retrieve the value of the matrices $m^{(k)}$ and $n^{(k)}$ that will be used to calculate the marginal cost, as shown in Figure 17.

The second step is a simple loop over all the regions. In each iteration the marginal cost is given to the production planning model to be the price as proved in the previous section and finally the optimal solution is obtained with a direct comparison of the total cost in each region.

![Algorithm using bilevel programming](image)

**Figure 17**: Algorithm using bilevel programming

6.4 Heuristic approaches

Two algorithms have been implemented to solve non-monolithic integration of energy management and production planning. These algorithms have been proven to give the optimal solution, but they are based on the multiparametric programming technique. There are many regions that are infeasible during the production scheduling. Therefore, it is inefficient to compute those regions in the first place.
Multiparametric programming performs badly when dealing with real world instances especially when binaries are involved in the problem. The computational time (see section 7.3.1) is growing exponentially which makes it difficult to run big instances.

To avoid the computational time difficulties using the multiparametric programming, a heuristic is developed, as shown in Figure 18. The idea is to provide one first load curve that is obtained by optimizing the makespan and the deviation penalties to the energy management model and then by using some cost signal between this model and the production planning model, it will be possible to steer the production scheduling towards the cheapest time slots.

The coordination signal used is the marginal cost because it is a good indicator of the price of electricity in each time slot. This signal can also be seen as a shadow price since it is the price of the electricity for a particular set up of the load.

In order to avoid jumping too far from a good solution; we used fixed point iteration method with which consists of taking a step as a linear combination between the previous price and the actual marginal cost. Therefore, the cost signal in each iteration is calculated as following:

$$p_s^{(k)}(k) = \gamma \cdot p_s^{(k-1)} + (1 - \gamma)MC_s^{(k)} \quad \forall s$$

Where $k$ is the number of iteration and $\gamma$ is a real value between 0 and 1.

In each iteration, the marginal cost is given to the production planning model as a price curve and this latter gives an optimal load distribution taking into consideration these prices, which will be again the input of EM. The solution obtained in each iteration, the load distribution and its optimal cost structure, is stored in order to find the best solution by performing a direct comparison of the total cost (makespan+deviation_penalties+Energy cost).

![Figure 18: Heuristic approach as an iterative framework to solve the problem of integration](image-url)

The algorithm is stopped when the load curve stays the same. This criterion is chosen because when the load curve is similar to the load in the previous iteration then the solution provided
by energy management model will be the same (the marginal cost will be the same as well). This will lead to the same load distribution again, so there is no improvement in the algorithm anymore. Besides, a maximum number of iteration is defined in case the algorithm does not converge towards a fixed solution.

In this heuristic, there is no insurance of convergence towards the optimal solution, in each iteration the total cost can be improved or it can be worsen. The marginal sots give only a direction where to move and do not tell the step to take in this direction.
7 Numerical case study results

In this section, all the tests are run using GAMS 24.1.2 and CPLEX 12.5.1 solver on a personal machine with Intel(R) Xeon(R) CPU (4 cores) 2.13GHz processor and 4 GB RAM.

7.1 Detailed Scheduling of electric arc furnace

In order to evaluate the advantage of using the detailed EAF model in the industry in terms of energy cost savings, this model is integrated to one of the monolithic models in the literature [7] including the other stages. Therefore, the numerical results displayed in this section are taking into consideration the energy cost of the other melt shop stages.

The model is tested on various problem sizes. We have also investigated how parameters affect the solution. Running the program with and without the detailed EAF model will allow us to see the potential behind having a detailed scheduling of EAF. Nevertheless, instead of using the exact calculation of the energy explained in the description of the EAF model, which involve many non-linearities, thus a complex program to run, we just used the heuristic implemented to evaluate the energy cost.

The problem was solved for different number of products. The optimization strategy used is electricity cost in order to see the potential of the model for cost savings. Besides, the model is solved fixing some binary variables (assignment variables and sequence variable which define the precedence between heats) in order to get the solution in a reasonable time.

Table 2 gives the values used for some of the parameters in EAF stage. Those parameters are basically the limitation on the power, the minimum and maximum duration of each smelting step and the load duration. They are presented to give an idea about the limitation made without being exactly accurate (different values are also used but they close to the ones shown in the table).

Table 2: The numerical value of the parameters

<table>
<thead>
<tr>
<th>$T_{load}$</th>
<th>$\rho_m^{max}$</th>
<th>$\rho_m^{min}$</th>
<th>$h_{p,m}^{min}$</th>
<th>$h_{p,m}^{max}$</th>
<th>$h_{p,m,i}^{min}$</th>
<th>$h_{p,m,i}^{max}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>5 min</td>
<td>100MW</td>
<td>0MW</td>
<td>71min</td>
<td>100min</td>
<td>16min</td>
<td>24min</td>
</tr>
</tbody>
</table>

Table 3 gives some statistics regarding the performance of the detailed scheduling of EAF model compared with a model without detailed EAF scheduling. The table shows a comparison in terms of number of variables and the computational time.

In this example, 7 products and 12 hours are taken and the same amount of energy is provided in both models. From the table it can be easily seen that the number of binary variable increased in the detailed EAF as well as the computational time. The model with detailed EAF is executed with the heuristic, therefore the energy cost given 204408.0276 is not the exact cost.
Table 3: Performance of detailed scheduling of EAF model

<table>
<thead>
<tr>
<th>Model</th>
<th>Without detailed EAF</th>
<th>With detailed EAF</th>
</tr>
</thead>
<tbody>
<tr>
<td>Binary variables</td>
<td>2128</td>
<td>2968</td>
</tr>
<tr>
<td>Continuous variables</td>
<td>3060</td>
<td>3284</td>
</tr>
<tr>
<td>Total variables</td>
<td>5188</td>
<td>6252</td>
</tr>
<tr>
<td>Equations</td>
<td>16807</td>
<td>18361</td>
</tr>
<tr>
<td>CPU time</td>
<td>6.219000416</td>
<td>6.49099967</td>
</tr>
<tr>
<td>MIP solution</td>
<td>206486</td>
<td>204408</td>
</tr>
<tr>
<td>Nodes</td>
<td>3767</td>
<td>3952</td>
</tr>
<tr>
<td>Iterations</td>
<td>39093</td>
<td>25897</td>
</tr>
</tbody>
</table>

Table 4 gives the optimization result of some test cases. From the results we conclude that depending on the model we have, we can make a profit or a loss by using the detailed EAF model. For example, if we take the model with 10 products and 12 hours, we make a profit of 1042€ and the error made by using the heuristic is 93.633€. The solutions for the model 4/12 are shown in the Gantt charts in Figure 19.

Table 4: Optimization results of detailed scheduling of EAF stage

<table>
<thead>
<tr>
<th>Model Products/hours</th>
<th>Without detailed EAF (cost1) [€]</th>
<th>With detailed EAF using heuristic, (cost2) [€]</th>
<th>Exact energy cost (cost3) [€]</th>
<th>Heuristic error (cost2-cost3) [€]</th>
<th>Cost savings (cost1-cost3) [€]</th>
</tr>
</thead>
<tbody>
<tr>
<td>4/12</td>
<td>4214.250</td>
<td>4458.313</td>
<td>4835.500</td>
<td>377.188</td>
<td>-621.25</td>
</tr>
<tr>
<td>7/12</td>
<td>9766.500</td>
<td>9302.973</td>
<td>9748.024</td>
<td>445.051</td>
<td>18.476</td>
</tr>
<tr>
<td>10/12</td>
<td>15267.875</td>
<td>14132.125</td>
<td>14225.758</td>
<td>93.633</td>
<td>1042.117</td>
</tr>
</tbody>
</table>

Figure 19: Production schedules for the model with detailed EAF (bottom) and without it (Top)

7.2 Monolithic model

7.2.1 “Duration variable” model’s performance

Two new monolithic models have been presented in the previous sections (5.1 and 5.2) in order to integrate energy to the production planning model. The first model is the “slot neighborhood” model, which seems to be comparable to the models developed in the literature in terms of the number of variables and equations. This model was implemented and its computational results are similar to the models in the literature, therefore there is no need to show its results because it does not provide any improvement to the existent models and it was just a step in order to come up with a more efficiently modeled version, which is the
“duration variable” model. Thus, in this section, only the results of running this second model are shown.

In this section, a performance comparison between “duration variable” model and one of the models in the literature will be presented. Say what is your experiment setup – that you want to run different test cases using different problems sizes in terms of number of heats and heat groups, different number of hours, the EM input data always stays the same etc. Please explain your strategy for different test cases.

Table 5 shows the obtained results and the statistics characterizing each model (i.e. total variables, total equations and CPU time). The comparison is done between the duration variable model (the new model) and a model taken from the literature [12](the old model). In the first part of the table all assignment variables are fixed except for the instances 12/6 and 8/3 where just some of them are fixed (assignment in the first three stages). Fixing assignment means that products with an even number (i.e. p2, p4…etc.) are assigned to machines with even number (i.e. EAF2, AOD2, AF2 and CC2) and the same goes for odd heats and odd machines. The sequence variables in the first part of the table ‘not-fixed binaries’ are left free.

In the second part of the table, the sequence variables are also fixed (each product is forced to proceed another product with a higher number). Different instances have been tested with different data sets and in all the tests both models give the same optimal solution and the same decision variables values.

Table 5: Performance comparison between monolithic models

| No. Of heats
(hours/products) | Model type | Total variables | Total equations | CPU Time(s) | Gap (%) | Makespan | Energy cost [€] |
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>24/20</td>
<td>New</td>
<td>10308</td>
<td>70771</td>
<td>19 269</td>
<td>0,06</td>
<td>1395,17</td>
<td>76832,82</td>
</tr>
<tr>
<td></td>
<td>Old</td>
<td>29508</td>
<td>102341</td>
<td>85 576</td>
<td>0,06</td>
<td>1391,37</td>
<td>35791,41</td>
</tr>
<tr>
<td>24/10</td>
<td>New</td>
<td>4948</td>
<td>33373</td>
<td>4 928</td>
<td>0,15</td>
<td>1391,37</td>
<td>50728,5</td>
</tr>
<tr>
<td></td>
<td>Old</td>
<td>14548</td>
<td>49155</td>
<td>22 871</td>
<td>0,6</td>
<td>720</td>
<td>50747,1</td>
</tr>
<tr>
<td>12/10</td>
<td>New</td>
<td>2836</td>
<td>15181</td>
<td>3 032</td>
<td>9,72</td>
<td>720</td>
<td>31536,49</td>
</tr>
<tr>
<td></td>
<td>Old</td>
<td>7636</td>
<td>25923</td>
<td>49 344</td>
<td>11,98</td>
<td>720</td>
<td>10905,58</td>
</tr>
<tr>
<td>12/6</td>
<td>New</td>
<td>1684</td>
<td>8656</td>
<td>8,70</td>
<td>0</td>
<td>720</td>
<td>53963,72</td>
</tr>
<tr>
<td></td>
<td>Old</td>
<td>4564</td>
<td>15100</td>
<td>675,38</td>
<td>0</td>
<td>720</td>
<td>10905,58</td>
</tr>
<tr>
<td>8/3</td>
<td>New</td>
<td>648</td>
<td>2745</td>
<td>2,37</td>
<td>0</td>
<td>473</td>
<td>10905,58</td>
</tr>
<tr>
<td></td>
<td>Old</td>
<td>1608</td>
<td>5079</td>
<td>17,30</td>
<td>0</td>
<td>473</td>
<td>10905,58</td>
</tr>
</tbody>
</table>

It can be easily seen how the new model performs compared to the old one. First, the number of the variables is reduced by at least 30%, whereas the number of equations is reduced by at least 60%, which will affect the computational time (as an example, for the instance 24 hours and 20 products without fixing the binaries, the number of variables is decreased by 35% and the number of equations is decreased by 69%). This can be seen by comparing the CPU time of each model, for example, the first instance in the table shows that the new model can reach a gap of 0.06% in just 19269.46 seconds while it takes 85576.77 seconds using the old model.
Thus, the new model performs much better than the old one and in some cases the solutions can be obtained in just a few minutes instead of hours. In general, for the investigated cases the new model shows an improvement of around one order of magnitude in terms of computational time.

The tests done in this section, are based on one particular input parameters, but different input parameters also should be taken into account when running different test cases because for MILP models the input parameters might have significant impact on the model performance (even if the model statistics stay the same).

7.2.2 Improvement’s effect on the model

The duration variable model has been numerically proven to be faster than the reference model available in the literature. But by looking at the computational time, it can be concluded that there is still a need for a strategy to speed up the program in order to compute the solution until reaching optimality. With this goal in mind, an algorithm, as described in Figure 14, is developed in order to perform some calculations before running the whole optimization problem. This algorithm eliminates some binaries upfront, using the process knowledge and the problem structure, by fixing them to specific values.

Table 6 shows how the algorithm performs and its effect on the model. As an example, for 24 hours and 20 products, the model runs either with or without this algorithm for 300 seconds, and it can be seen that 1025 binaries are eliminated, which corresponds to 27.4 % of the initial number of the binararies. The gap has decreased from 22% to 15.6%.

It can be concluded that this algorithm performs well and it speeds up the optimization program.

Table 6: Elimination of binary variable's effects on the monolithic model

<table>
<thead>
<tr>
<th>No. Of heats (hours/products)</th>
<th>Model type</th>
<th>Total binary variables</th>
<th>CPU Time (s)</th>
<th>Gap (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>24/20</td>
<td>Without eliminating binaries</td>
<td>5201</td>
<td>943.82</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Eliminating event binaries</td>
<td>3446</td>
<td>251.77</td>
<td>0</td>
</tr>
<tr>
<td>24/15</td>
<td>Without eliminating binaries</td>
<td>3738</td>
<td>301.35</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td>Eliminating event binaries</td>
<td>2713</td>
<td>301.57</td>
<td>15.6</td>
</tr>
</tbody>
</table>

7.3 Iterative framework

7.3.1 Multiparametric programming results

Multiparametric programming is the main core of the framework developed in this thesis. It is used as a first component in the two algorithms described in section 6. mp-MILP solver is available as a toolbox in Matlab and it can be used as a simple function that takes the model description as inputs and gives back the mp-MILP solutions. This toolbox is called YALMIP and it can be used to model and solve different optimization problems and provides the user with different functions to retrieve all the information needed [25]. Therefore, Matlab is used to run mp-MILP and then the solutions are stored in GDX files that will be used in GAMS.
Table 7 shows the computational time of mp-MILP on EM model shown in Appendix x, with different instances as well as the number of regions found during the solution process. The tests are done by taking the same input data as in the previous monolithic models, so that the solutions can be compared. A restriction is put on the load to be under 200 MWh before running the mp-MILP, which is practically a correct value because the specific consumption of each slot does not exceed this value due to production process constraints and input data.

One can easily see that the computational time and the number of regions are growing exponentially and it becomes hard to compute the solution when the dimension of the problem grows bigger. As an example, for 6 hours instance it took almost 8 days to get the total regions, which is far from being practical knowing that this program should be run each day in order to schedule the production accordingly to the hourly price that change every day and also knowing that the time horizon is 24 hours instead of 6 hours.

The computational problem is related to the nature of the algorithm behind mp-MILP and its ability to handle the binary variables (It is based on enumeration technique to find the best binary value in each region). Therefore, there is a need for a heuristic approaches to speed up the program and to be able to compute the real world instances.

One way to speed up the multiparametric programming is to provide a tight bound of the load. This idea has been practically proven with some test cases, but the bound should be carefully chosen according to the production planning constraints.

<table>
<thead>
<tr>
<th></th>
<th>2h</th>
<th>3h</th>
<th>4h</th>
<th>5h</th>
<th>6h</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU time (s)</td>
<td>54.606</td>
<td>67.462</td>
<td>1471.47</td>
<td>38302.20</td>
<td>629709.24</td>
</tr>
<tr>
<td>Nb of Regions</td>
<td>18</td>
<td>85</td>
<td>461</td>
<td>2809</td>
<td>10639</td>
</tr>
</tbody>
</table>

Multiparametric programming provides a map of the space of the varying parameters and in our case it corresponds to multiple optimal contract structures, which are characterized by an upper and lower bounds of the load where this contract structure applies while being the optimal one. To show the regions obtained in a graphical way, 2 slots example is presented while shutting down the onsite generation in order to have the energy management model as a linear program. Figure 20 shows the results found, where one can extract the load limitations that define each region. In this figure the number of regions is 9 instead of 18 because the onsite generation is removed.
To see the structure of the solution when using onsite generation, the binary variable that defines the slots where the onsite generation is running is plotted as a function of the load curve. Figure 21 in the left side shows how this decision in the first slot should be made accordingly with the load values (if the specific consumption is less than 40 MWh, then the onsite generation should be turned off. But if it is greater than 40 MWh, the onsite generation should be turned on). Similarly, Figure 21 in the right side shows the decision for the second slot.

Depending on the load position the decision to run the power plant or to shut it down can be different and mp-MILP provides the optimal decision that could be made for a particular set up of the load.

### 7.3.2 Algorithm’s results

After running mp-MILP and storing the information related to each cost structure in a GDX file, a loop over all regions is performed following the algorithm described in section 6.2.3. This algorithm has many components and in this part the focus will be on the component that
is done in GAMS since the results in the other component are already presented in the previous section.

In order to verify whether the algorithm gives the same optimal value as the monolithic model or not, both models are run using the same data inputs.

Table 8 shows different test results that have been made for a number of slots between 2 and 5. This number of slots is practically small to schedule the production; therefore different time steps will be used for each instance to get a feasible solution (i.e. 90min, 120min, 150min and 210min). The load distribution and the total cost including the makespan in each case. In the table, the CPU time is also presented and it corresponds to the whole second step (including the infeasible regions) of the algorithm described in section 6.2.3 (denoted Algo in the table). Therefore, to get the total CPU time, the time of multiparametric programming should be included. The monolithic model is also run in order to compare both solutions.

Table 8 also shows that the number of feasible regions is small compared to the number of the existing regions. As an example, for the instance 5h/2p there is 2809 regions and only 8 of them are feasible which is considered as an advantage and a disadvantage at the same time. It is an advantage because the production planning model will be run only for few times and as a disadvantage because it is a waste of time to compute the other regions using mp-MILP.

Table 8: Iterative framework's results

<table>
<thead>
<tr>
<th>Slots</th>
<th>Products</th>
<th>Model type</th>
<th>Time Step [min]</th>
<th>Feasible Rgs</th>
<th>CPU Time[s]</th>
<th>Total cost [€]</th>
<th>Load Distribution [MWh]</th>
</tr>
</thead>
<tbody>
<tr>
<td>5 slots</td>
<td>2p</td>
<td>Algo.</td>
<td>8</td>
<td>1397.68</td>
<td>0.51</td>
<td>45523.6</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Monolithic</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>3p</td>
<td>Algo.</td>
<td>90</td>
<td>1308.76</td>
<td>0.62</td>
<td>43420.4</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Monolithic</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>4p</td>
<td>Algo.</td>
<td>21</td>
<td>1308.38</td>
<td>3.15</td>
<td>42953.8</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Monolithic</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4 slots</td>
<td>2p</td>
<td>Algo.</td>
<td>8</td>
<td>170.2</td>
<td>0.56</td>
<td>36029.9</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Monolithic</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>3p</td>
<td>Algo.</td>
<td>120</td>
<td>200.5</td>
<td>0.66</td>
<td>35449.1</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Monolithic</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>4p</td>
<td>Algo.</td>
<td>4</td>
<td>221.41</td>
<td>1.00</td>
<td>46544.4</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Monolithic</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3 slots</td>
<td>2p</td>
<td>Algo.</td>
<td>5</td>
<td>30s</td>
<td>0.39</td>
<td>26278.6</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Monolithic</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>3p</td>
<td>Algo.</td>
<td>150</td>
<td>31</td>
<td>0.41</td>
<td>34367.5</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Monolithic</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2 slots</td>
<td>2p</td>
<td>Algo.</td>
<td>210</td>
<td>7</td>
<td>2.12</td>
<td>26138.5</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Monolithic</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

As expected, the algorithm gives the same optimal total cost as the monolithic model, but in terms of the computational time it is much worse. The load distribution given by the algorithm is slightly different than the one given by the monolithic model. This is normal because there is many load curves for a single solution (for the same total cost one can get different makespan and energy cost).
7.3.3 Heuristic Framework

The previous section shows that the algorithm for non-monolithic integration of the energy management and production planning is giving the optimal solution, but it is not possible to apply it to big instances due to computational time limitations. In this section, the results of the heuristic implemented as an alternative for this algorithm is presented.

Table 9: Heuristic's results for non-monolithic integration of EM and PP

<table>
<thead>
<tr>
<th>No of heats (hours/products)</th>
<th>Model type</th>
<th>No. of Iterations</th>
<th>EM cost [€]</th>
<th>Makespan</th>
<th>Total cost</th>
<th>CPU Time [s]</th>
<th>Tradeoff (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>24/20</td>
<td>Heuristic</td>
<td>4</td>
<td>76842.25</td>
<td>1395.17</td>
<td>78237.42</td>
<td>349</td>
<td>0.012</td>
</tr>
<tr>
<td></td>
<td>Monolithic</td>
<td></td>
<td>76832.82</td>
<td>1395.17</td>
<td>78227.99</td>
<td>22.21</td>
<td></td>
</tr>
<tr>
<td>12/6</td>
<td>Heuristic</td>
<td>10</td>
<td>32920.98</td>
<td>719.9</td>
<td>33640.88</td>
<td>21.3</td>
<td>4.11</td>
</tr>
<tr>
<td></td>
<td>monolithic</td>
<td></td>
<td>31536.49</td>
<td>720</td>
<td>32256.49</td>
<td>8.70</td>
<td></td>
</tr>
<tr>
<td>8/3</td>
<td>Heuristic</td>
<td>15</td>
<td>10919.92</td>
<td>475.1</td>
<td>11395.02</td>
<td>20.3</td>
<td>0.14</td>
</tr>
<tr>
<td></td>
<td>monolithic</td>
<td></td>
<td>10905.58</td>
<td>473</td>
<td>11378.58</td>
<td>2.37</td>
<td></td>
</tr>
</tbody>
</table>

Table 9 shows some of the tests that have been done. It provides the results of the monolithic model as well in order to see the performance of the heuristic and to compare the heuristic’s solution with the optimal one. Tradeoff is a rate to express the difference between the total cost (summation of energy cost and makespan) of the heuristic and the monolithic model whereas number of iterations is the total number of iterations performed. The value of the $\gamma$ parameter used in the fixed iteration point method is 0.6.

From the table, we conclude that the heuristic performs much better compared to the multiparametric programming algorithms in terms of computational time. One can see easily that the heuristic was able to find a good solution in a reasonable amount of time. As an example, for the instance 24 hours and 20 products, which is impossible to compute using mp-MILP, the computational time is just 349 seconds and the percentage of the difference between the total cost of the heuristic solution and the monolithic one is 0.012%, which is a small value. This conclusion can be seen for all other realized tests.

Figure 22: Heuristic behavior for 8 products and 3 hours
The convergence of the heuristic and its behavior depend on the data used in the model. For some instances, it gives a good solution in just 2 iterations whereas it takes more than that for other data to stabilize in one solution. Figure 22 shows the total cost for the instance 8 hours and 3 products for each iteration. The best solution is obtained in 12 iterations and it is close to the optimal solution.

It can be concluded that the heuristic performs very well and it gives a good solution if there is a need for a non-monolithic model. This heuristic is more practical in the industry and it can be integrated to the existent components without complex changes in the formulation.
8 Summary and conclusions

It has been recognized that the scheduling model needs to be extended with energy-awareness in order to achieve full profitability. Therefore, the main task of the thesis’s project is to combine the production planning with energy management in an intelligent way to get the solutions in a reasonable time. Besides, some improvements should be done to take a full advantage of all iDSM capacities.

First, a detailed Electric Arc Furnace of a stainless steel stage is developed for more control of the production process in the first stage. The detailed EAF model appears to be valuable in some cases in term of cost savings. This model offers the possibility to adjust the electricity power in order to finish the production earlier or later. This flexibility allows us to shift loading operations to off-peak in order to decrease the cost of the energy.

Nevertheless, if we compare between the computational time spent and the profit achieved we conclude that it is more practical to use the model without detailed EAF. In real life problems, the duration of a loading operation is up to 5min and with this small duration it is preferable to consider the whole product instead of looking at each smelting task separately. In the industry, a good solution is not always the optimal solution; a good solution could be a solution which is close to the global optimum obtained in a reasonable time. In addition, due to uncertainties in the production process itself the benefit of detailed EAF scheduling might not be achievable due to e.g. process disturbances.

As a second step for the integration of energy management and production planning, two monolithic models are developed. The first one seems to have a comparable performance compared to the models that exist in the literature whereas the second one has a considerable advantage in terms of computational time and the number of equations and variables, which are small. In some cases, the solution can be obtained in a few minutes instead of hours.

Since the size of the real-world instances is very large, there is a need for some kind of strategy to reduce the computational time. In contrast to traditional approach (decomposition, such as e.g. Benders) applied to the monolithic models to get the solutions in a reasonable amount of time, new strategies have been investigated. These strategies have the advantage of being non-monolithic, thus they can be integrated to similar models in the industry without changing the formulations of the problem. They have been implemented using bilevel programming, multiparametric programming in order to solve the problem in an iterative framework.

Two algorithms have been implemented to provide an iterative framework. But, both of them perform badly when solving the real world instances. In large instances, it is even impossible to solve without employing some heuristic methods. These algorithms provide a new approach that has never been implemented in the literature and they are based on strong mathematical background. Moreover, they could be further simplified by adapting the multiparametric programming technique to our specific problem.

Finally, a heuristic is implemented in an iterative framework based on coordination theory. This heuristic provides a good solution close to the optimal one in a reasonable time. Moreover, it has the advantage of being simple and easy to implement.
9 Further Work

9.1 Improving multiparametric programming

There are many improvements that could be integrated to the models implemented in this thesis. First, multiparametric programming has been used in the iterative framework without any knowledge about the constraints within the production process. Modifying the algorithm of multiparametric programming to adapt it to our specific problem could be one possible way to solve the computational time limitations. The goal is to obtain a small number of regions without enumerating the whole space of the varying parameter.

One mechanism that could reduce the number of regions is to tweak the mp-MILP algorithm in a way that it stops partitioning the regions (stops the enumeration at some point) and then gives the regions that are found.

One heuristic that could be developed to speed the computational time of mp-MILP is to run the production planning problem only with makespan optimization and then take some value up and down of the obtained load distribution. This will give a tight range on the parameter, which means less computational time. Besides, mp-MILP will provide few number regions where the production process could be optimized.

Another heuristic could be to fix the binary variables in EM to have a linear program, which can be solved using mp-LP much faster. This can be done by looking at the history of running EM and take the value of the binaries (decisions to run the onsite generation) that seem to have a benefit.

9.2 Approximation of multiparametric programming technique

One can also think about a way to approximate the multiparametric programming technique with a simple method that could provide the EM’s cost structures. This can be developed as a heuristic and it is valid under some assumptions.

We assume that the binary variables are fixed and that the time slots are independent to each other (modifying the load in one slot does not affect the other slots). The idea is to come up with a portfolio of different prices, which are equal to the marginal costs and the intervals where they can be applied. These intervals correspond to the regions where the contract structure stays the same.

Figure 23 shows an example with 3 sources of electricity, each one with maximum capacity, minimum capacity and the prices. The goal is to obtain the marginal cost and the intervals where they can be applied.

This can be done using the sensitivity information given by GAMS or CPLEX solver. For a given time slot, a load of 1 MWh is taken and then EM is run to get the corresponding cost structure, the price (marginal cost) and the upper bound of the load to stay within this cost structure (Upload). This upper bound of the load can be retrieved only using some modern solvers like Cplex or Gurobi. The load is then increased by Upload to cross the region where this cost structure is the same and then EM is run again. The condition to stop is when the load in all slots reaches the maximum load, which is calculated up front.
Figure 23: The marginal cost and its applied intervals for a particular example

This information is given to production planning model. The latter tries to optimize the schedule by taking the price equal to the obtained marginal cost and restrict the load to be inside the region.

### 9.3 Improvement of the heuristic approach in the framework

In section 6.4, a simple heuristic is implemented to steer the production planning towards the cheapest time slots using marginal cost signal. This heuristic can be further improved by adding another step to the algorithm.

In this section, it is assumed that the heuristic is run which provides the best solution, the bounds where the cost structure of this solution stays the same and the marginal cost. The idea is to put restrictions on the load in the production-planning model this time in order to find the optimal load distribution within one cost structure (i.e an improvement of the solution found in the heuristic). The total energy of this solution can be retrieved using EM a second time.
References


[7] Hubert Hadera and Iiro Harjunkoski, "Continuous-time Batch Scheduling Approach for Optimizing Electricity Consumption Cost," *ABB Corporate Research 2013*


[16] Pedro M. Castro and Ignacio E. Grossmann, "n efficient MILP model for the short-term scheduling of single stage batch plants," *Department of Chemical Engineering, Carnegie*
Mellon University, Pittsburgh, PA 15213, USA.


[37] Z. Caner Taşkın, "Benders Decomposition," *Department of Industrial Engineering, Boğaziçi University*.


[40] Johan Löfberg, "YALMIP : A toolbox for modeling and optimization in MATLAB," *Automatic Control Laboratory, ETHZ*. 


Appendix A : Production planning model

The notation and the equations of the model are presented for a better understanding of the model.

Sets:
- \( P \) – heats; \( HG \) – heat groups; \( HGP(P) \) – subset of heats \( p \) mapped to corresponding heat group \( hg \); \( L(P) \).
- \( F(P) \) – subsets of heats \( p \) cast last and first in a heat group \( hg \); \( M \) – equipment; \( S \) – time slots; \( E \) – electricity prices; \( ST \) – production stages; \( SM(ST, M) \) – production stages \( st \) mapped to corresponding equipment \( m \).

Variables:
- \( t_{m,p}^f, t_{m,p}^s \) – starting/finishing time of heat \( p \) on equipment \( m \); \( t_{st}^f, t_{st}^s \) – starting/finishing time of heat \( p \) at stage \( st \); \( q_p \) – electricity consumed in a time slot \( s \); \( A_{p,st}, B_{p, st}, C_{p, st}, D_{p, st}, E_{p, st}, F_{p, st} \) – binary, true when heat \( p \) is processed on equipment \( m \) in the time slot \( s \) (Fig. 1); \( h_{st, p, p'}, c_{p, st} \) – auxiliary continuous variables; \( X_{m, p} \) – binary, true when heat \( p \) is processed on equipment \( m \); \( V_{st, p, p'} \) – binary, true if heat \( p \) processed after \( p \) on stage \( st \); \( b_s \) – buffer level for allowed deviation from committed load; \( b_s^u, b_s^l \) – upper/lower bounds for the buffer; \( c_s^p, c_s^u \) – over/under consumption in a time slot.

Parameters:
- \( \tau_{m, p} \) – processing duration of heat \( p \) on equipment \( m \); \( t_{m}^{\text{setup}} \) – setup times for machine \( m \); \( e_s \) – electricity price boundary of time slot \( s \); \( e_s \) – electricity price of time slot \( s \); \( h_{p, m} \) – specific power consumption of heat \( p \) on equipment \( m \).

Table 10: The notation of Production planning model [7]

| \( \sum_{m \in SM_{st, m}} X_{m, p} = 1 \) \( \forall p \in P \) | \( t_{m, p}^f = t_{m, p}^s + X_{m, p} \cdot \tau_{p, m} \) \( \forall m \in M, p \in P \) | \( t_{m, p}^s \leq M \cdot X_{m, p} \) \( \forall m \in M, p \in P \) | \( t_{st}^s = \sum_{m \in SM_{st, m}} t_{m, p}^s \) \( p \in P, st \in ST \) | \( t_{st}^f = \sum_{m \in SM_{st, m}} t_{m, p}^f \) \( p \in P, st \in ST \) | \( t_{p, st}^f = t_{p, st}^s + w_{p, st} \) \( \forall p \in P, st \in ST, st < |ST| \) | \( t_{p, st}^s, t_{p, st}^f \) \( M \cdot X_{m, p} - 1 \) \( \forall m \in M, p \in P \) | \( V_{st, p, p'} = 1 \) \( \forall p, p' \in P, st \in ST, hg \in HG, p < p', p, p' \in HGP(P) \) | \( V_{st, p, p'} = 0 \) \( \forall p, p' \in P, st \in ST, p = p' \) | \( V_{st, p, p'} = V_{st+1, p, p'} \) \( \forall p, p' \in P, st \in ST, p < p', st < |ST| \) | \( t_{m, p}^s, t_{m, p}^f \) \( t_{m, p}^f + t_{m}^{\text{setup}} - M(3 - V_{st, p, p'} - X_{m, p} - X_{m, p'}) \) \( \forall p, p' \in P, m \in M, st \in ST, \{ st, m \} \in SM, p \neq p', st < |ST| \) | \( t_{m, p}^s, t_{m, p}^f \) \( t_{m, p}^f + t_{m}^{\text{setup}} - M(3 - V_{st, p, p'} - X_{m, p} - X_{m, p'}) \) \( \forall p, p' \in P, m \in M, st \in ST, \{ st, m \} \in SM, p \neq p', st = |ST| \) | \( X_{m, p} = X_{m, p+1} \) \( \forall p \in L(P), p' \in F(P), m \in M, st \in ST, \{ st, m \} \in SM, p \neq p', st = |ST| \) | \( t_{p, st}^f = t_{p, st}^s + w_{p, st} \) \( \forall p \in P \setminus L(P), st \in ST, st = |ST| \) | \( t_{ms} \geq t_{m, p}^s \) \( \forall p \in L(P), m \in M, \{ st, m \} \in SM, st = |ST| \) |
Appendix B: Energy management model

Table 10 shows the notation used in the model whereas the equations 1 to 12 show the constraints.

Sets:
- Node, i, j – nodes; Node_{pur} – purchase contracts node /long term contract (i1), TOU (i2), day ahead market (i3); Node_{gen} – onsite generation node /i4/; Node_{bal} – balancing node /i5/; Node_{dem} – process demand node /i6/; Node_{sal} – sales node /i7/; Arc_{Node,Node,s} defined arc between nodes.

Variables:
- \( g_{s,i,j} \) – binary, true when onsite generation is running in slot \( s \); \( g^s_{s,i,j} \) – continuous variables denoting startup of onsite generation in time slot \( s \); \( f_{s,i,j} \) – flow from node i to j in time slot \( s \); \( c^s_{i,j} \) – cost of onsite generation in slot \( s \).

Parameters:
- \( c_{s,i,j} \) – electricity cost in time slot \( s \); \( f^{min}_{s,i,j}, f^{max}_{s,i,j} \) – min and max flow between nodes i and j; minr, mind – minimum run and down time intervals of onsite generation; \( c^{start} \) – onsite generation startup cost; \( k \) – percentage reduction of onsite generated power due to startup.

Table 11: Energy management model’s notation [12]

The deviation penalties are integrated to the production planning model and they are calculated using the equations (13) to (15).

\[
\begin{align*}
\min \sum_{s \in S} (\sum_{i' \in \text{Node}, j' \in \text{pur}} f_{s,i',j'} + c^\text{gen}_{s,i,j} - \sum_{i \in \text{Node}, j \in \text{sal}} f_{s,i,j} \cdot c_{s,i,j}) & \quad (1) \\
\sum_{i \in \text{Node}} f_{s,i,j} = \sum_{j' \in \text{Node}} f_{s,i,j'} & \quad \forall (i,j') \in \text{Arc}, (j', j) \in \text{Bal}, s \in S \quad (2) \\
f^{\text{min}}_{s,i,j} \leq f_{s,i,j} \leq f^{\text{max}}_{s,i,j} & \quad \forall (i, j) \in \text{Arc}, i, j \in \text{Node}, s \in S \quad (3) \\
q_s = \sum_{i \in \text{Node}, j \in \text{Dem}} f_{s,i,j} & \quad \forall (i, j) \in \text{Arc}, s \in S \quad (4) \\
g_{s,i,j} \leq M \cdot G_{s,i,j} & \quad \forall (i, j) \in \text{Arc}, i \in \text{Node}, j \in \text{Gen}, s \in S \quad (5) \\
g^s_{s,i,j} \geq G_{s-1,i,j} & \quad \forall (i, j) \in \text{Arc}, i \in \text{Node}, j \in \text{Gen}, s \in S \quad (6) \\
g^s_{s,i,j} \leq G_{s,i,j} & \quad \forall (i, j) \in \text{Arc}, i \in \text{Node}, j \in \text{Gen}, s \in S \quad (7) \\
G^s_{s,i,j} \leq 1 - G_{s-1,i,j} & \quad \forall (i, j) \in \text{Arc}, i \in \text{Node}, j \in \text{Gen}, s \in S \quad (8) \\
c^s_{s,i,j} = \sum_{i \in \text{Node}, j \in \text{Gen}} f_{s,i,j} \cdot c_{s,i,j} + c^{\text{start}} \cdot g^s_{s,i,j} & \quad \forall (i, j) \in \text{Arc}, s \in S \quad (9) \\
f^{\text{max}}_{s,i,j} \cdot G_{s,i,j} - k \cdot f^{\text{max}}_{s,i,j} \cdot g^s_{s,i,j} & \quad \forall (i, j) \in \text{Arc}, i \in \text{Node}, j \in \text{Gen}, s \in S \quad (10) \\
\sum_{s' = s}^{s+\text{minr}-1} G_{s',i,j} \geq \text{minr} \cdot (G_{s,i,j} - G_{s-1,i,j}) & \quad \forall (i, j) \in \text{Arc}, i \in \text{Node}, j \in \text{Gen}, s \in S \quad (11) \\
\sum_{s' = s}^{s+\text{mind}-1} G_{s',i,j} \leq \text{mind} \cdot (1 + G_{s,i,j} - G_{s-1,i,j}) & \quad \forall (i, j) \in \text{Arc}, i \in \text{Node}, j \in \text{Gen}, s \in S \quad (12)
\end{align*}
\]