UNIVERSIDAD
NEBRIJA

Transmission Expansion Planning by Quantum Annealing

Author:
Sergio López Baños

Supervisors:
Dr. Álvaro Díaz Fernández
Dr. Oriol Raventós Morera

A thesis submitted in fulfilment of the requirements for the degree of MSc in Quantum Computing at Nebrija University.

The thesis was carried out at the

Institute of Networked Energy Systems
DLR - Deutsches Zentrum für Luft- und Raumfahrt

DLR **Deutsches Zentrum für Luft- und Raumfahrt**
German Aerospace Center

Declaration of Authorship

I, Sergio López Baños, hereby declare that this thesis and the work presented in it is entirely my own. Where I have consulted the work of others, this is always clearly stated.

Signed: Sergio López Baños.

Date: June 19, 2023.

NEBRIJA UNIVERSITY

Abstract

Research Center
DLR - Deutsches Zentrum für Luft- und Raumfahrt

MSc in Quantum Computing

Transmission Expansion Planning by Quantum Annealing

by Sergio López Baños

The *transmission expansion planning* problem (TEP) can be formulated as a *mixed-integer linear programming* (MILP) problem that aims at finding the optimal way to expand the capacity of an energy system. The solution provides the optimal layout of transmission lines that are to be built in order to satisfy the energy demand on a distributed energy system with a high share of renewable energy sources. The TEP scales badly using classical algorithms and, at the same time, energy system models are getting larger and more complex due to the integration of decentralized weather-dependent renewable energy sources, sector coupling and the increase of storage components. Currently, the problem is often linearized or the scope and granularity of the model are reduced using clustering algorithms. For this reason, any computational time reduction will have substantial implications in closing the granularity gap between what the current models can solve and the desired resolution needed by energy system operators. Quantum annealers are single-purpose quantum computers specialized in solving combinatorial optimization problems. Since quantum computers are still not sufficiently mature, large problems cannot be tackled purely with a quantum computer. We propose a decomposition protocol for the TEP problem which is similar in spirit to Benders' decomposition algorithm, which allows us to use a hybrid quantum-classical approach to tackle bigger problems by providing the binary master problem to a quantum annealer and a set of slave sub-problems to classical solvers. Therefore, our method can take advantage of cutting-edge classical algorithms and current quantum annealers. The ultimate goal is to find solutions that are closer to the optimum while achieving a speed-up.

Acknowledgements

I have no words to express my gratitude to my supervisor Álvaro Díaz Fernández not only for his lecture notes, corrections and feedback on the thesis but also for his academic advice. His expertise and attention to the details have played a crucial role in the success of my thesis.

Also, I am grateful for the working environment and the support that the German Aerospace Center (DLR) has offered me for the preparation of this thesis at the Institute of Networked Energy Systems. In particular, I am deeply thankful to Oriol Raventós Morera, for introducing me to the research world and for his insightful comments and suggestions.

Last, I would like to thank my family. For their support in every decision I have made and for their unconditional love.

Contents

List of Figures

List of Tables

List of Abbreviations

AQC	**A**diabatic **Q**uantum **C**omputing
BD	**B**ender **D**ecomposition
CPU	**C**entral **P**rocessing **U**nit
DDP	**D**ual **D**ynamic **P**rogramming
HQC	**H**ybrid **Q**uantum-**C**lassical
HPC	**H**igh **P**erformance **C**omputing
IP	**I**nteger **P**rogramming
KKT	**K**arush–**K**uhn–**T**ucker
LP	**L**inear **P**rogramming
MP	**M**aster **P**roblem
NISQ	**N**oisy **I**ntermediate-**S**cale **Q**uantum
SA	**S**imulated **A**nnealing
SP	**S**ub-**P**roblem
QA	**Q**uantum **A**nnealing
QPU	**Q**uantum **P**rocessing **U**nit
QUBO	**Q**uadratic **U**nconstrained **B**inary **O**ptimization
TEP	**T**ransmission **E**xpansion **P**lanning
TSP	**T**ravelling **S**alesman **P**roblem

Dedico esta Tesis de Máster a mis padres, cuyo esfuerzo ha permitido que estudiase en la universidad y cuyo ejemplo me ha guiado, inspirado y motivado en el camino.

CHAPTER 1

Introduction

Quantum computing is a new paradigm of computation whose importance is growing not only because of its promising speed-up over classical computers in some problems, such as combinatorial optimization problems, but also due to the fact that we are building transistors in a scale where quantum effects are starting to become relevant. Combinatorial optimization problems are common among industry albeit more research needs to be done to get insight about how quantum computing could help to solve industry challenges. There is already a large industry research in finance [1], route planning [2] and life sciences, among others. The research on applications of quantum computing to the energy sector has just began in the last years. There are only a few active research projects, like Q-Grid by e-On [3] or EnerQuant by Fraunhofer. The goal of this work to contribute to this field of research.

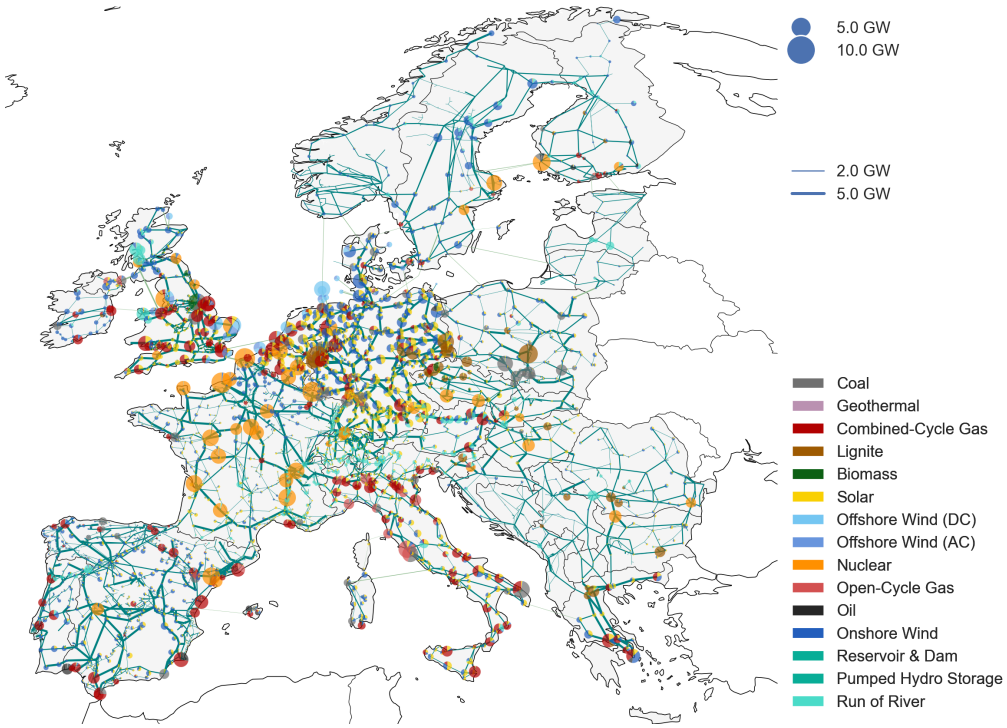


Figure 1.1: Clustered European transmission network model obtained from PyPSA-EUR [4].

Energy system models are getting larger and more complex due to the integration of decentralized weather-dependent renewable energy sources, intermittent loads, sector coupling and the increase of storage components. For instance, the renewable energy produced by the

integration of solar panels in dwellings is intended to be integrated to the grid if consumers are not using it. However, the grid infrastructure is not evolving accordingly to the new energy paradigm. As a consequence, energy from private sources cannot be added to the grid, which implies the efficiency of solar panels has to be decreased or the excess of energy has to be discarded. An accurate expansion planning would solve these problems by redirecting the excess of energy to where it is required or by storing it so that it can be used later, e.g., to charge an electric car. In this way, the energy would be efficiently distributed and energy companies could offer better prices to their consumers while safeguarding the environment by reducing their carbon footprints. Furthermore, an increment of the quantity of detailed data about energy consumption – smart meters – would extend the spatial and temporal resolution, i.e., the grid status of a region, so that better managing and expansion planning decisions can be made in order to satisfy the customer demand efficiently.

The problem we tackle in the present work is called *transmission expansion planning* (TEP) problem, which is a *mixed-integer linear programming* (MILP) problem with NP complexity, that aims at finding the optimal way to expand the capacity and connections of an energy system. Currently, the scope and granularity of the model are reduced using clustering algorithms. For this reason, any computational time reduction will have substantial implications in closing the granularity gap between what the current models can solve and the desired resolution needed by energy system operators.

Quantum computers are the candidates to solve NP-hard problems such as MILP, by making the complexity of the problem scale polynomically as the systems size grows. Concretely, quantum analog computers are special-purpose quantum computers specialised in solving combinatorial optimization problems. The size of quantum analog computers that is the number of qubits of the system is greater than the general purpose quantum computers such as the ones from IBM that are based in the successive applications of quantum gates, such as the recently released Osprey quantum processor with 433 qubits. Although quantum computers are improving every few years they are not mature enough for solving many real-world problems where the number of variables scales exponentially. Also, we have to take into account that quantum computers are hugely affected by the environmental conditions[1] – Noisy Intermediate-Scale Quantum (NISQ) era –, meaning that there will be errors during an algorithm execution, which implies that the execution time has to be short enough so that it does not exceed quantum decoherence. Because of the current maturity of quantum computers, hybrid quantum-classical approaches are required to tackle real-world problems [6]. The hybrid approach combines classical solvers and cutting-edge classical algorithms with quantum solvers that add the speed-up where it is possible.

A benchmark of how a TEP problem can be scaled up until current quantum solvers are not able to find a solution is provided among with a scheme to decompose a large TEP problem into a small enough master problem that can be addressed by a quantum computer and a subproblem addressed by a classical computer.

The present work is structured as follows. Chapter 2 guides through the foundations of adiabatic quantum computing. Chapter 3 describes Benders' decomposition techniques in the field of quantum computing. Finally, Chapter 4 solves a transmission expansion problem by starting with a small network solved by a pure quantum annealer and ending up with a bigger network that requires from hybrid solvers. Conclusions and outlook are drawn in Chapter 5.

[1] John Preskill called the current state of quantum technology as the Noisy Intermediate-Scale Quantum era, or NISQ for short [5].

CHAPTER 2

Adiabatic Quantum Computing

In the present chapter, we consider a paradigm of quantum computation known as *adiabatic quantum computing* (AQC). More details can be found in Ref. [7] and references therein. We start by sketching the rough idea of the adiabatic theorem to finally derive a formal proof of it. We also expose one application of AQC to solve *quadratic unconstrained binary optimization* (QUBO) problems, known as *quantum annealing* (QA) [8].

2.1 The Adiabatic Theorem

Quoting Sarandy and Lidar [9],

> *The theorem posits, roughly, that if a state is an instantaneous eigenstate of a sufficiently slowly varying Hamiltonian at one time, then it will remain an eigenstate at later times, while its eigenenergy evolves continuously.*

2.1.1 The Adiabatic Protocol

We start by constructing an initial Hamiltonian $\mathcal{H}(t=0) = \mathcal{H}_i$ whose ground state is known and whose time evolution, $t \in [0, T]$ – where T is the total evolution time – leads to a Hamiltonian $\mathcal{H}(t=T) = \mathcal{H}_f$ that encodes the solution to our problem. For simplicity, we can write a linear schedule between the initial and target Hamiltonian,

$$\mathcal{H}(t) = \left(1 - \frac{t}{T}\right)\mathcal{H}_i + \left(\frac{t}{T}\right)\mathcal{H}_f \,. \tag{2.1}$$

The adiabatic theorem, whose proof we give below, guarantees that if we start with an initial Hamiltonian $\mathcal{H}_i$ in a given eigenspace and the evolution is carried out sufficiently slowly – in further sections we demonstrate what slowly means in detail – then we end up in the equivalent eigenspace of the final Hamiltonian $\mathcal{H}_f$.

The adiabatic protocol can be summarized as follows:

- **Step 1 (Mapping):** map the problem into a Hamiltonian $\mathcal{H}_f$. Typically, the problem is encoded in the ground state.
- **Step 2 (Initialise $\mathcal{H}_i$):** initialise the system in the ground state of a Hamiltonian $\mathcal{H}_i$, easy to compute and to experimentally prepare. For instance, the ground state of $\mathcal{H}_i = -\sum_i^n \hat{\sigma}_i^x$ is the eigenvector $|+\rangle^{\otimes n}$.
- **Step 3 (Adiabatic Theorem):** slowly evolve the system from H_i to H_f. The adiabatic theorem guarantees that, under certain conditions that will be explained later, the system will end up in the ground state of H_f.

- **Step 4 (Measure):** measure the eigenstate of H_f. The result encodes a solution to our problem.

Notice that we wrote in the last step that the result of the measurement provides "a solution" not "the solution". This is because there are two possibilities for a finite-dimensional Hamiltonian:

- **Non-degenerate Hamiltonian:** we start with an initial Hamiltonian $\mathcal{H}_i$ in its ground eigenstate $|g(t=0)\rangle$ and end up in the equivalent eigenstate – ground state – of the final Hamiltonian $\mathcal{H}_f$ where the eigenvalue is $E_g(t=T)$ and the eigenvector is $|g(t=T)\rangle$.
- **Degenerate Hamiltonian:** we start with an initial Hamiltonian $\mathcal{H}_i$ in its ground eigenspace spanned by $\{|g^i(t=0)\rangle\}_{i\in[1,d]}$, where d is the degeneracy of the ground state, and end up in the equivalent eigenspace of the final Hamiltonian $\mathcal{H}_f$ with eigenvalue $E_g(t=T)$.

Intuitively, one can think that a problem can have multiple configurations that lead to the same minimum eigenvalue.

2.1.2 The Adiabatic Theorem: A First Approach

We now provide a simple argument to build intuition into the adiabatic theorem. The next section will be devoted to a formal proof.

In what follows, we consider an n-qubit system with a discrete and non-degenerate spectrum. A state of that system can be written as a linear combination of the instantaneous eigenstates of the Hamiltonian $|\psi(t)\rangle = \sum_i c_i(t)\,|i(t)\rangle$ as function of t. Its evolution is given by the time-dependent Schrödinger equation, see Appendix A,

$$i\hbar\,|\dot{\psi}(t)\rangle = \mathcal{H}(t)\,|\psi(t)\rangle\ . \tag{2.2}$$

In general, Eq. (2.2) represents a system of coupled differential equations for the evolution of the state $|\psi(s)\rangle$ which has a non-trivial solution. However, we can re-write the Hamiltonian in diagonal form using the diagonalization matrix $U(s)$,

$$\mathcal{H}_d(t) = U^{-1}(t)\mathcal{H}(t)U(t) = \begin{bmatrix} \lambda_0 & 0 & \dots & 0 \\ 0 & \ddots & & \vdots \\ \vdots & & \ddots & 0 \\ 0 & \dots & 0 & \lambda_{2^n-1} \end{bmatrix}\ . \tag{2.3}$$

We also define $|\psi_d(t)\rangle = U^{-1}(t)\,|\psi(t)\rangle$ as the state we get after applying the inverse transformation $U^{-1}(t)$ to the state $|\psi(t)\rangle$. Using the identity $\mathbb{I} = U(t)U^{-1}(t)$ and multiplying both sides of Eq. (2.2) by $U^{-1}(t)$ yields

$$i\hbar U^{-1}(t)\frac{\partial\left(U(t)U^{-1}(t)\,|\psi(t)\rangle\right)}{\partial t} = U^{-1}(t)\mathcal{H}(t)\left(U(t)U^{-1}(t)\right)|\psi(t)\rangle\ . \tag{2.4}$$

Rearranging terms

$$i\hbar U^{-1}(t)\frac{\partial U(t)}{\partial t}\,|\psi_d(t)\rangle + i\hbar\frac{\partial\,|\psi_d(t)\rangle}{\partial t} = \mathcal{H}_d(t)\,|\psi_d(t)\rangle\ . \tag{2.5}$$

If we assume $\mathcal{H}(t)$ varies slowly, then it is reasonable that $U(t)$ varies slowly as well, $\dot{U}(t) \simeq 0$, which implies

$$i\hbar\frac{\partial\,|\psi_d(t)\rangle}{\partial t} \simeq \mathcal{H}_d(t)\,|\psi_d(t)\rangle\ . \tag{2.6}$$

Now, the evolution of the state $|\psi_d(t)\rangle$ is led by a diagonal Hamiltonian $\mathcal{H}_d(t)$ so we have a set of uncoupled differential equations for each amplitude component of the state $|\psi_d(t)\rangle$. Furthermore, if the state of the system $|\psi_d(t)\rangle$ is an eigenstate of the Hamiltonian $|n(t)\rangle$, then the evolution of our system is conducted inside the eigenspace generated by $|n(t)\rangle$.

To sum up, under a general evolution such as Eq. (2.2) we get, in general, a system of coupled differential equations, but if the adiabatic approximation is satisfied, this evolution is led by a diagonal Hamiltonian, i.e., we get a system of uncoupled differential equations that enable us to evolve the ground state of an initial Hamiltonian into the ground state of a final Hamiltonian under the adiabatic conditions, see Figure 2.1.

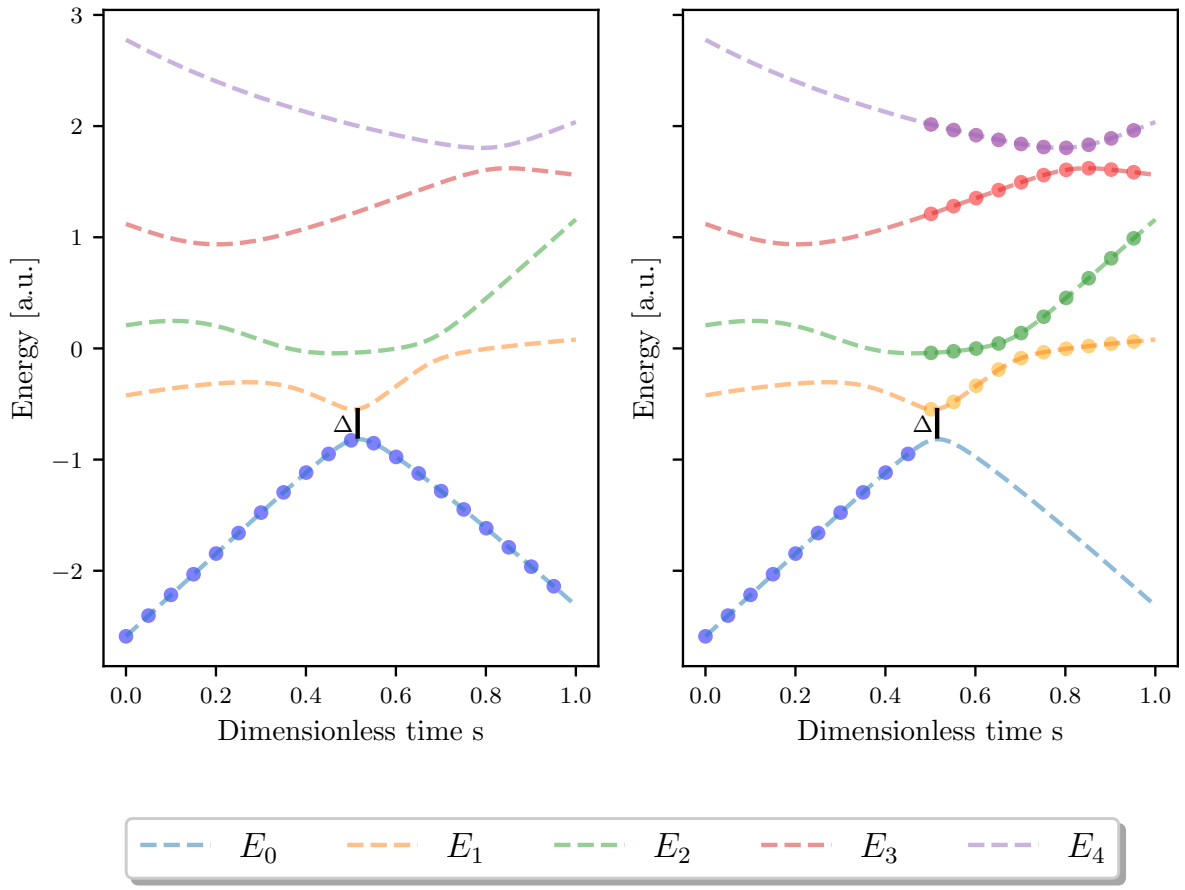


Figure 2.1: Eigenenergies of a random Hamiltonian as function of dimensionless time $s = t/T$. The dots indicate the state of the system at each point in time. **Left**: when adiabatic conditions are fulfilled, i.e., if the evolution is carried out sufficiently slowly, then our system evolves continuously in time along the ground state. **Right**: when the evolution is carried out not fulfilling the adiabatic conditions, it could happen that we end up in a different eigenstate if our system absorbs enough energy to jump to next level over the evolution, something that is more likely to happen close to the minimum gap Δ. If that happens we would end up in a different eigenstate of the target Hamiltonian which does not encode the optimal solution.

Quoting Goldstone *et al.* [7],

> *...if the gap between the two lowest levels, $E_1 - E_0$, is strictly greater than zero for all $0 \leq t \leq T$, then*
>
> $$\lim_{T \longrightarrow \infty} \| \langle g(t=T) | \psi(t=T) \rangle \|^2 = 1 \,. \tag{2.7}$$

Equivalently, the probability of being in the ground state, $|g(t=T)\rangle$, of the target Hamiltonian $\mathcal{H}_f$ at the end of evolution is 1 if and only if the total time required for the evolution is infinite. Otherwise, the probability of being in the ground state is

$$\| \langle g(t=T) | \psi(t=T) \rangle \|^2 = 1 - \epsilon^2 \,, \tag{2.8}$$

where $|\epsilon| \leq 1$ takes into account the error due to the adiabatic approximation, $\dot{U}(s) \simeq 0$.

2.1.3 The Adiabatic Theorem: A Formal Derivation

We start by writing down the Schrödinger equation [10]

$$i\hbar \frac{\partial \, |\psi(t)\rangle}{\partial t} = \mathcal{H}(t) \, |\psi(t)\rangle \,. \tag{2.9}$$

We assume that the instantaneous spectrum of $\mathcal{H}(t)$ is discrete and non-degenerate

$$\mathcal{H}(t) \, |n(t)\rangle = E_n(t) \, |n(t)\rangle \,, \tag{2.10}$$

where $|n(t)\rangle$ are the instantaneous eigenstates of the Hamiltonian and $E_n(t)$ are the eigenenergies labelled by n. Notice that we can label the eigenenergies with a single index because the spectrum is discrete and non-degenerate. For a discrete but degenerate spectrum, we would need an extra index to take into account the degeneracy of each state.

The eigenstates of the Hamiltonian form an orthonormal basis, so we can expand a given state in that basis

$$|\psi(t)\rangle = \sum_n c_n(t) e^{i\theta_n(t)} \, |n(t)\rangle \,, \tag{2.11}$$

where

$$\theta_n(t) = -\frac{1}{\hbar} \int_0^t E_n(t') dt' \,, \tag{2.12}$$

is the dynamic phase.
Substituting Eq. (2.11) into the Schrödinger equation yields

$$\sum_n \left[\dot{c}_n(t) \, |n(t)\rangle + c_n(t) \, |\dot{n}(t)\rangle \right] e^{i\theta_n(t)} = 0 \,. \tag{2.13}$$

Multiplying by $\langle m(t)|$ we get

$$\dot{c}_m(t) = -\sum_n c_n \, \langle m(t) | \dot{n}(t) \rangle \, e^{i(\theta_n(t) - \theta_m(t))} \,. \tag{2.14}$$

We need to re-write $\langle m(t)|\dot{n}(t)\rangle$ in terms of the Hamiltonian's derivative using Eq. (2.10). If we derive that expression with respect to time, we find

$$\frac{\partial \mathcal{H}(t)}{\partial t} \, |n(t)\rangle + \mathcal{H}(t) \frac{\partial \, |n(t)\rangle}{\partial t} = \frac{\partial E_n(t)}{\partial t} \, |n(t)\rangle + E_n(t) \frac{\partial \, |n(t)\rangle}{\partial t} \,. \tag{2.15}$$

Multiplying the last expression by $\langle m(t)|$, with $m \neq n$,

$$\langle m(t)|\frac{\partial \mathcal{H}(t)}{\partial t}|n(t)\rangle + E_m(t)\,\langle m(t)|\dot{n}(t)\rangle = E_n(t)\,\langle m(t)|\dot{n}(t)\rangle \ . \tag{2.16}$$

Finally,

$$\langle m(t)|\dot{n}(t)\rangle = \frac{1}{E_n(t) - E_m(t)}\,\langle m(t)|\frac{\partial \mathcal{H}(t)}{\partial t}|n(t)\rangle \ . \tag{2.17}$$

Substituting into Eq. (2.14) and defining $g_{nm}(t) \equiv E_n(t) - E_m(t)$ as the energy difference as a function of time t between the eigenstates $|m(t)\rangle$ and $|n(t)\rangle$ leads to

$$\dot{c}_m(t) = -c_m(t)\,\langle m(t)|\dot{m}(t)\rangle - \sum_{n\neq m} c_n(t)\frac{\langle m(t)|\dot{\mathcal{H}}(t)|n(t)\rangle}{g_{nm}(t)}e^{i(\theta_n(t)-\theta_m(t))} \ . \tag{2.18}$$

Adiabatic evolution is ensured if the coefficients $c_n(t)$ evolve independently from each other, i.e., if their dynamical equations do not couple. Mathematically,

$$\max_{0\leq t\leq T}\left|\frac{\langle m(t)|\dot{\mathcal{H}}(t)|n(t)\rangle}{g_{nm}(t)}\right| \ll \min_{0\leq t\leq T}|g_{nm}(t)| \ , \tag{2.19}$$

where T is the total evolution time.
Under the adiabatic approximation, the coupling term tends to zero, that is,

$$\sum_{n\neq m} c_n\frac{\langle m(t)|\dot{\mathcal{H}}(t)|n(t)\rangle}{g_{nm}(t)}e^{i(\theta_n(t)-\theta_m(t))} \to 0 \ . \tag{2.20}$$

Therefore, equation Eq. (2.18) turns into

$$\dot{c}_m(t) = -c_m(t)\,\langle m(t)|\dot{m}(t)\rangle \ , \tag{2.21}$$

whose solution is

$$c_m(t) = c_m(0)e^{i\gamma_m(t)} \ , \tag{2.22}$$

where

$$\gamma_m(t) = i\int_0^t \langle m(t')|\dot{m}(t')\rangle\,dt' \ , \qquad \gamma_m \in \mathbb{R} \ , \tag{2.23}$$

is the Berry phase.

2.1.4 Total Evolution Time

In previous sections, we stated that under a sufficiently slow Hamiltonian evolution the adiabatic theorem is satisfied. In this section, we define what "slow" means by deriving an expression to estimate the total time T required for the adiabatic evolution. The following derivation closely follows that of Ref. [9]. We also demonstrate that the total time not only depends on the energy gap between the ground state and the first excited state but also on the term $\langle m(s)|d\mathcal{H}(s)/ds|n(s)\rangle$, where s is a dimensionless time variable given by

$$s \equiv \frac{t}{T} \ , \quad s \in [0,1] \ . \tag{2.24}$$

We start by re-writing Eq. (2.18) in terms of the normalised time $s = t/T$ and Berry's phase

$$e^{i\gamma_m(sT)}\frac{1}{T}\frac{\partial}{\partial s}\left[c_m(sT)e^{-i\gamma_m(sT)}\right] = -\sum_{n\neq m} c_n(sT)\frac{\langle m(sT)|\dot{\mathcal{H}}(sT)|n(sT)\rangle}{g_{nm}(sT)}e^{-i(\theta_n(sT)-\theta_m(sT))} . \quad (2.25)$$

Notice we have added two Berry's phases terms with opposite sign so they cancel out. Integrating the last equation and re-arranging terms leads to

$$c_m(s)e^{-i\gamma_m(s)} = c_m(0) - \sum_{n\neq m}\int_0^s ds'\frac{F_{nm}(s')}{g_{nm}(s')}e^{-iT\int_0^{s'} ds''(g_{nm}(s''))} . \quad (2.26)$$

This equation expresses the coefficient $c_n(s)$ in terms of its initial value $c_n(0)$ and a summation term that keeps the rest of coefficients $c_m(s)$ with $m \neq n$, where

$$F_{nm}(s) = c_n(s)\,\langle m(s)|\dot{\mathcal{H}}(s)|n(s)\rangle\, e^{-i\gamma_m(s)} . \quad (2.27)$$

We can express the integrand of Eq. (2.26) as a difference between two terms,

$$\frac{F_{nm}(s')}{g_{nm}(s')}e^{-iT\int_0^{s'}(ds''g_{nm}(s''))} = \frac{i}{T}\left[\frac{d}{ds'}\left(\frac{F_{nm}(s')}{g_{nm}^2(s')}e^{-iT\int_0^{s'}(ds''g_{nm}(s''))}\right)\right. \quad (2.28)$$

$$\left. - e^{-iT\int_0^{s'}(ds''g_{nm}(s''))}\frac{d}{ds'}\left(\frac{F_{nm}(s')}{g_{nm}(s')}\right)\right] . \quad (2.29)$$

Substituting the previous result into Eq. (2.26) leads to

$$\begin{aligned} c_m(s)e^{-i\gamma_m(s)} = c_m(0) + \frac{i}{T}\left[\frac{F_{nm}(0)}{g_{nm}^2(0)} - \frac{F_{nm}(s)}{g_{nm}^2(s)}e^{-iT\int_0^s ds'g_{nm}(s'')}\right. \\ \left. + \int_0^s ds' e^{-iT\int_0^{s'}(ds''g_{nm}(s''))}\frac{d}{ds'}\left(\frac{F_{nm}(s')}{g_{nm}(s')}\right)\right] . \end{aligned} \quad (2.30)$$

Assuming the energy gap does not vanish when $T \to \infty$ and that $d\{F_{nm}(s')/g_{nm}^2(s')\}/ds'$ is integrable for all $s \in [0,1]$, the Riemann-Lebesgue lemma [11] guarantees that the last integral vanishes in the limit $T \to \infty$. So

$$c_m(s)e^{-i\gamma_m(s)} = c_m(0) + \frac{i}{T}\left[\frac{F_{nm}(0)}{g_{nm}^2(0)} - \frac{F_{nm}(s)}{g_{nm}^2(s)}e^{-iT\int_0^s ds'g_{nm}(s'')}\right] . \quad (2.31)$$

Under the adiabatic conditions there are no mixing terms, i.e., the coefficient $c_n(s)$ does not depend on the rest of coefficients. Mathematically,

$$c_m(m) = c_0(s)e^{i\gamma_m(s)} . \quad (2.32)$$

Therefore, we can simplify Eq. (2.27)

$$F_{nm}(s) = c_n(0)\,\langle m(s)|\dot{\mathcal{H}}|n(s)\rangle\, e^{-i[\gamma_m(s)-\gamma_n(s)]} . \quad (2.33)$$

With that simplification, we can estimate the total time for an adiabatic evolution by imposing a condition that minimises the second term of Eq. (2.31)

$$T \gg \frac{F}{g^2} , \quad (2.34)$$

where

$$F = \max_{0\leq s\leq 1} \left| c_n(0) \langle m(s)| \frac{d\mathcal{H}(s)}{ds} |n(s)\rangle \right|, \tag{2.35}$$

$$g = \min_{0\leq s\leq 1} |g_{nm}(s)| . \tag{2.36}$$

2.2 Quantum Annealing

Quantum annealing (QA) takes its name from a classical heuristic algorithm named s*imulated annealing* (SA), see Appendix B. It is a particularization of AQC where the Hamiltonian is the Ising's Hamiltonian

$$\mathcal{H}_f = -\sum_{ij} \mathcal{J}_{ij}\sigma_i^z\sigma_j^z - h\sum_i \sigma_i^z . \tag{2.37}$$

Spin information is given by the eigenvalues of Pauli Z-operators σ^z, where the possibles outcomes are $\{-1, 1\}$ corresponding to the eigenvectors $\{|\uparrow\rangle, |\downarrow\rangle\}$, respectively.

We also need an initial Hamiltonian whose ground state is easy to compute and prepare, e.g.,

$$\mathcal{H}_i = -\sum_i^n \sigma_i^x , \tag{2.38}$$

with ground state $|+\rangle^n$. This ground state considers all the possible configurations of our system with a uniform distribution.

The time-dependent Hamiltonian is given by

$$\mathcal{H}(s) = -\sum_{ij} \mathcal{J}_{ij}\sigma_i^z\sigma_j^z - h\sum_i \sigma_i^z - \Gamma(s)\sum_i \sigma_i^x , \tag{2.39}$$

where $\Gamma(s)\sum_i \sigma_i^x$ represents the quantum fluctuations – single-spin flip – and $\Gamma(s)$ plays the same role as temperature in SA. The way of controlling the quantum fluctuations depends on the hardware we are using. The hardware implementation and topology of the quantum computer we have used in the present work is discussed in Appendix C.

Initially, at $s \to 0$, the dominating term is $\Gamma(s)\sum_i \sigma_i^x$ because $\Gamma(s)$ takes a large value, but at later times $s \to 1$, $\Gamma(s)$ takes a value close to zero so the dominating Hamiltonian is $\mathcal{H}(s) = -\sum_{ij} \mathcal{J}_{ij}\sigma_i^z - h\sum_i \sigma_i^z$ which is the Hamiltonian we are interested in, as it contains the solution to our problem encoded in the ground state.

2.2.1 Formulation of Quadratic Unconstrained Binary Optimization Problems

Definition 2.1: ***Quadratic Unconstrained Binary Optimization*** **(QUBO)**

The quadratic unconstrained binary optimization problem can be written as

$$\min_{\vec{x}} \mathbf{x}^T Q\vec{x} , \tag{2.40}$$

where n is the total number of binary variables, $\mathbf{x} \in \{0, 1\}^n$ are the binary variables of the problem and Q is an $n \times n$ matrix whose entries encodes our problem.

Notice that the Ising Hamiltonian uses the binary variables $s_i = \{-1, 1\}$ but QUBO problems are formulated with the binary variables $x_i = \{0, 1\}$. Both formulations are equivalent if we consider the following mapping between Ising and QUBO variables

$$s_i = 2x_i - 1 \,. \tag{2.41}$$

When writing a given combinatorial problem into its QUBO formulation the constraints have to be mapped into its binary form. The following table summarises some of the most common binary constraints and how to map them into QUBO formulation.

Constraint	**QUBO penalty**
$x_1 = x_2$	$P\left(x_1 + x_2 - 2x_1x_2\right)$
$x_1 \leq x_2$	$P\left(x_1 - x_1x_2\right)$
$x_1 + x_2 = 1$	$P\left(1 - x_1 - x_2 + 2x_1x_2\right)$
$x_1 + x_2 \leq 1$	$P\left(x_1x_2\right)$
$x_1 + x_2 \geq 1$	$P\left(1 - x_1 - x_2 + x_1x_2\right)$

Table 2.1: Mapping of common binary constraints into QUBO formulation. The binary variables are represented by x_1 and x_2 and P is the penalty parameter, equivalently, it represents the Lagrange multiplier of the constraint.

There are constraints that require of additional binary variables – slack variables – in order to reformulate them into QUBO. These type of constraints can be written as a binary expansion as follows. Suppose we have the following constraint,

$$\sum_i w_i x_i \leq W \,, \tag{2.42}$$

where the W for simplicity is an integer value. The QUBO penalty of that constraint is

$$P\left(\sum_{k=0}^{M} c_k y_k - \sum_i w_i x_i\right)^2 \,, \tag{2.43}$$

where $M = \lfloor \log_2 W \rfloor$ is the floor binary logarithm of W, $c_k = 2^k$ is a binary expansion coefficient and $c_M = W + 1 - 2^M$ is the last term of that binary expansion.

Notice that slack variables have to be included into our problem, which implies an increase in the number of variables. Classically, the increment in the number of variables just affects the complexity of the problem but for a quantum annealer this increment could imply that the problem may not solvable with the current size of annealers. If that is the case, we have to simplify the model until the annealer is able to handle it, or we have to look for other techniques such as hybrid methods to tackle the problem, as we will describe in Chapter 3.

2.2.2 Example: Two Heirs Problem

As an example suppose we are given a 3-binary QUBO problem, see Table 2.2,where each variable $x_i \in \{0, 1\}$ represents an asset with an associated value v_i.

Index	Value
0	1
1	3
2	1

Table 2.2: Two heirs example given three assets.

We have to assign the assets to two heirs, Alice and Bob – represented by $\{0,1\}$ in the QUBO formulation and by $\{-1,1\} \equiv \{\uparrow,\downarrow\}$ in the Ising formulation, respectively – so that the difference between the value each heir receives is minimum, which represents our cost function. Assume that A is a subset of $S = \{s_0, s_1, s_2\}$ indicating which assets are given to Alice, $s = -1$. Analogously, $B = S - A$ indicates the assets received by Bob, $s = 1$. Then

$$f(s_0, s_1, s_2) = \left[\sum_{i\in A} s_i v_i + \sum_{i\in B} s_i v_i\right]^2 = \left[\sum_i s_i v_i\right]^2 . \tag{2.44}$$

Notice we are not interested in the sign of our cost function. Therefore, to avoid dealing with signs we take the square of the difference so we know we are minimising a positive quantity.

Expanding the square we arrive at the Ising formulation, $\vec{s}^T \mathcal{J} \vec{s}$. Notice that for $s_i \in \{-1,1\}$ the following relation holds $s_i^2 = 1$.

$$\left[\sum_i s_i v_i\right]^2 = \sum_i (s_i v_i)^2 + 2\sum_{i<j} s_i v_i v_j s_j = \underbrace{\sum_i v_i^2}_{\text{Constant}} + \sum_{i<j} s_i 2 v_i v_j s_j \, . \tag{2.45}$$

Therefore, the Ising matrix of this example is

$$\mathcal{J} = \begin{bmatrix} 0 & 2v_1v_2 & 2v_1v_3 \\ 0 & 0 & 2v_2v_3 \\ 0 & 0 & 0 \end{bmatrix} . \tag{2.46}$$

Notice that we have dropped the constant term as it only shifts the minimum value but it does not affect our solution, i.e., the minimum value of our cost function is going to change but the configuration that leads to that minimum remains the same.

Now, we initialise the evolution with the Hamiltonian $\mathcal{H}_i = -\sum_i^n \sigma_i^x$ whose ground state is $|+\rangle^n$. If the evolution is carried out under the adiabatic conditions,

$$\mathcal{H}(s) = -\sum_{ij} \mathcal{J}_{ij} \sigma_i^z \sigma_j^z - \Gamma(s) \sum_i \sigma_i^x \, , \tag{2.47}$$

then we end up in $\mathcal{H}_f$ with one of the following solutions

$$\mathcal{H}_f \, |g(s=1)\rangle = E_g(s=1) \, |{\downarrow\uparrow\downarrow}\rangle \, , \tag{2.48}$$

$$\mathcal{H}_f \, |g(s=1)\rangle = E_g(s=1) \, |{\uparrow\downarrow\uparrow}\rangle \, . \tag{2.49}$$

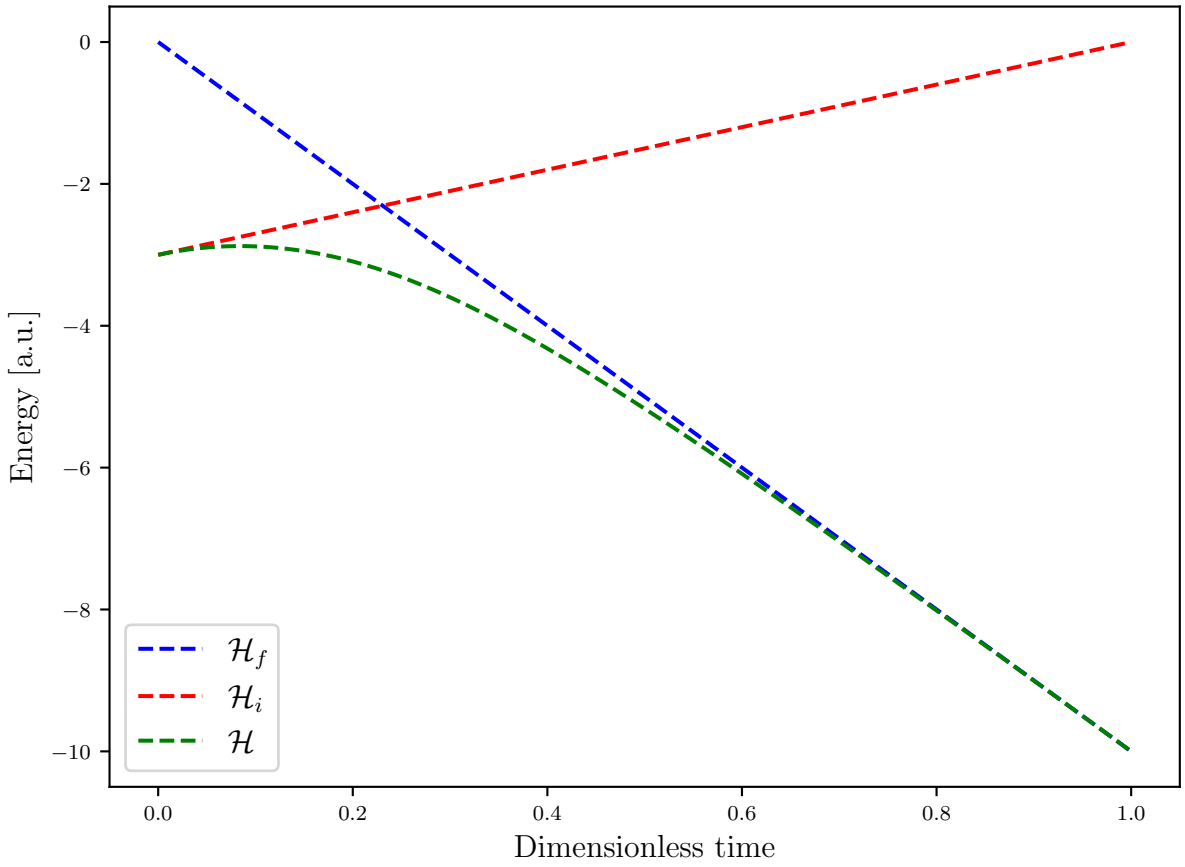


Figure 2.2: Eigenenergies of the ground state for the target, initial and total Hamiltonians. At $s = 0$ the Hamiltonian of the system is dominated by $\mathcal{H}_i$, as s increases the Hamiltonian is an interpolation of $\mathcal{H}_i$ and $\mathcal{H}_f$ and at $s = 1$ the dominant Hamiltonian is $\mathcal{H}_f$ which is the one that encodes the solution to our problem.

This means that we get a global minimum value $E_g(s = 1)$ associated with the ground states $\{|\downarrow\uparrow\downarrow\rangle, |\uparrow\downarrow\uparrow\rangle\}$. This is because in this case, the problem has a degenerate ground state and the eigenvectors correspond to different configurations of our binary variables that lead to the same absolute minimum value $E_g(s = 1)$. Intuitively, if a solution $|\uparrow\downarrow\uparrow\rangle$ indicates that Alice receives assets with indices $[0, 2]$ and Bob receive the asset with index $[1]$, the opposite is a solution $|\downarrow\uparrow\downarrow\rangle$ with the same cost, i.e., Bob receives assets with indices $[0, 2]$ and Alice receives the asset with index $[1]$.

2.3 Circuit Based AQC

In this section we sketch the idea of how to reproduce AQC in the gate model of computation with arbitrary precision. For a deeper understanding see Ref. [12].

According to Schrödinger equation, the evolution of a quantum system under a time-independent Hamiltonian is given by

$$|\psi(t)\rangle = e^{-i\frac{\mathcal{H}}{\hbar}t} |\psi(0)\rangle \ . \tag{2.50}$$

The previous equation represents a continuous evolution in time. This evolution can be simulated by a gate model circuit by discretizing the time evolution.

According to the Suzuki-Trotter expansion of first order we can approximate the exponential of a sum of matrices by

$$e^{\left(\hat{A}+\hat{B}\right)\Delta t} = e^{\hat{A}\Delta t}e^{\hat{B}\Delta t} + \mathcal{O}(\Delta t^2)\,. \tag{2.51}$$

In order to reproduce AQC with gates, we must discretize the time evolution in a set of time steps, N, of step size $\Delta t = T/N$,

$$e^{-i\hat{\mathcal{H}}\Delta t} = e^{-i\hat{\mathcal{H}}_i\beta(\Delta t)}e^{-i\hat{\mathcal{H}}_f\gamma(\Delta t)} + \mathcal{O}(\Delta t^2)\,, \tag{2.52}$$

where

$$e^{-i\hat{\mathcal{H}}_i\beta} = \prod_{k=1}^{N} e^{-i(T-k\Delta t)\hat{\mathcal{H}}_i}, \quad e^{-i\hat{\mathcal{H}}_f\gamma} = \prod_{k=1}^{N} e^{-i(k\Delta t)\hat{\mathcal{H}}_f} \tag{2.53}$$

represents the time discretization of the initial and target Hamiltonians respectively.

Hence, we can decompose the adiabatic evolution operator $\hat{U}(T,0) = e^{-i\mathcal{H}(t)}$ in a product of exponentials

$$|\psi(t=T)\rangle = \hat{U}(T,0)\,|\psi(t=0)\rangle \simeq \prod_{k=1}^{N} e^{-i(T-k\Delta t)\hat{\mathcal{H}}_i}e^{-i(k\Delta t)\hat{\mathcal{H}}_f}\,|\psi(t=0)\rangle\ . \tag{2.54}$$

In the limit $N \to \infty$ the gate model reproduces exactly the adiabatic evolution.

CHAPTER 3

Iterative Techniques for Mixed-Integer Linear Programming Problems

As we stated in previous chapters, quantum computers are not mature enough to solve real-world problems. For instance, the embedding of a QUBO problem into the architecture of a quantum annealer imposes an important constraint in the number of variables our problem can have. For this reason, a *hybrid quantum-classical* (HQC) approach is currently the best method one can use to tackle large-scale problems, by combining quantum and classical solvers.

The aim of HQC approaches is to decompose a problem into a *sub-problem(s)* (SP(s)) and a *master problem* (MP), with the hope that one of these problems is suitable for a quantum computer. On the one hand, a quantum annealer receives a QUBO problem or a problem that can be cast into it. On the other, the other problems are solved by the classical solver using cutting-edge algorithms. The MP and SP(s) are solved iteratively until a given stopping criterion is satisfied.

There are classical systematic approaches that always converge for convex functions, such as the Benders' decomposition algorithm [13], while other heuristic approaches do not guarantee the optimal solution, but they require less computational resources to find a sub-optimal solution by exploring the configuration space according to some criteria.

In this chapter, we describe the general formulation of a mixed-integer linear programming problem, then we present the most relevant results of dual dynamic programming theory, which is the underlying theory of problem decomposition. Finally, we show a quantum-classical Benders' decomposition protocol [14], a heuristic quantum-classical protocol [15] and another version inspired on it.

3.1 Mixed-Integer Linear Programming

In the present work, we are interested in mixed-integer linear programming (MILP) problems because they appear in a wide range of real-world applications and they are the corresponding mathematical formulation of TEP problems. More precisely, TEP problems contain real variables but these variables are often converted into integer values to solve the problem.

The general mathematical formulation of MILP problems is

$$\min_{\mathbf{x},\mathbf{y}} \quad f(\mathbf{x},\mathbf{y}) \qquad [f : \mathbb{R}^n \times \mathbb{Z}^m \to \mathbb{R}] \tag{3.1a}$$

$$\text{s.t.} \quad h(\mathbf{x},\mathbf{y}) = \mathbf{0} \qquad [h : \mathbb{R}^n \times \mathbb{Z}^m \to \mathbb{R}^q]\,, \tag{3.1b}$$

$$g(\mathbf{x}) \leq \mathbf{0} \qquad [g : \mathbb{Z}^n \to \mathbb{R}^p]\,, \tag{3.1c}$$

where q and p are the number of equality and inequality constraints and $(\mathbf{x},\mathbf{y})$ are the real and integer variables, respectively.

TEP problems scale badly as the problem size grows. Thus, large problems cannot be solved at the desired resolution by nowadays classical computers with solvers such as Gurobi [16], CPLEX [17] or cbc [18]. There are already benchmarks comparing classical solvers and quantum annealers for energy problems where even the fastest of these solvers – Gurobi – cannot find the optimal solution in a fixed run-time, see Ref. [3].

3.2 Classical Benders' Decomposition

The main idea behind classical *Benders' decomposition* (BD) – also known as *dual dynamic programming* (DDP) – is to decompose the primal problem into a master problem and a subproblem(s) once the complicated variables are detected. In the following sections, we introduce the most relevant results of dual dynamic programming. For a deeper understanding see Ref. [19].

3.2.1 Complicated Variables

We can consider a decision variable to be complicated if the decision variable is involved in most of the constraints or the decision variable produces non-convex optimization problems.

Definition 3.1

Complicated variables are those that allow us to split the original problem into subproblem(s) when they are fixed. These variables appear in constraints in such a way that the problem cannot be decomposed if they are not fixed. These type of constraints are known as linking constraints.

After splitting the original problem, the master problem is a relaxed version of the original problem. The global minimum is guaranteed if and only if the objective function as a function of the complicated variables is a convex envelope.

The BD method is used to decompose the general problem (3.1) into an MP with integer variables and an SP so that a clever rearrangement of the variables allows us to fix the complicated variables of the MP – once the MP is solved – and send these complicated variables as fixed

for the sub-problem. This process is performed iteratively until a stopping criterion is satisfied. The convergence of BD guarantees that we can satisfy an arbitrary threshold value after a number of iterations [20].

3.2.2 Dual Dynamic Programming

Suppose that an electricity market has to sell its energy production to different customers. However, the maximum amount of energy per customer is fixed to 10 units of energy where each unit of energy cost \$1. The electricity company decides to relax the constraints allowing customer Bob to buy more than 10 units of energy but paying \$2 for each extra unit of energy. Bob decides to buy extra units of energy because with one unit of energy he produces a chip that he sells for \$3. Then, Bob buys every extra energy unit he can, getting a profit of \$1 for each extra unit of energy. The electricity market notices that Bob is buying too much energy and it cannot provide the required energy to other customers, paying a fine because of that. In order to relax the constraints but at the same time be able to satisfy the demand of other customers, the electricity market decides to increase the price of the extra unit of energy to \$3. Now, the current price of extra unit of energy implies that Bob does not get profit so he is no longer interested in buying extra units of energy. In other words, Bob has no interest in violating the constraint, which he could do because it has been relaxed.

Notice that we have two points of view when solving combinatorial optimization problems. We can adopt either the customer's viewpoint, who wants to solve the optimization problem, or the market's, which defines the constraints and prices.

We can relax the primal problem (3.1) by associating a penalty to each of its constraints,

$$\lambda \in \mathbb{R}^q \text{ and } \mu \in \mathbb{R}^p, \tag{3.2}$$

also known as Lagrange multipliers. Then the Lagrangian of the primal problem is

$$\mathcal{L}(\mathbf{x}, \mathbf{y}, \lambda, \mu) = f(\mathbf{x}, \mathbf{y}) + \lambda^\intercal h(\mathbf{x}, \mathbf{y}) + \mu^\intercal g(\mathbf{x}) \,. \tag{3.3}$$

The Lagrangian allows us to reformulate the problem by augmenting the objective function with a set of constraint functions weighted by the Lagrange multipliers, so that our new problem has zero constraints.

Definition 3.2: Dual function

The dual function, $q : [\mathbb{R}^{q+p} \to \mathbb{R}]$, maps the set of parameters λ and μ to a minimization problem

$$q(\lambda, \mu) = \min_{\mathbf{x},\mathbf{y}} \mathcal{L}(\mathbf{x}, \mathbf{y}, \lambda, \mu) \,, \tag{3.4}$$

where q and p are the number of equality and inequality constraints and

$$\lambda \in \mathbb{R}^q \text{ and } \mu \in \mathbb{R}^p \,, \tag{3.5}$$

are the penalty parameters.

Notice that the dual function is an unconstrained problem because all the constraints have been moved into the Lagrangian. Considering the MILP problem (3.1), the inequality constraint $g(\mathbf{x}) \leq \mathbf{0}$ is violated if $g(\mathbf{x}) > \mathbf{0}$. To take into account this in the cost function we need to

introduce a positive cost, so $\mu^\intercal g(\mathbf{x}) > 0$ which implies μ has positive coefficients, that is

$$\mu \geq \mathbf{0}\,. \tag{3.6}$$

Theorem 3.1

The dual of a linear optimization problem is another linear optimization problem.

Theorem 3.2

The dual of the dual is the primal.

Theorem 3.3

Suppose that $(\mathbf{x}^*, \mathbf{y}^*)$ is not only a feasible solution, i.e., a solution satisfies the constraints, but an optimal solution of the primal. Then,

$$q(\lambda, \mu) \leq f(\mathbf{x}^*, \mathbf{y}^*) \tag{3.7}$$

is a lower bound of the primal problem, for

$$\lambda \in \mathbb{R}^q \text{ and } \mu \in \mathbb{R}^p\,. \tag{3.8}$$

Proof.

$$q(\lambda,\mu) = \min_{\mathbf{x},\mathbf{y}} \mathcal{L}(\mathbf{x},\mathbf{y},\lambda,\mu) \leq \mathcal{L}(\mathbf{x}^*,\mathbf{y}^*,\lambda,\mu) = f(\mathbf{x}^*,\mathbf{y}^*) + \underbrace{\lambda^\intercal h(\mathbf{x}^*,\mathbf{y}^*)}_{0} + \underbrace{\mu^\intercal g(\mathbf{x}^*)}_{\leq 0} \leq f(\mathbf{x}^*,\mathbf{y}^*) \tag{3.9}$$

QED

As stated before, a feasible solution satisfies the constraints, meaning that $h(\mathbf{x}^*, \mathbf{y}^*) = \mathbf{0}$ and $g(\mathbf{x}^*) \leq \mathbf{0}$. This implies that the dual function $q(\lambda, \mu)$ is a lower bound and the relaxed problem $f(\mathbf{x}^*, \mathbf{y}^*)$ is the upper bound. Notice that if $\mu = \mathbf{0}$, then the lower bound and upper bound would have the same value. In DDP we need to optimize not only the decision variables $\mathbf{x}$ but also the penalty parameters μ and λ.

Corollary 3.1

In particular, if $(\mathbf{x}^*, \mathbf{y}^*)$ is an optimal solution of the primal and $(\mathbf{x}, \mathbf{y})$ is a feasible solution of the primal, then

$$\underbrace{q(\lambda,\mu)}_{\text{Lower Bound}} \leq f(\mathbf{x}^*,\mathbf{y}^*) \leq \underbrace{f(\mathbf{x},\mathbf{y})}_{\text{Upper Bound}} \tag{3.10}$$

We want to find the best possible lower bound by solving the following optimization problem

$$\max_{\lambda,\mu} \quad q(\lambda,\mu) \qquad \left[q : \mathbb{R}^{q+p} \to \mathbb{R}\right] \tag{3.11a}$$

$$\text{s.t.} \quad \mu \geq \mathbf{0}, \tag{3.11b}$$

$$(\lambda,\mu) \in \{\lambda,\mu \mid q(\lambda,\mu) \text{ bounded}\}\,. \tag{3.11c}$$

Notice that we are choosing the penalties in such a way that the dual problem is bounded.

Theorem 3.4: Weak Duality

If $(\mathbf{x}^*, \mathbf{y}^*)$ is the optimal solution of the primal, and (λ^*, μ^*) is the optimal solution of the dual, then

$$q(\lambda^*, \mu^*) \leq f(\mathbf{x}^*, \mathbf{y}^*) \,. \tag{3.12}$$

This inequality holds even for non-convex problems.

The difference between the lower bound and the upper bound is known as the optimal duality gap which is always non-negative.

Theorem 3.5: Strong Duality

If $(\mathbf{x}^*, \mathbf{y}^*)$ is the optimal solution of a convex[a] primal problem, and (λ^*, μ^*) is the optimal solution of its dual, then

$$q(\lambda^*, \mu^*) = f(\mathbf{x}^*, \mathbf{y}^*) \,. \tag{3.13}$$

[a] There are conditions beyond convexity under which the strong duality holds [21]. However, we are not going to discuss them in the present work.

In other words, the strong duality theorem guarantees that under some iterative approach – as the one we describe in the next section – the upper bound and lower bound are going to converge into a point for the optimal solution of a convex problem, see Fig. 3.1 adapted from [14].

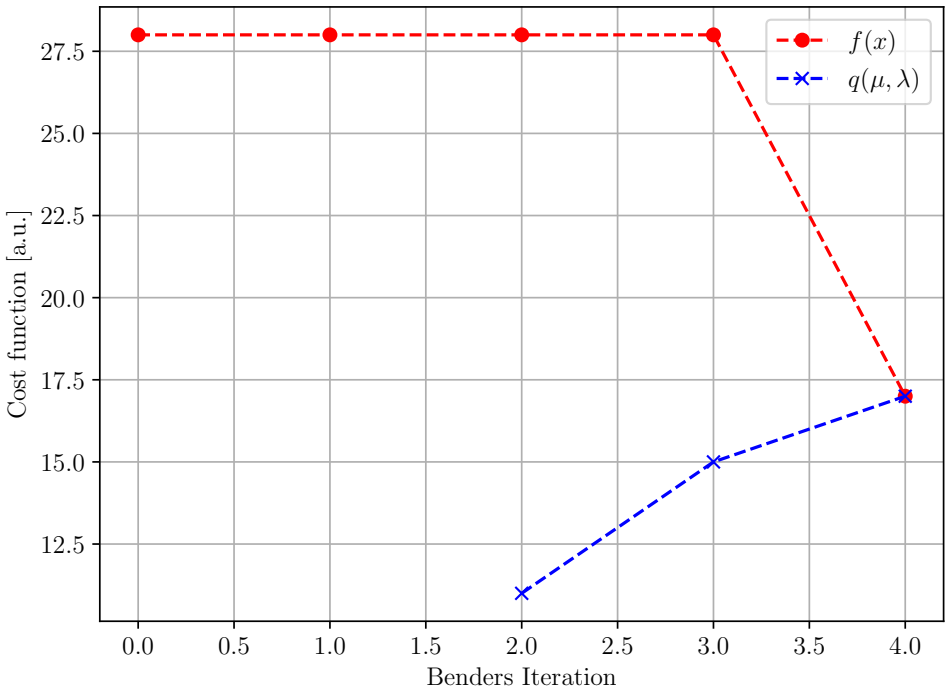


Figure 3.1: Benders' decomposition convergence for a combinatorial problem in which the strong duality theorem is satisfied. The x-axis indicates the number of iteration in the BD algorithm and y-axis indicates the value of the cost function.

Corollary 3.2

If either the primal problem or the dual problem are unbounded, then both problems are unbounded.

Proof. If the primal is unbounded, then according to the weak duality theorem

$$q(\lambda^*, \mu^*) \leq \infty \tag{3.14}$$

which implies the dual problem is unbounded. Analogously, if the dual problem is unbounded, then the primal problem is unbounded too. QED

Corollary 3.3

If there exist $\mathbf{x}^*, \mathbf{y}^*, \lambda^*$ and μ^* such that,

$$q(\lambda^*, \mu^*) = f(\mathbf{x}^*, \mathbf{y}^*) \tag{3.15}$$

then they are optimal.

Proof. Consider a feasible solution of the primal problem, $(\mathbf{x}, \mathbf{y})$, and a feasible solution of the dual problem, (λ, μ), then

$$f(\mathbf{x}, \mathbf{y}) \geq q(\lambda^*, \mu^*) = f(\mathbf{x}^*, \mathbf{y}^*) \geq q(\lambda, \mu)\,. \tag{3.16}$$

QED

3.2.3 Karush–Kuhn–Tucker Conditions

In this section we introduce a first-order necessary conditions for a solution of convex problems to be optimal, the *Karush–Kuhn–Tucker* (KKT) conditions. For a deeper understanding see Ref. [21].

Suppose we are given an LP problem, i.e., a problem with real variables $x_i \in \mathbf{x}$ whose functions are differentiable. Then its Lagrangian can be written as

$$\mathcal{L}(\mathbf{x}, \lambda, \mu) = f(\mathbf{x}) + \lambda^\intercal h(\mathbf{x}) + \mu^\intercal g(\mathbf{x})\,. \tag{3.17}$$

We can get the dual problem of (3.1) by writing a set of equations from the Lagrangian and the constraints of the primal problem. The following set of equations

$$\frac{\partial \mathcal{L}(\mathbf{x}, \lambda, \mu)}{\partial x_i} = 0, \quad \forall x_i \in \mathbf{x} \tag{3.18}$$

$$h(\mathbf{x}) = \mathbf{0} \tag{3.19}$$

$$\mathbf{0} \leq -g(\mathbf{x}) \perp \mu \geq \mathbf{0} \tag{3.20}$$

are known as the KKT conditions. Notice that if the optimization problem we are minimizing does not have any constraint, then we could consider just the first equation, i.e., the partial derivatives of the Lagrangian.

MILP problems mix real and integer variables which implies they are not convex and its functions are not differentiable so that KKT conditions are no longer valid. However, if we decompose the MILP problem into an *integer programming* (IP) problem and a *linear programming* (LP)

problem we could solve the LP using the KKT conditions and the IP with a standard solver or in our case with a quantum annealer.

From previous sections, we know that a MILP problem can be written as,

$$\max_{\lambda,\mu} q(\lambda,\mu) = \max_{\lambda,\mu} \left[\min_{\mathbf{x},\mathbf{y}} \mathcal{L}(\mathbf{x},\mathbf{y},\lambda,\mu) \right] \tag{3.21}$$

$$= \max_{\lambda,\mu} \underbrace{\left[\min_{\mathbf{x},\mathbf{y}} f(\mathbf{x},\mathbf{y}) + \lambda^\intercal h(\mathbf{x},\mathbf{y}) + \mu^\intercal g(\mathbf{x}) \right]}_{\text{MILP problem}} , \tag{3.22}$$

where the outer optimization problem is a maximization over the Lagrange multipliers (λ, μ) and the inner optimization problem is a minimization over the primal variables $\mathbf{x}$ and $\mathbf{y}$. Notice that the inner problem is an unconstrained problem since the constraints of the original problem are already taken into account as penalties in the Lagrangian. The inner problem has integer variables so we cannot use the KKT conditions but if the integer variables are fixed, $\mathbf{y}^*$, then the inner problem is a LP

$$\max_{\lambda,\mu} q(\lambda,\mu) = \max_{\lambda,\mu} \left[\min_{\mathbf{x}} \mathcal{L}(\mathbf{x},\mathbf{y}^*,\lambda,\mu) \right] \tag{3.23}$$

$$= \max_{\lambda,\mu} \underbrace{\left[\min_{\mathbf{x}} f(\mathbf{x},\mathbf{y}^*) + \lambda^\intercal h(\mathbf{x},\mathbf{y}^*) + \mu^\intercal g(\mathbf{x}) \right]}_{\text{LP problem}} , \tag{3.24}$$

Example: ***Quadratic Convex Minimization Problem***

Consider the following optimization problem,

$$\min_x \quad f(x) = mx^2 + nx \qquad [f : \mathbb{R} \to \mathbb{R}] \tag{3.25a}$$

$$\text{s.t.} \quad h(x) = ax - b = 0 \quad [h : \mathbb{R} \to \mathbb{R}] \ . \tag{3.25b}$$

The KKT conditions of this problem are

$$ax^* - b = 0 \tag{3.26}$$

$$2mx^* + n + a\lambda^* = 0 \, . \tag{3.27}$$

The solution of this set of equations are the optimal primal and dual variables,

$$x^* = \frac{b}{a} \tag{3.28}$$

$$\lambda^* = -\left(\frac{2mb}{a^2} + \frac{n}{a} \right) \ . \tag{3.29}$$

We can solve the problem in a different way by using the dual theory. First, reformulate the problem as an unconstrained inner minimization problem and an outer maximization problem

$$\max_{\lambda} \left[\min_{x} \mathcal{L}(x,\lambda) \right] \tag{3.30}$$

apply the KKT conditions in the inner problem,

$$\frac{\partial \mathcal{L}}{\partial x} = 2mx + n + a\lambda = 0 \tag{3.31}$$

$$x = -\frac{n + a\lambda}{2m}. \tag{3.32}$$

Inserting x in terms of λ in the inner problem yields

$$\max_{\lambda} \left[\frac{(n+a\lambda)^2}{4m} - \frac{n(n+a\lambda)}{2m} - \lambda \left(\frac{-a(n+a\lambda)}{2m} - b \right) \right]. \tag{3.33}$$

If we apply again the KKT conditions,

$$\frac{\partial \mathcal{L}}{\partial \lambda} = \frac{a(n+a\lambda)}{2m} - \frac{na}{2m} - \frac{na}{2m} - \frac{\lambda a^2}{m} - b = 0 \tag{3.34}$$

we get

$$\lambda = -\left(\frac{2mb}{a^2} + \frac{n}{a} \right). \tag{3.35}$$

If we now insert this in Eq. (3.32) we get the optimal solution of the primal problem.

This example shows two different ways of solving an optimization problem and how they are equivalent.

3.2.4 Benders' cuts

Consider the following two-variable MILP problem,

$$\min_{x,y} \quad f(x,y) = mx + ny \quad [f : \mathbb{R} \times \mathbb{Z} \to \mathbb{R}] \tag{3.36a}$$

$$\text{s.t.} \quad h(x,y) = ax - by = 0, \tag{3.36b}$$

$$g(x) = cx \leq 0, \tag{3.36c}$$

$$x \geq 0, \tag{3.36d}$$

$$y \in \{0,1\}, \tag{3.36e}$$

where $a, b, c, m, n \in \mathbb{R}$, Eq. (3.36b) is a linking constraint and the integer variable is binary. Fixing the complicated variable to a feasible value y^* would allow us to split the original problem into an SP, also called slave problem – which is an LP:

Slave problem

$$\min_{x} \quad \alpha \equiv mx \tag{3.37a}$$

$$\text{s.t.} \quad h(x,y^*) = ax - by^* = 0 \quad : \lambda, \tag{3.37b}$$

$$g(x) = cx \leq 0 \quad : \Pi, \tag{3.37c}$$

$$x \geq 0 \quad : \mu. \tag{3.37d}$$

We can write the problem as

$$\max_{\lambda,\mu} \left(\min_{x} \mathcal{L}(x, y^*, \lambda, \Pi, \mu) \right) \tag{3.38}$$

and using the KKT conditions in the inner optimization problem yields its dual optimization problem

$$\max_{\lambda, \Pi, \mu} \quad -by^*\lambda \tag{3.39a}$$

$$\text{s.t.} \quad \frac{\partial \mathcal{L}}{\partial x} = m + a\lambda + c\Pi - \mu = 0 \tag{3.39b}$$

Since the objective function only has λ and not Π and μ we can re-write the previous problem as an inequality by dropping the (Π, μ) Lagrange multipliers

Dual formulation of slave problem

$$\max_{\lambda} \quad -by^*\lambda \tag{3.40a}$$

$$\text{s.t.} \quad m + a\lambda \geq 0 \tag{3.40b}$$

The resolution of the dual problem leads to the candidate values x^* and λ^*. On the other hand, the MP, which is an IP, is given by

Master problem

$$\min_{y, \alpha} \quad z = ny + \alpha \tag{3.41a}$$

$$\text{s.t.} \quad \alpha \geq \alpha_{\text{down}}, \tag{3.41b}$$

$$\alpha \geq \underbrace{\lambda\,(-by)}_{\text{Benders' cuts}}, \tag{3.41c}$$

$$y \in \{0, 1\}. \tag{3.41d}$$

where α_{down} is a lower bound for α that we must fix so that the problem is bounded and $\lambda(-by)$ are the Benders' cuts. In every iteration of the algorithm, we add a Benders' cut with a different value of λ that comes from the resolution of the dual slave problem. These cuts reduce the configuration space for the candidate feasible solutions, so that in every iteration we are closer to the optimal solution.

Strong Duality condition

For lineal problems it is guaranteed that

$$mx^* + ny^* = -by^*\lambda^* \tag{3.42}$$

for the optimal values x^*, y^* and λ^*.

Notice that this process can be repeated, i.e., we can add an optimal cut per constraint in every iteration. Thus, reducing the configuration space. The iterative process is known as Benders' decomposition and we provide an example of it in Chapter 4.

3.3 Hybrid quantum-classical algorithms

In Appendix B, we show the foundations of *simulated annealing* (SA) and solve a travelling salesman problem to illustrate it. A simulated annealing algorithm does not guarantee an optimal solution but the results we can get with a clever annealing schedule can be good enough in accuracy and time. The same applies to a quantum annealing algorithm. In this section, we show how the decomposition of a QUBO problem allows us to use both quantum and classical solvers, where the classical solvers relax the original problem so that it can be tackled by the quantum solver.

3.3.1 Quantum Benders' Decomposition

The main idea behind the quantum Benders approach is to use the quantum annealer to solve either the primal problem or the dual problem [14]. We are interested in solving the primal problem, also known as master problem, with a quantum annealer.

3.3.2 Heuristic Protocols

As stated before, heuristic approaches do not guarantee to arrive at the optimal solution. However, these methods explore the configuration space in such a way that we do not get stuck in a local minimum. For a limited amount of computational resources and time, the solution that a heuristic approach can achieve is sub-optimal. Generally, these methods are better than a brute-force method when the problem is large enough so that there are no computational resources that are able to provide the optimal solution.

SA-QA Protocol

We present a hybrid quantum-classical algorithm protocol based on combining SA and QA [15] to solve the MP and SP(s), respectively. Figure 3.2 shows a graphical representation of the SA-QA protocol, which we describe in more detail below.

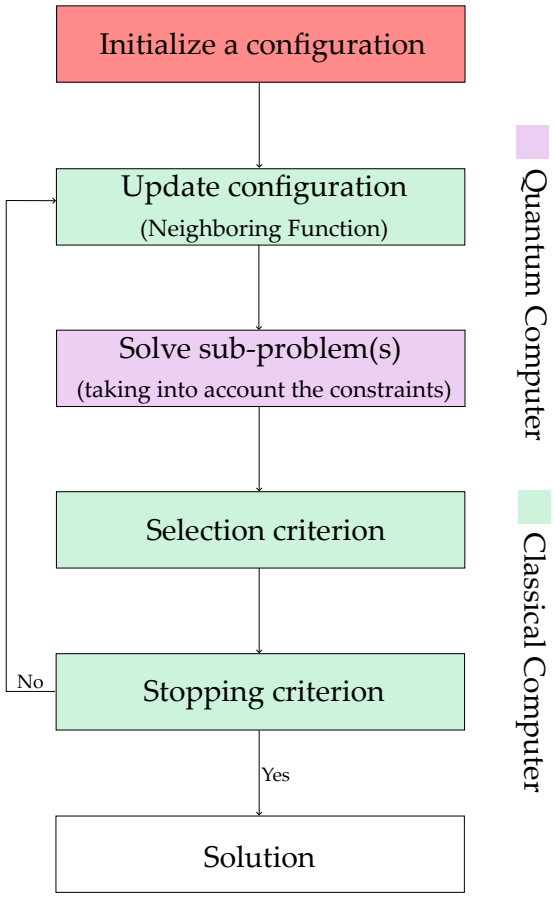


Figure 3.2: SA-QA scheme. See main text for details on this protocol.

1. Set the upper and lower bound of the problem – so that we avoid dealing with unbounded problems – and initialize the parameters of the problem, i.e., annealing schedule, initial and final temperature, and a selection criterion.

2. Randomly generate a configuration by changing the values of the binary variables according to a neighboring function that generates a new configuration in one of these ways:

   (a) Randomly pick a binary variable with value 1 and set it to 0.

   (b) Randomly pick a binary variable with value 0 and set it to 1.

   (c) Randomly pick two binary variables with different values and swap them.

   The new configuration is going to be selected or discarded according to a simulated annealing algorithm.

3. Given the new configuration, solve the sub-problem(s) – dual problems – with a quantum annealing algorithm.

4. Apply the selection criterion to keep or discard the current configuration.

5. If the selection criterion is not satisfied, repeat steps 2 to 4 until an iteration index is equal to the value set in step 1, then decrease the temperature and reset the iteration index. If the selection criterion is satisfied, the protocol outputs the current cost function value and its solution $\mathbf{x}$.

Notice that the algorithm solves the master problem with a simulated annealing algorithm in a classical solver and then the sub-problem(s) – dual problems – with the quantum computer,

more precisely with a quantum annealer. For this reason, the sub-problem(s) must have binary constraints or low integer values if we do not want to deal with discretization errors or adding a big set of slack variables.

In order to apply both simulated and quantum annealing to problems in which the master problem contains most of the binary and integer variables – as is the case with the TEP problem –, we have to reconsider or adapt the previous scheme.

QA-SA Protocol

As stated before, our sub-problem(s) has real constraints which implies it is not a clever approach to solve them with a quantum annealer. For this reason, we reverse the original protocol so that the quantum annealer tackles the master problem in which the binary and integer decision variables appear and the classical solver tackles the sub-problem(s). A scheme of the new protocol is shown in Figure 3.3.

In order to explain the protocol in accordance to what has been shown in the previous section we use the same notation $f(\mathbf{x}, \mathbf{y})$ to describe the upper bound of the primal problem and $q(\lambda, \mu)$ to describe the lower bound, where (λ, μ) represents the penalties of our problem, that is, the Lagrange multipliers, and $\mathbf{x}, \mathbf{y}$ represent the decision variables. We are showing a heuristic approach, meaning that the convergence is not guaranteed. That is, we are not getting the optimal values $(\mathbf{x}^*, \lambda^*, \mu^*)$ that maximize the lower bound and minimize the upper bound in each iteration. Instead, we are optimizing the dual problem by a SA algorithm and the master problem by QA, so that none of these methods guarantees the optimal solution to the combinatorial optimization problem.

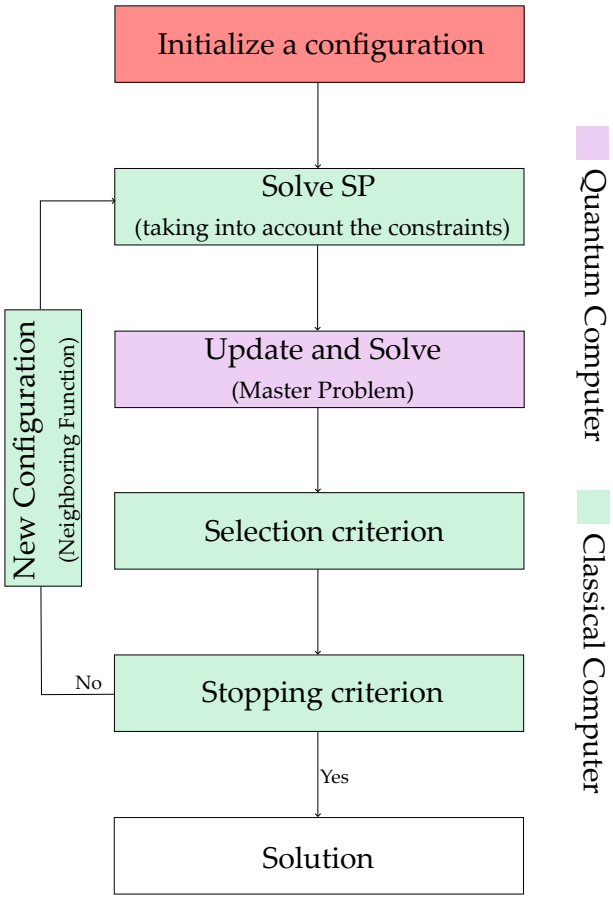


Figure 3.3: QA-SA protocol scheme. See the main text for details on this protocol.

1. Set an initial feasible configuration to the primal problem $\mathbf{x}$.

2. Solve the dual problem by SA,

$$\max_{\lambda,\mu} \quad q(\lambda,\mu) \qquad \left[q : \mathbb{R}^{m+p} \to \mathbb{R}\right] \tag{3.43a}$$

$$\text{s.t.} \quad \mu \geq \mathbf{0}, \tag{3.43b}$$

$$(\lambda,\mu) \in \{\lambda,\mu \mid q(\lambda,\mu) \quad \text{bounded}\}. \tag{3.43c}$$

   Notice that the parameters λ and μ do not have to be the optimal parameters. SA does not guarantee to obtain the optimal solution of the dual problem.

3. Update the master problem with $\{\lambda,\mu\}$

$$q(\lambda,\mu) = \min_{\mathbf{x}} \mathcal{L}(\mathbf{x},\lambda,\mu). \tag{3.44}$$

   QA does not guarantee to obtain the optimal solution $\mathbf{x}^*$ of the master problem.

4. Apply the selection criterion to keep or discard the current configuration.

5. If the stopping criterion is satisfied, then the protocol outputs the solution. Otherwise, insert the values $\mathbf{x}$ of step 3 into step 2 and repeat until step 4.

CHAPTER 4

Transmission Expansion Planning by Quantum Annealing

4.1 Statement of the Problem

The *Transmission expansion planning* (TEP) [22] problem determines the number of new lines to be installed in order to achieve a specific goal, optimizing the performance of an energy system subject to the assumptions of a future scenario. It is classically formulated as a *mixed integer linear programming* (MILP) problem that aims at finding the optimal way to expand the capacity of an energy system by minimizing the cost function, which is the sum of the investment cost of transmission lines and the operational cost of generators. It decides how many transmission lines to build and where to connect them in order to satisfy the energy demand on a distributed energy system with a high share of renewable energy sources. There are other components to build apart from transmission lines such as generators or storages, but considering them implies not only an increment in the number of variables of the problem but also an increment in its complexity. For this reason, the problem we are solving is not completely realistic but it encapsulates the most important features of it. We should consider it just as a starting point to get an insight into the problem formulation and challenges.

We can control the number of variables of the problem according to

- **Number of nodes:** clustering techniques allow us to fix the number of nodes of the problem so that some loads of a given network are grouped acting as a single big load.
- **Snapshots of time**: in expansion problems, it is common to consider hourly snapshots of time of one year, i.e., a total of 8760 snapshots. However, we can restrict the problem to a subset of snapshots without loss of generality.
- **Connectivity of transmission lines**: we can control the connectivity between nodes so that we do not allow a node the possibility of being fully connected with the rest of the nodes. Furthermore, because of legal procedures to build new lines in areas where there are no transmission lines, TEP focuses on building lines in existing connections without changing the topology of the network.
- **Adding targets**: in this chapter, we minimize the cost function as a function of the investment cost of transmission lines, the operational cost of generators and the load shedding cost, which is going to be explained later. But, there are other targets such as the increment of renewable energy sources in a given region or the reduction of the carbon footprint that we are not taking into account.

The resolution of MILP associated with TEP scales badly using classical algorithms [23] and, at the same time, energy system models are getting larger and more complex due to the integration of decentralized weather-dependent renewable energy sources, intermittent loads, sector

coupling and the increase of storage components. Currently, the problem is often linearized or the scope and granularity of the model are reduced using clustering algorithms. For this reason, any computational time reduction will have substantial implications in closing the granularity gap between what the current models can solve and the desired resolution needed by energy system operators. However, since quantum computers are still not sufficiently mature, large TEP problems cannot be solved fully by a quantum annealer. For this reason, we require hybrid methods [24]–[28] to decompose large problems into a master problem, which can be solved by a quantum annealer, and a sub-problem for which cutting-edge classical algorithms are going to be applied.

4.1.1 Nomenclature

In this section, we introduce the notation of the TEP problem. Let N be the set of nodes of a given network, H the set of snapshots, C the set of candidate transmission lines and E the set of existing lines. Then x_{kl}, with $kl \in C$, represents the binary variable of the transmission line from node $k \in N$ to $l \in N$ – it decides if that transmission line is built $x_{kl} = 1$ or not $x_{kl} = 0$ –, $d_k(h)$ represents the demand at node $k \in N$ in snapshot $h \in H$, f^0_{kl} with $kl \in E$ represents the power flow being transmitted in existing line from node $k \in N$ to $l \in N$, $\bar{f}^0_{kl}$ with $kl \in E$ represents the maximum power flow being transmitted in existing line from node $k \in N$ to $l \in N$, f^1_{kl} with $kl \in C$ represents the power flow being transmitted in candidate line from node $k \in N$ to $l \in N$, $\bar{f}^1_{kl}$ with $kl \in E$ represents the maximum power flow being transmitted in existing line from node $k \in N$ to $l \in N$, g_k represents the energy produced at node $k \in N$, $\bar{g}_k$ represents the maximum energy production at node $k \in N$ and r_k represents the load shedding, which can be thought as an artificial generator with a very large operational cost. Lastly, c_{kl} is the coefficient of investment cost of the transmission line from node $k \in N$ to $l \in N$, c^{oc}_k is the operational cost of generator $k \in N$ and c_k is the load shedding cost associated with r_k.

Symbol	Description	Type
N	Set of nodes of the network	Set
H	Set of snapshots	Set
C	Set of candidate transmission lines	Set
C_k	Set of candidate transmission lines from all nodes to node k	Set
E	Set of existing transmission lines	Set
E_k	Set of existing transmission lines from all nodes to node k	Set
x_{kl}	Transmission line from node k to l	Binary
f^0_{kl}	Power flow in existing line from node k to l	Integer
$\bar{f}^0_{kl}$	Maximum power flow in existing line from node k to l	Integer
f^1_{kl}	Power flow in candidate line from node k to l	Integer
$\bar{f}^1_{kl}$	Maximum power flow in candidate line from node k to l	Integer
r_k	Shedding load at node k	Integer
$d_k(h)$	Demand of node k at snapshot h	Integer
$g_k(h)$	Current generation at node k at snapshot h	Integer
$\bar{g}_k$	Maximum generation at node k	Integer
c_{kl}	Investment cost of transmission line from node k to l	Real
$c^{(\text{oc})}_k$	Annualised operational cost per MWh of generator g_k	Real
c_k	Cost of shedding load at node k	Real

Table 4.1: Description of variables involved in TEP problems [24].

4.1.2 Brownfield and Greenfield Models

There are two ways of formulating the TEP, shown in Figure 4.1. On the one hand, the brownfield model considers the power flow of existing lines in a network f_{kl}^0 among with the power flow of a set of candidate lines f_{kl}^1. On the other hand, the greenfield model considers an empty network and a set of candidate lines f_{kl}^1. For simplicity, we are going to consider the greenfield approach so that we do not have to take into account the constants associated with the existing lines. We can map two indices ij to a single index k by using a bijective function for a given number of nodes N, see Ref. [29],

$$k(i,j,N) = \begin{cases} iN - i\,(i+1)\,/2 + j - (i+1)\,, & i < j \\ \text{None}\,, & i = j \\ k(j,i,N)\,, & i > j \end{cases} \tag{4.1}$$

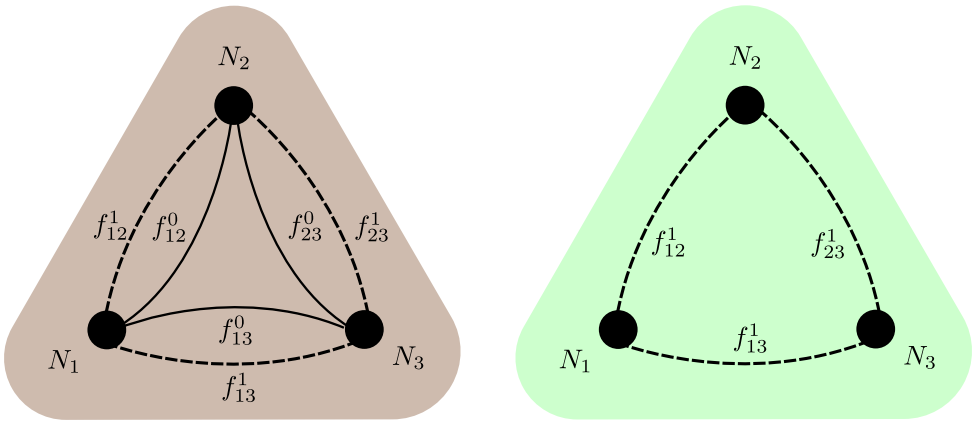


Figure 4.1: Three node example. **Left:** Network considering current transmission lines – solid lines – and candidate lines – dashed lines –, brownfield model. **Right:** Network considering just candidate lines, greenfield model.

Luckily, the *dimod* python package from D-Wave allows us to work with two indices so that we do not need to use the previous function to reformulate the problem.

4.1.3 Network Topology

The TEP problems usually consider the building of new lines in existing connections so that we do not change the network topology. One of the reasons for adding lines in existing connections is because the legal permissions are granted. However, in a future scenario with a high share of decentralized renewable energies, new connections have to be considered. Hence, we are going to enable new connections by changing the topology. Moreover, we allow all lines to be connected to each node, i.e., we are considering fully connected networks.

4.1.4 Formulation

We formulate the TEP problem according to the work of Dilwali *et al.* [24] as follows

$$\min_{\mathbf{x},\mathbf{g},\mathbf{r},\mathbf{f}^0,\mathbf{f}^1} \quad \underbrace{\sum_{kl\in C} c_{kl}x_{kl}}_{\text{Investment cost}} + \underbrace{\sum_{h\in H}\sum_{k} c_k^{(\text{oc})} g_k(h)}_{\text{Operational cost}} + \underbrace{\sum_{h\in H}\sum_{k} r_k(h)c_k}_{\text{Load shedding cost}} \tag{4.2a}$$

$$\text{s.t.} \quad d_k(h) - \left(\sum_{l\in E_k} f_{kl}^0(h) + \sum_{l\in C_k} f_{kl}^1(h) + g_k(h) + r_k(h)\right) = 0\,, \quad \forall k\in N, h\in H, \tag{4.2b}$$

$$\left|f_{kl}^0(h)\right| - \bar{f}_{kl}^0(h) \le 0\,, \qquad \forall\, kl\in E, h\in H, \tag{4.2c}$$

$$\left|f_{kl}^1(h)\right| - \bar{f}_{kl}^1(h)x_{kl} \le 0\,, \qquad \forall\, kl\in C, h\in H, \tag{4.2d}$$

$$g_k(h) - \bar{g}_k(h) \le 0\,, \qquad \forall\, k\in N, h\in H, \tag{4.2e}$$

$$r_k(h) - d_k(h) \le 0\,, \qquad \forall\, k\in N, h\in H, \tag{4.2f}$$

$$\mathbf{d},\mathbf{g},\mathbf{r} \ge 0, \tag{4.2g}$$

$$x_{kl}\in\{0,1\}\,, \qquad \forall\, kl\in C, \tag{4.2h}$$

where the variables are described in Table 4.1. We next briefly describe each equation:

- **Objective function Eq.** (4.2a): it is the sum of investment cost of transmission lines, operational cost of generators and load shedding cost.
- **Power balance Eq.** (4.2b): it represents the power balance constraints, i.e., if the demand $d_k(h)$ at node k and snapshot h is fulfilled by the generator g_k at node k and the power flow incoming from other nodes $f_{kl}^0(h)$ and $f_{kl}^1(h)$. The term r_k is the load shedding. It can be thought of as an artificial generator whose operational cost is very large. It represents the fine for not fulfilling the demand in a node and it guarantees that the problem is feasible. To ensure that mathematically the demand is always fulfilled one can consider a load shedding term that is zero when demand at node k is satisfied by the elements of the network and $r_k \neq 0$ when it is not fulfilled. If $r_k \neq 0$ the demand is not fulfilled, i.e., the electricity markets cannot offer the required energy to a given node and then pay a fine, which is an extra cost in the objective function. If the load shedding cost is high enough the demand is going to be always fulfilled since the load shedding term is the dominant term we want to minimize.
- **Existing circuit flow limits Eq.** (4.2c): the power flow of an existing transmission f_{kl}^0 line cannot exceed its maximum capacity, $\bar{f}_{kl}^0$.
- **Candidate circuit flow limits Eq.** (4.2d): the power flow of a candidate transmission line f_{kl}^1 cannot exceed its maximum capacity, $\bar{f}_{kl}^1$.
- **Node generation limits Eq.** (4.2e): the energy produced in a node k, g_k, cannot exceed its maximum energy production, $\bar{g}_k$.
- **Node loads limits Eq.** (4.2f): the load shedding at node k, r_k, cannot exceed the load d_k of that node.

4.1.5 Remarks

The previous model is called a transport model since we are not considering Kirchoff's equations. Notice that we include the operational cost which is usually not included in the literature [30]. The reason for usually not including the operational cost is that other authors consider

a single snapshot so that the dominant term is the investment cost. However, if one wants to consider a large number of snapshots, then the operational cost is the dominant term. On the other hand, if we minimize the investment cost, then there is a change of building transmission lines that connect the most expensive generators to the nodes. In summary, extremal solutions lead to a high value of the cost function in one of the following ways:

- **Under-investment** leads to a high value of the total cost function because the system is not able to fulfil the demand $d(h)$.
- **Over-investment** leads to a high value of the total cost function even though it satisfies the demand. Intuitively, we are creating more lines $\{x_{kl}\}$ than we need. Usually, there is an upper bound due to the capital budget.

The investment cost c_{kl} is the cost of building the transmission line x_{kl} and it is a constant – we are not considering time-dependent costs. Analogously, the operational cost $c_j^{(\mathrm{oc})}$ represents the annualised cost per MWh of the produced energy $g_k(h)$ at snapshot h. For simplicity, we set the operational cost of each carrier – wind turbine, solar or run of river among others – to a fixed value – mean cost of that carrier – so that we do not have to consider the variation of the cost as a function of the node or the time.

Notice that when the solution is feasible, i.e., $r_k = 0$ for all $k \in N$, then the total cost depends on the investment cost and the operational cost. As stated before, the number of possible configurations of our problem scales badly. For this reason, knowing if we are considering a large set of snapshots – large investment planning - or a small subset of snapshots gives us clues about what region of the configuration space we should explore. There are two regions to be considered:

- **Large expansion planning:** for instance if we consider a 10 year expansion planning problem and we produce the energy with the cheapest carriers – e.g. wind turbines – we will reduce the operational cost drastically as compared with producing the energy with other carriers. However, it is possible that the fact of producing the energy with the cheapest carriers implies the need of building the most expensive transmission lines, hence increasing the investment cost.
- **Short expansion planning:** for instance if we consider a single-snapshot. Then, it is clear that the minimum total cost function implies a minimum in the investment cost regardless of the operational costs, since the operational costs are lower as compared with the investment cost when there is only one snapshot to optimize over.

To summarise, given the energy demand for a set of nodes N for a set of snapshots H, the goal is to optimise the network so as to build the minimum amount of transmission lines – minimizing in that way the investment cost – while satisfying the energy demand for each snapshot. We are also taking into account the operational cost of each carrier so there is a trade-off between the investment cost of transmission lines and the operational cost of generators used to produce the energy to satisfy the demand.

4.2 Small Network solved by D-Wave's simulators

We are considering a single snapshot of a three-node fully-connected network with nonexistent lines with a single demand at node 2 and without the load shedding term. Furthermore, each node has a single generator g_k with a nominal power – maximum capacity – of 10 energy units.

Symbol	Description	Value
$\mathbf{c} = [c_{12}, c_{13}, c_{23}]$	Investment cost	$[10, 20, 30]$
$\mathbf{c}^{\text{oc}} = [c_1^{\text{oc}}, c_2^{\text{oc}}, c_3^{\text{oc}}]$	Operational cost	$[10, 5, 2]$
$\mathbf{d} = [d_1, d_2, d_3]$	Demands	$[0, 12, 0]$
$\bar{\mathbf{g}} = [\bar{g}_1, \bar{g}_2, \bar{g}_3]$	Maximum capacity	$[10, 10, 10]$
$\bar{\mathbf{f}}^1 = [f_{12}^1, f_{13}^1, f_{23}^1]$	Maximum Power Flow in candidate line	$[10, 10, 10]$

Table 4.2: Investment cost, operational cost, demand, maximum capacity per generator and maximum power flow per candidate line for a three node network.

We use synthetic data for our TEP problem described in Table 4.2.
The objective function for the three-node fully-connected network described above is

$$\min_{\mathbf{x},\mathbf{g}} \quad \underbrace{\mathbf{C}^\intercal \mathbf{x}}_{\text{Investment Cost}} + \underbrace{\mathbf{C}_{\text{oc}}^\intercal \mathbf{g}}_{\text{Operational Cost}} \tag{4.3a}$$

$$\text{s.t.} \quad g_k \leq \bar{g}_k \qquad \forall\, k \in N, \tag{4.3b}$$

$$d_k = g_k + \underbrace{\sum_{l \neq k, l \in N} g_k x_{kl}}_{\text{Quadratic Constraint}} \qquad \forall\, k \in N, \tag{4.3c}$$

$$\mathbf{d}, \mathbf{g} \geq 0\,, \tag{4.3d}$$

$$x_{kl} \in \{0, 1\}, \qquad \forall\, kl \in C. \tag{4.3e}$$

Notice that we are not considering the power flow limits f_{kl}^1 of candidate lines. Any candidate line can transmit any amount of energy. In other words, we wrote the problem in such a way that none of the power flow limits is going to be violated, see Figure 4.2.

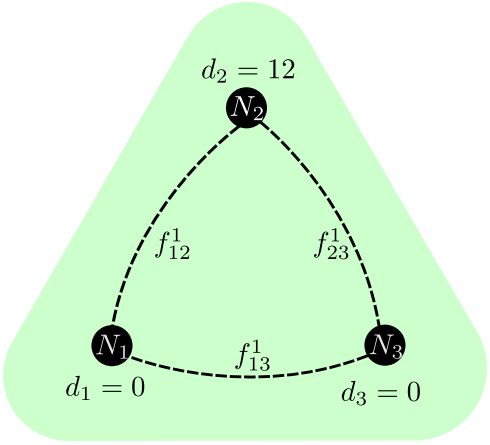


Figure 4.2: Graphic formulation of three node problem.

The demand at node 2 is 12 energy units, so it cannot be fulfilled by its own generator and requires energy from at least one of the neighboring nodes which means we need to build at least one transmission line connected to node 2.

4.2.1 D-Wave ExactSolver

Since the problem we are tackling has just a few variables we can sample all the configuration space. For this task we use the *ExactCQMSolver* class. One of the advantages of CQM is that it writes down if the solution is feasible, i.e., if that solution satisfies all the constraints of our problem. The solution of the problem is represented graphically in Figure 4.3.

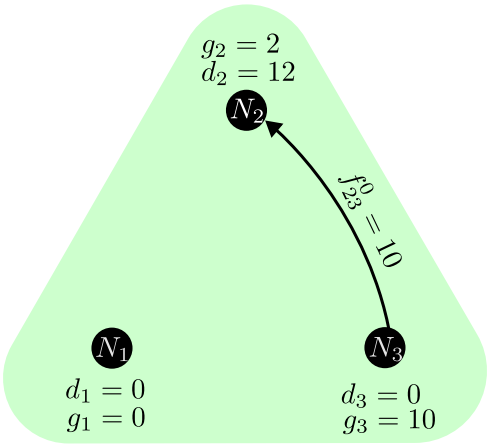


Figure 4.3: Graphical solution of three node problem.

The Table 4.3 represents only feasible solutions which are sorted from lower energy to higher energy. The first row is the solution that minimizes the objective function by fulfilling the constraints, i.e., the demand.

x_{12}	x_{13}	x_{23}	g_1	g_2	g_3	Prices	Feasible
0	0	1	0	2	10	60	True
0	0	1	0	3	9	63	True
0	0	1	0	4	8	66	True
0	0	1	0	6	7	69	True

Table 4.3: D-Wave's feasible solutions to the TEP combinatorial optimization problem.

The optimal solution decides to build the most expensive transmission line, i.e., the one connecting nodes 2 and 3, and to use the maximum capacity of the cheapest generator, the one at node 3, see Table 4.2.

4.2.2 D-Wave LeapHybridCQMSampler

The complete implementation of TEP problems – taking into account integer or real variables – requires of hybrid solvers. For this reason, we implemented the TEP problem from (4.2) in Python and used a D-Wave hybrid solver. We would expect the solution to the problem to be the same as above. However, in order to exploit the capabilities of the hybrid solver, we have taken into account every variable of the TEP problem which were not considered in the previous section. This is why Table 4.4 displays more columns than Table 4.3.

x_{12}	x_{13}	x_{23}	f_{12}	f_{13}	f_{21}	f_{23}	f_{31}	f_{32}	g_1	g_2	g_3	r_1	r_2	r_3	**Prices**
0	0	1	0	0	0	10	0	0	0	2	10	0	0	0	60
1	1	0	0	10	10	0	0	0	0	2	10	0	0	0	60

Table 4.4: D-Wave's feasible solutions to the TEP combinatorial optimization problem.

Notice that there are two solutions with the same cost. The first one was shown in the previous section. However, the new solution produced by the hybrid solver builds two lines whose accumulated cost is the same as the one connecting nodes 3 and 2. Taking into account the complete formulation of the problem and not just a simplified version allows us to get new solutions that otherwise would not appear. In this case, the complete formulation of the problem allows the current to flow between nodes until it reaches the destination node.

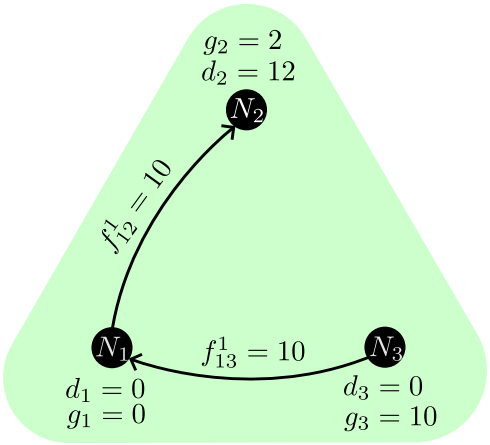


Figure 4.4: Graphical solution of three node problem.

Execution Times

Last, we run ten tests for a different number of nodes in a network with a D-Wave hybrid solver known as *LeapHybridCQMSampler*. Although the execution time of the hybrid solvers can be set, the LeapHybridCQMSampler will calculate a minimum time based on the size of our problem. As we can observe, the execution times are nearly identical for any number of nodes Figure 4.5. This is because the execution time limit is set internally. Although it can be modified, we were not able not perform as many trials as desired because of the limitations of Leap's free account. However, it would be more interesting to consider other quantities such as time-to-solution [31], which will surely increase with the number of nodes. This will be the subject of future work.

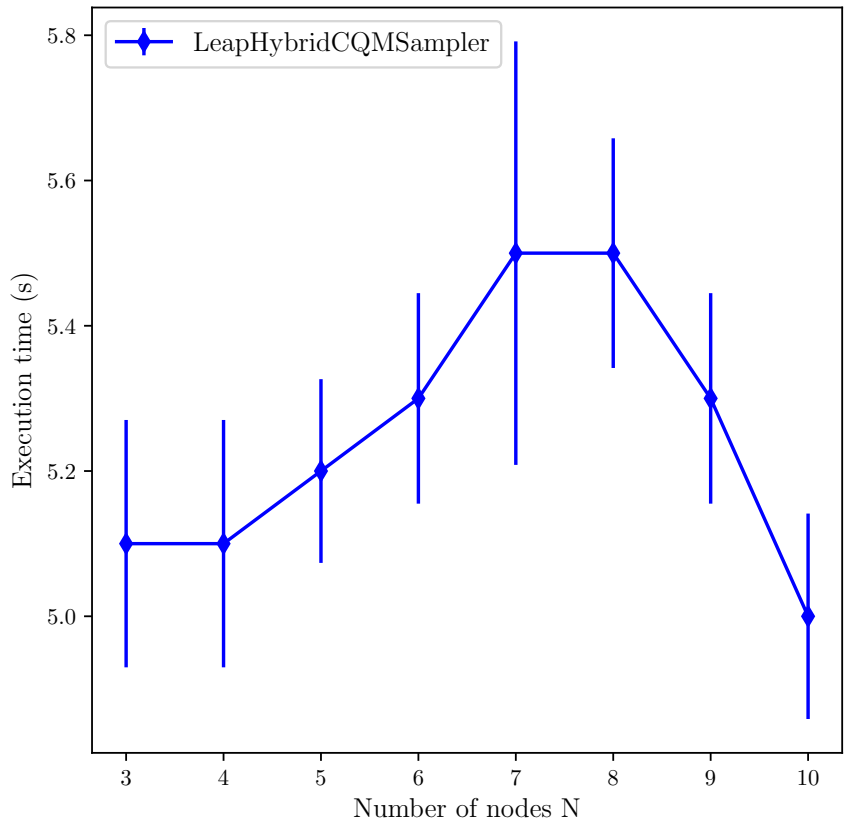


Figure 4.5: Execution times for different test cases of small networks with D-Wave *LeapHybridCQMSampler*.

4.3 Benders Decomposition in TEP

In Chapter 3 we showed three hybrid approaches that combine classical and quantum solvers. In this work, we decided to explore the Benders' decomposition approach due to two main points. Firstly, the convergence properties of Benders' decomposition allows us to know how good a solution is. Secondly, TEP problems have diagonal structure which means they are suitable for a Benders' decomposition approach. We propose a quantum Benders' decomposition scheme for a general formulation of TEP problems, but its implementation is left for a future work. However, the results of Figure 4.5 will be compared in the future with the results we get from our quantum BD algorithm.

In order to simplify the problem we consider a single snapshot so that we can drop the h index. As discussed in Ch. 3, we can formulate the problem by maximizing a dual function or by minimizing the primal problem with the dual as a penalty. The Benders' decomposition algorithm that we propose consists of four steps:

- **Step 1** (*Initialization of Benders' decomposition*)**:** we initialize the iteration counter t, the upper bound $\bar{z}^0$, the lower bound $\underline{z}^0$, the load shedding and operational cost α^0 and the penalty term associated with the violation of power flow constraints of candidate lines $\Pi_{f_{kl}^1}$.

- **Step 2** (*Master problem solved by a quantum annealer*)**:** we increase the lower bound $\underline{z}^t$ by taking the minimum value of

$$\min_{\mathbf{x}} \quad \underline{z}^t \tag{4.4a}$$

$$\text{s.t.} \quad \underline{z}^t \geq \sum_{kl \in C} c_{kl} x_{kl}^t + \alpha^{t-1} - \underbrace{\sum_{kl \in C} \Pi_{f_{kl}^1}^{\tau} \left(x_{kl}^t - x_{kl}^{\tau} \right)}_{\text{Benders' cuts}} = 0, \quad \forall \tau = 1, \ldots, t-1. \tag{4.4b}$$

In that equation α^{t-1} represent the summation of the objective function of the slave problem(s) and $\Pi_{f_{kl}^1}^{\tau}$ represent the Benders' cut of iteration τ where $\tau < t$, for the linking constraint of Eq. (4.2d). Notice that we have a Benders' cut for every candidate circuit flow constraint f_{kl}^1. In every iteration of Benders' algorithm we are adding cuts, in other words, we are restricting the region of feasible solutions.

- **Step 3** (*Slave problem solved by a classical solver*)**:** we minimize α^t which includes the load shedding cost and the operational cost.

$$\min_{\mathbf{g}, \mathbf{r}, \mathbf{f}^0, \mathbf{f}^1} \quad \alpha^t \equiv \underbrace{\sum_k c_k^{(\text{oc})} g_k}_{\text{Operational cost}} + \underbrace{\sum_k r_k c_k}_{\text{Load shedding cost}} \tag{4.5a}$$

$$\text{s.t.} \quad d_k - \left(\sum_{l \in E_k} f_{kl}^0 + \sum_{l \in C_k} f_{kl}^1 + g_k + r_k \right) = 0, \quad \forall k \in N, \tag{4.5b}$$

$$\left| f_{kl}^0 \right| - \bar{f}_{kl}^0 \leq 0, \qquad \forall\, kl \in E, \tag{4.5c}$$

$$\left| f_{kl}^1 \right| - \bar{f}_{kl}^1 x_{kl}^t \leq 0, \qquad \forall\, kl \in C, \tag{4.5d}$$

$$g_k - \bar{g}_k \leq 0, \qquad \forall\, k \in N, \tag{4.5e}$$

$$r_k - d_k \leq 0, \qquad \forall\, k \in N, \tag{4.5f}$$

$$\mathbf{d}, \mathbf{g}, \mathbf{r} \geq 0\,. \tag{4.5g}$$

- **Step 4** (*Stopping criterion*)**:** after minimizing α^t we reduce the upper bound

$$\bar{z}^t = \min\{\bar{z}^{t-1}, \sum_{kl \in C} c_{kl} x_{kl}^t + \alpha^t\}. \tag{4.6}$$

If the difference between the upper and lower bound is less or equal than a given threshold

$$\bar{z} - \underline{z} \leq \epsilon, \tag{4.7}$$

then the Benders' algorithm stops and outputs the value of $\{\mathbf{x}, \mathbf{g}, \mathbf{r}, \mathbf{f^1}\}$ as the (sub-)optimal solution. Otherwise, it goes to step 2 with the new candidate solution as input.

CHAPTER 5

Conclusions and Future Work

5.1 Conclusions

The present work provides a well-known iterative scheme – classical Benders' decomposition – to solve large combinatorial optimization problems. Specifically, we provided an implementation of BD that combines quantum and classical solvers. We formulated a general TEP problem by taking into account investment cost, load shedding cost and operational cost and split the original problem into a master problem in QUBO formulation, tackled by the quantum solver, and a slave problem, tackled by the classical solver. The quantum solver is a quantum annealer and the classical solver is any of the well-known classical methods such as simplex or the interior point method to solve the slave problem. Benders' decomposition has convergence properties so that we could control how good a solution to a combinatorial problem is by comparing the difference between an upper and a lower bound. Furthermore, we could control the degree to which the quantum solver is employed in the algorithm, opening up the possibility of increasing the complexity of the problem assigned to the quantum solver according to the hardware evolution. Currently, hybrid approaches, such as the ones from D-Wave, provide the solution to large combinatorial optimization problem where there are binary, integer and even real variables involved. However, we cannot know how much quantum solver we used in the resolution of our problem. The problems solved in this thesis with those hybrid solvers are intended to be the test cases for the implementation of the quantum BD algorithm, which is left for future work. It is our belief that a specific hybrid implementation would solve the problem faster and better than a general hybrid approach as that of D-Wave.

The potential speed-up would help to close the granularity gap between what the current models can solve and the desired solutions needed by energy system operators. Tackling real-time problems by using the smart grid data is still far from being solved efficiently by classical and quantum solvers but iterative methods open a new path to explore the resolution of large problems in shorter times by gaining speed-up from the quantum solvers.

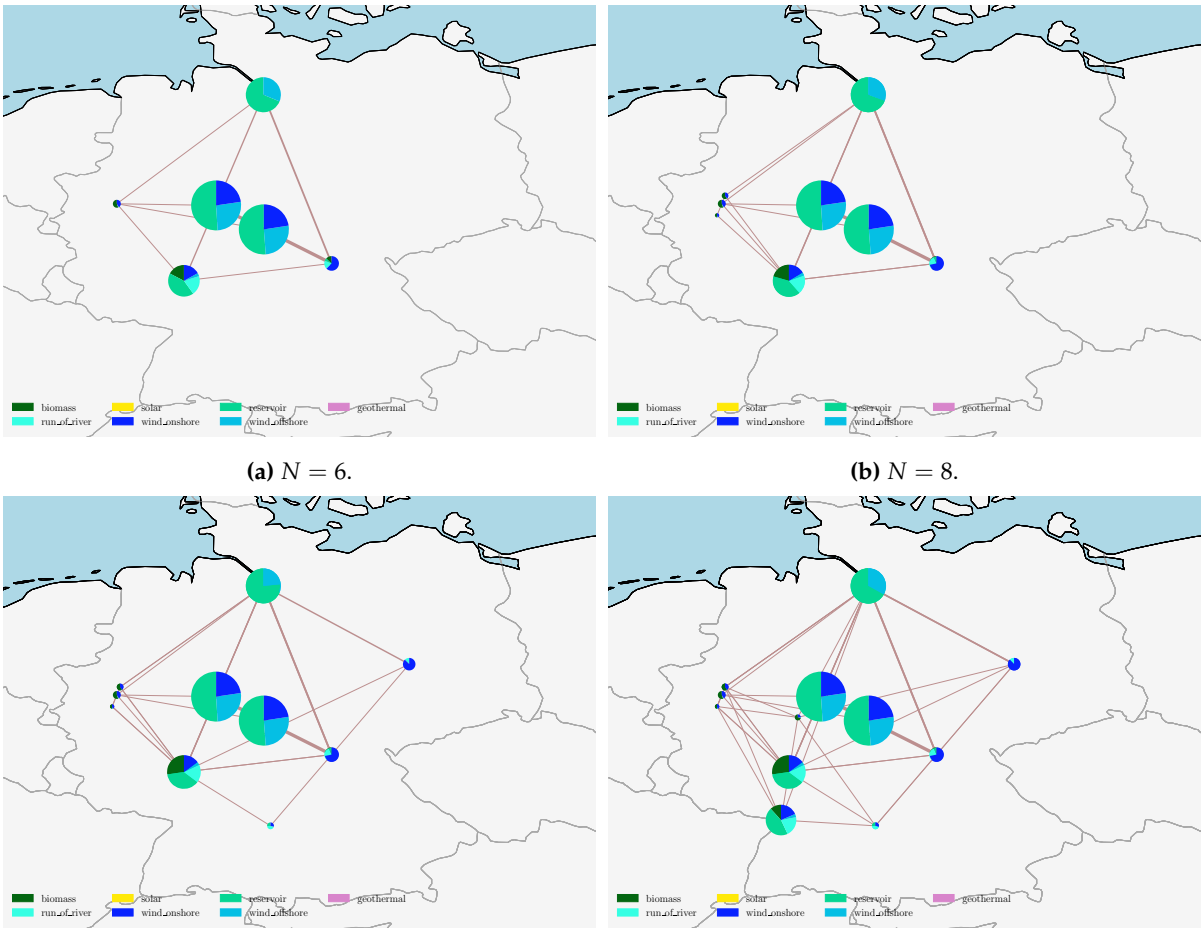
This work aims to show the challenges of applying hybrid quantum-classical methods to energy sector problems. It is our belief that, although quantum solvers are still in their early stages, hybrid approaches will allow researchers and different industry players to unlock the potential of current quantum solvers.

5.2 Future Work

This master thesis has presented an iterative approach to solve TEP problems. We plan to implement the quantum BD in Python and use real data from different sources of energy models to develop the test cases. Depending on the source from which we load the data we need to

process it accordingly. We will use the PyPSA python package [4] to take care of that issue.

Dealing with real data implies not only dealing with real numbers – that can be rounded to integers – but also with a large number of variables which increase the complexity of the problem. These points do not change the implementation, but if we plan to add new components such as storages – which implies taking into account a multi-snapshot dependence – the linear equations that describe our problem are going to change. PyPSA can simplify the linear model for any number of targets considered in the original problem and to plot the associated network for a given number of nodes N, as in Figure 5.1. This enables us to read these linear equations and include them into our Benders' decomposition scheme. For this task we plan to use QUARK [32], a reformulation kernel that allows us to send the algorithm to different quantum machines.



(a) $N = 6$.

(b) $N = 8$.

(c) $N = 10$.

(d) $N = 12$.

Figure 5.1: Germany network generated with PyPSA for N clusters from eGo100 data [33].

Last, we could compare our hybrid quantum-classical algorithm with the ones provided by the quantum solvers. We think that a specific hybrid quantum-classical approach should provide a better solution in terms of computational time and optimal value than the general hybrid solvers from D-Wave. Furthermore, we should be able to determine the exact usage of quantum solver we are employing in the problem, which is not possible with solvers such as D-Wave.

APPENDIX A

Introduction to Quantum Computing

This appendix serves as a quick introduction to quantum computing. We explain the basic concepts without going deeply into their physical meaning or historical approach. For a deeper discussion of the topics described herein, we refer the reader to Refs. [34], [35] and [36].

A.1 Hilbert space

We start by defining the vector space where quantum computing takes place, the Hilbert space, $\mathbb{H}$.

Definition A.1: Hilbert Space

A Hilbert space is a special type of linear vector space whose elements are complex-valued *square integrable*[a] functions, $\psi(x)$, of a real variable x, defined on the closed interval $[a, b]$, equipped with a *complete inner product*, $(\cdot, \cdot)$, in $\mathbb{C}$.

[a] A function $\psi(x)$ is said to be *square integrable* on a given interval $[a, b]$ if $\int_a^b \|\psi(x)\|^2 dx$ exist with a finite value.

Definition A.2: Inner Product

Given two functions of the Hilbert space, ψ_1 and ψ_2, the inner product is defined by

$$(\psi_1, \psi_2) \equiv \int_a^b \psi_1^*(x)\psi_2(x)dx. \tag{A.1}$$

Corollary A.1

Given the functions $\{\psi_1,\psi_2,\psi_3\} \in \mathbb{H}$ and $\{\alpha,\beta\} \in \mathbb{C}$, the inner product of the associated Hilbert space satisfies:

- Closed operation: $(\psi_1,\psi_2) \in \mathbb{C}$.
- Conjugate symmetry: $(\psi_1,\psi_2) = (\psi_2,\psi_1)^*$.
- Linear with respect to the second vector: $(\psi_1,\lambda\psi_2+\beta\psi_3) = \lambda(\psi_1,\psi_2)+\beta(\psi_1,\psi_3)$.
- Anti-linear with respect to the first vector: $(\lambda\psi_1+\beta\psi_2,\psi_3) = \lambda^*(\psi_1,\psi_3)+\beta^*(\psi_2,\psi_3)$.
- Positive definiteness: $(\psi,\psi) = \|\psi\|^2 \in [0,\infty)$.[a]

[a] The quantity $\|\psi\|^2$ is called the norm of ψ. Notice that $\|\psi\|^2 = 0$ that does not imply $\psi(x) = 0$ for all x in $[a,b]$. The function can have nonzero values at some points and the integral will remain zero. The integral roughly computes the area in a given interval, so, if there is just a point not null in the interval the area captured by a point is zero, so its contributions to the integral is zero. In our case the quantity $\|\psi\|^2$ represents the probability density of a given state ψ, i.e., if we integrate the whole interval of definition $x \in D$, $\int_D \|\psi\|^2 dx = 1$.

Definition A.3: Distance

The distance defined by Hilbert space's inner product is given by

$$d \equiv |\psi_2 - \psi_1| = \sqrt{(\psi_2-\psi_1,\psi_2-\psi_1)}. \tag{A.2}$$

Definition A.4: Completeness of a space

A complete space is one in which any Cauchy sequence – of that space – is convergent, i.e., tends towards a value inside the given space.

We can show why completeness of space is a necessary condition. Suppose that our space is not complete, i.e., some Cauchy sequences are not convergent. Then, the evolution of an initial state – explained in later sections –, $|\psi(0)\rangle$, under a given constant Hamiltonian is not guaranteed. In other words, the following equation may not hold because it is not guaranteed that the Cauchy sequence $\sum_{i=0}^{n}(-it)^k/\left(k!h^k\right)\mathcal{H}^k$ converges.

$$|\psi(t)\rangle = e^{-\frac{i\mathcal{H}}{\hbar}t}|\psi(0)\rangle = \lim_{n\to\infty}\sum_{i=0}^{n}\frac{(-it)^k}{k!h^k}\mathcal{H}^k|\psi(0)\rangle\,. \tag{A.3}$$

Definition A.5: Completeness of an orthonormal set of functions

A set of orthonormal functions $\{\psi_i\}$ is complete if any function $\psi(x)$ in Hilbert space can be written as a linear combination of the $\psi_i(x)$:

$$\lim_{n\to\infty}\|\psi(x) - \underbrace{\sum_{i=1}^{n}c_i\psi_i(x)}_{\psi_n}\|^2 = 0\,. \tag{A.4}$$

Theorem A.1: Riesz-Fischer

Assume the functions $\psi_1(x), \psi_2(x), \ldots$ are elements of a Hilbert space. If

$$\lim_{n,m\to\infty} \|\psi_n - \psi_m\|^2 \equiv \lim_{n,m\to\infty} \int_a^b \|\psi_n - \psi_m\|^2 dx = 0, \tag{A.5}$$

then there exist a square (Lebesgue) integrable function $\psi(x)$ to which the sequence $\psi_n(x)$ converges such that

$$\lim_{n\to\infty} \int_a^b \|\psi - \psi_n\|^2 dx = 0. \tag{A.6}$$

Equivalently, let ψ_n be a Cauchy sequence and ψ a value inside the given space. Then, the Cauchy sequence converges to $\psi(x)$ in the mean[a].

[a] Here we do not ask for point convergence, we weaken the convergence criteria to converge in the mean, i.e., we allow the difference $\|\psi(x) - \psi_n(x)\|^2 \neq 0$ at some points x, so that there exists an $N(\epsilon)$ for which the integral $I = \lim_{n\to\infty} \int_a^b \|\psi - \psi_n\|^2 dx$ is arbitrarily close to zero $I < \epsilon$ for $n > N(\epsilon)$. Otherwise, there would not exist a complete set of orthonormal functions in the Hilbert space.

Definition A.6: Orthonormality

A given set of functions $\{\psi_i\}$ is said to be orthonormal if

$$(\psi_i, \psi_j) \equiv \int_a^b \psi_i^*(x)\psi_j(x)dx = \delta_{ij}. \tag{A.7}$$

In the present work we work with a particular Hilbert space, a finite Hilbert space. This implies our space is *separable*.

Definition A.7: Separable

A Hilbert space is said to be *separable* if an only if it has a countable orthonormal basis.

So for a finite Hilbert space – which is a countable space – an orthonormal basis is guaranteed.

A.2 Notation

Definition A.8: Dirac's Bra-Ket Notation

The inner product of an n-dimensional Hilbert space defines a linear map from $\mathbb{H}$ to $\mathbb{C}^n$

$$\begin{aligned} \tau : \mathbb{H} &\longrightarrow \mathbb{C}^n \\ \tau(\psi) &\longrightarrow \begin{bmatrix} \alpha_1 \\ \vdots \\ \alpha_n \end{bmatrix}. \end{aligned}$$

Conversely,

$$\begin{aligned} \bar{\tau} : \mathbb{H}^* &\longrightarrow \mathbb{C}^n, \\ \bar{\tau}(\psi) &\longrightarrow \left[\alpha_1^*, \ldots, \alpha_n^*\right], \end{aligned}$$

where $\mathbb{H}^*$ denotes the dual space. The dual space $\mathbb{H}^*$ is also a Hilbert space with the same dimension as $\mathbb{H}$.

- Elements of a Hilbert space $\mathbb{H}$ are called *ket-vectors*

$$\psi \equiv |\psi\rangle = \begin{bmatrix} \alpha_1 \\ \vdots \\ \alpha_n \end{bmatrix}. \tag{A.8}$$

- Elements of a dual Hilbert space $\mathbb{H}^*$ are called *bra-vectors*

$$\psi^* \equiv \langle\psi| = \left[\alpha_1^*, \quad \ldots \quad, \alpha_n^*\right]. \tag{A.9}$$

Notice that a bra-vector is just the transpose conjugate of a ket-vector. This is a useful way of mapping a ket-vector of a given Hilbert space into its bra-vector on the associated dual Hilbert space.

Corollary A.2

In bra-ket notation, a set of vectors $\{|\psi_j\rangle\}$ is said to span $\mathbb{H}$ if we can express any vector $|\psi\rangle$ of that space as a linear combination of the vectors in the given set

$$|\psi\rangle = \sum_j \alpha_j |\psi_j\rangle, \tag{A.10}$$

where the coefficients of the combination α_j are complex numbers. In particular, if the set of vectors $\{|\psi_j\rangle\}$ are linearly independent and the number of vectors in that set is equal to the dimension of our space, then this set of vectors is a basis set of our space and the previous expression is the so-called basis expansion of $|\psi\rangle$.

A.3 Quantum bits

A *bit* is the smallest unit of information of classical computing. Analogously, a *qubit* is the smallest unit of information for quantum computing. The following section treats bits and qubits as abstract mathematical objects without their physical implementation. In Appendix C,

we describe some implementations of a physical qubit.

A classical bit has two possibles states commonly named $|0\rangle$ or $|1\rangle$. Notice that the name of the states of a bit or qubit is not important. It is just a way of labelling those states. However, the state of a quantum bit is a linear combination of states – *superposition* – $\{|0\rangle, |1\rangle\}$. The general state of a qubit can be conceived as a vector in a two-dimensional complex vector space, with $\alpha, \beta \in \mathbb{C}$,

$$|\psi\rangle = \alpha |0\rangle + \beta |1\rangle = \alpha \begin{bmatrix} 1 \\ 0 \end{bmatrix} + \beta \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \tag{A.11}$$

where $\|\alpha\|^2 + \|\beta\|^2 = 1$, i.e, the state of a qubit is normalized.
The last expression can be written in terms of two parameters, θ and ϕ,

$$|\psi\rangle = \cos\frac{\theta}{2} |0\rangle + e^{i\phi} \sin\frac{\theta}{2} |1\rangle \tag{A.12}$$

that represent the angles of the Bloch Sphere. A Bloch sphere is a geometric representation of the state of a single qubit where each axis has two orthogonal states, e.g., the Z-axis has the orthogonal states $\{|0\rangle, |1\rangle\}$.
The states $|0\rangle$ and $|1\rangle$ form a computational basis (orthonormal basis) for the $\mathbb{C}^2$ vector space. There are infinite single-qubit basis sets for $\mathbb{C}^2$, albeit the ones known as *computational basis* are the ones that build the axis of the Bloch Sphere.

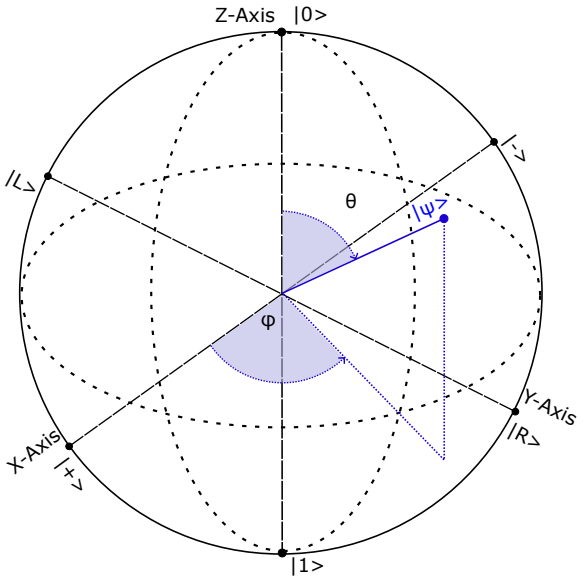


Figure A.1: A Bloch Sphere displaying the state $|\psi\rangle$ of a single qubit.

The Pauli matrices represent a rotation around a given axis $\{X, Y, Z\}$. Pauli matrices are generators of SU(2) group, so a complex exponential can be understood as a rotation around a vector $\vec{n}$ where $e^{i\frac{\theta}{2}(\vec{n}\cdot\vec{\sigma})} = \mathbb{I}\cos\frac{\theta}{2} + i(\vec{n}\cdot\vec{\sigma})\sin\frac{\theta}{2}$.The matrix representation is given by,

$$X \equiv \sigma_x = \sigma_1 = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix},$$

$$Y \equiv \sigma_y = \sigma_2 = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix},$$

$$Z \equiv \sigma_z = \sigma_3 = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}.$$

The eigenvectors of the Pauli matrices are the computational basis vectors,

$$\left\{\begin{bmatrix}1\\0\end{bmatrix},\begin{bmatrix}0\\1\end{bmatrix}\right\} \equiv \{|0\rangle, |1\rangle\} \in Z_{basis},$$
$$\left\{\begin{bmatrix}\frac{1}{\sqrt{2}}\\\frac{1}{\sqrt{2}}\end{bmatrix},\begin{bmatrix}\frac{1}{\sqrt{2}}\\-\frac{1}{\sqrt{2}}\end{bmatrix}\right\} \equiv \{|+\rangle, |-\rangle\} \in X_{basis},$$
$$\left\{\begin{bmatrix}\frac{1}{\sqrt{2}}\\\frac{i}{\sqrt{2}}\end{bmatrix},\begin{bmatrix}\frac{1}{\sqrt{2}}\\-\frac{i}{\sqrt{2}}\end{bmatrix}\right\} \equiv \{|R\rangle, |L\rangle\} \in Y_{basis}.$$

In general, we deal with multiple qubits. The basis for an n-qubit system is just the tensor product of n-single-qubit basis.
Suppose we have two qubits. Then, the $Z \otimes Z - basis$ is $\{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$ and the state of a two-qubit system is described by the linear combination

$$|\psi\rangle = \alpha_{00}|00\rangle + \alpha_{01}|01\rangle + \alpha_{10}|10\rangle + \alpha_{11}|11\rangle, \tag{A.13}$$

where $\|\alpha_{00}\|^2 + \|\alpha_{01}\|^2 + \|\alpha_{10}\|^2 + \|\alpha_{11}\|^2 = 1$, i.e., the state is normalized.
The normalization condition for an n-qubit system can be written as

$$\sum_{x\in\{0,1\}^n} \|\alpha_x\|^2 = 1, \tag{A.14}$$

where the expression $x \in \{0,1\}^n$ indicates all the possible tensor product combinations of the n-single-qubit states. These combinations form a basis for the n-qubit system.

A.4 Measurements and Operators

Quantum Mechanics makes predictions about microscopic objects taking into account their statistics by measuring in an ensemble of equally prepared states which produces quantities distributed around a mean value with a given frequency. The predictions have implications for the macroscopic world. In classical computing a system is mostly unaltered by tiny interactions such as light or heat. However, in quantum computing this environmental noise is quite important as it modifies the state of the system. This noise is a doubled-edged sword as it can be used to modify intentionally the state of a quantum system.

Definition A.9: Operator

An operator $\hat{A}$ is a linear map between Hilbert spaces that satisfy,

$$\langle \hat{A}^\dagger \psi | \varphi \rangle = \langle \psi | \hat{A} \varphi \rangle \; \forall \psi, \varphi \in \mathbb{H} \tag{A.15}$$

where $\hat{A}$ admits a matrix representation and † means transpose conjugate

Corollary A.3

Given an operator $\hat{A}$, on a finite Hilbert space, the operator is said to be Hermitian if

$$\hat{A}^\dagger = \hat{A}. \tag{A.16}$$

Hermitian operators play an important role in quantum physics because their eigenvalues are real and represent measurable physical quantities.

Definition A.10: Observable

Given the state of a system $|\psi\rangle$, an *observable* – represented with an operator $\hat{A}$ – is the physical quantity we can measure associated with the Hermitian operator $\hat{A}$. The possible observables of an operator are its eigenvalues.

As an example consider the *Hamiltonian* of a system, $\hat{\mathcal{H}}$. For the present work, the Hamiltonian of a system represents the total energy of the system. So, the associated eigenvalues are the eigenenergies of the system.

Definition A.11: Unitary Operator

An operator U on $\mathbb{H}$ is unitary if

$$U^\dagger U = \mathbb{I}. \tag{A.17}$$

where $\mathbb{I}$ is the identity operator.

Definition A.12: Expectation Value

The expectation value, $< \cdot >$, of an observable is the mean value we get after a sequence of measurements of that observable in an ensemble of equally prepared states. Mathematically,

$$\langle \hat{A} \rangle_{|\psi\rangle} \equiv \langle \psi | \hat{A} | \psi \rangle \,. \tag{A.18}$$

We can measure a classical bit to check if it is in the state 0 or 1. However, when we measure – in the Z-basis – a quantum bit we do not get the parameters α, β that describe the qubit state. Instead, we get either $|0\rangle$ or $|1\rangle$ with probabilities $\|\alpha\|^2$ and $\|\beta\|^2$ respectively. The logic of classical computing is Boolean, which means that if the system is not in the state 0 it must be in state 1. In quantum computing, if the system is not in the state 0 it does not have to be in state 1. In a measurement, the general state of the qubit collapses to one of the states of the basis we are using to measure. After this measurement the state of our qubit is fixed and successive measurements will give the same state with probability 1, i.e., measurements collapse a qubit into one of the basis states destroying the superposition.
Quoting Nielsen and Chuang [36],

> *This dichotomy between the unobservable state of a qubit and the observations we can make lies at the heart of quantum computation and quantum information.*

A.5 Schrödinger Equation

One of the paradigms of universal quantum computing is the gate model. This approach substitutes classical logic gates – such as OR, XOR, AND or NOT – by its quantum analog. These quantum gates are represented by unitary matrices, so that the inner product is preserved.

Equivalently, a quantum gate is represented by a complex exponential of the form $\exp\left(-i\hat{\mathcal{H}}t/\hbar\right)$ where a given Hamiltonian, $\mathcal{H}$ – controlled by external fields such as magnetic fields – leads the evolution of the qubit in such a way that the associated matrix of the complex exponential – that admits a series expansion – match the matrix representation of the quantum gate.

Precisely, the evolution of a quantum system is governed by the Schrödinger equation which cannot be derived from first principles, i.e., it is an experimental fact. Mathematically, it can be expressed as

$$\hat{\mathcal{H}}\left|\psi(t)\right\rangle = i\hbar\frac{\partial}{\partial t}\left|\psi(t)\right\rangle, \tag{A.19}$$

where $\hat{\mathcal{H}}$ is the Hamiltonian operator, $\hbar$ is the reduced Planck constant and $|\psi(t)\rangle$ represents the state of a system. The Schrödinger equation governs the time evolution of a quantum system. It plays the same role that Newton's equations do in classical mechanics.

A.6 Speed-up Advantage

We end up this appendix by discussing the main features of quantum behavior that provide a speed-up for some algorithms versus their classical approach.

The main power of quantum computers can be summarized in three properties,

- **Superposition:** a qubit can be a in a linear combination of states. A quantum memory register can exist in a superposition of states. If we perform an operation on this quantum register, this operation is applied to all states of that superposition. Notice that we have applied an operation to all the states in that superposition by performing the function just once.

- **Entanglement:** an entangled state is a state that cannot be written in term of the tensor product of pure states, i.e., states whose trace $Tr(\rho^2) = 1$, where ρ is the density matrix. Entangled states store information exponentially instead of linearly (as classical computing does). As an example, suppose we have the following two-qubit state, called *Bell state*

  $$|\psi\rangle = \frac{1}{\sqrt{2}}\left(|00\rangle + |11\rangle\right). \tag{A.20}$$

  If we measure the first qubit in the Z-basis, we can get either the state $|0\rangle$ or $|1\rangle$. We can see that both qubits are entangled in the sense that knowing the state of the first qubit allows us to determine the state of the second qubit. In this case, if the first qubit measurement is $|0\rangle$ we know that the second qubit must be in state $|0\rangle$.[1]

- **Interference:** we can build quantum algorithms that cross out those terms that are not interesting for us and increase the amplitudes of the ones we are interested in.

Even though the advantage of using quantum computing with respect to classical computing has been proven theoretically for some specific problems, see Ref. [37], the real world problems are yet far away from being addressed fully by quantum computing. The best we can do nowadays is to solve problems by splitting them into a quantum part that can be done by a quantum computer and a classical part that is solved by using an HPC.

[1] Einstein termed this effect *spooky action at a distance* in an attempt to ridicule quantum mechanics.

A.6.1 Grover's Algorithm

In this section we show the advantages of quantum computing by considering a specific example, that of Grover's algorithm.
Grover's algorithm is a search algorithm that uses qubits in superposition adjusting their phases to make the computations by quantum parallelism. Given a table with a number of items N, we are interested in a particular item ω. The best known classical algorithm can solve the problem in $O(N)$ complexity, which means that in the worst case a classical algorithm has to check each entry of the data-set until finding the item ω in the last query. Grover's algorithm can solve the problem in $\sim O(\sqrt{N})$ complexity, which represent a quadratic speed up compared with the classical approach, i.e., after $\sqrt{N}$ evaluations it is guaranteed to provide the desired state.

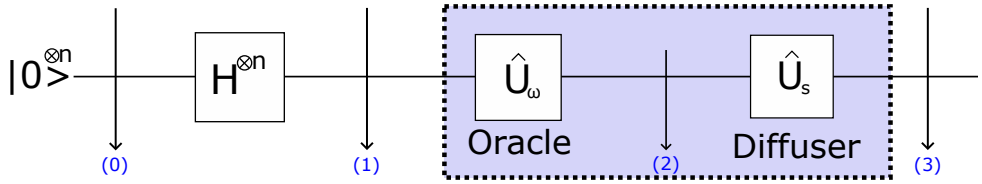


Figure A.2: Circuit scheme of Grover's algorithm for an n-qubit system.

Grover's Algorithm Steps for a Two-qubit System ($N = 4$)

In this section we show what is the state of the system at each step – after the application of the quantum gates – and a geometrical representation based on Ref. [38].

Binary Index	00	01	10	11
Item	0	1	2	3

Table A.1: Grover's search example. Different items described by its binary expansion.

As an example consider the Table A.1, and suppose we want to find the item 2, where the index – binary expansion of decimal index starting from zero – represents the eigenstate of the system we are looking for. Then, if we are interested in finding the item 2, we are looking for the state $|\omega\rangle = |10\rangle$.

(0)Initialize single qubit states: all the qubits are initialized at $|0\rangle$ so the state of the system is given by the tensor product of the qubits,

$$|\psi_0\rangle = |0\rangle \otimes |0\rangle = |00\rangle\,. \tag{A.21}$$

(1)Equiprobability: apply a Hadamard gate to each qubit so we create a configuration of equiprobable states. The reason for this is that at the beginning we do not have a good guess of where the item can be so each possibility is equally probable,

$$|\psi_1\rangle = \hat{H}^{\otimes 2}\,|00\rangle = \frac{1}{\sqrt{N}} \sum_{x\in\{0,1\}^2} |x\rangle \equiv |s\rangle\,. \tag{A.22}$$

One can label the states using decimal notation, $i = 0, \ldots, 2^n - 1$, where n is the number of qubits.
For a two-qubit system,

$$\{|x\rangle\} = \{|00\rangle\,, |01\rangle\,, |10\rangle\,, |11\rangle\} \equiv \{|0\rangle\,, |1\rangle\,, |2\rangle\,, |3\rangle\} = \{|i\rangle\}. \tag{A.23}$$

Therefore, we can re-write the state $|s\rangle$ as

$$|s\rangle = \frac{1}{\sqrt{N}} |\omega\rangle + \frac{1}{\sqrt{N}} \sum_{i \neq \omega} |i\rangle . \tag{A.24}$$

Due to the orthonormality of the basis, $|\omega\rangle$ is orthogonal to $\sum_{i \neq \omega} |i\rangle$. So, we can write

$$|s\rangle = \sin\theta \, |\omega\rangle + \cos\theta \, |s'\rangle , \tag{A.25}$$

where

$$\sin\theta = \frac{1}{\sqrt{N}}, \qquad |s'\rangle = \frac{1}{\cos\theta} \frac{1}{\sqrt{N}} \sum_{i \neq \omega} |i\rangle . \tag{A.26}$$

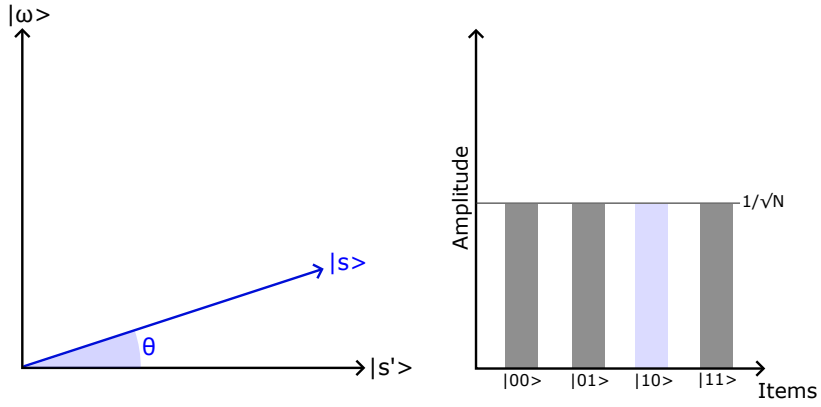


Figure A.3: Amplitude distribution of states after applying a Hadamard gate. The vector $|\omega\rangle$ is our desired state and $|s'\rangle$ is a orthogonal vector. In this way we can write $|s\rangle = \sin\theta \, |\omega\rangle + \cos\theta \, |s'\rangle$.

(2)Reflection about the state $|s'\rangle$: apply an operator $\hat{U}_\omega$ defined by

$$\begin{cases} \hat{U}_\omega |\omega\rangle = - |\omega\rangle \\ \hat{U}_\omega |i\rangle = |i\rangle , \qquad \forall |i\rangle \neq |\omega\rangle \end{cases} \tag{A.27}$$

so the operator can be written as,

$$\hat{U}_\omega = \mathbb{I} - 2 |\omega\rangle \langle\omega| . \tag{A.28}$$

Applying the operator to the state $|s\rangle$ yields,

$$|\psi_2\rangle = \hat{U}_\omega |s\rangle = |s\rangle - \frac{2}{\sqrt{N}} |\omega\rangle \equiv |\bar{s}\rangle . \tag{A.29}$$

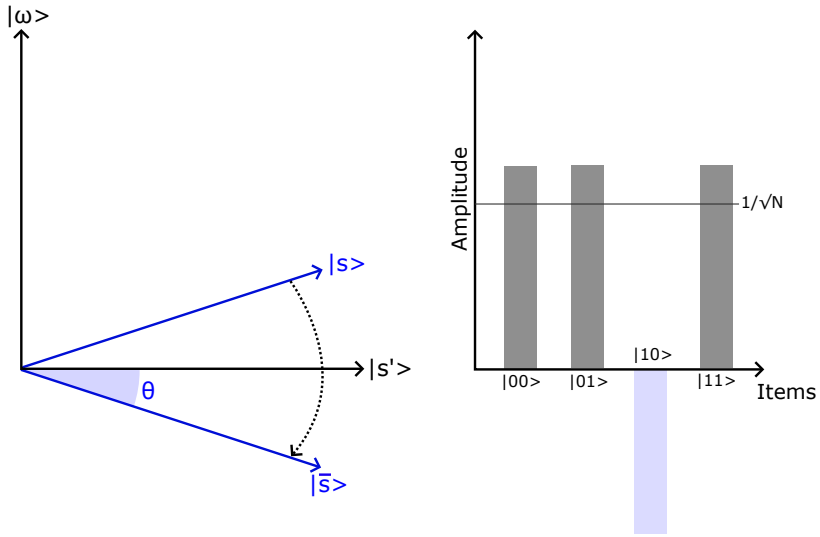


Figure A.4: Amplitude distribution of states after applying a U_ω. Notice that we have applied a phase to the state we are looking for. Geometrically this means we have applied a reflection about a state orthonormal to $|\omega\rangle$.

(3)Reflection about the state $|s\rangle$: the operator $\hat{U}_s$ is defined as,

$$\hat{U}_s \equiv 2\,|s\rangle\,\langle s| - \mathbb{I}. \tag{A.30}$$

Applying the operator to the state of our system yields,

$$|\psi_3\rangle = \hat{U}_s\left(|s\rangle - \frac{2}{\sqrt{N}}\,|\omega\rangle\right) = (2\,|s\rangle\,\langle s| - \mathbb{I})\left(|s\rangle - \frac{2}{\sqrt{N}}\,|\omega\rangle\right) = \frac{N-4}{N}\,|s\rangle + \frac{2}{\sqrt{N}}\,|\omega\rangle$$
$$= \frac{1}{N\sqrt{N}}\left[(N-4)\sum_{x\neq\omega}|x\rangle + (3N-4)\,|\omega\rangle\right] \overset{N=4}{=} \frac{1}{8}\left[0\cdot\sum_{x\neq\omega}|x\rangle + 8\,|\omega\rangle\right] = |\omega\rangle\,.$$

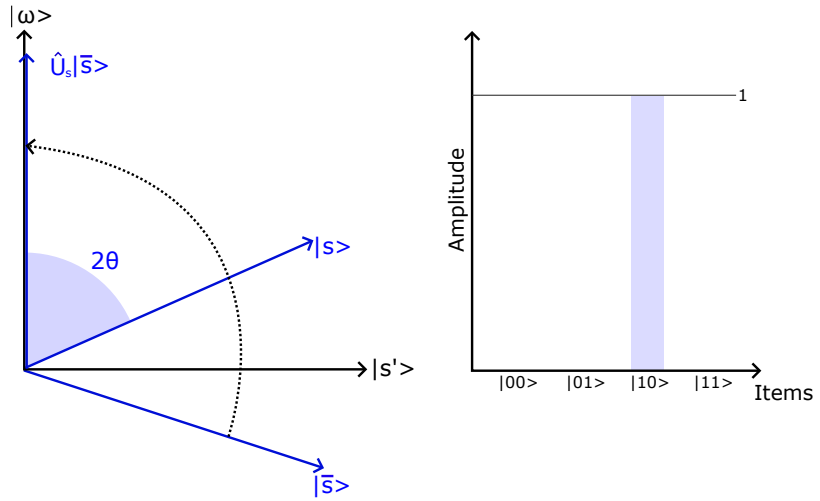


Figure A.5: Amplitude distribution of states after applying $\hat{U}_s$. For the case of two qubits, the amplitudes of non-desired states go to zero so we get the desired state $|\omega\rangle$ with a 100% of probability in an ideal quantum computer. This does not happen if the number of qubits is increased. In that case the amplitudes of non-desired states are reduced compared with the amplitude of the state we are looking for.

We have rotated the original state of our system towards the eigenvector $|\omega\rangle$ that represents the item we are looking for in the data-set. The closer we want to be to $|\omega\rangle$ the more times we have to repeat (2) and (3), so that we increase the probability of getting the desired state $|\omega >$ when we measure the system. The quadratic speed up comes from the fact that Grover's algorithm raises the probability amplitude of the item we are looking for by a factor of $\frac{1}{\sqrt{N}}$.

APPENDIX B

Simulated Annealing

This appendix justifies the name of the quantum approach we use in the present work – *Quantum Annealing* (QA) – to solve *Quadratic Unconstrained Binary Optimization* (QUBO) problems, see Chapter 2. In the next sections, we show that the expressions we get in the classical approach known as *Simulated Annealing* (SA), have the same functional form as the ones we get with QA [8]. To do that we are going to develop the solution for a stochastic problem. For a better understanding about stochastic problems, see Refs. [39] and [40].

B.1 Master Equation

A master equation is used to characterize the time-evolution of a given system that switches between states according to a transition rate for a given distribution. These switches between states are going to generate a new configuration that would be chosen as the current configuration if the energy difference between the current configuration and the new one is negative, $\Delta E = E_{\text{new}} - E_{\text{current}} < 0$. However, if $\Delta E > 0$, there is still a possibility for the new configuration to be chosen given by a probability distribution $P(\Delta E, T)$ that depends on the energy difference and the temperature T of the system.

B.1.1 Discrete Processes

Suppose we have a space of states $\Gamma = \{\uparrow\uparrow, \downarrow\uparrow, \uparrow\downarrow, \downarrow\downarrow\} \equiv \{1, 2, 3, 4\}$ and a discrete stochastic process, e.g., $\{X_t, t = 0, 1, 2, ...\} \to \{1, 4, 2, 3, 1, ..\}$. Furthermore, assume that the actual setting of our system only depends on the previous one, i.e., the probability of being in state j at time $t + 1$, $X_{t+1} = j$, given that the current state is i, $X_t = i$, does not depend on previous configurations. Mathematically,

$$P(X_{t+1} = j | X_t = i) = P(X_{t+1} = j | X_t = i, X_{t-1} = i_{t-1}, ..., X_0 = i_0) \ . \tag{B.1}$$

The equation Eq. (B.1) is known as **Markov chain** condition and the conditional probabilities are named *transition probabilities*.
The normalization condition for probabilities is also fulfilled, i.e., starting from a state i any state j – where j can be also the current state – has a transition probability such that

$$\sum_j P(X_{t+1} = j | X_t = i) = 1, \quad \forall i \ . \tag{B.2}$$

Theorem B.1: Total Probability

Consider an event A in a discrete set of events $\{B_i\}$, the probability of occurrence of event A is given by,

$$P(A) = \sum_i P(A|B_i)\, P(B_i) \ , \tag{B.3}$$

where $P(A|B_i)$ is the probability of occurrence of event A given that event B_i has already happened.

Renaming variables,

$$\Pi_i(t) \equiv P(X_t = i) \tag{B.4}$$

$$p_{i\leftarrow j} \equiv P(X_{t+1} = i | X_t = j) \ . \tag{B.5}$$

From equations Eq. (B.4) and Eq. (B.5), we can write,

$$\Pi_i(t+1) = p_{i\leftarrow i}\Pi_i(t) + \sum_{j\neq i} p_{i\leftarrow j}\Pi_j(t) \tag{B.6}$$

$$p_{i\leftarrow i} = 1 - \sum_{j\neq i} p_{j\leftarrow i} \ . \tag{B.7}$$

Combining the last two equations,

$$\Pi_i(t+1) = \left[1 - \sum_{j\neq i} p_{j\leftarrow i}\right] \Pi_i(t) + \sum_{j\neq i} p_{i\leftarrow j}\Pi_j(t), \tag{B.8}$$

$$\Pi_i(t+1) - \Pi_i(t) = -\Pi_i(t) \sum_{j\neq i} p_{i\leftarrow j} + \sum_{j\neq i} p_{i\leftarrow j}\Pi_j(t), \tag{B.9}$$

$$d\Pi_i(t) \equiv \Pi_i(t+1) - \Pi_i(t) = -\Pi_i(t) \sum_{j\neq i} p_{i\leftarrow j} + \sum_{j\neq i} p_{i\leftarrow j}\Pi_j(t) \ . \tag{B.10}$$

we get the master equation of a Markov stochastic discrete process.

B.1.2 Continuous Processes

In the case where we deal with a continuous process[1] we need to work with transition rates defined by

$$\mathcal{L}_{i\leftarrow j} = \begin{cases} \mathcal{L}_{i\leftarrow j} & i \neq j \\ -\sum_{k\neq i} \mathcal{L}_{k\leftarrow i} & i = j \ , \end{cases} \tag{B.11}$$

so that the master equation can be written as

$$\frac{d\Pi_i(t)}{dt} = \sum_j \mathcal{L}_{i\leftarrow j}\Pi_j(t) \ . \tag{B.12}$$

If we know $\Pi_i(t_0)$ we can know the probability at later times. Assume our system has reached the steady state, i.e., $d\Pi_i/dt = 0$, which implies

$$\Pi_i^{\text{eq}} \sum_{j\neq i} \mathcal{L}_{j\leftarrow i} = \sum_{j\neq i} \mathcal{L}_{i\leftarrow j}\Pi_j^{\text{eq}} \ . \tag{B.13}$$

[1] In this thesis, a continuous process is understood as a stochastic process that is continuous in time but whose state space is discrete.

A sufficient but not necessary condition to fulfill Eq. (B.13) is,

$$\Pi_i^{\text{eq}} \mathcal{L}_{j\leftarrow i} = \mathcal{L}_{i\leftarrow j} \Pi_j^{\text{eq}} \,, \tag{B.14}$$

which is known as the **detailed balanced condition**.
The evolution of our system is determined by knowing an initial state and the transition rate expression. Metropolis *et al.* [41], in connection to statistical mechanics, chose the Boltzmann distribution

$$\Pi_i^{\text{eq}} = \frac{1}{Z} \exp\left(-\frac{\mathcal{H}_i}{k_B T}\right), \quad Z = \sum_i \exp\left(-\frac{\mathcal{H}_i}{k_B T}\right) \,, \tag{B.15}$$

which implies,

$$\frac{\mathcal{L}_{j\leftarrow i}}{\mathcal{L}_{i\leftarrow j}} = \exp\left(\frac{\mathcal{H}_i - \mathcal{H}_j}{k_B T}\right) \,. \tag{B.16}$$

The last expression does not define uniquely the transition rate so we need a criterion for that. There are two common criteria:

- **Metropolis criterion:** $\mathcal{L}_{j\leftarrow i} = \min\left[1, \exp\left(\mathcal{H}_i - \mathcal{H}_j / \left(k_B T\right)\right)\right]$. This condition guarantees a transition into states with lower energy without forbidding a transition to higher energy states, where this transition rate depends on the energy difference.
- **Heat bath criterion:** $\mathcal{L}_{i\leftarrow j} = \frac{\Pi_j^{\text{eq}}}{\Pi_i^{\text{eq}} + \Pi_j^{\text{eq}}} = \left[1 + \exp\left(\frac{\mathcal{H}_j - \mathcal{H}_i}{k_B T}\right)^{-1}\right]$.

The transition rates do not guarantee the optimal value of our objective function. Hereafter, we will consider $k_B = 1$.

B.1.3 Annealing Schedules

The annealing approach considers a temperature that starts from a high value and decreases gradually with time. The dependence of temperature with respect to time t is named annealing schedule. There are many annealing schedules but the most common ones are:

- **Geman-Geman:** guarantees that the absolute minimum is reached when $t \to \infty$ [42]. It is given by

$$T(t) = \frac{a}{b + \log t} \,. \tag{B.17}$$

Since we do not have infinite time to perform our simulations, we need schedules that are faster despite their not guaranteeing convergence to the global minimum.

- **Linear cooling:** $T(t) = a - bT$ where $a = T_0$ – initial temperature – and $b \in (0.01, 0.2)$.
- **Exponential cooling:** $T(t) = ab^t$ where $a = T_0$ – initial temperature – and $b \in (0.8, 0.999)$.

B.2 Traveling Salesman Problem

The traveling salesman problem (TSP) is an NP-hard combinatorial optimization problem. That means it does not have a polynomial-time solution. The goal of our traveling salesman is to find the shortest path to visit all the n regions in a given list R, which are represented by nodes in a graph, by annotating the position P that each region has in the path, returning to the initial region. The distance between nodes $u, v \in R$, $D(u, v)$, measured in units of length, is calculated using the Euclidean metric

$$D(u, v) \equiv D_{uv} = \sqrt{|x_v - x_u|^2 + |y_v - y_u|^2} \,. \tag{B.18}$$

The objective function to minimize for a graph with n nodes can be written as

$$\min_{\vec{x}} f(\vec{x}) = \min_{\vec{x}} \sum_{v} \sum_{u} \sum_{i=0}^{i<n} D_{u,v} x_{v,i} x_{u,i+1} \,, \tag{B.19}$$

where the coefficients $x_{v,i}$ are binary variables that indicate if node $v \in R$ is in the position $i \in P$ of the tour, represented by the vector $\mathbf{x}$. Furthermore the distance between nodes is symmetric $D_{uv} = D_{vu}$ which is taken into account when creating the matrix D_{uv}.

B.2.1 Constraints

The traveling salesman has to visit each node just once in the whole path

$$\sum_{i=0}^{i<n} x_{u,i} = 1 \quad \forall\, u \in R \,, \tag{B.20}$$

and each stop of the tour has one node

$$\sum_{u} x_{u,i} = 1 \quad \forall\, i \in P \,. \tag{B.21}$$

The constraints can be taken into account in the cost function by using the Lagrange multipliers $\{\lambda_i\}_{i\in P}$ and $\{\lambda_u\}_{u\in R}$

$$\begin{aligned} \min_{\vec{x}} f(\vec{x}) = \min_{\vec{x}} \sum_{v} \sum_{u} \sum_{i=0}^{i<n} D_{u,v} x_{v,i} x_{u,i+1} \\ - \sum_{i=0}^{i<n} \lambda_i \left(\sum_{u} x_{u,i} - 1 \right)^2 - \sum_{u} \lambda_u \left(\sum_{i=0}^{i<n} x_{u,i} - 1 \right)^2 . \end{aligned} \tag{B.22}$$

The problem can be re-written to have the functional form of a QUBO problem $\min_x x^T Q x$, where Q is a matrix that stores the Hamiltonian coefficients. For this particular problem it is necessary to map the two indices of our binary variables $x_{u,i}$ into a single index, see Ref. [29]. If the constraints are taken into account in the loops of our program, then the function to minimize is just Eq. (B.19) which produces Figure B.1.

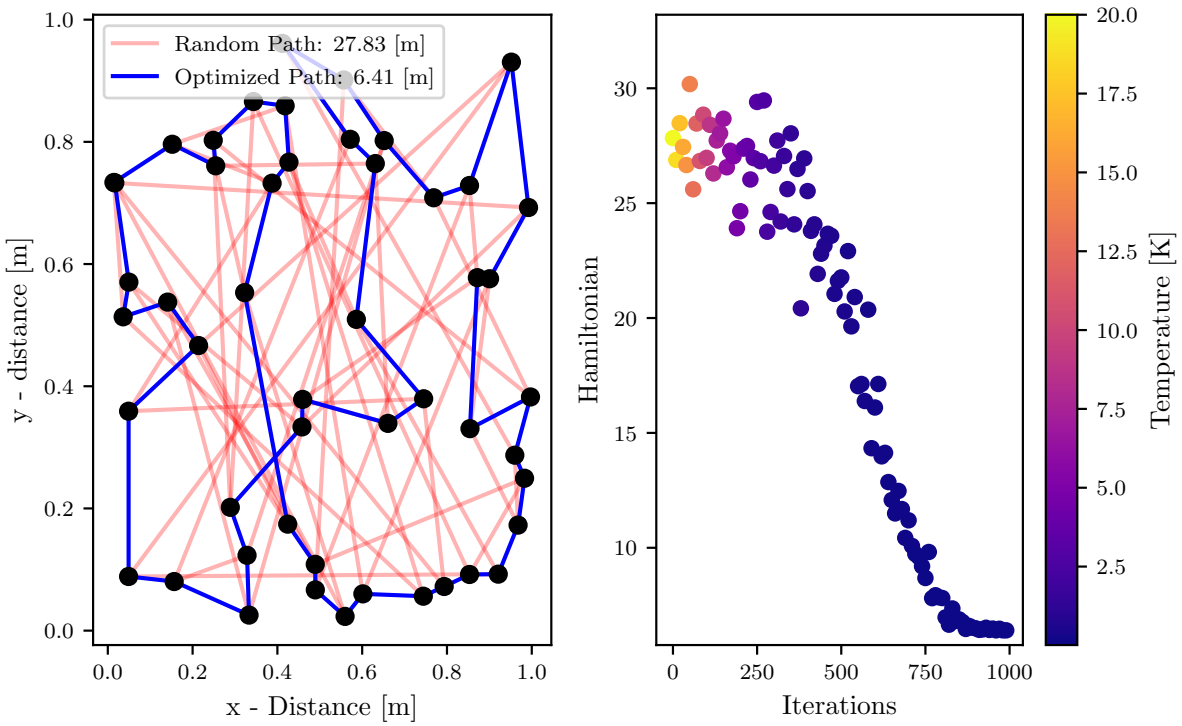


Figure B.1: Simulated annealing process for a 20-nodes travelling salesman problem, where nodes are represented by black dots. The code to produce the figure can be found in Appendix D. **Left:** Plot showing a random path (red) to travel the nodes and the path (blue) obtained for an instance of simulated annealing. **Right:** Hamiltonian as a function of iterations and temperature.

APPENDIX C

Hardware

In this Appendix we give an overview of the current quantum hardware without going deeply into details of its implementation. For a better understanding of the implementation of a superconducting qubit – the ones used by D-Wave – see Ref. [40].

C.1 Quantum Technologies

In classical computing, the *central processing unit* (CPU) is the part of a classical computer that processes the data and performs calculations. The key components of a CPU are called transistors. According to Moore's law, the number of transistors integrated in a micro-controller is doubled every two years. Equivalently, the size of transistors is getting smaller every two years. However, we are approaching the size where quantum effects are starting to appear and have to be taken into account. For this reason, a *quantum processing unit* (QPU) is required. There are different ways of implementing a qubit physically. We summarise some of them in the next table.

Qubit implementation technologies	
Name	**Companies**
Superconducting qubits	IBM, D-Wave, Rigetti, IQM
Trapped ions	IONQ, Quantinuum, Oxford Ionics, EleQtron
Photons	Xanadu, Orca Computing, PsiQuantum, Quandela
Anyons	Microsoft

Table C.1: List of current ways of implementing a qubit and the companies that are are developing them..

Concretely, we are interested in the implementation of D-Wave's superconducting qubits. However, a detailed explanation would be lengthy and we will only provide a simple overview of the historical evolution of D-Wave's annealers in terms of number of qubits and connectivity. More details can be found in Ref. [40].

Quoting DiVicenzo [43], the following criteria have to be fulfilled for the realisation of a quantum computer:

- *A scalable physical system with well characterized qubits.*
- *The ability to initialize the state of the qubits to a simple fiducial state, such as* $|000...\rangle$.
- *Long relevant decoherence times, much longer than the gate operation time.*
- *A "universal" set of quantum gates.*
- *A qubit-specific measurement capability.*

C.2 Quantum Annealers: An Overview

Quantum annealers are single-purpose quantum computers that solve Ising/QUBO problems. D-Wave started using using d-wave superconductors – from where D-Wave takes its name – but in 2011 D-Wave's researchers demonstrated a way of implementing synthetic qubits [44], i.e., qubits that interact between each other and whose behaviour is that of the Ising model. Here we show a list of D-Wave's quantum annealers.

D-Wave quantum annealers				
Name	**Release**	**No. of physical qubits**	**Architecture**	**Connectivity**
D-Wave One	2011	128	Chimera	6
D-Wave Two	2013	512	Chimera	6
D-Wave 2χ	2015	1152	Chimera	6
D-Wave 2000Q	2017	2048	Chimera	6
D-Wave Advantage	2020	5640	Pegasus	15
D-Wave Advantage 2	$\sim$2024	7440	Zephyr	20

Table C.2: Evolution of D-Wave quantum annealers.

Notice that the number of qubits is increasing every few years, almost duplicating its number. However, the number of qubits is not the only factor to care about but the connectivity of those qubits. The connectivity is given by the architecture, it represents the number of different qubits to which a given qubit is connected. Once a QUBO problem is formulated, D-Wave maps it into a Ising problem and it tries to embed the problem into the hardware. If a possible embedding is found, then the problem can be executed in a quantum annealer. This embedding depends on the number of variables our system has and on the entangled qubits required. In order to be able to solve a problem with the same number of binary variables as the number of physical qubits of a quantum annealer, the quantum annealer must have fully-connected qubits. We are yet far away from this fully-connected architecture, even though the connectivity between qubits is increasing every few years as Table C.2 shows.

APPENDIX D

Python Scripts

This appendix contains all the scripts used in the thesis. These scripts can be found at at the author's Github repository [45].

D.1 Two Heirs Problem

```
#=============================================================================#
# Two-Heirs problem
# Author: Sergio Lopez Banos
# GitHub: /LopezBanos
# Citation: If you use this code in your project please cite the following
# thesis:
'''
@Thesis{SerAlOr2023,
    author      = {Sergio López Baños},
    title       = {Transmission Expansion Planning by Quantum Annealing},
    type        = {diplomathesis},
    institution = {Nebrija University},
    year        = {2023},
}
'''
#=============================================================================#
#                            Importing packages                               #
#=============================================================================#
import numpy as np
import matplotlib.pyplot as plt
from numpy import linalg as LA

#=============================================================================#
#                                Ising Matrix                                 #
#=============================================================================#
# Two-Heirs problem parameters
v0=1 ; v1=3 ; v2=1
# Pauli Matrix and Identity
I = np.eye(2)
X = np.matrix([[0,1],[1,0]])
Z = np.matrix([[1,0],[0,-1]])

# Mapping of our problem into Ising
```

```
Ising = np.matrix([[0, 2*v0*v1, 2*v0*v2],[0, 0, 2*v1*v2],[0,0,0]])

#=============================================================================#
#                                 OUTER PRODUCTS                              #
#=============================================================================#
# Z
Z0Z1 = np.kron(Z, np.kron(Z, I))
Z0Z2 = np.kron(Z, np.kron(I, Z))
Z1Z2 = np.kron(I, np.kron(Z, Z))

# X
X0 = np.kron(X, np.kron(I, I))
X1 = np.kron(I, np.kron(X, I))
X2 = np.kron(I, np.kron(I, X))


#=============================================================================#
#                                 Hamiltonians                                #
#=============================================================================#
H_Ising = Ising[0,1]*Z0Z1 + Ising[0,2]*Z0Z2 + Ising[1,2]*Z1Z2
H_ini = -X0 - X1 - X2

# Check eigenvectors of Ising Hamiltonian
w, v = LA.eig(H_Ising)
print("eigenvalues:", w)
min_indices = np.where(w == w.min())[0]
print("min_eigenvalues indices:", min_indices[0])
print("The associated eigenvectors")
for i in range(len(min_indices)):
    print(v[min_indices[i]])

T = 1
time = np.linspace(0, T, 100)
#H = (1-t/T)*H_ini + (t/T)*H_Ising
eigen_H = []
eigen_H_Ising = []
eigen_H_ini = []
for t in time:
    # Compute the Hamiltonian at that snapshot
    H = (1-t/T)*H_ini + (t/T)*H_Ising

    # Compute the ground eigenstate
    # Total Hamiltonian
    w, v = LA.eig(H)
    min_index = np.argmin(w)
    eigen_H.append(w[min_index])
    # Ising Hamiltonian
    w, v = LA.eig((t/T)*H_Ising)
    min_index = np.argmin(w)
    eigen_H_Ising.append(w[min_index])
    # Initial Hamiltonian
```

```
    w, v = LA.eig((1-t/T)*H_ini)
    min_index = np.argmin(w)
    eigen_H_ini.append(w[min_index])

#=============================================================================#
# Plot with your preferred framework
#=============================================================================#
```

D.2 Traveling Salesman Problem (TSP)

```
#=============================================================================#
# Traveling salesman problem by simulated annealing
# Author: Sergio Lopez Banos
# GitHub: /LopezBanos
# Citation: If you use this code in your project please cite the following
# thesis:
'''
@Thesis{SerAlOr2023,
    author      = {Sergio López Baños},
    title       = {Transmission Expansion Planning by Quantum Annealing},
    type        = {diplomathesis},
    institution = {Nebrija University},
    year        = {2023},
}
'''
#=============================================================================#


#=============================================================================#
#                              Importing packages                             #
#=============================================================================#
import copy
import math
import matplotlib.pyplot as plt
import numpy as np


#=============================================================================#
#                                Coordinate Class                             #
#=============================================================================#
class Coordinates:
    def __init__(self, x, y):
        """
        Store a given coordinates array x, y
        """
        self.x = x
        self.y = y

    @staticmethod
```

```
40      def get_distance(a, b):
41          """
42          Get distance from two nodes (currrent node and next node):
43          param a: vector (x,y) of node 'a'
44          param b: vector (x,y) of node 'b'
45          """
46          return np.sqrt((a.x - b.x)**2 + (a.y - b.y)**2)
47
48      @staticmethod
49      def get_total_distance(coords):
50          """
51          Get distance of the path for the current configuration.
52          param coords: numpy.array with the coordinates of all the nodes
53          arranged in the travelled order.
54          """
55          dist = 0   # Initialize the distance
56          # Zip combine all coordinates x,y of nodes except the first and last
57          # node
58          for first, second in zip(coords[:-1], coords[1:]):
59              dist += Coordinates.get_distance(first, second)
60          # Distance from first node to last node
61          dist += Coordinates.get_distance(coords[0], coords[-1])
62          return dist
63
64
65  #============================================================================#
66  #            Fill up the coordinates and connecting nodes randomly           #
67  #============================================================================#
68  coords = []
69  N = 50  # Number of nodes
70  for i in range(N):
71      # Create a random list of instances (Nodes): Each instance with x = self.x
72      # and y = self.y coordinates
73      coords.append(Coordinates(np.random.uniform(low=0.0, high=1),
74       np.random.uniform(low=0.0, high=1)))
75  random_coords = copy.deepcopy(coords)
76
77
78  #============================================================================#
79  #                        Simulated annealing algorithm                       #
80  #============================================================================#
81  print("Starting annealing...")
82  #============================================================================#
83  #                               Annealing Schedule                           #
84  #============================================================================#
85  def annealing_schedule(T, factor, cost_init, number_of_swaps, coords,
86   annealing_type=None):
87      """
88      Annealing Schedule
89      """
90      # Type of Schedule (more types can be added)
```

```
    if annealing_type is None or annealing_type == "linear":
        T *= factor

    for j in range(number_of_swaps):
        # Exchange two coordinates and get a new neighbour solution
        # Get Index of two coordinates to swap
        r1, r2 = np.random.randint(0, len(coords), size=2)

        temp = coords[r1]
        coords[r1] = coords[r2]
        coords[r2] = temp

        # Get the cost of the new path
        cost1 = Coordinates.get_total_distance(coords)
        # Acceptance of new Candidate path
        # Check if neighbor is best so far
        cost_diff = cost1 - cost_init
        # if the new solution is better, accept it
        if cost_diff < 0:
            cost_init = cost1
        # if the new solution is not better, use Metropolis criterion
        else:
            if np.random.uniform() < math.exp(-cost_diff/T):
                cost_init = cost1
            else:
                temp = coords[r1]
                coords[r1] = coords[r2]
                coords[r2] = temp
    return [T, cost_init]


# List to store values
cost_list = []
T_list = []
steps_list = [0]

cost0 = Coordinates.get_total_distance(coords) # Current Cost
cost_list.append(cost0)
# Number of swaps between nodes before decreasing the temperature
number_of_swaps = 500

# Iterations equiv. decrease temperature by "factor",
# number_of_iterations (1000 times)
iteration = 0
number_of_iterations = 1000
# Every 10 steps we store the cost
steps = 10

# Annealing parameters
T = 20       # Current Temperature
T_init = T
```

```
T_end = 0.01
factor = (T_end/T_init)**(1/number_of_iterations)
T_list.append(T)

# Main Loop
while T > T_end:
    iteration += 1
    print("Temperature:", T, "[K]  ", 'cost = ', cost0,
     'Iteration = ', iteration)
    # Append Cost and temperature Every 100 iterations
    if (iteration % steps) == 0:
        steps_list.append(iteration)
        cost_list.append(cost0)
        T_list.append(T)
        if cost_list[-1] == cost_list[-2]:
            break
    list_annealing_schedule = annealing_schedule(T, factor, cost0,
     number_of_swaps, coords, "linear")
    T = list_annealing_schedule[0]
    cost0 = list_annealing_schedule[1]

#=============================================================================#
# Plot with your preferred framework
#=============================================================================#
```

D.3 Transmission Expansion Planning (TEP) Problem: A Three Node Fully Connected Network

```
#=============================================================================#
# Transmission Expansion Planning (TEP) Problem:
# A Three Node Fully Connected Network
# Author: Sergio Lopez Banos
# GitHub: /LopezBanos
# Citation: If you use this code in your project please cite the following
# thesis:
'''
@Thesis{SerAlOr2023,
    author      = {Sergio López Baños},
    title       = {Transmission Expansion Planning by Quantum Annealing},
    type        = {diplomathesis},
    institution = {Nebrija University},
    year        = {2023},
}
'''
#=============================================================================#


#=============================================================================#
#                            Importing packages                               #
#=============================================================================#
```

```
import dimod
import neal
import numpy as np
import pandas as pd
from dimod import ConstrainedQuadraticModel,BinaryQuadraticModel, Binary
from dimod import Integer, quicksum


#=============================================================================#
#                             Fill up the coordinates                         #
#=============================================================================#
C_iv = np.array([10, 20, 30], dtype=int)      # Investment Cost Coefficients
C_oc = np.array([10, 5, 2], dtype=int)        # Operational Cost Coefficients
g_max = np.array([10, 10, 10], dtype=int)     # Generator nominal power
S =  int(np.shape(C_iv)[0])                   # Number of rows of the matrix
D = np.array([10, 12, 10], dtype=int)         # Demand at each node


#=============================================================================#
#                                Binary Variables                             #
#=============================================================================#
#-----------------------------------------------------#
# Discretize generator variables (slack variables) #
#-----------------------------------------------------#
M_G = [int(np.floor(np.log2(g_max[i])))for i in range(S)]
c_g = []
for i in range(S):
    coeff_list = [2**j for j in range(M_G[i])]
    coeff_list.append(g_max[i] + 1 - 2**M_G[i])
    c_g.append(coeff_list)
#-----------------------------------------------------------#
# Creating the binary variables required for the CQM model #
#-----------------------------------------------------------#
# Transmission Lines
X = [Binary('X_{}_{}'.format(i+1, j+1))
     for i in range(S) for j in range(i+1,S)]

# Generators with slack variables
G = [Binary('g_{}_{}'.format(i+1, j+1))
     for i in range(len(c_g)) for j in range(4)]

# Group the transmission lines in list of list [[LinesofNode1],...]
x = [[X[0], X[1]],[X[0],X[2]],[X[1],X[2]]]

# Group generators
g = []
list_gen = []
for item in G:
    list_gen.append(item)
    if len(list_gen)==4: #?
        g.append(list_gen)
```

```
74          list_gen = []
75
76  #============================================================================#
77  #                                Build the CQM model                         #
78  #============================================================================#
79  cqm = ConstrainedQuadraticModel()
80  # Objective Function
81  obj_weight_value = 1
82
83  # Investment Cost
84  investment_objective = obj_weight_value * quicksum(C_iv[i] * X[i]
85                                                     for i in range(S))
86
87  # Operational cost
88  operational_objective = obj_weight_value * quicksum(C_oc[i]*c_g[i][j]*g[i][j]
89                                                      for i in range(S)
90                                                      for j in range(4))
91
92  # Total Objective
93  objective = investment_objective + operational_objective
94
95  # Add the objective to the CQM
96  cqm.set_objective(objective)
97
98  def pop_index(index, size):
99      """
100     Remove an index from a list.
101     """
102     relaxed_list = [i for i in range(size)]
103     relaxed_list.remove(index)
104     return relaxed_list
105
106 # Single Demand Constraint at Node 2
107 i=1
108 cross_index  = pop_index(i, 3)
109 cqm.add_constraint(quicksum(c_g[i][j]*g[i][j]
110             + (c_g[cross_index[0]][j]*g[cross_index[0]][j])*x[i][0]
111             + (c_g[cross_index[1]][j]*g[cross_index[1]][j])*x[i][1]
112             for j in range(4)) == D[i], label = 'Demand_{}'.format(i+1))
113
114
115 #============================================================================#
116 #                   Select a dimod Solver and solve the problem              #
117 #============================================================================#
118 cqm_exactsolver = dimod.ExactCQMSolver()
119 results = cqm_exactsolver.sample_cqm(cqm)
```

Bibliography

[1] D. Herman, C. Googin, X. Liu, *et al.*, *A survey of quantum computing for finance*, 2022. DOI: 10.48550/ARXIV.2201.02773.

[2] T. D. Tambunan, A. B. Suksmono, I. J. M. Edward, and R. Mulyawan, *Quantum annealing for vehicle routing problem with weighted segment*, 2022. DOI: 10.48550/ARXIV.2203.13469.

[3] M. Fernández-Campoamor, C. O'Meara, G. Cortiana, V. Peric, and J. Bernabé-Moreno, *Community detection in electrical grids using quantum annealing*, 2021. DOI: 10.48550/ARXIV.2112.08300.

[4] *PyPSA-Eur: A Sector-Coupled Open Optimisation Model of the European Energy System — PyPSA-Eur*. [Online]. Available: https://pypsa-eur.readthedocs.io/en/latest/.

[5] J. Preskill, "Quantum computing in the NISQ era and beyond," *Quantum*, vol. 2, p. 79, Aug. 2018. DOI: 10.22331/q-2018-08-06-79.

[6] A. Callison and N. Chancellor, "Hybrid quantum-classical algorithms in the noisy intermediate-scale quantum era and beyond," *Physical Review A*, vol. 106, no. 1, Jul. 2022. DOI: 10.1103/physreva.106.010101.

[7] E. Farhi, J. Goldstone, S. Gutmann, and M. Sipser, *Quantum computation by adiabatic evolution*, 2000. DOI: 10.48550/ARXIV.QUANT-PH/0001106.

[8] T. Kadowaki and H. Nishimori, "Quantum annealing in the transverse ising model," *Physical Review E*, vol. 58, no. 5, pp. 5355–5363, Nov. 1998. DOI: 10.1103/physreve.58.5355.

[9] M. S. Sarandy and D. A. Lidar, "Adiabatic approximation in open quantum systems," *Physical Review A*, vol. 71, no. 1, Jan. 2005. DOI: 10.1103/physreva.71.012331.

[10] D. J. Griffiths, *Introduction to Quantum Mechanics*, 2nd ed. Upper Saddle River, NJ: Pearson, Mar. 2004.

[11] J. W. Brown and R. V. Churchill., *Fourier Series and Boundary Value Problems*. New York: McGraw-Hill, 1993.

[12] E. Farhi, J. Goldstone, and S. Gutmann, *A quantum approximate optimization algorithm*, 2014. DOI: 10.48550/ARXIV.1411.4028.

[13] M. V. F. Pereira and L. M. V. G. Pinto, "Multi-stage stochastic optimization applied to energy planning," *Mathematical Programming*, vol. 52, no. 1-3, pp. 359–375, May 1991. DOI: 10.1007/bf01582895.

[14] Z. Zhao, L. Fan, and Z. Han, "Hybrid quantum benders' decomposition for mixed-integer linear programming," in *2022 IEEE Wireless Communications and Networking Conference (WCNC)*, IEEE, Apr. 2022. DOI: 10.1109/wcnc51071.2022.9771632.

[15] Y. Ding, X. Chen, L. Lamata, E. Solano, and M. Sanz, "Implementation of a hybrid classical-quantum annealing algorithm for logistic network design," *SN Computer Science*, vol. 2, no. 2, Feb. 2021. DOI: 10.1007/s42979-021-00466-2.

[16] Gurobi Optimization, LLC, *Gurobi Optimizer Reference Manual*, 2023. [Online]. Available: https://www.gurobi.com.

[17] I. I. Cplex, "V12. 1: User's manual for cplex," *International Business Machines Corporation*, vol. 46, no. 53, p. 157, 2009.

[18] J. Forrest, T. Ralphs, H. G. Santos, *et al.*, *Coin-or/cbc: Release releases/2.10.10*, version releases/2.10.10, Apr. 2023. DOI: 10.5281/zenodo.7843975. [Online]. Available: https://doi.org/10.5281/zenodo.7843975.

[19] M. Bierlaire, *Optimization: Principles and Algorithms*, 2nd. Lausanne: EPFL Press, 2018, ISBN: 978-2-940-22278-0.

[20] N. Sahinidis and I. Grossmann, "Convergence properties of generalized benders decomposition," *Computers and Chemical Engineering*, vol. 15, no. 7, pp. 481–491, Jul. 1991. DOI: 10.1016/0098-1354(91)85027-r.

[21] S. Boyd and L. Vandenberghe, *Convex Optimization*, en. Cambridge, England: Cambridge University Press, Mar. 2004, ch. 5, pp. 215–287.

[22] F. Neumann and T. Brown, "Transmission expansion planning using cycle flows," in *Proceedings of the Eleventh ACM International Conference on Future Energy Systems*, ACM, Jun. 2020. DOI: 10.1145/3396851.3397688.

[23] D. Oertel and R. Ravi, "Complexity of transmission network expansion planning," *Energy Systems*, vol. 5, no. 1, pp. 179–207, Aug. 2013. DOI: 10.1007/s12667-013-0091-3.

[24] K. Dilwali, H. Gunnaasankaraan, A. Viswanath, and K. Mahata, "Transmission expansion planning using benders decomposition and local branching," in *2016 IEEE Power and Energy Conference at Illinois (PECI)*, IEEE, Feb. 2016. DOI: 10.1109/peci.2016.7459265.

[25] S. Binato, M. Pereira, and S. Granville, "A new benders decomposition approach to solve power transmission network design problems," *IEEE Transactions on Power Systems*, vol. 16, no. 2, pp. 235–240, May 2001. DOI: 10.1109/59.918292.

[26] S. Huang and V. Dinavahi, "A branch-and-cut benders decomposition algorithm for transmission expansion planning," *IEEE Systems Journal*, vol. 13, no. 1, pp. 659–669, Mar. 2019. DOI: 10.1109/jsyst.2017.2775610.

[27] C. MacRae, A. Ernst, and M. Ozlen, "A benders decomposition approach to transmission expansion planning considering energy storage," *Energy*, vol. 112, pp. 795–803, Oct. 2016. DOI: 10.1016/j.energy.2016.06.080.

[28] Z. Zhao, L. Fan, and Z. Han, "Hybrid Quantum Benders' Decomposition For Mixed-integer Linear Programming," Dec. 2021. [Online]. Available: http://arxiv.org/abs/2112.07109.

[29] S. Jain, "Solving the traveling salesman problem on the d-wave quantum computer," *Frontiers in Physics*, vol. 9, Nov. 2021. DOI: 10.3389/fphy.2021.760783.

[30] P. V. Gomes and J. T. Saraiva, "State-of-the-art of transmission expansion planning: A survey from restructuring to renewable and distributed electricity markets," *International Journal of Electrical Power and Energy Systems*, vol. 111, pp. 411–424, Oct. 2019. DOI: 10.1016/j.ijepes.2019.04.035.

[31] S. Mandrà and H. G. Katzgraber, "A deceptive step towards quantum speedup detection," *Quantum Science and Technology*, vol. 3, no. 4, 04LT01, Jul. 2018. DOI: 10.1088/2058-9565/aac8b2.

[32] DLR-SC, `Quark`, version 0.1, 2023. [Online]. Available: https://gitlab.com/quantum-computing-software/quark.

[33] U. P. Mueller, L. Wienholt, D. Kleinhans, *et al.*, "The eGo grid model: An open source approach towards a model of german high and extra-high voltage power grids," *Journal of Physics: Conference Series*, vol. 977, p. 012 003, Feb. 2018. DOI: 10.1088/1742-6596/977/1/012003.

[34] F. W.Bryon and R. W.Fuller, "Hilbert Space – Complete Orthonormal Sets of Functions," in *Mathematics of Classical and Quantum Physics*, Revised ed., Dover Publications, 1992, ch. 5, pp. 212–303.

[35] W. Scherer, *Mathematics of Quantum Computing*. Cham: Springer International Publishing, 2019, ISBN: 978-3-030-12357-4. DOI: 10.1007/978-3-030-12358-1.

[36] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information: 10th Anniversary Edition*. Cambridge University Press, 2011, ISBN: 978-1-107-00217-3.

[37] L. K. Grover, "A fast quantum mechanical algorithm for database search," in *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing - STOC '96*, ACM Press, 1996. DOI: 10.1145/237814.237866.

[38] C. Lavor, L. R. U. Manssur, and R. Portugal, *Grover's algorithm: Quantum database search*, 2003. DOI: 10.48550/ARXIV.QUANT-PH/0301079.

[39] J. J. Schneider and S. Kirkpatrick, *Stochastic Optimization*. Berlin Heidelberg, Germany: Springer-Ver, 2006.

[40] Á. Díaz-Fernández, "Lecture notes on quantum annealing," *unpublished*, 2023.

[41] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, "Equation of state calculations by fast computing machines," *The Journal of Chemical Physics*, vol. 21, no. 6, pp. 1087–1092, Jun. 1953. DOI: 10.1063/1.1699114.

[42] S. Geman and D. Geman, "Stochastic relaxation, gibbs distributions, and the bayesian restoration of images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-6, pp. 721–741, 1984. DOI: 10.1109/tpami.1984.4767596.

[43] D. P. DiVincenzo, "The physical implementation of quantum computation," *Fortschritte der Physik*, vol. 48, no. 9-11, pp. 771–783, Sep. 2000. DOI: 10.1002/1521-3978(200009)48:9/11<771::aid-prop771>3.0.co;2-e.

[44] M. W. Johnson, M. H. S. Amin, S. Gildert, *et al.*, "Quantum annealing with manufactured spins," *Nature*, vol. 473, no. 7346, pp. 194–198, May 2011. DOI: 10.1038/nature10012.

[45] S. López Baños, *MSc Thesis Nebrija-DLR*, https://github.com/LopezBanos/MsCThesis_Nebrija_DLR, 2023.