UNIVERSITÀ DEGLI STUDI
DI GENOVA

Department of Computer Science, Bioengineering, Robotics and System Engineering (DIBRIS)

Artificial Intelligence techniques for solving the hydro generation scheduling problem

Master's Degree in Computer Engineering
- Artificial Intelligence and Human-Centered Computing -

Candidate:
Riccardo Poli

Supervisor:
Prof. Marco Maratea

Co-supervisor:
Dott. Davide Mini

Academic year: 2021/2022

Abstract

Electricity generation from renewable sources is a key element for the future due to its ecological properties and endless resources. To improve to the best of its potential, in recent years numerous computer techniques and an increasing focus on data analysis have been exploited in this area. In particular, reservoir hydropower plants can be of decisive support in ensuring the balance of the energy grid, acting when needed to cover energy shortages, and having the ability to be scheduled and programmed over time horizons. Companies that manage the dispatching, buying and selling of electricity, such as EGO, need to schedule the output of the power plants they operate in the best possible way, so as to take advantage of the water available in the reservoir at the times when electricity is most in demand and the price of energy is higher. In this thesis we aim to exploit Artificial Intelligence formalisms from Knowledge Representation and Reasoning, which have been successful in numerous fields including planning and scheduling, to find an efficient technique for solving the hydro generation scheduling problem.

Contents

List of Figures

1 Introduction

In this chapter, we present the background and motivations behind the problem under study in this thesis, the *Hydro Generation Scheduling problem*, which has remained a constant focus of attention for researchers in the last fifty years and which could be crucial for energy development in the years to come.

Next, we briefly present the state of the art, which will then be addressed more exhaustively in a later chapter, listing in particular the various methodologies that have already been used to address the problem we discuss.

Finally, we present the objectives of the thesis and its structure.

1.1 Introduction to the Concepts

In order to best describe the work done in this thesis, let us first introduce some important concepts for the problem environment.
In particular, here will be briefly explained the status of the Energy Crisis, the functioning of the Italian energy market, the characteristics of a generic hydropower plant, the different types of optimization that can occur in this field and a brief description of the EGO company, from which we obtained the specifics of the problem and the real-world data.

1.1.1 Energy crisis and the related research

There has always been a worldwide awareness of the problem of demand for energy supplies; traditional energy sources such as crude oil and natural gas are present in about 10 percent of the world's countries, forcing other countries to supply the needs with imports. The gradual depletion of sources and countries' geopolitical and diplomatic strategies are driving energy prices to unsustainable levels.

In Europe, alternative power generation solutions will have to be sought to address rising costs, for example by harnessing renewable energy production.
According to the European Environment Agency, European countries have set a goal of having 32% of energy be supplied from renewable sources by 2030. In order to succeed in this task, it is necessary to analyze and improve renewable solutions.

Energy solutions are part of several research funds; in particular, "Horizon Europe," the world's largest transnational research and innovation program, was renewed for 7 years in 2021, with funding of 100 billion euros.
HorizonEU aims to achieve scientific, technological, economic and societal impact from Eu investments in research and innovation and, in particular, encapsulates a deep interest in Energy Innovation. Among the objectives we have:

- Boosting cost performance and reliability of renewable energy solutions and making the energy grid more flexible and secure.

- Creating inclusive growth and employment in Europe, bringing down costs for consumers and reducing EU's energy import dependency by developing energy efficient demand side solutions.
- Delivering technological and socio-economic breakthroughs necessary to achieve climate neutrality and zero pollution of the building stock by 2050 and support the recovery, upgrade and/or conversion of industrial excess (waste) heat, as well as electrification of heat generation in industry.
- Achieving new cross-sectoral energy solutions enabling the clean energy transition and more secure and competitive energy supply.

1.1.2 The electricity market

Since 2004, in Italy, a system of interconnected power auctions determines the price of energy and this is fundamental for understanding how it is established the allocation of power production.

At the roots of this structure, we found of high importance the concept of "MGP" or "Mercato del Giorno prima" (Day ahead market), which is including the prices for the auction of energy sales for the next day. In these fields, we can find many aspects and rules, in particular, based on the definitions from GME "Gestori Mercati Elettrici" we can summarize that in MGP:

- hourly energy blocks are traded for the next day.
- Participants submit bids where they specify the quantity and the minimum/maximum price at which they are willing to sell/purchase.
- The MGP sitting opens at 8 a.m. of the ninth day before the day of delivery and closes at 12 p.m. of the day before the day of delivery. The results of the MGP are made known within 12.58 p.m. of the day before the day of delivery.
- Bids/asks are accepted after the closure of the market sitting based on the economic merit-order criterion and taking into account transmission capacity limits between zones. Therefore, the MGP is an auction market and not a continuous-trading market.
- The price is determined, for each hour, by the intersection of the demand and supply curves.

The paper "Guida al Mercato Elettrico" [32], can provide some additional insights on the topic of the Italian power exchange market.

1.1.3 Hydroelectric power plant

Since ancient times, attempts have been made to exploit water sources, for example, with the introduction of mills and water wheels in the grain milling process. For the high interest in this area, it was an easy development in the late 19*th* century to introduce hydropower as a new electricity source when the British-American Engineer James Francis developed the first modern water turbine.

As described in [40], today hydropower provides about 16 percent of the world's electricity, in 2020 was reached a new record in production with a generation of 4,300 terawatt hours of clean electricity, with the world leader China producing more than 850 billion kilowatt hours a year and having an installed capacity of over 370 GW. Moreover, this field is still in development with projects around 21 GW put into operation in the last year. In figure 1, we can see a global review of development in 2018.

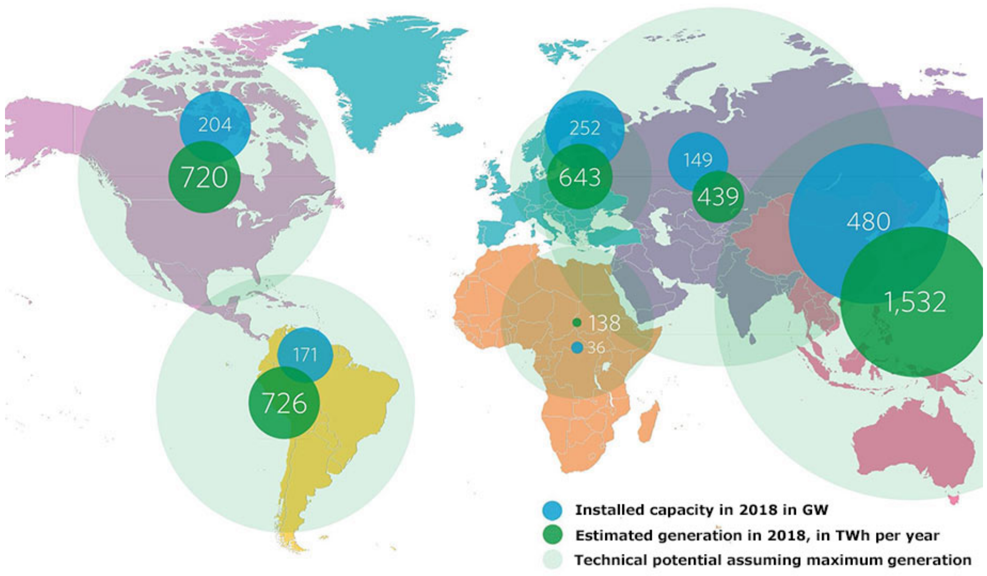

204
720
252
643
149
439
480
1,532
138
36
171
726
Installed capacity in 2018 in GW
Estimated generation in 2018, in TWh per year
Technical potential assuming maximum generation

Figure 1: Global review of hydropower developments (2018).

There are several advantages to using hydropower. Energy is produced from naturally occurring waterways with a resource that is definitely clean and that can be renewed through precipitation and melting ice; In addition, hydropower plants can be remotely controlled and programmed, to be adjusted according to energy demand.

"A typical hydroelectric power plant is a system with three parts: a reservoir where the water is stored, a dam that can be opened or closed to control water flow, and a power plant where the energy is produced" [49]. A simple version of a hydroelectric system is represented in figure 2.

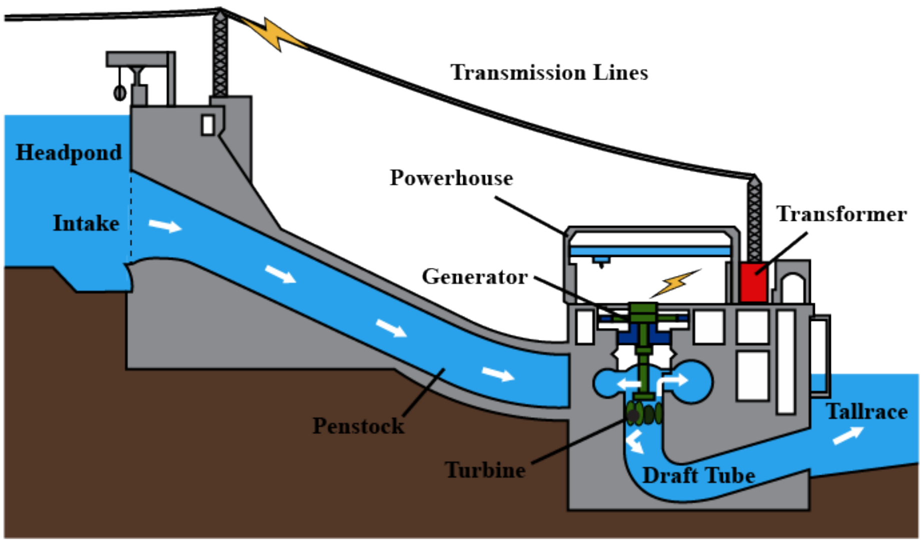


Figure 2: Overview of a simple reservoir-based hydroelectric power plant.

In a reservoir hydropower plant, one or more streams are directed to a water collection area, usually a reservoir or dam, where the water can be stored. This allows the resource needed for electricity production to be available at all times and to be able to produce regardless of the season or weather conditions.

The basin is connected by a intake gate to a penstock, which can carry the water to the power plant's production area. The gate can block some unwanted objects and can be opened and adjusted to let a certain variable amount of water through.

In the area of the power plant designated for production, the water is guided to a certain number of turbines disposed separately or grouped as a unit; these turbines are designed to rotate by effect of the flow of the water and to transmit this motion to the alternators for the production of electricity. The generating power is due to the amount of water that is granted to the turbines.

The electricity generated is collected and passed through transformers, to be thus sent to the transmission lines of the national power grid.

The water already harnessed for the production is sent outside from a draft tube.

1.1.4 Optimizations of energy production

Electric energy production in water power plants usually regards different kinds of optimizations considering different targets (e.g. decrease the water consumption, increase the total production or the production efficiency, optimize the production frequency, improve ecological impact on the environment, management of multiple reservoirs and/or power stations).

In the field of Hydro Scheduling [36] it is also important to consider the differences regarding time horizons and we can have short, medium or long-term optimizations:

- Short-term hydro scheduling (STHS) is based on the hydro generation of energy in a short-term period of a few days (usually 3 to 7 days).

  This type of problem is the most related to the one we are considering in this thesis. In particular, this type of optimization can be used with fresh data sets and/or basic forecasts of them, coming directly from the power plants sensors This type of analysis is often employed in the analysis regarding the profit derived from energy production with hourly considerations.

- Mid-term hydro scheduling (MTHS) is also referred to as the problem of generation of energy and water management of the power plant.

  Concerning the the STHS, this type of problem has a larger horizon of 3-18 months and, this way, it manages to capture the seasonal properties of the watercourses and the climatic and meteorological changes that can occur in the different months.

  Due to uncertainty and probability distribution factors, given by the increased size of the horizon (e.g., natural inflow and price distributions), it is intractable to formulate and solve those problems without approximation or simplification (e.g., the mean of annual natural inflow is formulated as input). Moreover, because of the high number of possibilities that we can have in assigning the production, we have also an increase in the computational cost.

- Long-term hydro scheduling (LTHS) is referred to a very large horizon of 1-5 years.

  In this case, the focus is on the annual cyclicity of the problem, as the study of the mean behavior of the power plant; for this reason, it is used for economical reasons as a predictor of good reservoir operation management and to draw some statistics on the variables of the problem.

  As for the MTHS, we need to use approximation or simplification and we have a considerable amount of stochastic and inaccurate components.

1.1.5 EGO: trading, production and dispatching of energy

EGO Group is an Italian group of companies based in Genoa that deals with trading, production, dispatching and consulting in the energy field. The Group has always been a key player in the energy markets, with a history that has followed the main opportunities offered by technological evolution and the various phases of liberalization of the electricity market.

Today, as a group, it operates more than 1,500 plants with 2,000 MW of rated capacity and more than 5 TWh per year of physical dispatched generation. EGO also participates in the new dispatching services market, aggregating more than 250 MW from virtually aggregated mixed units scattered in different Italian regions.

In an increasingly decentralized model of energy generation, ever more open to small producers, the ability to manage large amounts of data is a key differentiating factor. EGO has been investing for many years in innovative data analysis and artificial intelligence technologies; for this reason, it is able to capitalize on the full range of opportunities the energy sector currently offers.

EGO Data S.r.l. is EGO Group's digital technology service provider, managing millions of data points per day to power business processes with next-generation IT infrastructure. The technology architecture used is highly scalable and resilient, based on machine learning systems and compatible with the industry's leading communication protocols.

EGO Data has offered itself as a partner for this thesis; in this project, specifications and data from a plant operated by their group will be used, thanks to which we would have the opportunity to analyze the efficiency of our solution for the hydro generation scheduling problem.

1.2 Context and Motivations

Power generation methods have been in continuous development in the last decades, but it is with the advent of the modern energy crisis that we are fully realizing the importance, not only scientific but also political and economic, of finding new solutions for energy generation and achieving "energy independence". Today, a viable solution can be offered by the exploitation of renewable resources. Wind, solar, geothermal, marine, and hydroelectric plants are applicable in different territories and contribute to the supply of about 30% of the world's electricity. To improve this statistic, efforts have been made in recent years to refine power plants and to offer automated and technological systems that can make the best possible use of the available sources.

In countries such as Italy, following the liberalization of the energy market, the buying and selling of energy are carried out through an auction system, with bids that must be scheduled days in advance. For this reason, producers need to know in time the quantity of energy their plants are capable of producing. [section 1.1.2]

While technologies such as photovoltaic and wind power generation require inductive data analysis in order to predict and plan the production intervals (e.g., due to weather events); power generation through hydropower has some particularities that distinguish it from others and also makes considerations of short-, medium-, and long-term planning and operational controls possible.

Hydroelectric plants [section 1.1.3] are divided into three main types:

- Flowing water plants
- Reservoir plants
- Storage plants

The particularity of the 2nd and 3rd categories is the possibility of storing a certain amount of water within a reservoir (typically with a dam) in order to be able to exploit this supply of water at the time of need and thus overcome the temporal and seasonal component that affects the amount of water available.

Due to its storability, flexibility, and controllability, hydropower generation is of critical importance in ensuring safety in the system. A significant fraction of the capacity of hydropower units acts as a bonus operating reserve, to meet the frequent fluctuations and shortages in the power system that may occur during a particular period (e.g., at night when solar/photovoltaic power is not available).

Reservoir-based hydropower plants have the ability to be scheduled, therefore, it is important to be able to define algorithms to try to optimize production schedules. The standard goal is to regulate energy production at times when it is most convenient, considering the level of water entering the system, the amount of water that can be contained by the reservoir, and the operational properties of the turbines that will be converting the flow of water into electricity; as well as some information about the production demand and the value of energy at a particular time of the day (e.g., "MGP" prices [section 1.1.2]).

This problem is characterized by the large number of variables involved, the nonlinear system dynamics and it also has some stochastic components (e.g., the price of energy and the water inflow to the system). For this reason, it has been in recent years in the interest of researchers and is known in the literature as the "Hydro Generation Scheduling Problem" or "long-mid-short-term hydro scheduling problem" (LMSTHS) [36].

1.3 State of the art overview

Recent development in computer technologies has made possible the deployment of complex and sophisticated methods for solving hydro generation scheduling problems and more generally optimizations in hydroelectric problems, for example, with the establishment of methodologies with linear and nonlinear programming, dynamic programming, and some meta-heuristic algorithms.

Each of these proposed solutions has its own particular upside, which makes it particularly applicable in a specific area of optimization, e.g., the methodology based on Dynamic Programming is the most suitable for solving problems with stochastic and nonlinear physical variables, however, it suffers from "the curse of dimensionality" [see section 7.4] and thus is inefficient in solving schedules for plants with multiple reservoirs or multiple production units.

To evaluate techniques based on mathematical programming, certain factors such as the complexity of the problem definition must be taken into account, whereas, in the case of meta-heuristic techniques we have a methodology that best fits the nature of the problem but may not often lead to the optimal solution.

Given the large number of existing problem instances related to hydroelectric power plants, we can find several reviews and summaries of algorithms and methodologies for solving the hydro generation scheduling problem, for example in the papers [36, 39, 42] and in the thesis [48].

Although these techniques succeed theoretically, and sometimes practically, in solving sufficiently the problems for which they were designed, there is still a need for research in this area. A generic methodology that can successfully handle the most complex optimizations and is also adaptable to multiple variants of the problem has not yet been found.

1.4 Objective

The goal of this thesis is to exploit an Artificial Intelligence methodology based on Knowledge Representation and Reasoning, to attempt to solve the problem of short-term hydro generation scheduling in a way that should not only be effective but also adaptable to different instances of the problem.

In realizing our goal, we will follow the following steps:

- Starting from a formal description of the specification and data of the problem instance provided by EGO, an encoding of the problem under study will be carried out, so as to make it applicable to the given methodology.
- Different scenarios in which the problem is solved by dedicated tools will be analyzed, showing the performance of the applied solution, even compared to the solution currently used by EGO.
- Finally, a web application will be developed to facilitate the target user in performing the scheduling problem applications in the real world.

1.5 Outline

This thesis is structured as follows:

In the second chapter we have a formal definition in which we will identify the structure of the problem environment. Moreover, we will analyze the data and the different characteristics of the power plant object of study and the targets of the optimization.

In the third chapter we will provide the definition of our methodology: the "Answer Set Programming" paradigm, both with a theoretical explanation and some examples.

The fourth chapter is about encoding, we will explain how to encode our problem using the typical Guess&Check&Optimize Answer Set Programming methodology. Moreover, we will also try to improve the structure of the rules and constraints to make the best possible declarative ASP structure and to obtain good performances. Here we also provide some additional rules and heuristics to give an example of possible adjustments to the problem environment that are allowed thanks to the usage of ASP.

In the fifth chapter we will show the results derived from trials on real data provided by EGO, with the usage of statistics and data visualization techniques.

In the sixth chapter we will describe the web application that we developed to make a real example of a user-friendly solution, that can be used in the production environment by companies like EGO.

In the seventh chapter we will describe some related works on the topic of hydropower optimizations and, in particular, of the hydro generation scheduling problem.

In the last chapter we will conclude by explaining the results and considerations carried out from this thesis and some future enhancements.

2 Problem Definition

In this chapter we will describe the problem of the thesis.

Each hydroelectric power plant has different structural characteristics: there are plants that have multiple reservoirs and/or multiple units of turbines; moreover, due to physical conditions of the plant or operational choices of the operators, sometimes there are some special conditions to be met, that must be included in the problem environment description.

From now on we are going to consider the hydroelectric power plant system and problem formulation that has been described to us by EGO, in this way we have the real data of the plant at our disposal.

2.1 Description of the hydroelectric power plant

In this thesis, we are considering an example of a reservoir-based hydroelectric power plant [see section 1.1.3 and figure 3] with a single basin and four different turbines in a single unit.

A canal of about 10 km, partly open air and partly in a tunnel, collects water from several streams and sends this flow to a large reservoir. This basin is also connected through a gate with a pressure tunnel and different steel penstocks; these canals feed four different horizontally disposed turbines.

In the power generation area of the power plant, turbines rotate in contact with water and this movement allows electricity to be generated, which is then sent into different stages of transformers and delivered to Terna's national network.

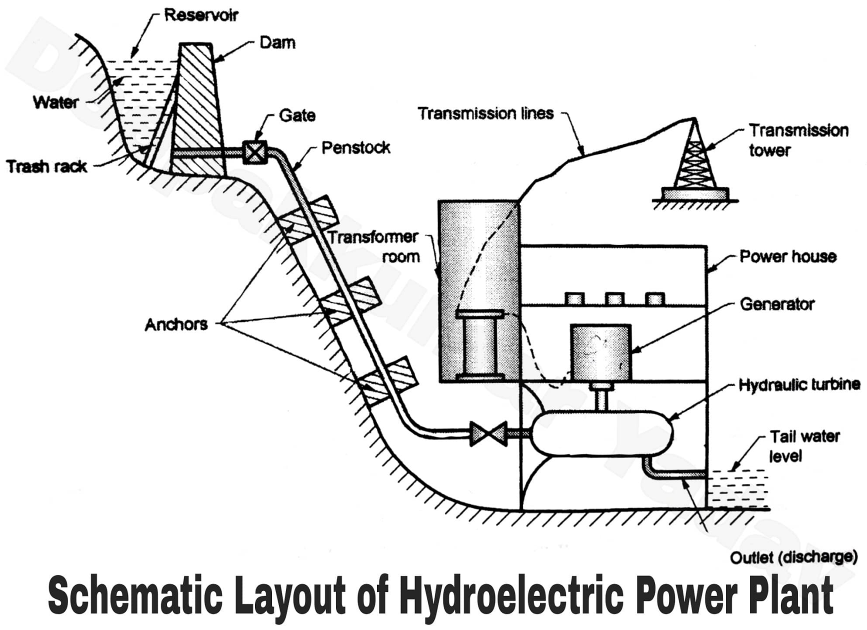


Figure 3: A reservoir-based hydroelectric power plant (D.Yadav).

2.1.1 Water Inflow

The hydropower plant system has as its input a certain amount of water from a natural watercourse such as a river.

The water flow is usually unpredictable and depends on various climatic and meteorological conditions; however, it can be considered slowly variable in nature ($< 50\%$) in most cases.

Under rainy conditions, the river level connected to the basin may rise with some temporal delay. As indicated by some researchers in related works it is difficult to estimate the river level accurately.

Some particular operational choices of the power plant may be based on the seasonal properties of watercourses, having for example:

- Months such as June and July when there are the phenomena related to ice melting, where consequently there is quite a lot of water available and continuous production is encouraged.
- Dry summer months in which there is very little water available and production is restricted to the periods of the day when energy is most needed.

In this thesis, we are assuming the river flow constant over the few days representing the domain of optimization.

2.1.2 Water Reservoir

The reservoir of the hydroelectric plant contains a certain amount of water; the capacity depends on the size of the reservoir and can be measured by volume (m^3) or by height level (m).

The water level in the basin varies over time and this variation depends almost entirely on the water flow from the inlet channel and on the outflow to the power plant production area.
The quantity of water in the reservoir at time t is therefore defined as:

$$\text{water}(t) = \text{water}(t_0) + \text{waterIn}(t - t_0) - \text{waterOut}(t - t_0)$$

where waterIn is the water inflow entering the reservoir and waterOut is the water exiting the reservoir, directed to the production area.

Water increases/decreases by a certain amount, related to the value of:

$$\Delta\text{water}(t - t_0) = \text{waterIn} - \text{waterOut}$$

In the hydroelectric power plant system described by EGO, there are some additional reservoir rules imposed by the plant operators. Indeed, there are two water level threshold values (m) that must be met to avoid the occurrence of two particular situations:

- If the water in the reservoir falls below a certain threshold (minimum level), the pressure channel starts to transport air to the turbines.
- If the reservoir level rises above a certain threshold (maximum level), water overflow can occur, resulting in wasted resources.

These thresholds are related to the physical capacity of the plant and it is important to set the minimum and maximum values properly, to maintain a safe condition with respect to the data uncertainties and approximations.

2.1.3 Power Production

Power plants can produce a variable amount of electricity, proportional to the quantity of water fed to the turbines.

In the considered power plant, the hourly power output ranges from 0 to 23 MW, with a granularity of 0.1 MW.

It is possible to indicate to the plant how much energy you intend to produce for each hourly time slot, so the water input to the turbines is adjusted accordingly.

We have also the possibility to introduce 2 advanced operational properties, which are usually encouraged by the power plant operators:

- Power generation, and consequently the turbines, should not be stopped for more than 3 times a day.
- There are different power configurations, which can be selected every day and they select the proper and most efficient configuration of turbines to produce the desired amount of power.

2.1.4 Water Usage

To properly define the hydro generation scheduling problem, it is important to understand how the level of the reservoir varies and, consequently how much water flows through the pressure channel directed to the turbines.

Considering that we have a "driven production", in which we are the ones who specify how much power needs to be produced by the plant, we can then use some equations and estimates to find the relationship between the power produced and the required water "used" by the turbines.

In the hydroelectric power plant, any amount of power (while still respecting the threshold defined in the 2.1.3) can be produced by the turbines by swirling a certain amount of water. However, we have to consider that it is almost never possible to immediately set the output of the plant to the desired power level; for example, one of the reasons has to do with the fact that the penstocks carrying the water to the turbines are quite long and this will cause delay. Its needed some time, for example, to increase the output from PW1 to PW2.

As a result of the previous consideration, we need to calculate and consider in our problem the water needed, not only to produce a certain amount of power, but also the amount of water needed for the ramp taking place between PW1 and PW2.

2.2 Energy Selling Profit

The resource to be optimized is the expected economic gain from the sale of energy [see section 1.1.2].

Selling prices are almost always proportional to energy needs, in general there can be very different energy prices depending on temporal factors (e.g. time, day of the week, seasonality, ...) and also on particular events (e.g. weather events, energy and economic crisis, ...). For example:

- In the middle hours of the day, prices are lowered due to the possible use of solar energy or other alternative sources of production.
- On weekends, since many factories are closed, prices may be lower compared to other days of the week.

Price optimization is related to a short-term distribution of energy, e.g., each day one would need to schedule the energy to be put on the grid the next day (about 24h).
In the exchange-type market of energy, prices are not known in advance, therefore an estimation of market forecasts is necessary, from which hourly price tables can then be extracted, e.g., with a daily or a few days' horizon.

The key metric is the "MGP" Price (Day-Ahead Market Price), which is obtained the day before the market transaction. MGP is often defined as time series which includes for each time slot the price of energy per MWh.

Having the MGP price in €$/MWh$ available for each hour of the day, we can then simply multiply this value by the amount in MW of power produced. Considering h as an hourly time slot, we can define:

$$\text{expectedProfit}(h)\ [€] = \text{priceMGP}(h)\ [€/MWh] * \text{energyProduced}(h)\ [MWh]$$

2.3 Procedure Properties

It is important to solve this hydro generation scheduling problem quickly by meeting certain requirements.

- In order to minimize inaccuracies on data and forecasts, the algorithm should be run with fresh data.
- We need to wait until we have all the required data available.
- The deadlines of the day-ahead market and the delivery of the schedule to the hydroelectric plant must be met.

Since we already have some constraint related to these time requirements, the methodology developed will have to be rather fast in finding the solution to the problem; E.G. a maximum run-time of 120 min.

2.4 Problem Overview

The hydro generation scheduling problem we want to explore in this thesis is defined as follows:

Description:

Power plant turbines can produce a specific amount of power each hour.

$$\text{power} \in [0 - 23\ MW]$$

The reservoir can hold a certain amount of water, the level of which must be maintained between a minimum and maximum threshold.

$$\text{waterLevel} \in [3.5 - 7\ m]$$

The water in the reservoir increases/decreases over time, depending on the water inflow as input and the amount of water sent to the turbines.

$$\text{water}(t) = \text{water}(t_0) + \text{waterIn}(t - t_0) - \text{waterOut}(t - t_0)$$

The expected profit from the selling of the energy is given by the product between the MGP price of energy and the quantity of power that we are able to sell.

$$\text{expectedProfit}(h) \ = \text{priceMGP}(h) \ * \text{energyProduced}(h)$$

In such short-term optimization, it is important to avoid using all the available water so as not to be in an unfavorable situation with the next schedule. For this reason, among the data to be considered, there is also the minimum desired water level to be maintained at the end of the schedule horizon.

$$\text{waterLevel} \geqslant X \ \text{(at the end of the schedule horizon)}$$

Additional Requirements:

The production should not be stopped more than 3 times a day.

A specific power configuration, which affects the production capacity of the plant, must be selected from those available, at the beginning of each day.

2.5 Input Data

We have at disposal some data that could be given as input to the scheduling problem:

- Estimated prices of energy:

  Prices estimated by market analysts, relative to the sells of energy in the period defined by the scheduling horizon.

- Inlet water flow:

  Water flow entering the reservoir for each hour of the scheduling horizon, data coming from the observation of the previous days.

- Initial reservoir water level:

  Reservoir water level measured or estimated for the beginning of the scheduling period.

And regarding the power plant and optimization parameters:

- Time horizon of the schedule.
- Minimum and maximum power output.
- Minimum and maximum threshold of water.
- Minimum final water level.
- Utilization of water for energy production.
- Maximum number of stops to production.
- Power configurations.

2.6 Expected Results

The optimizer should calculate and return as output the best possible schedule for the production of the hydroelectric plant, concerning the short term of 4 days.

We are also interested in bringing out from the result some additional data, that could be used for the analysis of the performances of the method employed, evaluating the quality of the result and deriving additional information related to the production schedule. For example:

- Production:

  The temporal data with the scheduled production for each hour.

- Final water level:

  The water level in the reservoir at the end of the schedule.

- Expected profit:

  The value of the expected profit resulting from the schedule.

- Water level:

  The height in meters or the volume of the water level in the reservoir, for each hour of the scheduled horizon.

3 Specification language - ASP

Answer Set Programming (ASP) is a programming paradigm developed in the field of logic programming and non-monotonic reasoning. In making this overview, we assume that the conventions of logic programming are known, more information about ASP can be found at https://potassco.org/.

For more insights about the syntax and semantics of ASP and some related descriptions, see [12, 5].

3.1 Syntax

The syntax of ASP is similar to that of Prolog. Variables are strings with the first letter capitalized and constants are non-negative integers or strings with the first letter lowercase.

A *term* is either a variable or a constant. A *standard atom* is an expression $p(t_1, ..., t_n)$, where p is a predicate with n arguments and $t_1, ..., t_n$ are the terms. An atom $p(t_1, ..., t_n)$ is ground if $t_1, ..., t_n$ are constants.

A *ground set* is a set of pairs in the form $\{consts : Conj\}$, where $consts$ is a list of constants and $Conj$ is a conjunction of ground atoms. A *symbolic set* is a set defined syntactically as $\{Terms_1 : Conj_1;\ ...\ ; Terms_t : Conj_t\}$, where $t > 0$, and for all $i \in [1, t]$, each $Terms_i$ is a list of terms such that $|Terms_i| = k > 0$, and each $Conj_i$ is a conjunction of standard atoms.

A *set term* is either a ground set or a symbolic set.
Intuitively, a set of terms $X : a(X, c), p(X);\ Y : b(Y, m)$ represents the union of two sets: the first contains the values of X that make the conjunction $a(X, c), p(X)$ true, and the second contains the values of Y that make the conjunction $b(Y, m)$ true.

An *aggregate function* is of the form $f(S)$, where S is a set of terms, and f is a symbol of an aggregate function. Aggregate functions map multisets of constants to a constant. [e.g. section 3.6.2]

The most common functions implemented in ASP systems are as follows:

- #*count*, number of terms.
- #*sum*, sum of integers.
- #*min*, minimal term.
- #*max*, maximal term.

An *aggregate atom* is of the form $f(S) \prec T$, where $f(S)$ is an aggregate function, $\prec \in \{<, \leqslant, >, \geqslant, \neq, =\}$ is a comparison operator, T is a term called guard. An aggregate atom $f(S) \prec T$ is ground if T is a constant and S is a ground set. An *atom* is either a standard atom or an aggregate atom.

A *rule* r has the following structure:

$$a_1 \vee ... \vee a_n \ \text{:-}\ b_1, ..., b_k, not\ b_{k+1}, ..., not\ b_m.$$

where $a_1, ..., a_n$ are standard atoms, $b_1, ..., b_k$ are atoms, $b_{k+1}, ..., b_m$ are standard atoms, and $n, k, m \geqslant 0$. A *literal* is either a standard atom a or its negation *not* a. The disjunction $a_1 \vee ... \vee a_n$ is the head of r, while the conjunction $b_1, ., b_k, not\ b_{k+1}, ..., not\ b_m$ is its body.

Rules without a body are called *facts*. Rules without a head are called *constraints*.

A variable that only appears in the terms of a rule r is called a *local* in r, otherwise, it is a *global* variable in r.

An ASP program is a set of *safe* rules, where a rule r is *safe* if it satisfies the following conditions:
(i) for each global variable X of r there is a standard atom ℓ in the body of r such that X appears in ℓ;
(ii) each local variable of r that appears in a symbol set $\{Terms : Conj\}$ also appears in a positive atom in $Conj$.

A *weak constraint* [11] (Buccafurri et al.) ω is in the form:

$$:\sim b_1, ..., b_k, not\ b_{k+1}, ..., not\ b_m.\ [w@l]$$

where w and l are the weight and level ω respectively. (Intuitively, $[w@l]$ reads "weighted w, at level l", where the weight is the "cost" paid to violate the condition in the body of w, while the level may be used to indicate a different priority). [e.g. section 3.6.4]

A standard atom, literal, rule, program or weak constraint is *ground* if no variables appear in it.

An ASP program with weak constraints is $\Pi = \langle P, W \rangle$, where P is a program and W is a set of weak constraints.

3.2 Semantics

Considering the ASP program P. The Herbrand universe U_P and the Herbrand base B_P of P are defined. The ground instance G_P of P is the set of all the ground instances of the rules of P that can be obtained by replacing variables with constants from U_P.

An interpretation I for P is the subset I of B_P.

A literal ground ℓ (resp., *not* ℓ) is true with respect to I if $\ell \in I$ (resp., $\ell \notin I$), or false (resp., *true*) otherwise.

An aggregate atom is true with respect to I if the value of the aggregate function (i.e., the result of applying f in the sets S) with respect to I satisfies the guard; otherwise, it is false.

A ground rule r is satisfied by I if at least one atom in the head is true with respect to I while all conjunctions in the body of r are true with respect to I.

A model is an interpretation that solves all the rules of a program.

Given a ground program G_P and an interpretation I, the reduced [25] of G_P with respect to I is the subset G_P^I of G_P obtained by removing from G_P the rules in which a literal in the body is false with respect to I.

An interpretation I for P is an *answer set* (or stable model) for P if I is a minimal model of G_P^I (i.e., I is a minimal model of G_P^I) [25].

Given a program with weak constraints $\Pi = \langle P, W\rangle$, the semantics of Π extends that of the standard model explained above. Let $G_\Pi = \langle GP, GW\rangle$ be the instance of Π; a constraint $\omega \in G_W$ is violated by an interpretation of I if all literals in ω are true with respect to I. An optimal answer set for Π is a G_P answer set that minimizes the sum of the weights of the weak constraints violated in G_W with priority order.

3.3 Programming Methodology

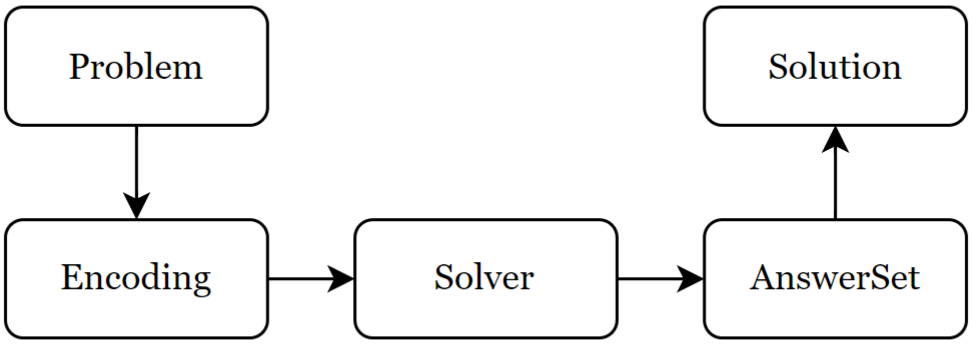


Figure 4: Programming methodology for ASP based solutions.

ASP has been exploited in several domains, from classical deductive databases to artificial intelligence. It can be used to encode problems declaratively; In fact, the power of disjunctive rules allows more complex NP problems to be expressed, and the (optional) separation of a fixed, non-ground program from an input database allows for uniform solutions on variable instances.

Software solutions implemented with the ASP language are based on a well-known programming methodology; the diagram in figure 4 shows us what are the basic steps.

We will also follow this path within the thesis, to show how each of these points was carried out

Problem:

At the top we have the problem, as was done in chapter 2, it is necessary to analyze the description of the specifics and constraints related to the topic to be addressed. Understanding the problem in depth is essential in order to create a reliable encoding that can serve the problem.

Encoding:

This step will be explored in chapter 4; after defining the problem, it is necessary to encode the specifications and constraints with the syntactic definitions and logical language required by ASP. For this purpose, we followed a known encoding methodology called Guess&Check&Optimize and explained in section 3.4.

Solver:

Once the coding of the problem is complete, we can give it as input to a solver, such as Clingo, in order to be able to obtain the solution.

Clingo can acquire an input program with first-order variables, expand it to the equivalent ground (variable-free) version, and finally process it, to be able to return the optimal answer set with the solution.

AnswerSet:

Once the solver has completed its execution, it is possible to obtain an answer set, see 3.2, which is the stable model that solves all the rules of the program. In our case, the answer set, or optimal answer set, is representing the solution.

Solution:

At the end of the procedure, we can decode the answer set back from the logical syntax of ASP and obtain the data for the schedule.
We should then analyze these results and make considerations about possible improvements to the encoding.

3.4 Guess,Check,Optimize Methodology

ASP can be used to encode problems in a declarative way usually employing a *Guess&Check&Optimize* methodology [44]. This method requires that a database of facts is used to specify an instance of the problem; a set of rules, called "*guessing part*", is used to define the search space; admissible solutions are then identified by other rules, called "*checking part*", which impose some admissibility constraints; finally weak constraints are used to single out solutions that are optimal with respect to some criteria, in the "*optimize part*" [e.g. section 3.6.3].

3.5 Syntactic abbreviation

Within the ASP model we can also use choice rules in the form $\{p\}$, where p is an atom. Choice rules can be interpreted as a syntactic abbreviation for the rule $p \vee p'$, where p' is a new atom that does not appear elsewhere in the program; this means that p can be chosen as true. [e.g. section 3.6.3]

3.6 ASP Examples

3.6.1 An example of rules and constraints

To present an example of a *rule*, let us imagine that we need to create a simple model in which we want to produce energy in the hydropower plant at time T if there is water available in the reservoir:

$$production(T) \text{ :- } waterLevel(L,\ T),\ L > 0.$$

Where $waterLevel(L,\ T)$ is true when there is L water in the reservoir at time T; In this way, the atom $production(T)$ will be true if there is L water available in the reservoir and L is greater than 0.

For the *constraints*, let us consider that we want to keep at most the equivalent of 5 meters of water in the reservoir of the hydropower plant, to prevent water from overflowing:

$$\text{:- } waterLevel(L,\ T),\ L > 5.$$

With the constraint, we impose that at least one of the two atoms is false. So in our example, if we have L water in the reservoir at time T, for sure it is not possible that L is greater than 5.

3.6.2 An example of aggregate functions

Consider that we built a model in which we always produce energy, in each hour of the day, if we have water available in the reservoir. Now we want to calculate how many hours per day we produce energy.

As seen before [section 3.6.1] we can define the rule for the production, assuming that we have a water level atom for each hour of the day:

$$production(T) \text{ :- } waterLevel(L,\ T),\ L > 0.$$

To obtain the number of hours of production, we can use the aggregate function *#count* like this:

$$productionHours(N) \text{ :- } N = \ \#count\{T \ : \ production(T)\}.$$

In this way, N would be equal to the number of production atoms that are true (the number of time in which we produce energy among the 24 hours of the day).

Now we define a model in which we express the economic aspects of power plant production.
The expected profit from the selling of energy is equal to the product of the MGP energy price and the amount of energy produced in each hour T:

$$gain(G,\ T) \text{ :- } price(PR,\ T),\ production(Q,\ T),\ G = PR * Q.$$

Where $price(PR,\ T)$ means that at time T the energy price is PR and $production(Q,\ T)$ means that at time T the power plant is producing at Q (MW) of power.

To calculate the total gain for the day we can use the aggregate function $\#sum$:

$$totalGain(TG)\ \text{:-}\ TG = \ \#sum\{G,\ T\ :\ gain(G,\ T)\}.$$

This way we add up all the values of G present in the gain atoms.

3.6.3 An example of Guess and Check and Choice rules

Consider as an example that a hydroelectric power plant can produce a specific integer amount of energy, ranging from 0 to 5 MWh. We now want to define this property in a new model, along with the fact that in order to produce energy, we must have the necessary amount of water in the reservoir.

Let us design this model by following the steps of Guess and Check:

Guessing phase:

$$production(0,\ T)\ |\ production(1,\ T)\ |\ ...\ |\ production(5,\ T)\ \text{:-}\ time(T).$$

Which only makes one of the six alternative atoms production true.

As you can see above, we need to list each different atom of production, for each value of power that can be produced by the plant. This can be avoided by using the choice rules, which allow us to rewrite the guess phase into:

$$1 \leqslant \{production(Q,\ T) : power(Q)\} \leqslant 1\ \text{:-}\ time(T).$$

Where the atom $power(Q)$ defined for each value from 0 to 5 (e.g. $power(0..5)$.) In this way, for each time T, we assign a production quantity Q from among the possible ones.

Checking phase:

$$\text{:-}\ production(Q,\ T),\ waterLevel(L,\ T),\ waterRequired(WR,\ Q),\ L < WR.$$

With the atom $waterRequired(WR,\ Q)$ defining that we need WR water to produce Q (MW) of power.
Thus we check that we have the necessary quantity of water in the reservoir to produce energy.

3.6.4 An example of weak constraints

Consider the model defined before in section 3.6.2, in which we have the definition of the expected gain during each hour of the day. Our goal, as usually in hydroelectric scheduling problems, is to maximize the total gain to increase the profit.

To maximize the gain, we can use the weak constraint:

$$:\sim gain(G,\ T).\ [-G@1,\ T]$$

In which the maximization weight is the total value of G for each T, the minus sign before G indicates that we are doing maximization instead of minimization (standard). In this case the level is 1 and it is irrelevant because we have an unique optimization.

4 Encoding

This section will present the second step for our problem-solving methodology.

The encoding procedure is based on defining the logical rules and constraints of the problem environment, defined in section 2, with the syntax of ASP. In this way, we can provide the grounder-solver program with a formal definition of the search space of our hydro generation scheduling problem.

4.1 Data Model

We begin our specification of the problem encoding by starting with the data and parameters we have available.

As mentioned in section 2.5, for the solution of the hydro generation scheduling problem, we can use some data coming directly from the plant and then processed by ETL, estimation and forecasting processes, for instance by EGO data scientists.
These data must be selected with respect to the target scheduling horizon and then converted into the syntactic format of ASP, to be readable and compatible with the rules and constraints we will define.

Time Slot:

Instances of the predicate $time(T)$ represent each time slot that we consider for the problem. Due to the required time granularity, in our problem each slot will be of 1 hour.

E.G. in a scheduling horizon of 5 hours we will have:
$time(1)$. $time(2)$. $time(3)$. $time(4)$. $time(5)$.
and in this case we can simplify the syntax as $time(1..5)$.

Power:

Instances of the predicate $power(Q)$ represent the powers that can be produced by the hydropower plant.

E.G. considering a maximum production of 5 MW (granularity of 1 MW):
$power(1)$. $power(2)$. $power(3)$. $power(4)$. $power(5)$.
and in this case we can simplify the syntax as $power(1..5)$.

Final time slot:

We define as input an instance of the predicate $finalTime(FT)$ representing the final time slot for the schedule horizon.

E.G. in a scheduling horizon of 12 hours we will have:
$finalTime(12)$.

Minimum threshold of water:

We define as input an instance of the predicate $minWaterLevel(L)$ representing

the minimum level of water allowed in the reservoir.
E.G. $minWaterLevel(35)$.

Maximum threshold of water:

We define as input an instance of the predicate $maxWaterLevel(L)$ representing the maximum level of water allowed in the reservoir.
E.G. $maxWaterLevel(70)$.

Starting water level:

We define as input an instance of the predicate $startingLevel(L)$ representing the level of water in the reservoir at the beginning of the schedule.
E.G. $startingLevel(44)$.

Final water level (minimum):

We define as input an instance of the predicate $finalLevel(L)$ representing the desired minimum level of water in the reservoir at the end of the schedule.
E.G. $finalLevel(50)$.

Price of energy:

Instances of the predicate $price(PR,\ T)$ represent the value of the estimated price of energy for the time slot T.
E.G. for the first 5 time slots we could have:
$price(95, 1)$. $price(87, 2)$. $price(82, 3)$. $price(51, 4)$. $price(58, 5)$.

Inlet water flow:

Instances of the predicate $inWaterFlow(IF,\ T)$ represent the estimated amount of water entering the reservoir during the time slot T.
E.G. for the first 3 time slots we could have:
$inWaterFlow(8, 1)$. $inWaterFlow(8, 2)$. $inWaterFlow(9, 3)$.

Water usage:

To avoid overloading the complexity of the asp program, we have chosen to calculate water usage (the measure of the amount of water required to produce a certain amount of energy) as part of a preprocessing step before the encoding. This data can be imported directly using the ASP syntax.
Instances of the predicate $waterUsage(WU,\ PW1,\ PW2)$. represent water usage per hour, in relation to the power variation ($PW1 \rightarrow PW2$).
E.G. $waterUsage(0, 0, 0)$. $waterUsage(11, 0, 5)$. $waterUsage(23, 0, 10)$.

Power configurations:

Instances of the predicate $config(C,\ P1,\ P2)$ represent the different power configurations available, where C is a label and $P1$, $P2$ are the minimum and max-

imum power available for that configuration.
E.G. $config(1,\ 0,\ 13)$. $config(2,\ 1,\ 17)$.

Hours of the day:
We need to know, as an aid for the encoding of some rules, how to divide the time slots according to the day they are part of:
Instances of the predicate $day(D,\ T)$ defines that the day D includes the time-slot T.
E.G. $day(1,\ 1)$. $day(1,\ 2)$. $day(1,\ 24)$. $day(2,\ 25)$.

Production stops:
An instance of the predicate $maxStops(SN)$ represents the maximum number of stop to the production we can have during the daily schedule.
E.G. $maxStops(3)$.

4.2 Problem Encoding

Now we describe the ASP rules used for solving the short-time hydro generation scheduling problem of the considered power plant. We proceed by following the Guess&Check&Optimize programming methodology defined in section 3.4.

4.2.1 Guess Phase

In the guess phase, we use ASP rules to describe the search space of the problem.

Production:
Power plant turbines can produce a specific amount of power each hour.

We can use a choice rule to define the different productions that we can have for each time slot:

$$\{production(Q,\ T)\ :\ power(Q)\} = 1\ \text{:-}\ time(T). \tag{1}$$

that means to set in the predicate $production(Q,\ T)$, for each time slot T, a certain amount of power Q among all the available powers that can be produced.

Expected Gain:
The expected profit from the selling of the energy is given by the product between the MGP price of energy and the quantity of power that we are able to sell.

We can define the following rule to express the expected gain we might have for each time slot: [see section 3.6.2]

$$gain(G,\ T)\ \text{:-}\ price(PR,\ T),\ production(Q,\ T),\ G = PR * Q. \tag{2}$$

Production & Expected Gain:

To improve the quality of the ASP definition and reduce the time for finding the solution, it may sometimes be important to consider merging multiple rules together.

This change was also made in our case, in which rules (1) and (2) were rewritten in the form of the following rule:

$$\{production(Q,\ Q * PR,\ T)\ :\ price(PR,\ T),\ power(Q)\} = 1\ \text{:-}\ time(T). \tag{3}$$

In this rule, we set within the predicate *production*$(Q,\ G,\ T)$, both how many MW of power can be produced and how much is, in that case, the relative expected gain, for each time slot of the schedule.

Reservoir Water Variation:

The water variation in the reservoir depends on the water inflow as input and on the amount of water sent to the turbines.

The following rule defines the variation of water in the reservoir:

$$\begin{aligned}&waterVariation(D,\ T)\ \text{:-}\\ &time(T),\ production(Q1,\ \_,\ T{-}1),\ production(Q2,\ \_,\ T),\\ &inWaterFlow(IF,\ T),\ waterUsage(WU,\ Q1,\ Q2),\ D = IF - WU.\end{aligned} \tag{4}$$

The predicate $waterUsage(WU,\ Q1,\ Q2)$ contains the amount of water that needs to be "used" to produce $Q2$ [MW] of power, however, as described in section 4.1 it is also necessary to indicate the initial power output of the plant since the measurement also depends on the variation in power output.

For the above reason we have to add in the "Reservoir Water Variation" rule the 2 predicates *production*$(Q1,\ G,\ T*)$ (where the gain is not relevant), which considers the current time slot (T) and the previous time slot (T-1).

Reservoir Water Level:

The water in the reservoir increases/decreases over time, depending on the water variation.

We introduce a $waterLevel(L,\ T)$ predicate to describe the quantity of water over time in the reservoir:

We define the starting water level for the reservoir by adding a rule with the predicate *startingLevel*:

$$waterLevel(L,\ 1)\ \text{:-}\ startingLevel(L). \tag{5}$$

Then, we add a new rule to express the change over time of the water in the reservoir:

$$\begin{aligned}&waterLevel(LN,\ T+1)\ \text{:-}\\ &waterLevel(L,\ T),\ waterVariation(D,\ T),\ time(T),\ LN = L + D.\end{aligned} \tag{6}$$

This rule describes the quantity of the water at the beginning of the next time slot T based on the current level and the change in water that occurs.

Power plant configurations - full schedule horizon:

A specific power configuration, which affects the production capacity of the plant, must be selected from those available, at the beginning of the schedule.

We introduce a choice rule to select one among the available configurations.

$$\{selConfig(C)\ :\ config(C,\ \_,\ \_)\} = 1. \tag{7}$$

We introduce a rule to define the predicate $power(Q)$ according to the selected configuration. (It is necessary to remove the power predicate from the inputs).

$$power(P1..P2)\ \text{:-}\ selConfig(C),\ config(C,\ P1,\ P2). \tag{8}$$

Power plant configurations - daily:

A specific power configuration, which affects the production capacity of the plant, must be selected from those available, at the beginning of each day.

This is the same concept as before , but this time we increase the complexity by allowing a daily selection of the configurations.

We need to add to the predicates of the rules ”7” and ”8” a term referring to the day and adapt the predicate ”1”:

$$\{selConfig(C,\ D)\ :\ config(C,\ \_,\ \_)\} = 1\ \text{:-}\ day(D,\ \_). \tag{9}$$

$$power(P1..P2,\ D)\ \text{:-}\ selConfig(C,\ D),\ config(C,\ P1,\ P2). \tag{10}$$

$$\{production(Q,\ T)\ :\ power(Q,\ D)\} = 1\ \text{:-}\ time(T),\ day(D,\ T). \tag{11}$$

4.2.2 Check Phase

In the check phase, we introduce constraints to cut off the unwanted solutions.

Water Level Constraints:

The reservoir can hold a certain amount of water, the level of which must be maintained between a minimum and maximum threshold.

We introduce the following 2 constraints to ensure, in valid solutions, that the minimum and maximum water level thresholds in the basin are respected.

$$\text{:-}\ waterLevel(L,\ T),\ minWaterLevel(MIN),\ L < MIN. \tag{12}$$

$$\text{:-}\ waterLevel(L,\ T),\ maxWaterLevel(MAX),\ L > MAX. \tag{13}$$

If we have a predicate $waterLevel(L,\ T)$ true, then L cannot be lower/higher than the minimum/maximum allowed threshold.

Minimum Final Water Level:

In such short-term optimization, it is important to avoid using all the available water so as not to be in an unfavorable situation with the next schedule.

We introduce a constraint to prevent the water in the reservoir from falling below a minimum level at the end of the time horizon.

$$:\!-\ waterLevel(L,\ FT+1),\ finalLevel(FL),\ finalTime(FT),\ L < FL. \quad (14)$$

Maximum number of production stops:

The production should not be stopped for more than N times a day.

We introduce a new rule to define the concept of daily stops in production:

$$\begin{aligned}&stop(T,\ D)\ :\!- \\ &production(0,\ \_,\ T),\ production(Q,\ \_,\ T{-}1),\ day(D,\ T),\ Q > 0.\end{aligned} \quad (15)$$

Where the predicate $stop(T,\ D)$ means that we have a stop in the production at time T of day D.

Finally, we add a new constraint to impose the maximum number of stops:

$$:\!-\ S = \ \#count\{T\ :\ stop(T,\ D)\},\ day(D,\ \_),\ maxStops(MS),\ S > MS. \quad (16)$$

4.2.3 Optimize Phase

In this phase, we add weak constraints to define the target of the optimization.

Maximize the profit from energy selling:

Optimize the short-term production (scheduling) of the hydropower plant, so as to produce more energy at times when it is most profitable; such as when the prices are high.

The only optimization we need to do in this hydro generation scheduling problem is to maximize the expected profit from the sale of electricity.
We introduce the following weak constraint:

$$:\sim production(\_,\ G,\ T).\ [-G@1,\ T] \quad (17)$$

We consider for the optimization the predicate $production(\_,\ TG,\ T)$, which includes the value of the expected gain G, and we maximize by using the syntax of the weak constraint $(-G)$.

The encoding shown successfully enables the hydro generation scheduling problem under study to be solved. Assuming that all rules and constraints have been defined correctly it is possible to obtain the best possible production scheduling related to the maximum expected gain achievable from the production.

4.2.4 Improvements

The problem under analysis is complex, because of the size of the search space and the presence of physical variables. As is evident from the analysis of related work, many computer systems may not have the computing power to solve the optimization in a time compatible with the application.

In our case, for example, the scheduling of hydropower generation has to be executed every day, and in addition, the time available for execution is short and limited because we have some deadlines to meet [see section 2.3]. In these situations, it may be necessary to introduce some additional constraints, that can simplify the procedure.

Adding these rules means changing the problem environment and there is the risk of losing some (even optimal) solution from the result, so it is important in this case to proceed with caution and check that the trade-off between execution time and optimization performance is still positive for the purpose of the results.

Additional Constraint for the production:
We can introduce a new constraint to drive the production schedule using a heuristic approach:

$$\begin{aligned} &:\!-\ \mathit{production}(Q1,\ \_\ T1),\ \mathit{price}(PR1,\ T1),\ \mathit{price}(PR2,\ T2),\\ &\mathit{production}(Q2,\ \_\ T2),\ \mathit{time}(T1),\ \mathit{time}(T2),\\ &\mathit{day}(D,\ T1),\ \mathit{day}(D,\ T2),\ PR2 > PR1,\ Q2 < Q1. \end{aligned} \tag{18}$$

In other words, considering two time slots on the same day, if one of them (T2) has an higher energy price than the other (T1), then it is not possible (according to this constraint) that, between the two, we produce more energy in slot T1.

This comes from real-world experience and observation of data, where the common and immediate choice is to focus production during periods when energy prices are highest.

This constraint may cut off some desired and optimal solutions, especially in small reservoirs, due to water management or for particular price distributions.

Many times, if there are high price spikes extended over several consecutive times, the water in the reservoir is not sufficient to cover the entire period, thus there will be, a forced lowering of production in an instant when the price is nevertheless high. The constraint introduced in this section may consequently induce a limitation in production in the other separate periods.

To tackle this problem, we introduced a predicate day inside the rules so to apply the heuristic to only a limited horizon of 24 hours, in which the price distribution should be more regular.

On the next page, we have a full view of the defined ASP encoding.

```
% ASP - HYDROELECTRIC POWER PLANT - STHS PROBLEM
% Poli Riccardo - 11/2022
% ----------------------------------------------------------------------

% -------------------------------------
% GUESS STEP

        production(Q,0,0) :- production(Q,_,1).

        %{ production(Q,Q*PR,T) : price(PR,T), power(Q) } = 1 :- time(T).

        waterVariation(D,T) :- time(T), production(Q1,_,T-1), production(Q2,_,T),
                inWaterFlow(IF,T), waterUsage(WU,Q1,Q2), D = IF-WU.

        waterLevel(L,1) :- startingLevel(L).

        waterLevel(LN,T+1) :- waterLevel(L,T), waterVariation(D,T), time(T), LN = L+D.

% -------------------------------------
% CHECK STEP

        :- waterLevel(L,T), minWaterLevel(MIN), L < MIN.

        :- waterLevel(L,T), maxWaterLevel(MAX), L > MAX.

        :- waterLevel(L,FT+1), finalLevel(FL), finalTime(FT), L < FL.

% -------------------------------------
% IMPROVEMENTS

        :- day(D,T1), day(D,T2), production(Q1,_,T1), price(PR1,T1), price(PR2,T2),
                production(Q2,_,T2), time(T1), time(T2), PR2>PR1, Q2<Q1.

% -------------------------------------

% MAX DAILY STOPS

        stop(T,D) :- production(0,_,T), production(Q,_,T-1), day(D,T), Q>0.
        :- S = #count{T : stop(T,D)}, day(D,_), maxStops(MS), S > MS.

% POWER CONFIGURATIONS

        { selConfig(C,D) : config(C,_,_) } = 1 :- day(D,_).
        power(P1..P2,D) :- selConfig(C,D), config(C,P1,P2).
        { production(Q,Q*PR,T) : price(PR,T), power(Q,D) } = 1 :- time(T), day(D,T).

% -------------------------------------
% OPTIMIZE STEP

        :~ production(_,G,T). [-G@1,T]

% -------------------------------------

        #show price/2.
        #show production/3.
        #show waterLevel/2.
        #show minWaterLevel/1.
        #show maxWaterLevel/1.
        #show minPower/1.
        #show maxPower/1.
        #show startTime/1.
        #show finalTime/1.
        #show selConfig/2.
        #show config/3.
        #show stop/2.

% -------------------------------------

% ----------------------------------------------------------------------
```

5 Comparative experimental analysis

At the beginning of this chapter, we present the setting in which we performed the tests and an introduction to the scenarios we considered. Then, we analyze the results obtained from our solution in terms of the efficiency of the procedure and validation of the scheduling data.

5.1 Experimental settings

In our experiments, we leveraged the tools offered by Amazon Web Services (AWS), to align our work with the serverless technologies used by EGO. This allowed us to test and deploy our code on a production-like environment having high computing capacity and state-of-the-art backend structure.

We used the following services in particular:

Amazon EC2:

EC2 enables software and application deployment by providing a web service, in which a user can start an Amazon Machine Image (AMI) to create a distributed virtual machine, which will contain the desired software.
A user can create, launch and close any instance at any time.
Specifically, in our case, we used a "c5.4xlarge" instance that features an Intel Xeon Platinum 8000 processor with 8 cores, 16 threads, clock speed of 3.5 GHz and a memory of 32 GiB.

Amazon ECS - Fargate:

AWS Fargate is a technology that you can use together with Amazon ECS to run containers without having to manage servers or clusters of Amazon EC2 instances.
With this solution, it was possible to deploy a container that can read the necessary data and run a Clingo wrapper at startup to perform the search for scheduling procedure.
It is possible to select a specific amount of vCPUs and ram to be allocated to the Fargate task; moreover, each task is isolated and does not share resources with the others. In our case, we have an Intel Xeon Platinum 8375C processor with 8 cores, 16 threads and a clock speed of 2.90 GHz; the memory allocated is 32 GiB.

We can choose some additional parameters for Clingo, the effectiveness of these parameters depends on the type of problem and so we had to make a selection and choose the following ones that are most suitable for our instance:

- maximum timeout: 3600 seconds

  Allows us to set a maximum execution time for the solver.

- parallel mode: 16, split

Allows us to take advantage of multiple parallel threads in a "splitting-based" search, during the search for the optimum.

5.2 Experiments and Results

In this section, we will describe different types of experiments that we performed with the real-world data of the hydroelectric power plant provided by EGO.

Each attempt is related to a specific scenario of a typical hydro generation scheduling problem and which follows the definition of chapter 3. Scenarios are designed in a way that makes it possible to evaluate different features of the solution, such as the validation of the model and the efficiency of the methodology and results.
Since the problem is related to time series and time-dependent data, for reasons of data availability each experiment will be conducted considering past situations.

- In the **first scenario**, we will start by evaluating the assigned schedule for the hydropower production of a single day of the year and an horizon size of 24 hours; then, we will try to improve the quality of the results by adding an heuristic rule (defined in chapter 4.2.4) and a procedure based on the "divide-and-conquer" paradigm; at the end we will try to increase the horizon size, to see if there are changes on the performances.

- In the **second scenario**, we will try to find the solutions for many different instances, to see if the execution time and/or the quality of the assigned schedule for the production are different, while having different instances of the problem.

- In the **third scenario**, we will test a real world based application of the problem, to replicate a situation of execution of the optimizer in the production environment; moreover, we will try to focus on the actual result, making a comparison with the schedule obtained by the program used by EGO.

In each experiment, we will solve the problem encoding using Clingo with the technical settings given in section 5.1. For each attempt, we will use the encoding of the problem defined in chapter 4.

5.2.1 Scenario 1

In the first experiments, we will test the solution developed in this thesis by introducing simple instances of an hydro generation scheduling problem.
The optimization will be set up so that we can validate the model of the solution and make an evaluation on the performance of the optimizer.

Scenario Description

Let us consider as an example, that we are on the date "27/01/2022", and we want to optimize the power production of the next hours by following the rules and insights defined in 3.

In the next attempts, we want to obtain, as a result, the schedule that our optimizer has found to be optimal for the hydropower production in a specified amount of time and by considering an horizon of interest.

For this scenario, we will consider real data and parameters provided by EGO, in our case we took all the necessary data for the schedule of the 27/01/2022:

- Prices of Energy

  The price of the energy for each hour of the horizon (see figure 5).

- Inlet Water Flow

  The flow of water in input to the reservoir for each hour of the horizon.

- Water usage

  The water usage for each variation of the power produced, that is a matrix of length 24x24, representing the quantity of water required to the turbines for producing at a certain power. (Read section 4.1 for more insights.)

- Starting Water Level

  Initial level of water in the reservoir (level on "27/01/2022" at 00:00, when the schedule horizon start).

As for other parameters, we used the values inserted in the table 1:

Parameter	Value
Time Slots	beginning from: 27/01/2022 (horizon size: 24h - 48h - 72h - 96h)
Power Levels	0 to 23 MW
Water Level Thresholds	3.5 to 7 m
Final Water Level	5 m (at the end of the schedule)

Table 1: Parameters for the scenario of the first experiment.

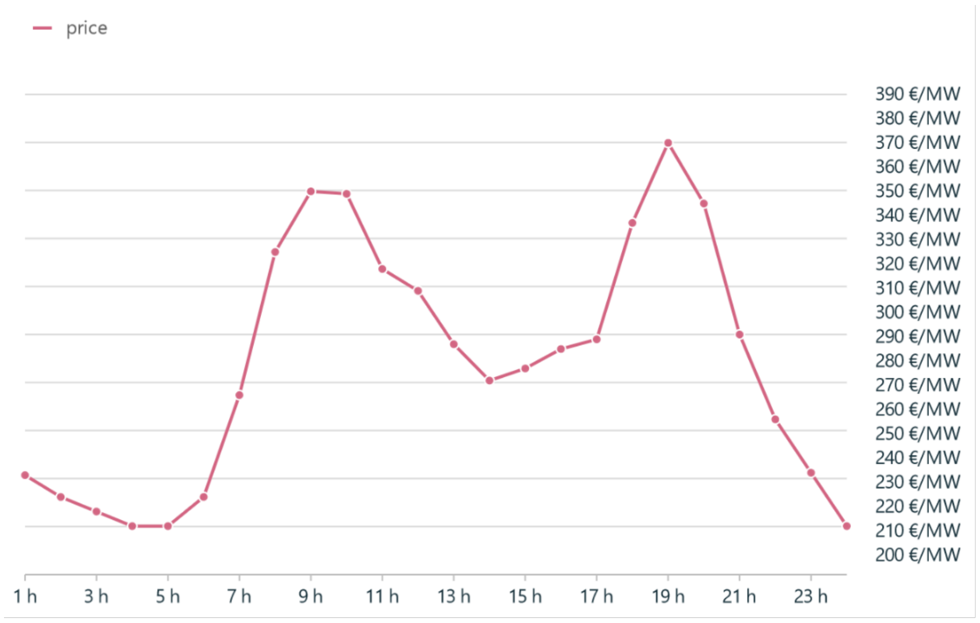


Figure 5: Expected prices of energy for the day of 27/01/22.

Experiment 1 - basic 24h schedule

As a first attempt, we ran the optimization for a schedule of 24h, this way we can see a first example of the results provided by the procedure:

In the following table, we can see the different types of parameters of the result considered during this analysis: the execution time spent for the solving phase and 3 rows with data regarding the schedule obtained (Profit, Total production and Schedule Horizon).

In this case, we can see that we found a schedule of 24 hours, with an expected profit of 17817 € and a total production of 58 MWh of energy. However, in this case, the solver did not manage to find the optimum and the execution was interrupted with the time limit of 3600 seconds, referred to as "TIME" in the table, that we imposed in section 5.1.

Schedule for the 27/01/22

Execution Time	TIME
Profit	17817 €
Total Production	58 MWh
Schedule Horizon	24 h

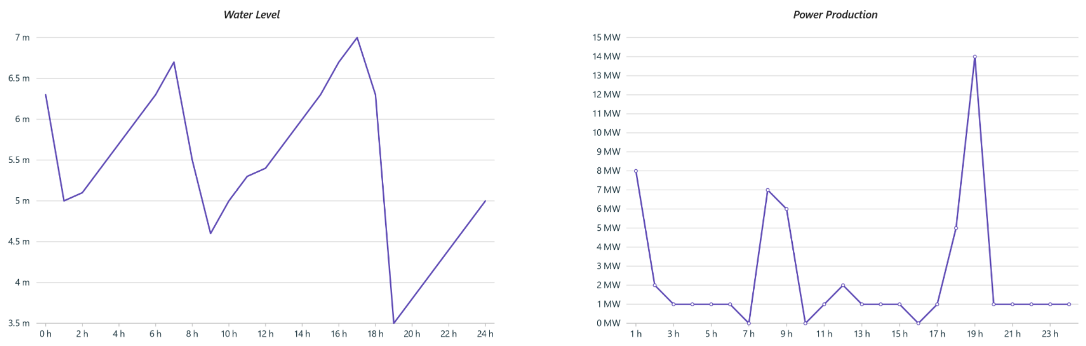


Figure 6: Water level and Production schedule (27/01/22).

The charts in figure 6 can show to us two pieces of relevant information about the schedule: on the left, we have a line chart representing the variation of water in the reservoir during the time horizon, with the hours (0-24) that we are considering in the x-axis and the values of water level in meters in the y axis. The second chart, on the right, represents the effective production of the power plant according to the schedule obtained, we have in this one the 24 hours (1-24) in which to produce and the amount of power in MW in the y-axis.

Let us look, first, at the line chart of the reservoir water level; as can be seen, the optimizer found a production schedule that, in the 24h horizon of the schedule, never exceeded the two minimum and maximum threshold limits (3.5 and 5 m, respectively).

Another validation of the model is given to us by the fact that the reservoir, at the end of the day, has a water level of 5 m; recall in this sense that, in the encoding, we imposed to have at least this amount of water at the end of the schedule.

To better analyze the allocated production, let us look instead at the following graph in figure 7 which has the two slopes of power produced and the expected price of energy for each hour of the schedule, showing us what is the relationship between these two fundamental data.

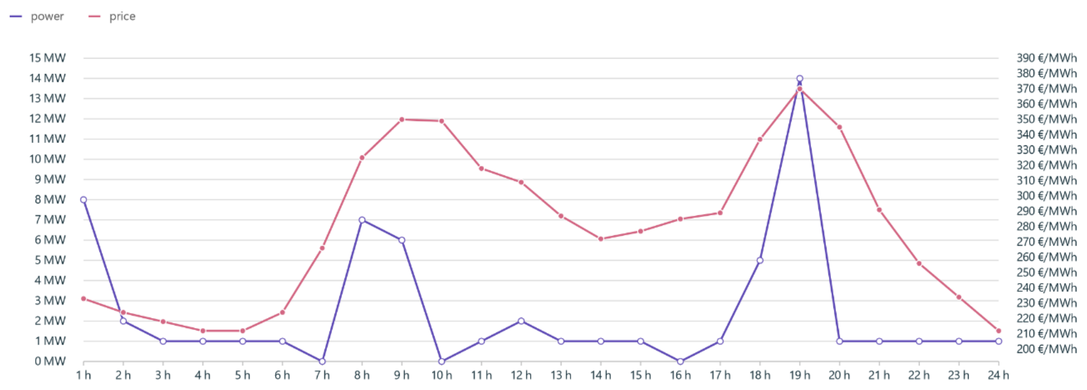


Figure 7: Power production and Expected prices for the schedule (27/01/22).

In this schedule, energy production has been allocated to the hydropower program in quantities between 0 and 14 MW, certainly far below the maximum amount that can be produced by the plant (23 MW). However, this fact does not worry us because, as we will see in the next scenario, this can happen in some instances, based on the quantity of water available

The result itself is consistent with the suggestion of encouraging production when prices are the highest; in fact, in this case, the maximum production (14 MW) is reached at the peak price of 370 €/MWh in the 19th hour.

We can see that the first two peaks of prices in hours 9 and 10 are not well managed by this schedule, probably it needs more time to find a better solution.

Experiment 2 - Heuristic Encoding

In the section 4.2.4, we defined an additional constraint for the problem under study, to try to improve our solution. Let us try now to understand what pros we have in using that heuristic encoding, rather than the standard encoding used in the first experiment.

For this attempt, we took the situation in scenario 1 and tried to solve it, using 2 different ASP encodings: the standard one and the one based on heuristics, introduced in section 4.2.4.

Execution Time

Horizon Size	Standard Encoding	Heuristic Encoding
24h	TIME	2 s
48h	TIME	TIME
72h	TIME	TIME
96h	TIME	TIME

Expected Profit

Horizon Size	Standard Encoding	Heuristic Encoding
24h	17817 €	18643 €
48h	31856 €	36499 €
72h	45044 €	52036 €
96h	58846 €	66317 €

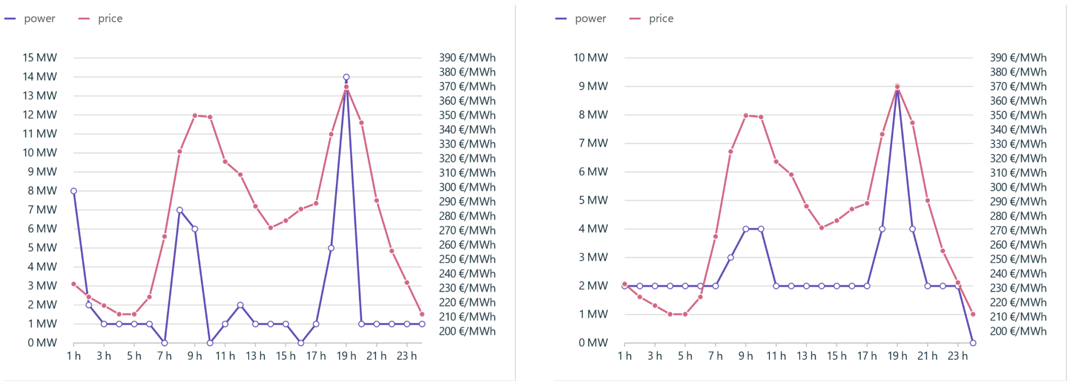
Here above we have two tables; one for the execution time and the other for the expected profit of the scheduling, each of them is showing to us the value which was recorded by the two types of encoding, with a variable horizon of 24, 48, 72 and 96 hours.

As can be seen from the tables, standard encoding takes much longer to execute and does not manage to finish before the time limit of 3600 seconds (TIME) also while considering the smallest horizon of 24 hours. The rule introduced, on the other hand, changes the problem and makes it more feasible to find early a better solution in terms of profit.

Since the heuristic encoding is based on the idea, not always beneficial, of producing less when the prices of energy are lower, we can have a situation in which we are losing some admissible solutions. We can eventually consider to relax this constraint, making it less heavy with respect to the model or to chain together the two different optimizations.

The heuristic encoding result may not get the best global possible result for the production schedule; however, in this scenario, it still manages to provide us with a better solution than the simpler model for large time horizons. In particular, it is able to do so in a shorter time and even with computationally weaker systems.

In Figure 8, we can see the two charts of energy price and production that were attributed to the schedules using the two different encodings.



(a) Standard encoding (3600s). (b) Heuristic encoding (2s).

Figure 8: Standard and Heuristic encoding results (27/01/22) - 24h.

Experiment 3 - Time segmentation

As seen above, there are situations in which the time needed to find a good solution is longer than the time limit we set for ourselves, especially while using standard encoding. To solve this issue, a procedure based on the divide-and-conquer paradigm was also tested, which can improve the results achievable by the optimizer (we will introduce this concept also basing by the study of Nobile et. al. in the thesis [48]).

This methodology starts by finding the production schedule for the entire horizon of interest, then the discretization of the problem is shifted to the optimization of sub-problems that contain the overall structure of the original problem but singularly over a smaller portion of the original optimization horizon. For these sub-problems, the boundary conditions are provided by the solution of the full optimization, in fact, good profiles for production and the water level curve are known from the standard procedure and these will be considered as the base curve on which to extrapolate the initial and final levels of water reservoirs for the different sub-problems.

In this procedure:

- The first optimization, which we call "global", will assign the production with visibility to the entirety of the price trend for the scheduling horizon.
- Subsequent optimizations, which we call "local", on the other hand, will attribute the production curves of smaller time intervals.

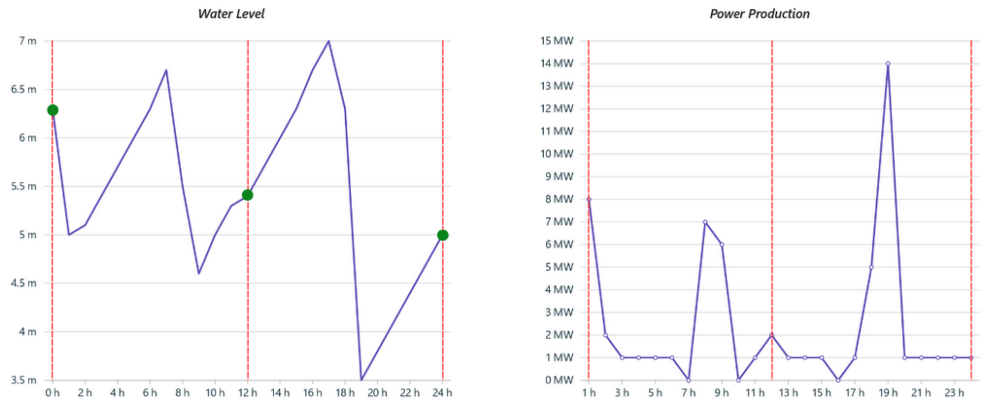


Figure 9: Result and subdivision for the time segmentation (27/01/22) - 24h.

As an example, let us take the result of the optimization with the standard encoding. We can see the result as production scheduled and reservoir water level in Figure 9.

To proceed with the introduced methodology let us consider, for example, 2 smaller 12-hour sub-intervals, as indicated by the red lines in the chart. The value of the water level, in correspondence with the green point, was determined by the global optimization and is decisive for the subdivision of the problem.

So in the first sub-problem, we will start the solver, but include only the data pertaining to the first 12 hours and considering the corresponding values of the green dots as the values of the initial and minimum-final water level.

In the second sub-problem, we will repeat the execution with an initial water level equal to the final water level returned by the first sub-problem and with a final minimum level equal to the value of the green dot of the 24 hours.

By recomposing the two results just obtained, as in figure 10, we can obtain the final result which, using the necessary time in the solving phase for the sub-problems, would be better than or equal to the result obtained globally.

We can perform this time segmentation technique in several ways: for example, we can divide the problem into smaller intervals of 6 hours leading to a shorter running time, or into larger 24-hour intervals, since, as we will see, for the purposes of our use case only the first 24 hours are of interest and so we can stop and compute only the first sub-problem.

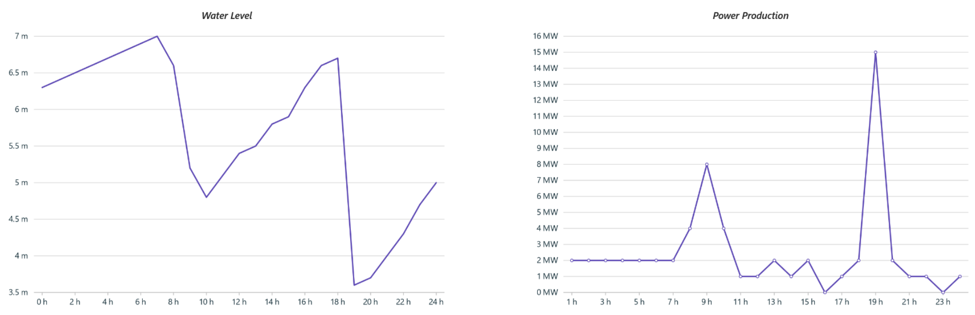


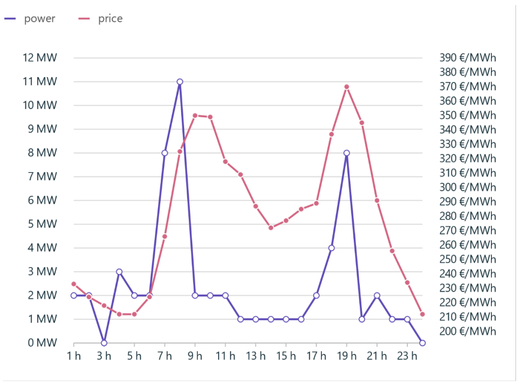
Figure 10: Results using the time segmentation methodology (27/01/22) - 24h.

In our attempt, we set 3600 seconds of maximum execution time (time limit, TL) for the standard optimization and 900 seconds each for the sub-problems.

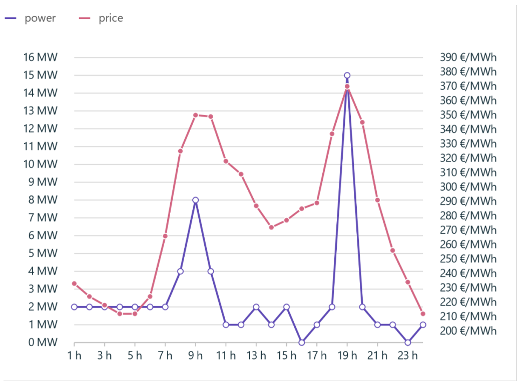
The total maximum execution time for the methodology with time segmentation will be 5400 seconds (3600 s + 900 s + 900 s). Therefore, to make a fair comparison, we will compare the result with a standard encoding execution having a time limit of 5400s.

Results from the Time Segmentation

	Time	Profit	Production	Horizon size
Standard Encoding	TL-3600 s	17817 €	58 MWh	24h
Sub-problem 1	TL-900 s	9301 €	32 MWh	12h
Sub-problem 2	TL-900 s	9362 €	28 MWh	12h
Time Segmentation	5400 s	18663 €	60 MWh	24h
Standard Encoding	TL-5400 s	17892 €	60 MWh	24h



(a) Standard encoding (5400s).



(b) Time segmentation (5400s).

Figure 11: Time Segmentation results (27/01/22) - 24h.

As we can see from the results in the table, by using the time segmentation procedure, we managed to find a better result with the same time available for the standard encoding solution.

In the charts in figure 11, we can see how the slope of the production obtained by time segmentation is different from the other and in particular, how this production assignment exploits better the peak of prices.

This technique can be applied also to the heuristic encoding and can be a way for obtaining better results, especially in complex problems with long schedule horizons.

In the following experiments, we will not use this technique because we are more interested in the solution provided directly from a single ASP solving phase and we want to evaluate the execution time of that phase without the interferences given by the sub-problems.

Experiment 4 - variation of the schedule horizon

In this fourth experiment, we will try to perform optimization with a broader scheduling horizon. In doing so, we will use the heuristic encoding; the goal will be to consider the short-term period of 4 days, as in the needs explained in section 2.6, while verifying that the performances and the time required for getting a good result remain stable.

Schedule for the 27/01/22

	24h	48h	72h	96h
Execution Time	2 s	TIME	TIME	TIME
Profit (first day)	18643 €	19920 €	19180 €	18855 €
Total Profit	18643 €	36499 €	52036 €	66317 €
Production (first day)	62 MWh	66 MWh	64 MWh	63 MWh
Total Production	62 MWh	118 MWh	174 MWh	229 MWh

As can be seen from the results, the time needed for solving the optimization problem is increasing with the number of hours in the horizon; in particular, the solver fails in this case to find the optimal encoding solution and is stopped at the end of the 3600-seconds time limit (TIME).

This does not mean that the result of a program with four days as the horizon size is not good; to prove this fact, we will now look at the results and comment on them.

From the table, we can see the values of total power production and profit for the full schedule horizon considered and also only for the first day. In this case, since we have different sizes of the horizons, it is fair to analyze the performances of only the first days.

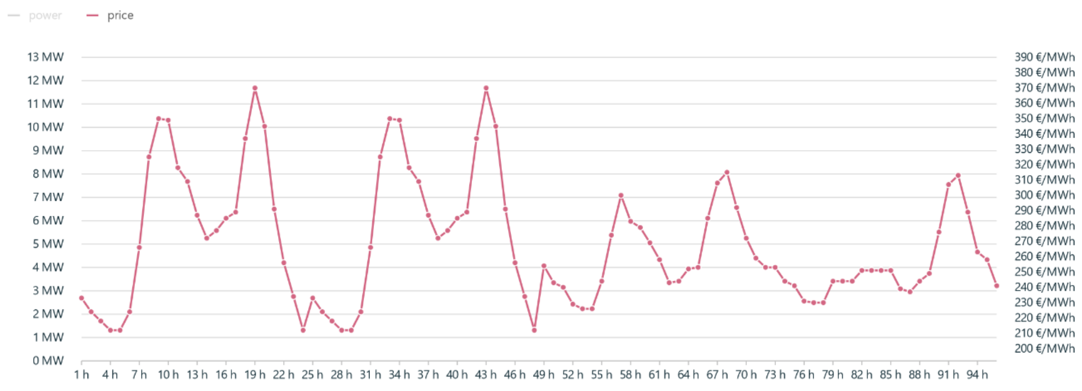


Figure 12: Expected prices of energy for the period from 27/01 to 30/01.

From the chart of prices of the four days in figure 12, we can see that the most favorable hours for the production are all grouped in the first two days, this means that we should produce more in these hours. Let us check now the quality of the optimization with some consideration for each horizon size that we have:

- In the **24h horizon**, the optimizer has a visibility only on the trend of prices of the first day, as we can see in figure 13, the two slopes of prices and productions are similar and we can see an increase in the production at the highest peaks of prices.
- In the **48h horizon** (figure 14), there is then an increase on the visibility of the price trends. The second day prices' slope is equal to the one of the first day, this should allow the scheduler to choose to produce energy in the best possible way, for example distributing itself similarly over the two available days.
- In the **72h and 96 horizon** (figure 15 and 16), the visibility on the third and fourth day allow to arrange the production in the best way: the first 48h still remain the ones with the best prices, with the trend being the same on each day. Production should therefore be more distributed in this period, with a behavior similar to the previous case, the 3rd and 4th days' production should then be adjusted accordingly to the residual or incoming water at the end of the 48h.

The following charts show the schedule found by our automatic solution, for the time horizon size of 24, 48, 72 and 96 hours.

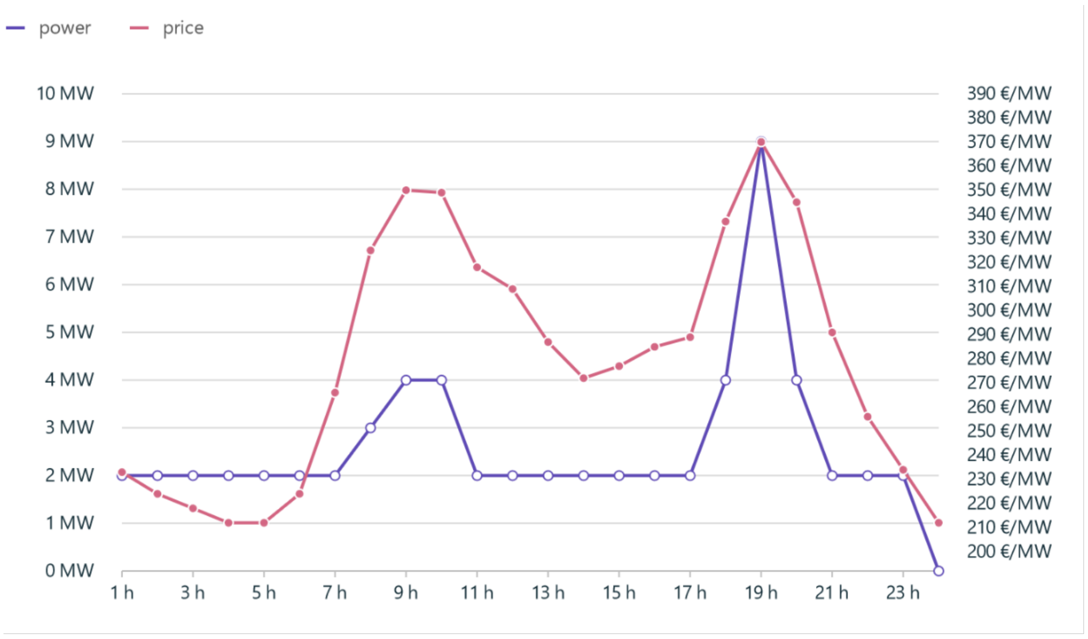


Figure 13: Production and Prices for the schedule (27/01/22) - 24h.

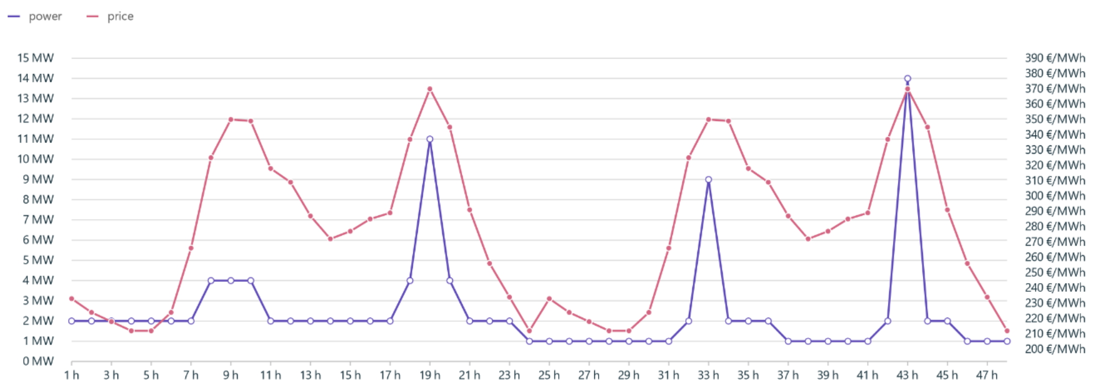


Figure 14: Production and Prices for the schedule (27/01/22) - 48h.

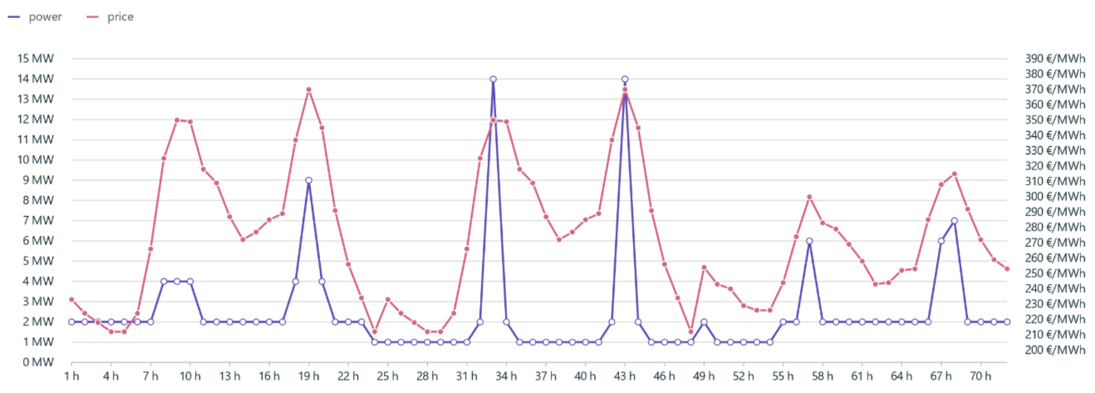


Figure 15: Production and Prices for the schedule (27/01/22) - 72h.

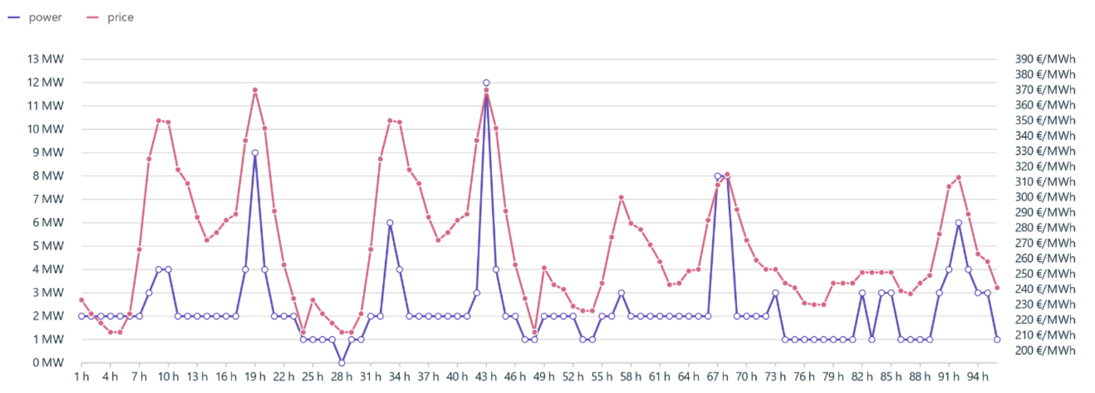


Figure 16: Production and Prices for the schedule (27/01/22) - 96h.

5.2.2 Scenario 2

In this second scenario, we want to test our solution by trying it on different types of problem instances.

Depending on the input data we have, there may be situations that can make the quest for the solution easier or harder. To verify this factor we tried to see the results we have from running several different instances, comparing the running time and the schedule obtained.

The three input data acting on the problem are:

1. The initial water level of the reservoir.
2. The time series of expected prices.
3. The water inflow into the reservoir.

Input 1 affects the production capacity of the plant, it depends on the past production performed.

Inputs 2 and 3 are more decisive for the purpose of this analysis; for example, if we have a lot of water available, we have the possibility to produce a greater amount of power and based on the slope of the price curve, we can have more or less optimal production choices.

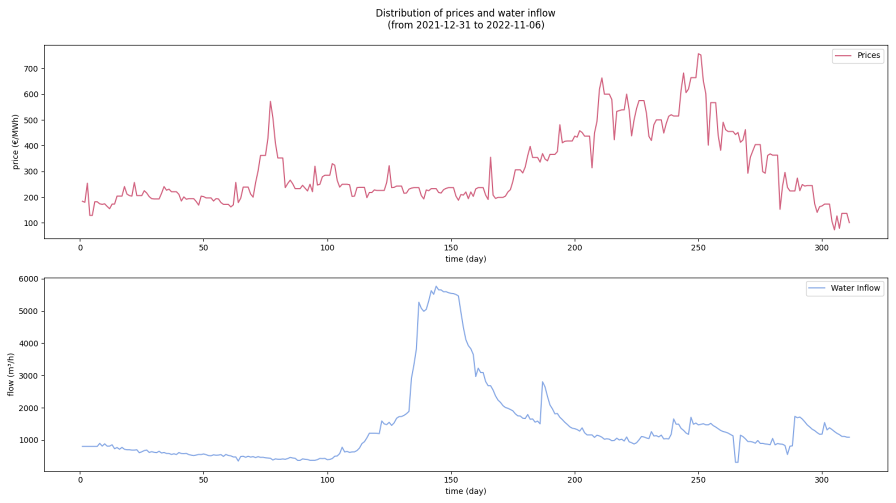


Figure 17: Distribution of prices and water inflow (January to October).

The instances that we are going to test were chosen so that we have different types of these data; in particular, we tried scheduling dates of different months of the year, from those available, since both water inflow and prices have seasonal fluctuations. [see sections 2.1.1 and 2.2 and image 17]

We selected the following dates:

1. 04/01/2022
2. 17/02/2022
3. 17/03/2022
4. 14/04/2022
5. 19/05/2022
6. 26/06/2022
7. 06/07/2022
8. 18/08/2022

For each of these instances, we collected the real-world data and then, we tried to start the search for the optimal schedule with the heuristic encoding and the time limit (TIME) set at 3600 seconds.

Execution Times

Problem Instance	24h	48h	72h	96h
1	3 s	TIME	TIME	TIME
2	2 s	TIME	TIME	TIME
3	3 s	387 s	TIME	TIME
4	4 s	TIME	TIME	TIME
5	2 s	TIME	TIME	TIME
6	6 s	TIME	TIME	TIME
7	13 s	TIME	TIME	TIME
8	11 s	TIME	TIME	TIME

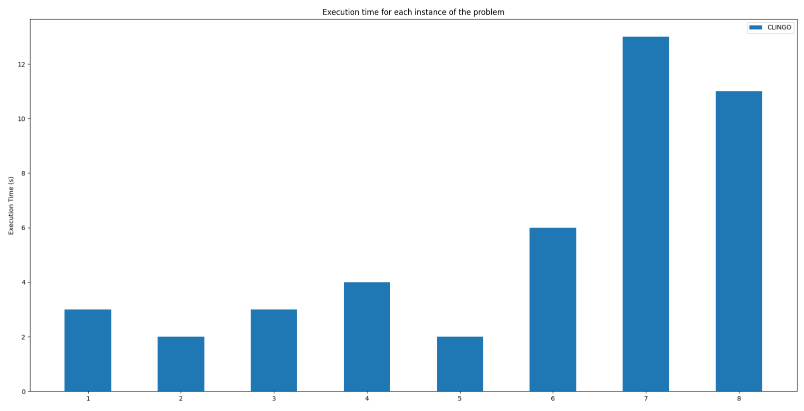


Figure 18: Execution time for each instance of the problem (24h).

From the execution times of the 24 hours horizon, shown in the table and in the figure 18, we can see that our solution performs in different ways, depending on the instance of the problem we are considering.
With instances 7 and 8 in particular, Clingo required more solving time to find the available optimal solution; moreover, we can see how, for the 48h hours horizon, the 3rd instance needed only 387 seconds to find the solution, differently from each other schedule.

From here, we will show some charts related to particular events/slopes obtained from these trials in the full horizon of 4 days. We will see that there are still good results for time-limited executions.

Instance 3 - low water available

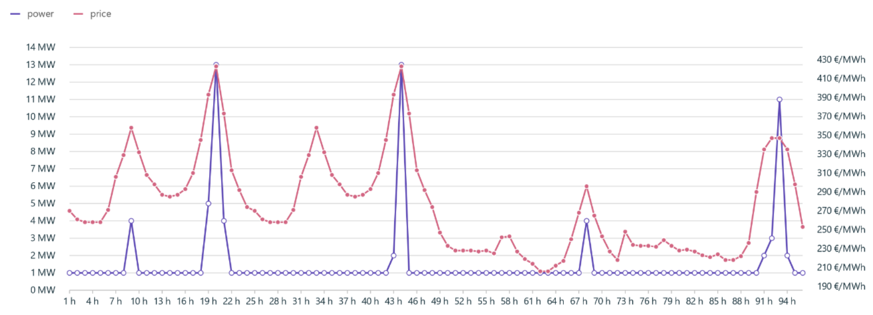


Figure 19: Production and Prices for the schedule (17/03/22) - 96h.

As can be verified from the graph of annual water inflow distribution in figure 17, the amount of water available during this schedule dates is low and therefore it is not possible to take advantage of all the price peaks that we have. In figure 19, the reduced flow of water creates a situation in which the possible output is narrow and should therefore be easily assigned to the high price spikes present in the interval; we note in this sense that the execution time for a small horizon in this scheduling was shorter than for the others, particularly with regard to the 48 hour horizon (387 seconds).

Instance 5 - high water available

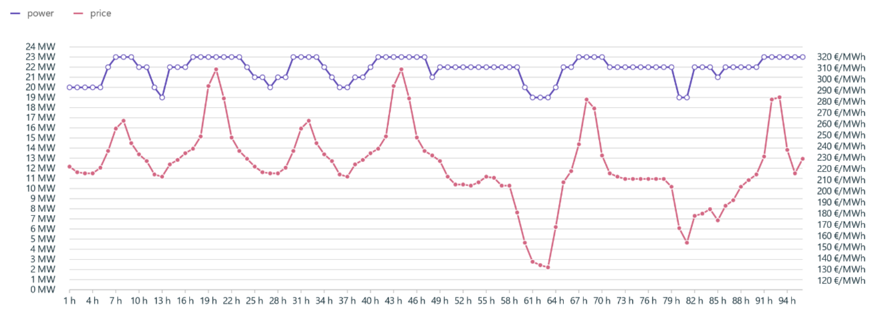


Figure 20: Production and Prices for the schedule (19/05/22) - 96h.

Instance number 5, in figure 20, is related to a May scheduling and gives a new result in which we see, for the first time in our analysis, a very high production of 23 MW (maximum power) prolonged over time and spaced by a small decrease, for few hours, when the price of energy falls.

If we look at the graph of the annual distribution of water inflow, we notice that just in the month of May there is a huge amount of water available for the hydropower plant, compared to the other times of the year. This phenomenon is well known and is due to the melting of ice with the arrival of the warm seasons.

Our solution managed to assign high production to this schedule, perhaps, also driven by the fact that the reservoir is filling up quickly and lower powers may cause the "minimum water level threshold" constraint to be broken. This condition, upside down but analogous to that of the previous instance for the low water, causes the problem to have a reduced number of allowed schedules.

Instance 7 - medium water inflow

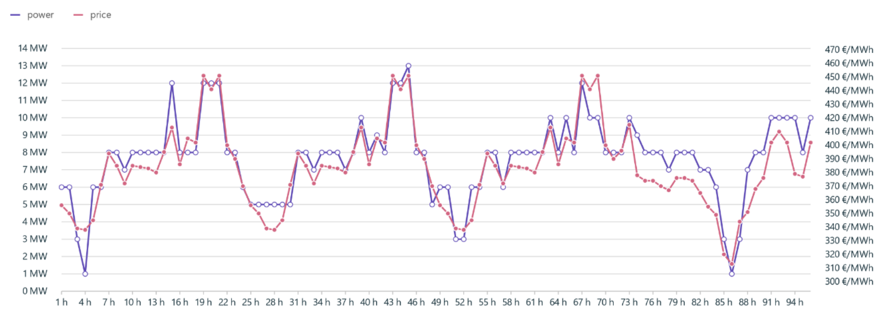


Figure 21: Production and Prices for the schedule (06/07/22) - 96h.

In the 06/07/2022 schedule, (instance 7), as we can see from the image 21, the hourly production curve tries to follow the behavior of price trends more closely; there are water savings and accumulation phases at low price peaks, alternating with some production increases at strategic points.

In this scenario there is more water available than in the 3rd instance but less respect to the 5th, in fact the power plant is able to maintain a stable production around 8 MW without lowering the reservoir level. This factor increases the possible choices in terms of power produced per hour.

The shape of the slope of the prices, like in this instance, is one of the focus of our encoding with heuristics, due to its irregularity, it tends to carry with it the assignment of the production. It can be seen that, for example, production above 8 MW is focused almost uniquely at the high price peaks, without exploiting other time slots. But, in this case, the heuristic encoding is important for the search of a solution, because it help the solver choose between the big amount of combinations that are allowed, for the schedule of a period with medium water inflow.

5.2.3 Scenario 3

The third scenario we have chosen to consider is the typical use case of the hydro generation scheduling problem:

Assume that we are in the morning of 16/03/2022, what we should do is preparing all the input data at our disposal and schedule in time the hydroelectric production for the next day ("17/03/2022"). From experience, to evaluate the short-term trend of prices and obtain the best schedule possible, the schedule should consider the reference horizon of the 4 successive days; for example, in our case from "17/03/2022 00:00" to "20/03/2022 24:00" local time.

The procedure described here will need to be repeated every day of the year from EGO, in order to have available all the schedules that are used to drive the power plant and market estimates.
As we said, the particularity of this methodology is that we are considering a 4-day horizon, that allows us to adjust production according to price trends considering multiple days of the week; however, in terms of production, we are only interested in knowing what is the schedule of the first day, because the followings are used only as a metric.

In this scenario, we will use real-world data provided by EGO and we will try the 4 scheduling operations with the use of the heuristic encoding. The results obtained will be then compared to the ones obtained in the same period from the program of EGO.

We will assign the following parameters:

- The "Runtime" Day
  (16/03/2022): the day on which we collect data and execute the optimization.
- The Full period of the experiment.
  (17th, 18th, 19th, 20th of March 2022): the days whose scheduling we want to optimize, each will correspond to a different execution of the solver.
- A full short-term Schedule horizon
  (96h): the number of hours we will consider as the schedule horizon for each optimization.

Results:

This experiment was carried out considering the schedules of the days from the 17th to the 20th of March and in this period, as seen in section 5.2.2, the time for each solving phase is equal to the time limit we had chosen of 3600 seconds. As can be seen from the results, however, the scheduling is still very good and is still competitive, in terms of energy production and profit, compared to the scheduler currently used by EGO.

Expected Profit from the schedule (4 days)

	Day 1	Day 2	Day 3	Day 4	Total
ASP Solution	16223	12406	5882	6532	41043 €
EGO Optimizer	14518	12329	5339	7425	39611 €
Difference	1705	77	543	-893	1432 €

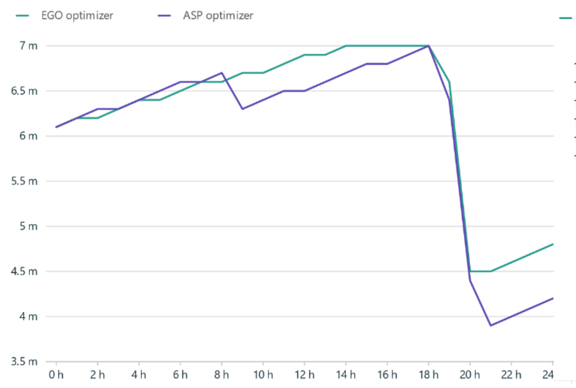
Expected Power Production from the schedule (4 days)

	Day 1	Day 2	Day 3	Day 4	Total
ASP Solution	46	39	28	29	142 MWh
EGO Optimizer	42	39	26	35	142 MWh
Difference	4	0	2	-6	0 MWh

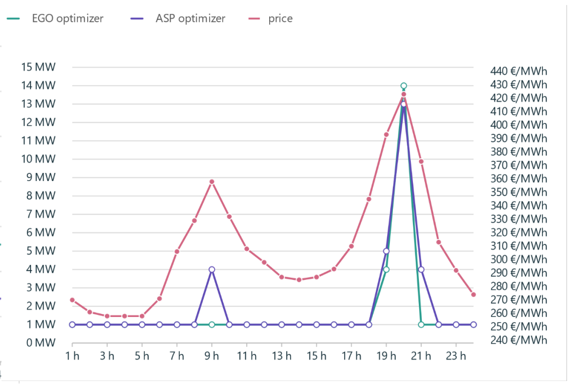
Looking at the charts in the figures 22, 23, 24 and 25, we can see the change in reservoir level and scheduled energy production for the four previously selected dates.

Our solution (in blue) seems to have behaved in the right way, we can see that, for example, the behavior, which is recorded in the hours with low energy prices, is to keep the production at low power levels (1 MW) and is also the same as implemented by the EGO scheduler (in green).
One difference we notice, probably due to the use of heuristics, is that in our case there are sometimes small unneeded energy productions at the price peaks at the beginning of the day.
Water is handled differently, EGO's scheduler tries to recharge the reservoir as much as possible, sometimes even causing some unintended water overflow, to try to have enough water near the maximum price peaks; on the other hand, our solution forces itself to keep the maximum water level around 7 meters and, to do this, it is sometimes "forced" to produce in disadvantageous situations. There is also a specific time (1st hour of the 20/03), in which the EGO scheduler choose to produce to reduce the water in the reservoir, we do not need to this instead in our result and the water was kept for the production in the most advantageous time.

Despite some inaccuracies due to the approximation of the data, the schedule we obtained still has high performance, even compared to the reliable EGO scheduler.

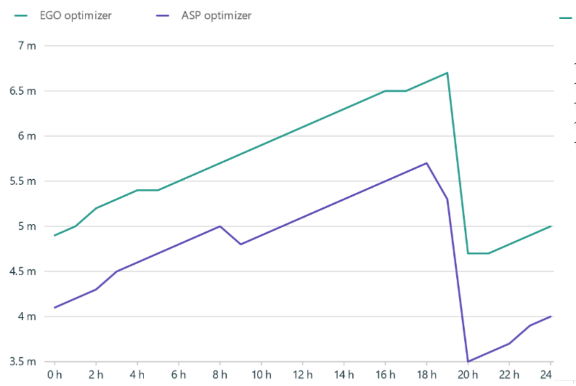


(a) Water Level

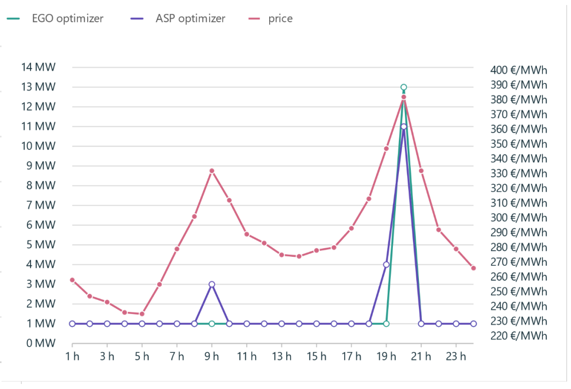


(b) Energy Production

Figure 22: Schedule comparison (17/03/22) - first 24h.

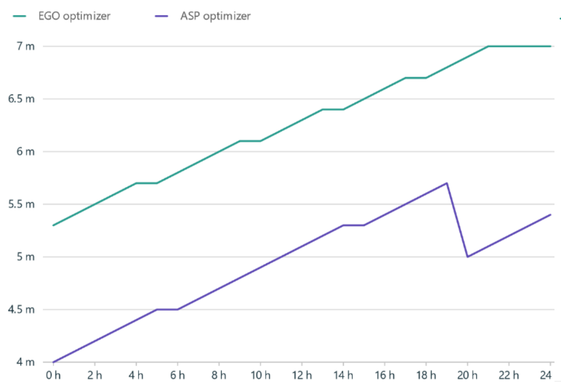


(a) Water Level

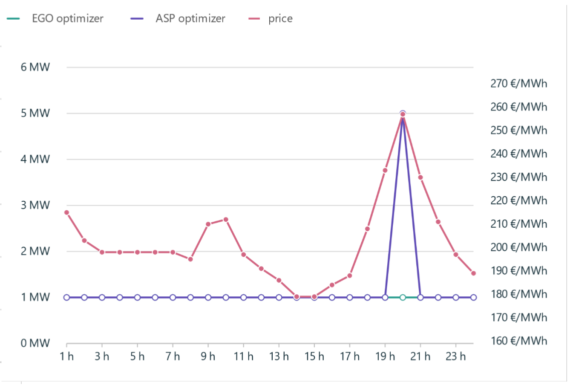


(b) Energy Production

Figure 23: Schedule comparison (18/03/22) - first 24h.

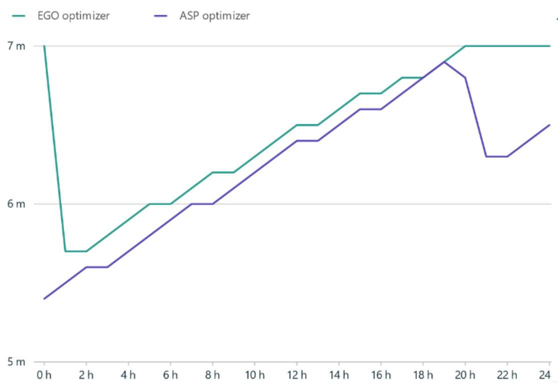


(a) Water Level

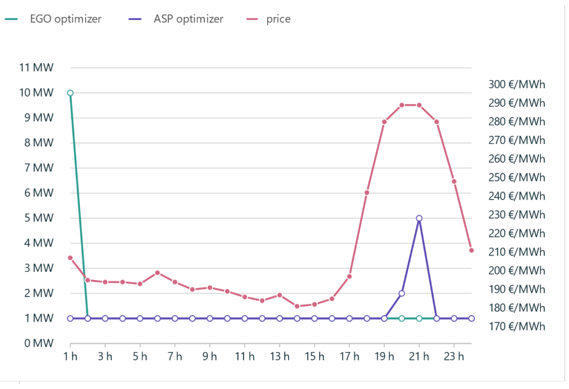


(b) Energy Production

Figure 24: Schedule comparison (19/03/22) - first 24h.



(a) Water Level



(b) Energy Production

Figure 25: Schedule comparison (20/03/22) - first 24h.

6 Web Application

Among the objectives of this thesis was to build a web application, which allows us to express the potential of our solution and provide EGO with a new system for managing and scheduling the production of hydropower plants.

The web application was designed and developed following some cutting-edge technologies available to EGO, in particular we tried to develop the backend of the application and its features with the tools provided by Amazon Web Services (AWS).

The Interface was built from scratch using together React, Typescript and the Ant-Design library.

In our case the web app is useful for managing the entire life-cycle of the process of hydropower optimization. This is a tool designed to test the new solution proposed in this thesis and see the results, also in comparison to the existing optimizer of EGO.

6.1 Hydroelectric Optimizer service

In order to make the data, encoding, solver and, more generally, the optimization procedure developed in this thesis accessible to a web application, we had to develop a complex backend structure.

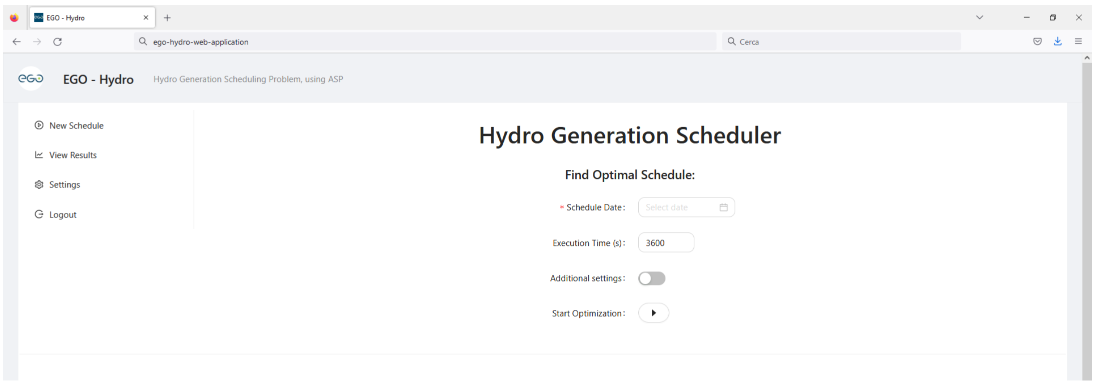
The optimizer we implemented can access data from EGO databases, process it, and then encode it in ASP format.

The scheduling procedure can be performed with Clingo completely autonomously and independently within the distributed containers, described in section 5.1. In this way, we can search for multiple schedules at the same time, without affecting the computational power we have available.

The service described here is also available as a stand-alone, in the form of a Web API. In this way, the hydro scheduling search procedure can be performed by automated services.

6.2 Description of the web application

In this chapter, we will have a look at the user interface of the web application we have developed, with the intention of showing the different visual and instrumental features that have been introduced (see figure 36).


EGO - Hydro

ego-hydro-web-application

Cerca

EGO - Hydro Hydro Generation Scheduling Problem, using ASP

New Schedule

View Results

Settings

Logout

Hydro Generation Scheduler

Find Optimal Schedule:

* Schedule Date: Select date

Execution Time (s): 3600

Additional settings:

Start Optimization:

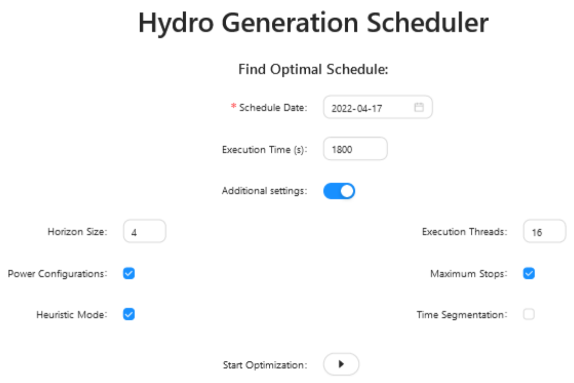
Figure 26: First look at the web application main page.

Find a new optimal production schedule:

By logging into the web application, at the top of the page, it is possible with a form to set up our proposed solution, to solve the hydro generation scheduling problem for a certain day of the calendar (see figure 26).

Here, we can select the scheduling date with an appropriate form, in this case, a popup calendar will appear and we can easily choose the day we want to schedule. The date must be valid and the data needed to schedule that day must be available.

The first field, containing the scheduling objective is the most important, however, we gave the possibility to the user to set also the execution time for the schedule and some additional settings.


Hydro Generation Scheduler

Find Optimal Schedule:

* Schedule Date: 2022-04-17

Execution Time (s): 1800

Additional settings: on

Horizon Size:	4	Execution Threads:	16
Power Configurations:	✓	Maximum Stops:	✓
Heuristic Mode:	✓	Time Segmentation:	

Start Optimization:

Figure 27: Web application form.

As we can see in figure 27, if we click the toggle button, some additional settings will appear and we can set them for the schedule we want to run.

At the top, we find two selectors, to choose the number of days we want to consider within our scheduling horizon and the number of parallel threads (see section 5.1) we want to leverage within the Amazon ECS container, for the solving phase.

In this example of a web application, to show how we can interact with the ASP encoding of the procedure, we also introduced the possibility of choosing to consider or ignore within the scheduling, two operational rules that we have already discussed and added to the encoding in the section 4 (15 and 9), the one of the power configuration and the one of the maximum number of shutdowns. In this way, the user can choose whether they want to have these two plant conditions valid in the solution that will be returned. It could be possible in the future, if needed, to give the possibility to choose the parameters of these rules as well, for example, what configurations should be used or how many maximum stops the plant production can undergo.

Finally, at the bottom, we have the possibility to choose to use one or both of the methodologies introduced in this thesis, to improve the scheduling results.

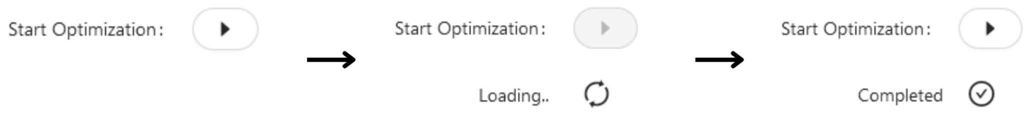
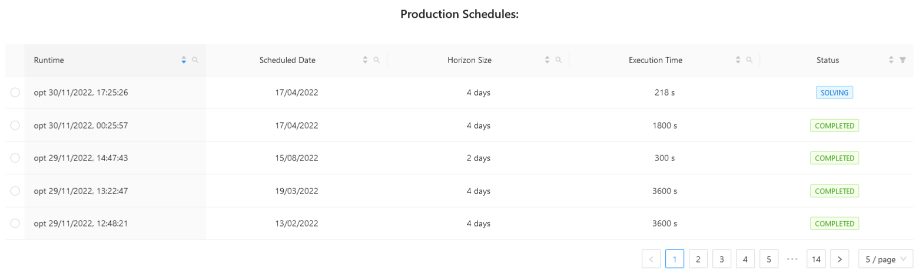


Figure 28: Running the scheduler.

It is sufficient to press the button of figure 28, to start the procedure for the search for the optimal schedule. At this point, the fields will be validated and we will see first a loading bar, and then a check mark when the job is effectively running inside the distributed service described in section 6.1.

See results from previous run:

In the center of the web page, we can see a table containing all the schedules launched or completed with our optimizer service. In this way we can, both observe the results of schedules started by this same app and also have a way to keep an eye on scheduling services initialized by automated systems (e.g., a hypothetical timed service that launches the optimizer once a day).

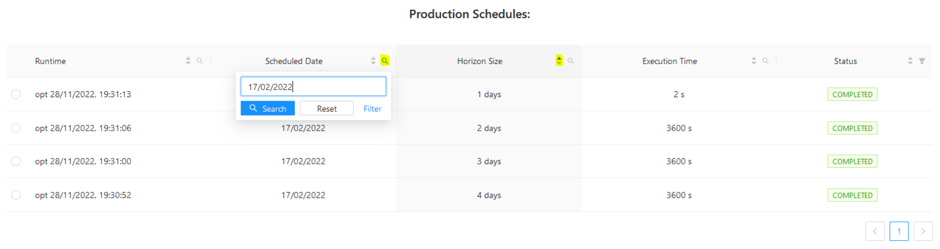


Production Schedules:

Runtime	Scheduled Date	Horizon Size	Execution Time	Status
opt 30/11/2022, 17:25:26	17/04/2022	4 days	218 s	SOLVING
opt 30/11/2022, 00:25:57	17/04/2022	4 days	1800 s	COMPLETED
opt 29/11/2022, 14:47:43	15/08/2022	2 days	300 s	COMPLETED
opt 29/11/2022, 13:22:47	19/03/2022	4 days	3600 s	COMPLETED
opt 29/11/2022, 12:48:21	13/02/2022	4 days	3600 s	COMPLETED

Figure 29: Table for the view of schedules.

In figure 29, we can see how the table shows some relevant information such as the time the process was executed, the scheduled date, the horizon of the schedule, the execution time, and a colored box showing us what the execution status is (optimization completed or solving).
We can also sort, filter or even search the items in the table. For example, in figure 30, we search for the schedule of the day 17/02/2022 and we sort the horizon size column in ascending order.

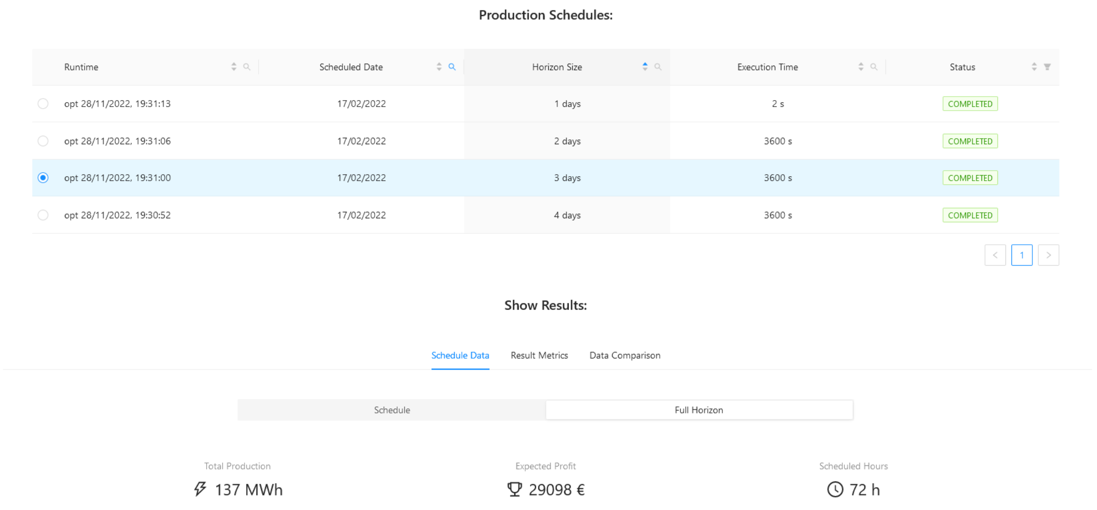


Production Schedules:

Runtime	Scheduled Date	Horizon Size	Execution Time	Status
opt 28/11/2022, 19:31:13		1 days	2 s	COMPLETED
opt 28/11/2022, 19:31:06	17/02/2022	2 days	3600 s	COMPLETED
opt 28/11/2022, 19:31:00	17/02/2022	3 days	3600 s	COMPLETED
opt 28/11/2022, 19:30:52	17/02/2022	4 days	3600 s	COMPLETED

Figure 30: Web application table filters.

If a scheduling process is completed, we can select its row in the table to be able to analyze the results obtained (see figure 31).



Production Schedules:

Runtime	Scheduled Date	Horizon Size	Execution Time	Status
opt 28/11/2022, 19:31:13	17/02/2022	1 days	2 s	COMPLETED
opt 28/11/2022, 19:31:06	17/02/2022	2 days	3600 s	COMPLETED
● opt 28/11/2022, 19:31:00	17/02/2022	3 days	3600 s	COMPLETED
opt 28/11/2022, 19:30:52	17/02/2022	4 days	3600 s	COMPLETED

Show Results:

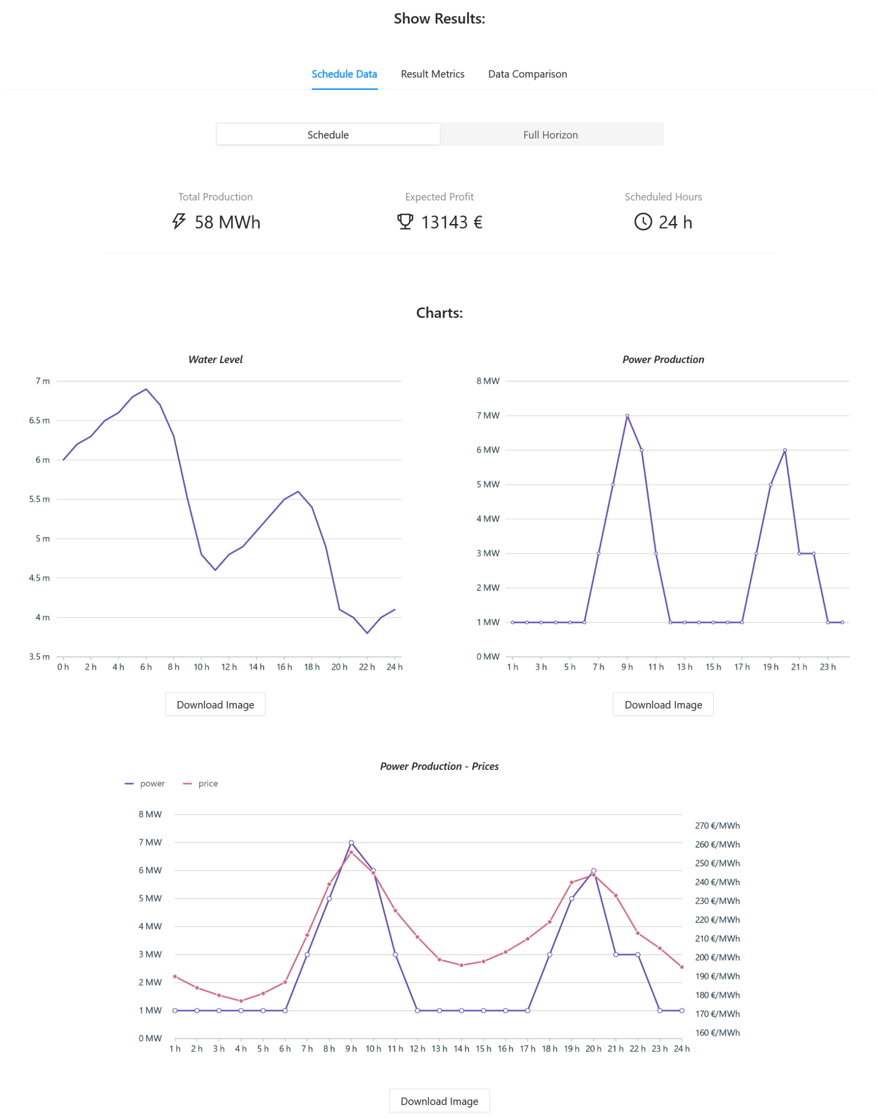
Total Production: 137 MWh
Expected Profit: 29098 €
Scheduled Hours: 72 h

Figure 31: Web application table selection and tabs.

The results are arranged below the table and are divided into three main categories:

- Schedule Data
- Result Metrics
- Data Comparison

Schedule Data:


Show Results:

Total Production: 58 MWh

Expected Profit: 13143 €

Scheduled Hours: 24 h

Charts:

Figure 32: First tab of the web application, Schedule

In this tab we can see the data related to the performed schedule: we have the possibility to see the total energy produced and the expected profit from the 24h of the production (see figure 32).

Here, we can also see some charts similar to those already analyzed in this thesis, at the top we have 2 charts for the water level of the reservoir and for power production, while at the bottom we have a more complex graph showing the values of power production and prices together. Each chart can be hovered to show tooltips, with data values for each hour of the day. Moreover, it is also possible to download each chart image with the dedicated buttons.

It is possible, to click on the button on the top of the tab and select the view on the full horizon, in this case, the types of charts of the data showed will be the same but the values will change according to the full size of the schedule horizon (see figure 33).

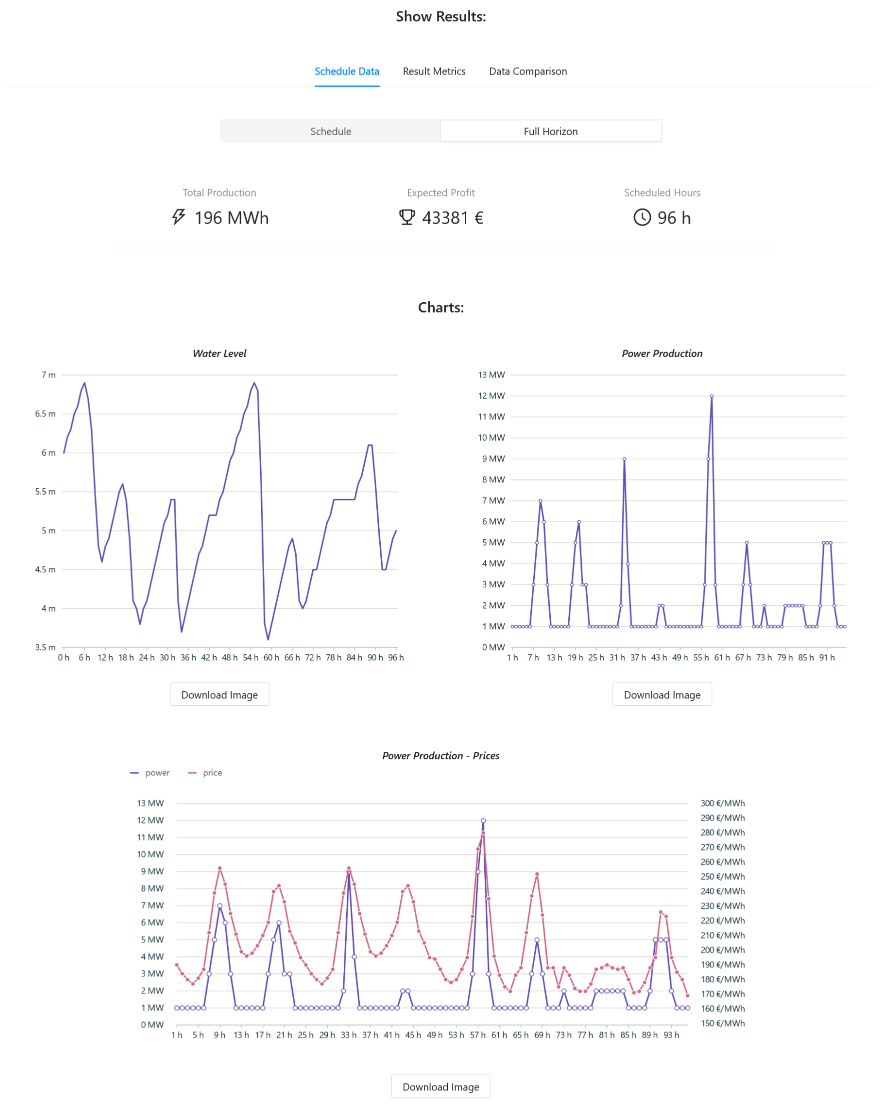


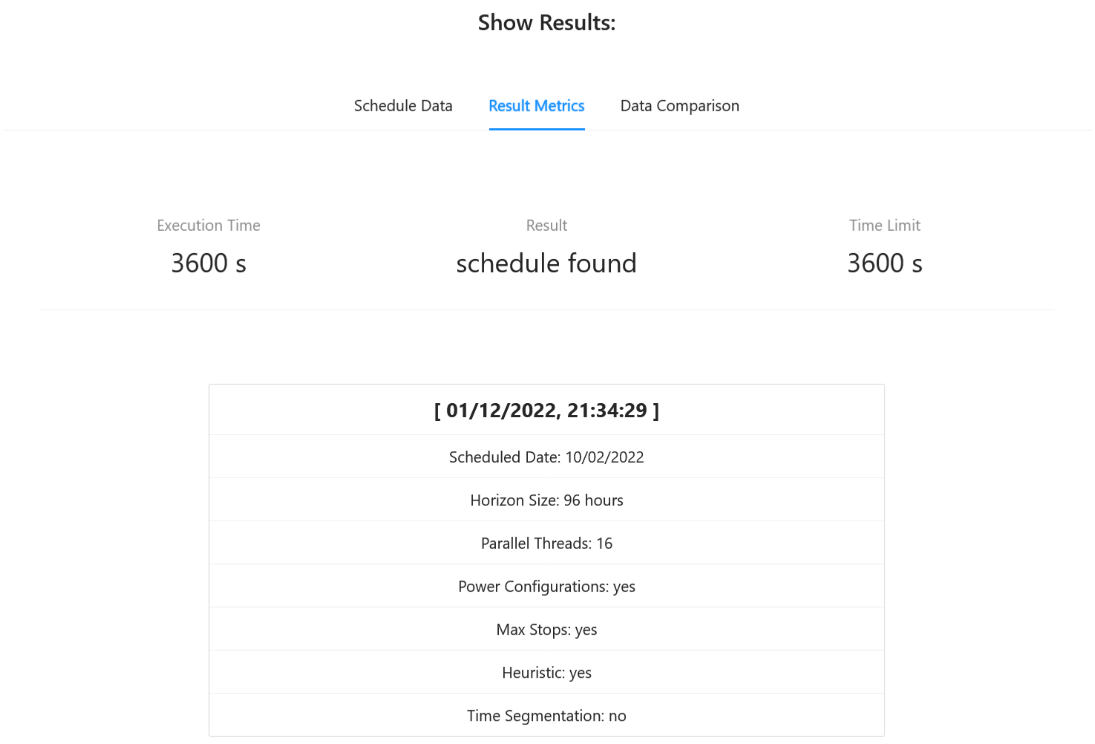
Figure 33: First tab of the web application, Full Horizon.

Results Metrics:

As you can see from the image 34, this tab is focusing on the properties of the executions.

For example, at the top, we can see the execution time and the time limit assigned to it. In the middle, we can see whether the execution went well and whether a result was found (schedule found or optimal schedule found).

Further down we find the process details listed, with the date and time of when the solution was found, the settings selected for the startup, and some other information already in the table.



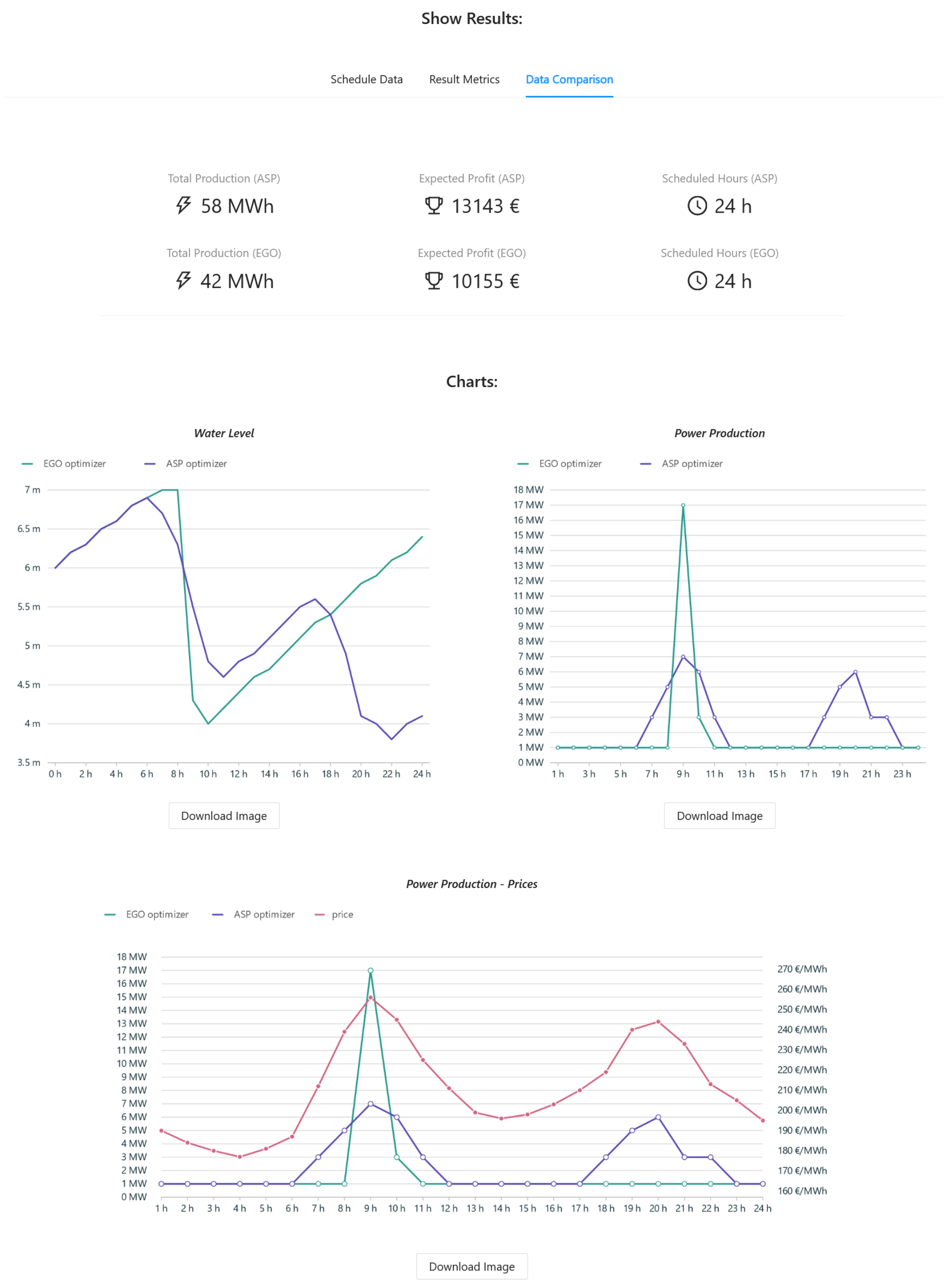
Execution Time	Result	Time Limit
3600 s	schedule found	3600 s

[01/12/2022, 21:34:29]
Scheduled Date: 10/02/2022
Horizon Size: 96 hours
Parallel Threads: 16
Power Configurations: yes
Max Stops: yes
Heuristic: yes
Time Segmentation: no

Figure 34: Second tab of the web application.

Data Comparison:

The last tab (figure 35) introduces a comparison with the scheduling performed by the optimizer used by EGO. Here we can check the differences of the two procedures via charts and data results.



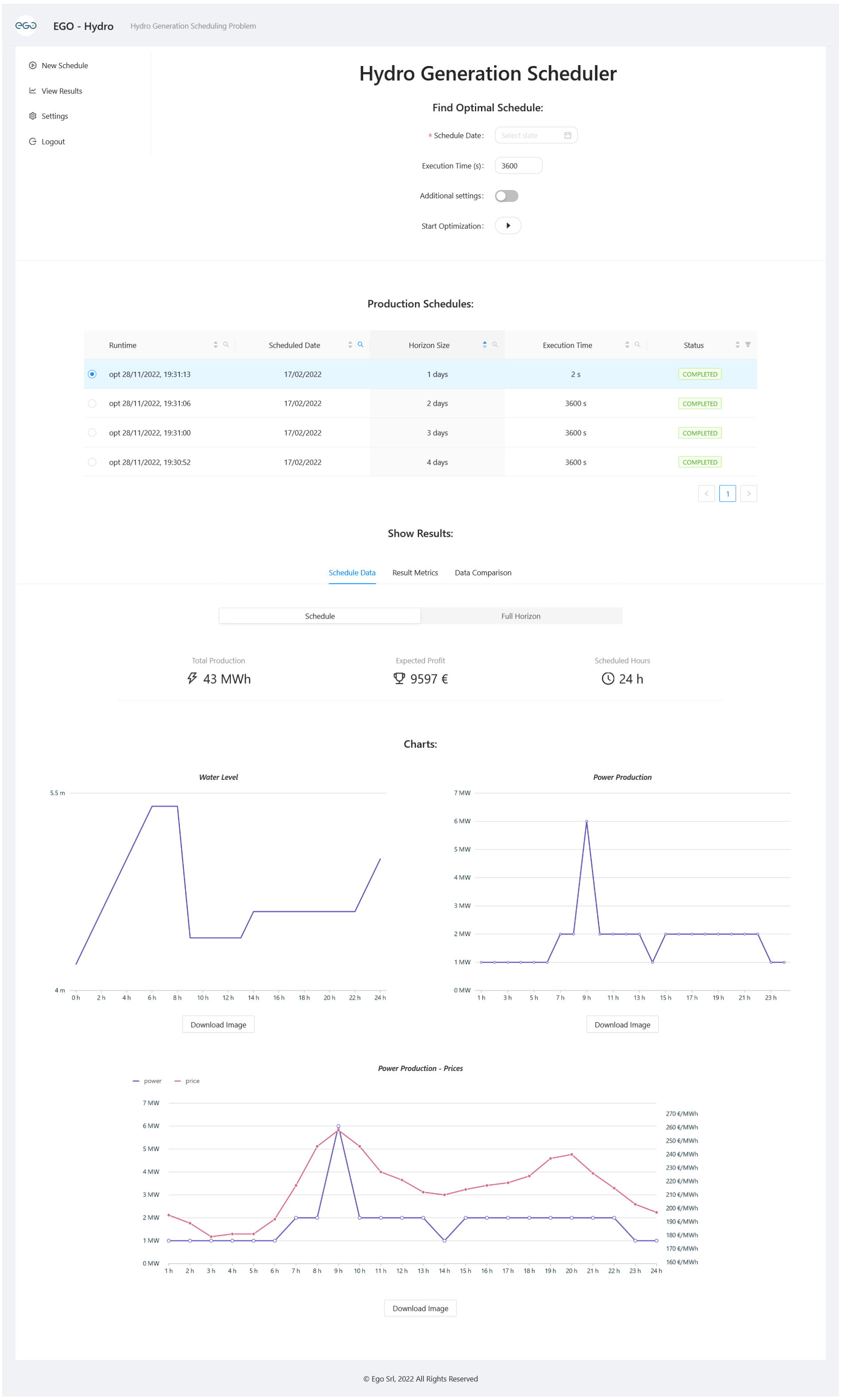
Show Results:

Schedule Data | Result Metrics | Data Comparison

Total Production (ASP)	Expected Profit (ASP)	Scheduled Hours (ASP)
58 MWh	13143 €	24 h
Total Production (EGO)	**Expected Profit (EGO)**	**Scheduled Hours (EGO)**
42 MWh	10155 €	24 h

Charts:

Figure 35: Third tab of the web application.



Hydro Generation Scheduler

Find Optimal Schedule:

* Schedule Date: Select date

Execution Time (s): 3600

Additional settings:

Start Optimization:

Production Schedules:

	Runtime	Scheduled Date	Horizon Size	Execution Time	Status
●	opt 28/11/2022, 19:31:13	17/02/2022	1 days	2 s	COMPLETED
○	opt 28/11/2022, 19:31:06	17/02/2022	2 days	3600 s	COMPLETED
○	opt 28/11/2022, 19:31:00	17/02/2022	3 days	3600 s	COMPLETED
○	opt 28/11/2022, 19:30:52	17/02/2022	4 days	3600 s	COMPLETED

Show Results:

Total Production: 43 MWh

Expected Profit: 9597 €

Scheduled Hours: 24 h

Charts:

Figure 36: Web application interface.

7 Related Work

7.1 State of the art

Different mathematical programming methods, coming from Operational Research and Optical Control Techniques studies, have been developed for the computation of optimal operating strategies in water systems. Many of these methods produce satisfactory results, at least from a theoretical point of view.

According to the paper [39] here are listed the most commonly used optimization techniques regarding this field:

- Linear Programming (LP)
- Non-linear Programming (NLP)
- Dynamic Programming (DP)
- Meta-heuristics and Genetic Algorithms
- Simulations

In related works, we can find also the usage of techniques like Artificial Neural Networks [37], Real-time optimization algorithms [16] and heuristics [1].

In the following sections, we will propose an overview of the characteristics of each technique, in terms of applicability and advantages, and we will take inspiration from the reviews [39, 53, 58, 56, 48, 42].

7.2 Linear Programming

Linear programming is certainly the simplest technique for optimization and has been tested in multiple areas such as transportation, telecommunications, manufacturing and, as in our case, energy.

As expressed by its name, this rule applies in problems where specifications can be expressed in terms of linear objective functions and linear constraints, for this reason, it is often required to apply linearization to the complex rules of hydroelectric systems.

This method can provide a global optimal result for the problem.

We can find some tentative of solving hydro generation scheduling problems and water resources optimizations in different papers, such as [45] (1962) with the resolution of a water resource problem; [33] (1982) about the design of HYDROSIM, the simulation of the Tennessee Valley Authority reservoir system and [51] (1988) for the respect of operational rules in the field of a drought-management system. The article described in [52] (2012), in which was defined a linear programming application with the use of penalty functions and the Nelder-Mead algorithm, is instead one of the most interesting because, similarly of what we have done, it provides an instance of computation using real-world data with approximations and analysis on the performances and result obtained.

Many other tests and results can be seen in the reviews of Husain [39], Labadie [42] and Nobile et. al. [48]; moreover, we can find some works related to other different physical and operational aspects of hydropower, such as with the inclusion of ecological and mechanical policies.

7.3 Non-linear Programming

When a problem is too complex and nonlinear relationships come into play, such as those related to plant production, linearization cannot be used effectively and instead the solution must be sought using the methodology called non-linear programming.
We can find some applications in this area in the review [39, 53]; however, in this case, non-linear programming is not very popular because of the computational complexity and, also, for related difficult theoretical terms.

7.4 Dynamic Programming

Dynamic programming concepts are derived from the definitions of the "principle of optimality" of Bellman [10, 9] (1957). The procedure consists of the decomposition of the original problem into sub-problems, then solved by a recursive procedure. This structure is often considered both a mathematical optimization and a computer programming method.

The pro of using this technique in the field of hydro generation scheduling problems is that we have a problem definition that behaves similarly to the procedure, because it can be expressed as a sequential structure in which each decision depends on the past choices that we made.

Dynamic programming is the most used technique, we can find some applications and studies in the works of Hall and Buras [35] (1961), Young [59] (1967), Opricovic and Djordjevic [50] (1976),Collins [15] (1977) and Allen and Bridgeman [2] (1986). Additional insights can be gathered in the reviews of Yakowitz [57] (1982) and Yeh [58] (1985).

This optimization technique has many advantages; it is suitable in the same way for linear or non-linear problems, had many successful results in various fields and is ideal for the resolution of multi-stage based and sequential-decision problems.
In the same way, however, we can find some high issues and disadvantages, we have in particular a method that needs to store a huge amount of data for each stage that we want to analyze and, in problems of large sizes, we can crush in the "curse of dimensionality" of Bellman [10, 9] (1957).
Dynamic programming offers also a very specific procedure, that is different for each problem, has to be solved in its own way and makes it hard to update with new operational rules.

7.5 Meta-Heuristic Algorithms

As stated by Husain in [39], the result that is still missing in the search for optimizations techniques in hydropower systems is to have a generic technique for optimizing complex systems; in fact, many algorithms or techniques need to focus on a specific instance of the problem and are not easily adaptable to every type of problem. To find a solution, in the last years, were made some trials on the applicability of the so-called meta-heuristic techniques.

Meta-heuristic techniques are high-level procedures that try to explore the search space of a problem model by using heuristics rules or approximation. These techniques can manage to find solutions for problems of huge size because they try to cut some valuable possibilities and find a solution in a fast way.

The disadvantage of using these techniques is of having a procedure that focuses on finding good solutions but non-optimal (not the best possible solutions), because they are, as in the definition, based on an approximated non-deterministic structure.
As for the advantages, we have the fact that they make it possible to find the solution for hard and large-sized problems, too large to be fully explored; that they usually require less time; and manage to solve the problem of dynamic programming, with the definition of a methodology that is not problem-specific.

When we are talking about meta-heuristics, we are referring to different categories of techniques, such as:

- Evolutionary algorithms.
- Physics-based algorithms.
- Swarm-Based algorithms.
- Population-Based algorithms.
- Nature Inspired algorithms.

In the following subsection, we are going to see one of the most used meta-heuristic techniques regarding hydropower optimizations and the hydro-generation scheduling problem.

7.5.1 Genetic Algorithms:

Genetic algorithms (GA) are the most known among the meta-heuristics techniques defined as population-based; these approaches try to define sets of solutions and then improve or cut off some of them, based on some events or phenomena.
The concept of this procedure, first introduced by Holland [38] (1975), is to follow the concept of evolution provided by Darwin [17] (1859) and to replicate it in the search for the optimal solutions of a problem.

Like in the concept expressed by Darwin, each GA is considering abstract representations of the solutions as species that need to pass different stages of stochastic selections, such as combinations, mutations and selections; when a solution is effectively kept until the end, we can have a sub-optimal solution for the problem.

We can have some insights on GA technique in the works of Goldberg [34] (1989), Michalewicz [47] (1992), Davis [19] (1991) and Dasgupta and Michalewicz [18].

7.6 Simulations

The last type of technique, referred to in related works, consists in build a model and performing a simulation of the real-world functioning of a hydroelectric power plant system
As stated by Husain in [39], our aim in this experiment is to approximate some behavior of real-world phenomenon with the use of computer techniques and a mathematical description of the problem and "aim to identify the optimal decision for system operation" (e.g. the production at each interval of time) "capable to maximize certain given objectives (e.g. the maximization of the profit from the energy sales)".

We had in the years different trials on these procedures, for example, the ones described in the papers and reviews of Fleming [27] (1975), Karamouz and Houck [41] (1982), Yeh [58] (1985), Labadie et al. [43] (1987) and Wurbs [56] (1993).

In this case, the technique offers a specific model for the problem to be solved plus the disadvantage of needing some level of approximations and being so unable to necessarily obtain the best possible (optimal) solution.

7.7 Differences among the Techniques:

In Table 2 we can see a summary of the most commonly used techniques within Hydroelectric optimization problems, with their advantages and disadvantages already described within the above papers.

Method	Benefits	Drawbacks
Linear Programming	simple, good computational efficiency, provide the optimal global solution	it requires linear or linearizable objective function and constraints.
Non-linear Programming	capable of handling complex nonlinear problems, describes accurately the characteristics of hydro power production.	computational complexity, difficult theoretical formulation.
Dynamic Programming	capable of treating non-convex, non-linear and discontinuous objective and constraints functions.	curse of dimensionality, huge requirements in terms of computer memory and execution time.
Meta-heuristic algorithms	generic technique, good solution in an affordable computing time.	may not necessarily lead to the best possible solution.
Simulations	provide a good description of the problem's variables.	global optimal solution may not be necessarily archived.

Table 2: Techniques for solving Hydro Optimizations Problems.

Problem formulations

Our problem is a specific case of hydro generation scheduling problem with a single reservoir and a single generating unit consisting of four turbines; for this reason, unlike some cases mentioned above in linear and non-linear programming, we have a problem that is less concerned with the physical structure of the power plant.

In our problem, the objective was to maximize the gain derived from production rather than to maximize the output of the plant itself; this definition of the objective traces back to the fact that the recipient of the problem is an energy trading firm, rather than purely the plant operator.

Many of the problems described above are also devoted to finding long-term optimizations and schedules, this does not fit well with our problem where it is necessary to have available fresh data of the price forecast and the channel flow rate at the reservoir inlet.

Differences with the ASP methodology

ASP provides a solution expressed in logical language, with a model that is easily understandable and provide a clear view on the description of the specific of the problem; while, with other methodologies we need to encode the problem into more complex mathematical definitions or programming code. Moreover, the problem formulation is also compact because we can use some constructs such as choice rules and aggregate functions.

Through the use of weak constraints, it is embedded the possibility to express objective functions with multiple objectives and multiple levels, which makes it easy to optimize different problem features with different weights.

Another advantage to using ASP, from a performance standpoint, is the presence of very efficient solvers such as Clingo.

7.8 Solving scheduling problems with ASP.

ASP has been successfully used for solving hard combinatorial and application scheduling problems in several research areas; however, it is new for applications in the field of Energy and production management.

In the Healthcare domain, we have many successful applications with the use of ASP methodology:
In 2017, the Nurse Scheduling Problem, described in [4, 24, 46], was the first problem, with the aim to create scheduling for nurses working in hospital units.
In 1918, the problem of assigning operating rooms to patients, denoted as Operating Room Scheduling [21, 22], further extended to include bed management [23].
More recent problems include the Chemotherapy Treatment Scheduling problem [20], in which patients are assigned a chair or a bed for their treatments, and the Rehabilitation Scheduling Problem [14], which assigns patients to operators in rehabilitation sessions.

Concerning the Job assignment problem, we have an example of the usage of ASP in the following documents:
[54], in which the available personnel were allocated in the seaport for serving the incoming ships.
[7], in which efforts were made to optimize the allocation of jobs to different devices.
[5], talking about the problem of assigning reviewers in the Program Committee to submit conference papers.

Other types of problems are cited in the papers by Gebser et. al. [31] and Falkner et. al. [26].

8 Conclusions

8.1 Thesis Conclusions

In this thesis, we analyzed the hydro generation scheduling problem, which deals with the automatic allocation of the production of hydroelectric power plants, based on economic price assessments.

The work we performed, began with an initial phase of gathering the requirements and specifications of the problem we were going to attempt to solve. For this phase we consulted EGO, an Italian company that deals with the management, dispatching and trading of electricity from renewable sources. Under the company's guidance, we wrote a problem description that could define the operation of the hydroelectric power plant and the specifications of the objective of the problem.

Various mathematical programming and meta-heuristic algorithm methodologies have been used in recent years to solve the hydro generation scheduling problem. The goal of this thesis was to find and test an artificial intelligence-based methodology that could provide new advantages, compared to the existing solutions. In particular, our solution had to provide a solving style suitable for several different instances of the problem and with the ability to easily accept any future specifications that may be introduced into the power plant system.

We made a solution based on the Answer Set Programming (ASP) paradigm, in which problem definitions can be declared in logical form and then given to a solver, in our case Clingo, to obtain the solution to the problem.

Initially, we aimed to introduce an encoding that could best describe the defined problem; in doing this step, we took advantage of the organization provided by the Guess&Check&Optimize encoding methodology, commonly followed in related work, while using ASP.

We then evaluated the results obtained from the execution, checking their consistency with respect to the definition in words of the problem, the performance of the result in terms of time and value, and the robustness of the solution in dealing with different types of instances of the problem.

At the end of the analytical process, we developed a web application, which is able to exploit the potential of the solution with a robust serverless backend system and provide users with a new tool for the analysis and management of production workflow in hydropower plants.

At the end of the project we are able to draw some conclusions about the success of the objective:

The methodology exposed in this thesis has allowed us to find a new solution for the hydro generation scheduling problem, able to find good results in time and being elastic with respect to the inclusion of new rules and constraints of the power plant.

Some computational difficulties, that we found in terms of time, were addressed in this project by introducing heuristics that can simplify the production allocation problem and a "divide and conquer" methodology to help solve schedules with time horizons as long as more than a few days.

Comparison, with the results obtained from the software currently employed by EGO, gave good results; the web application developed can be used to further verify and compare the strength of our solution.

8.2 Future Works

In this thesis, the Answer Set Programming methodology was applied for the first time in the Energy domain. We wanted to introduce the application of this scheduling technique, in a simplified hydro generation scheduling problem, and do so, in particular, using real data for the validation.

In this area, there are, of course, still many aspects left open that we have not been able to discuss. We will try in this section to expose particular attempts and future developments that can be carried out to improve the efficiency and utilization of deductive artificial intelligence techniques in the next years.

Improvements for the methodology

Regarding ASP, we could try to use a different solver to obtain the solution, in particular, we could use WASP, which has already been tested in articles such as [3] (2019) and which could allow us to obtain a better solution in less time.

Modelling of physical aspects of the power plant

In this thesis, the hydropower plant has been analyzed as an atomic system that can consume water as input and present in the reservoir, to produce an output of electric power.

Many papers of related work try to focus on a more physical modeling, with a better definition of each turbine and each penstock channel, to introduce constraints and specifications related to the presence and optimization of multiple reservoirs and/or multiple production units. Obviously in this case the problem can become more complex in terms of constraints and variables to be considered but, it can also provide for the solution of optimizations for some more specific factors of the power plant, such as the efficiency of power generation and up to the increase of the ecological value of the power plant.

In the future, the ASP solution should be tested for these problems as well. For example, to look specifically at each individual turbine, we can introduce predicates that contain information about each of them, such as the production capacity and the amount of water required; at this point, the production rules can refer to and adapt to the different characteristics involved. Or again, we can introduce constraints that are valid differently for each reservoir within the system.

Additional problem specifications

As we have shown, we have also been able to add 2 additional plant operational constraints in our solution: power configurations and the maximum number of shutdowns. In this sense, a possible future development is to consider some additional rules of the plant and test the effect these cause on the execution time and performance of the scheduler.

For example, storage-based hydroelectric power plants are those that have been gaining more popularity in recent times, capable of pumping outgoing water, already used by generation, back to the reservoir, to be reused during dry times, when there would otherwise be no water.
In this type of power plant, the model is similar to that of a battery, in which we can best decide when to discharge the energy into the national power lines. It will certainly be interesting, in the coming years to go and try to introduce the additional rules brought by the different structures of these power plants. For example, to decide what is the appropriate time to re-fill the reservoir and how much water in particular should be transported.

References

[1] Ahmed Kheiri. “Multi-stage Hyper-heuristics for Optimisation Problems”. In: *Thesis submitted to the University of Nottingham* (2014).

[2] Allen R. B. and Bridgeman S. G. “Dynamic Programming in Hydropower Scheduling”. In: *Journal of Water Resources Planning and Management* 112.3 (1986).

[3] Alviano Mario, Amendola Giovanni, Dodaro Carmine, Leone Nicola, Maratea Marco, and Ricca Francesco. “Evaluation of Disjunctive Programs in WASP”. In: *LPNMR*. Ed. by Marcello Balduccini, Yuliya Lierler, and Stefan Woltran. Vol. 11481. LNCS. Springer, 2019, pp. 241–255.

[4] Alviano Mario, Dodaro Carmine, and Maratea Marco. “An Advanced Answer Set Programming Encoding for Nurse Scheduling”. In: *AI*IA*. Vol. 10640. LNCS. Springer, 2017, pp. 468–482. DOI: `https://doi.org/10.1007/978-3-319-70169-1_35`.

[5] Amendola Giovanni, Dodaro Carmine, Leone Nicola, and Ricca Francesco. “On the Application of Answer Set Programming to the Conference Paper Assignment Problem”. In: *AI*IA*. Vol. 10037. Lecture Notes in Computer Science. Springer, 2016, pp. 164–178.

[6] Azad Abdus Samad, Rahaman Shokor A., Watada Junzo, Vasant Pandian, and Vintaned Jose Antonio Gamez. “Optimization of the hydropower energy generation using Meta-Heuristic approaches: A review”. In: *Energy Reports* 6 (2020), pp. 2230–2248. DOI: `https://doi.org/10.1016/j.egyr.2020.08.009`.

[7] Balduccini Marcello. “Industrial-Size Scheduling with ASP+CP”. In: *Logic Programming and Nonmonotonic Reasoning - 11th International Conference, LPNMR 2011, Vancouver, Canada, May 16-19, 2011. Proceedings*. Ed. by Delgrande James P. and Faber Wolfgang. Vol. 6645. Lecture Notes in Computer Science. Springer, 2011, pp. 284–296. DOI: `https://doi.org/10.1007/978-3-642-20895-9\_33`.

[8] Beg A.H. and Islam M.Z. “Advantages and limitations of genetic algorithms for clustering records”. In: *Conference on Industrial Electronics and Applications* (2016).

[9] Bellman Richard. “On the Theory of Dynamic Programming”. In: *PNAS* 38.8 (1952). DOI: `https://doi.org/10.1073/pnas.38.8.716`.

[10] Bellman Richard and Kalaba Robert. “Dynamic Programming”. In: *PNAS* 43.8 (1957). DOI: `https://doi.org/10.1073/pnas.43.8.749`.

[11] Buccafurri F., Leone N., and Rullo P. “Enhancing Disjunctive Datalog by Constraints”. In: *IEEE Transactions on Knowledge and Data Engineering* 12.5 (2000), pp. 845–860.
DOI: `https://doi.org/10.1109/69.877512`.

[12] Calimeri Francesco, Faber Wolfgang, Gebser Martin, Ianni Giovambattista, Kaminski Roland, Krennwallner Thomas, Leone Nicola, Maratea Marco, Ricca Francesco, and Schaub Torsten. “ASP-Core-2 Input Language Format”. In: *Theory and Practice of Logic Programming* 20.2 (2020), pp. 294–309.

[13] Calimeri Francesco, Gebser Martin, Maratea Marco, and Ricca Francesco. “Design and results of the Fifth Answer Set Programming Competition”. In: *Artificial Intelligence* 231 (2016), pp. 151–181.

[14] Cardellini Matteo, De Nardi Paolo, Dodaro Carmine, Galata Giuseppe, Giardini Anna, Maratea Marco, and Porro Ivan. “A Two-Phase ASP Encoding for Solving Rehabilitation Scheduling”. In: *Proceedings of the 5th International Joint Conference on Rules and Reasoning (RuleML+RR 2021).* Ed. by Sotiris Moschoyiannis, Rafael Peñaloza, Jan Vanthienen, Ahmet Soylu, and Dumitru Roman. Vol. 12851. Lecture Notes in Computer Science. Springer, 2021, pp. 111–125.

[15] Collins M. A. “Implementation of an Optimization Model for Operation of a metropolitan Reservoir System”. In: *JAWRA Journal of the American Water Resources Association* 13.1 (1977), pp. 7–70.
DOI: `https://doi.org/10.1111/j.1752-1688.1977.tb01990.x`.

[16] Cordova M.M., Finardi E.C., Ribas F.A.C., Matos V.L. de, and Scuzziato M.R. “Prediction of small hydropower plant power production in Himreen Lake dam (HLD) using artificial neural network”. In: *Electric Power Systems Research* 116 (2014), pp. 201–207.
DOI: `https://doi.org/10.1016/j.epsr.2014.06.012`.

[17] Darwin Charles. *On the origin of species by means of natural selection, or, The preservation of favoured races in the struggle for life.* London : John Murray, 1859.

[18] Dasgupta Dipankar and Michalewicz Zbigniew. “Evolutionary Algorithms — An Overview”. In: *Evolutionary Algorithms in Engineering Applications* (1997), pp. 3–28.
DOI: `https://doi.org/10.1007/978-3-662-03423-1_1`.

[19] Davis L. *Handbook of Genetic Algorithms.* Van Nostrand, Reinhold, 1991.

[20] Dodaro Carmine, Galata Giuseppe, Grioni Andrea, Maratea Marco, Mochi Marco, and Porro Ivan. “An ASP-based Solution to the Chemotherapy Treatment Scheduling problem”. In: *Theory and Practice of Logic Programming* 21.6 (2021), pp. 835–851.

[21] Dodaro Carmine, Galata Giuseppe, Maratea Marco, and Ivan Porro. "Operating Room Scheduling via Answer Set Programming". In: *AI*IA*. Vol. 11298. LNCS. Springer, 2018, pp. 445–459.

[22] Dodaro Carmine, Galata Giuseppe, Maratea Marco, and Porro Ivan. "An ASP-based framework for operating room scheduling". In: *Intelligenza Artificiale* 13.1 (2019), pp. 63–77.

[23] Dodaro Carmine, Galata Giuseppe, Muhammad Kamran Khan, Maratea Marco, and Porro Ivan. "An ASP-based Solution for Operating Room Scheduling with Beds Management". In: *Proceedings of the Third International Joint Conference on Rules and Reasoning (RuleML+RR 2019)*. Ed. by Paul Fodor, Marco Montali, Diego Calvanese, and Dumitru Roman. Vol. 11784. Lecture Notes in Computer Science. Springer, 2019, pp. 67–81.

[24] Dodaro Carmine and Maratea Marco. "Nurse Scheduling via Answer Set Programming". In: *LPNMR*. Vol. 10377. LNCS. Springer, 2017, pp. 301–307.

[25] Faber W., Pfeifer G., and Leone N. "Semantics and complexity of recursive aggregates in answer set programming". In: *Artificial Intelligence* 175.1 (2011), pp. 278–298. DOI: `https://doi.org/10.1016/j.artint.2010.04.002`.

[26] Falkner Andreas A., Friedrich Gerhard, Schekotihin Konstantin, Taupe Richard, and Teppan Erich Christian. "Industrial Applications of Answer Set Programming". In: *Künstliche Intelligenz* 32.2-3 (2018), pp. 165–176.

[27] Fleming George. *Computer simulation techniques in hydrology.* Elsevier, 1975.

[28] Ge Xiaolin, Xia Shu, and Lee Wei Jen. "An efficient stochastic algorithm for mid-term scheduling of cascaded hydro systems". In: *Journal of Modern Power Systems and Clean Energy* 7 (2019), pp. 163–173. DOI: `https://doi.org/10.1007/s40565-018-0412-6`.

[29] Gebser Martin, Maratea Marco, and Ricca Francesco. "The Design of the Seventh Answer Set Programming Competition". In: *LPNMR*. Ed. by Marcello Balduccini and Tomi Janhunen. Vol. 10377. Lecture Notes in Computer Science. Springer, 2017, pp. 3–9.

[30] Gebser Martin, Maratea Marco, and Ricca Francesco. "The Sixth Answer Set Programming Competition". In: *Journal of Artificial Intelligence Research* 60 (2017), pp. 41–95.

[31] Gebser Martin, Obermeier Philipp, Schaub Torsten, Ratsch-Heitmann Michel, and Runge Mario. "Routing Driverless Transport Vehicles in Car Assembly with Answer Set Programming". In: *Theory and Practice of Logic Programming* 18.3-4 (2018), pp. 520–534. DOI: `https://doi.org/10.1017/S1471068418000182`.

[32] Gestore dei Mercati Energetici S.p.A. *Guida al Mercato Elettrico*. 2003.

[33] Gilbert KG and Shane RM. "TVA Hydro Scheduling Model: Theoretical Aspects". In: *Journal of the Water Resources Planning and Management Division* 108.1 (1982), pp. 21–36. DOI: `https://doi.org/10.1061/JWRDDC.0000245`.

[34] Goldberg David E. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Longman Publishing Co, 1989.

[35] Hall Warren A. and Buras Nathan. "The dynamic programming approach to water-resources development". In: *Journal of Geophysical Research (1896-1977)* 66.2 (1961), pp. 517–520. DOI: `https://doi.org/10.1029/JZ066i002p00517`.

[36] Hammid Ali Thaeer, Awad Omar I., Sulaiman Mohd Herwan, Gunasekaran Saraswathy Shamini, Mostafa Salama A., Kumar Nallapaneni Manoj, Khalaf Bashar Ahmad, Al-Jawhar Yasir Amer, and Abdulkareem Abdulhasan Raed. "Review of Optimization Algorithms in Solving Hydro Generation Scheduling Problems". In: *Energies* 13.11 (2020). DOI: `https://doi.org/10.3390/en13112787`.

[37] Hammid Ali Thaeer, Bin Sulaiman Mohd Herwan, and Abdallac Ahmed N. "Prediction of small hydropower plant power production in Himreen Lake dam (HLD) using artificial neural network". In: *Alexandria Engineering Journal* 557.1 (2018), pp. 211–221. DOI: `https://doi.org/10.1016/j.aej.2016.12.011`.

[38] Holland J.H. "Adaptation in Natural and Artificial Systems". In: *MIT Press, Cambridge, Mass* (1975).

[39] Husain Azhar. "An Overview of Reservoir Systems Operation Techniques". In: *International Journal of Engineering Research and Development* 4.10 (2012), pp. 30–37.

[40] hydropower.org. *2021 Hydropower Status Report*. hydropower.org. 2022. URL: `https://www.hydropower.org/publications/2021-hydropower-status-report`.

[41] Karamouz Mohammad and Houck Mark H. "Annual and monthly reservoir operating rules generated by deterministic optimization". In: *Water Resources Research* 18.5 (1982), pp. 1337–1344. DOI: `https://doi.org/10.1029/WR018i005p01337`.

[42] Labadie John W. "Optimal Operation of Multireservoir Systems: State-of-the-Art Review". In: *Journal of Water Resources Planning and Management* (2004). DOI: `https://doi.org/10.1061/(ASCE)0733-9496(2004)130:2(93)`.

[43] Labadie John W., Fontane Darrell G., Members ASCE, Q. Tabios Guillermo, and Fang Chou Nine. “Stochastic Analysis of Dependable Hydropower Capacity”. In: *Journal of Water Resources Planning and Management* 113.3 (1987), pp. 422–437.

[44] Leone N., Pfeifer G., Faber W., Eiter T., Gottlob G., Perri S., and Scarcello F. “The DLV system for knowledge representation and reasoning”. In: *ACM Trans. Comput. Log.* 7.3 (2006), pp. 499–562.

[45] Maass A., Hufschmidt M., Dorfman R., Thomas H., Marglin S., and Fair G. *Design of water resource systems*. Harvard University Press, 1962.

[46] Mario Alviano, Carmine Dodaro, and Marco Maratea. “Nurse (Re)scheduling via answer set programming”. In: *Intelligenza Artificiale* 12.2 (2018), pp. 109–124.

[47] Michalewicz Zbigniew. *Genetic Algorithms + Data Structures = Evolution Programs*. Springer, 1992.

[48] Nobile Luca, Tarabini Marco, Franzò Simone, and Marzaroli Pietro. “Linear and dynamic optimization of multi-reservoirs Hidroelectric Systems”. In: *Master of science in mechanical engineering and management engineering, Master thesis, Politecnico di Milano* (2019).

[49] Nunez Christina. *Hydropower explained*. nationalgeographic.com. 2019. URL: `https://www.nationalgeographic.com/environment/article/hydropower`.

[50] Opricović Serafim and Djordjević Branislav. “Optimal long-term control of a multipurpose reservoir with indirect users”. In: *Water Resources Research* 12.6 (1976), pp. 1286–1290. DOI: `https://doi.org/10.1029/WR012i006p01286`.

[51] Palmer Richard N. and Holmes K. John. “Operational Guidance During Droughts: Expert System Approach”. In: *Journal of Water Resources Planning and Management Division* 114.6 (1988), pp. 647–666. DOI: `https://doi.org/10.1061/(ASCE)0733-9496(1988)114:6(647)`.

[52] Ribeiro A.F., Guedes M.C.M., Smirnov G.V., and Vilela S. “On the optimal control of a cascade of hydro-electric power stations”. In: *Electric Power Systems Research* 88 (2012), pp. 121–129. DOI: `https://doi.org/10.1016/j.epsr.2012.02.010`.

[53] Ribeiro A.F., Guedes M.C.M., Smirnov G.V., and Vilela S. “Reservoir Management and Operations Models: A State-of-the-Art Review”. In: *Water Resources Research* 21.12 (1985), pp. 1797–1818. DOI: `https://doi.org/10.1029/WR021i012p01797`.

[54] Ricca Francesco, Grasso Giovanni, Alviano Mario, Manna Marco, Lio Vincenzino, Iiritano Salvatore, and Leone Nicola. “Team-building with answer set programming in the Gioia-Tauro seaport”. In: *Theory and Practice of Logic Programming* 12.3 (2012), pp. 361–381. DOI: `https://doi.org/10.1017/S147106841100007X`.

[55] Tengberg Oskar. “Implementation of Hydro Power Plant Optimization for Operation and Production Planning”. In: *Mechanical Engineering, master’s level* (2019).

[56] Wurbs R.A. “Reservoir System Simulation and Optimization Models”. In: *Journal of Water Resources Planning and Management* (1993). DOI: `https://doi.org/10.1061/(ASCE)0733-9496(1993)119:4(455)`.

[57] Yakowitz Sidney. “Dynamic Programming applications in water resources”. In: *Water Resources Research* 18.4 (1982), pp. 673–696. DOI: `https://doi.org/10.1029/WR018i004p00673`.

[58] Yeh William W-G. “Reservoir Management and Operations Models: A State-of-the-Art Review”. In: *Water Resources Research* 21.12 (1985), pp. 1797–1818. DOI: `https://doi.org/10.1029/WR021i012p01797`.

[59] Young Jr. George K. “Finding reservoir operating rules”. In: *Journal of the Hydraulics Division* 93.6 (1967).

Ringraziamenti

(Acknowledgement)

Voglio ringraziare, in questo spazio, tutte le persone che mi hanno supportato nello svolgimento di questo progetto e in questi cinque anni di cammino universitario.

Ringrazio il Professor Marco Maratea per avermi dato la possibilità di svolgere questo progetto di tesi; grazie per tutti i consigli che ho ricevuto e per aver creduto sempre nelle mie capacità.

Ringrazio il Dott. Davide Mini per tutto l'impegno impiegato nel suo ruolo di correlatore, per essere stato sempre disponibile e per tutte le spiegazioni sul mondo dei dati e dell'energia.

Ringrazio EGO per avere reso possibile questa tesi e per la possibilità che mi ha dato di apprendere diversi aspetti dell'ambiente lavorativo. In particolare, voglio ringraziare Matteo Fattore, Fabio Garagiola e il gruppo EGO Data perchè mi hanno insegnato tanto e hanno espresso, fin da subito, fiducia nei miei confronti, facendomi sentire a tutti gli effetti parte del loro gruppo.

Il ringraziamento più grande lo voglio dedicare alla mia famiglia.
Più di tutti voglio ringraziare i miei genitori, che mi hanno sempre sostenuto e dato la possibilità di realizzare i miei sogni. Grazie per tutte le attenzioni e per il vostro affetto, vi voglio bene.
Ringrazio con affetto i miei nonni che ogni giorno pensano a me, per tutti gli importanti consigli ricevuti e poi tutti gli zii e i cugini che mi sono sempre stati vicini.

Voglio ringraziare tutti i miei amici, in particolare Jacopo, Michele, Ivan, Simone, Marco B., Davide e Marco M. per tutti i momenti divertenti che abbiamo passato assieme. Non vedo l'ora di festeggiare con voi questo grande traguardo.

Ringrazio Lorenzo, amico e compagno di studi fin dalle scuole superiori, che oggi si laurea con me. Voglio ringraziarlo per tutto l'aiuto e la compagnia che mi ha offerto nei tanti progetti e nelle esperienze che abbiamo condiviso.

Infine, ringrazio tutti i compagni di università e i professori che, in questi ultimi anni, mi hanno aiutato e hanno reso più prezioso questo percorso di laurea.

Riccardo