Universidade Federal de Minas Gerais

Ph.D. Thesis

Permutation-based Optimization for the Load Restoration Problem with Improved Time Estimation of Maneuvers

Author:
Fillipe Goulart

Advisor:
Dr. Felipe Campelo
Co-advisor:
Dr. Eduardo G. Carrano

Tese submetida à banca examinadora designada pelo Colegiado do Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal de Minas Gerais, como parte dos requisitos necessários para a obtenção do título de Doutor em Engenharia

January 2019

Declaration of Authorship

I, Fillipe GOULART, declare that this thesis, titled "Permutation-based Optimization for the Load Restoration Problem with Improved Time Estimation of Maneuvers", and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

"Permutation-based Optimization for the Load Restoration Problem with Improved Time Estimation of Maneuvers"

Fillipe Goulart Silva Mendes

Tese de Doutorado submetida à Banca Examinadora designada pelo Colegiado do Programa de Pós-Graduação em Engenharia Elétrica da Escola de Engenharia da Universidade Federal de Minas Gerais, como requisito para obtenção do grau de Doutor em Engenharia Elétrica.

Aprovada em 26 de novembro de 2018.

Por:

Prof. Dr. Felipe Campelo França Pinto
DEE (UFMG) - Orientador

Prof. Dr. Eduardo Gontijo Carrano
DEE (UFMG) - Coorientador

Prof. Dr. Michel Bessani
DEE (UFMG)

Prof. Dr. Wallace do Couto Boaventura
DEE (UFMG)

Prof. Dr. Paulo Eduardo Maciel de Almeida
DECOM (CEFET-MG)

Prof. Dr. Joao Bosco Augusto London Junior
DEEC (USP - SC)

"Statistics: The only science that enables different experts using the same figures to draw different conclusions."

Evan Esar

UNIVERSIDADE FEDERAL DE MINAS GERAIS

Abstract

There is an old saying that two things are inevitable in life: death and taxes. Let me take the liberty to include another one: blackouts in the system, or, more technically, faults, which can interrupt the regular energy supply to customers and leave them out of service. While it is not practical to completely prevent these events, it is possible to minimize their impacts by employing load restoration techniques. This work deals with the load restoration problem in radial distribution systems, which consists in implementing a sequence of switch opening and closing operations such that the resulting configured network restores service to the most loads in the shortest possible time. We formulate this optimization problem in terms of two complementary objectives: minimizing simultaneously the energy not supplied and the power not restored. The search space is encoded as a set of permutation vectors containing all maneuverable switches, and the decoding mechanism always guarantees feasibility and allows for multiple solutions per vector. In order to cope with the possibly large search space, an efficient reduction mechanism is proposed to decrease the number of allowed permutations. Also, an initial step considering only remote switches is employed to return solutions implementable within the no-penalties time window that many companies have. The resulting optimization problem is solved using Simulated Annealing followed by a Local Search refinement. A decoding mechanism is proposed to return a proper sequence of maneuvers from this permutation vector, and this sequence not only respects predefined rules of precedences but also provides an estimation of the time of maneuvers and energy not supplied for multiple dispatch teams, if available. The goal is to provide more realistic solutions when compared to the usual approach of considering only the number of switch operations. The complete method is validated using known optimal results in small problem instances, and is able to return significantly better results when compared against a Branch and Bound method with a pruning heuristic in a more complex scenario.

Keywords: multi-objective optimization, electric distribution systems, load restoration problem, time of maneuvers, energy not supplied.

UNIVERSIDADE FEDERAL DE MINAS GERAIS

Resumo

Existe um antigo ditado que diz que há duas coisas na vida que continuam inevitáveis: a morte e os impostos. Muitos irão concordar se eu, porém, tomar a liberdade de acrescentar mais um item à lista: a luz "acabar", ou, de forma mais técnica, o sistema elétrico sofrer faltas, interrompendo o suprimento normal de energia aos clientes e deixando-os fora de serviço. Apesar de ser impraticável impedir por completo estes infortúnios, é possível aplicar técnicas de restauração de carga para amenizar os seus impactos. Este trabalho lida com o problema de restauração em sistemas de distribuição, o qual se resume a executar uma sequência de aberturas e fechamentos de chaves de forma a levar o sistema a uma nova configuração que recupere o máximo de cargas no menor tempo possível. Este problema é aqui formulado como a minimização da energia não suprida e da potência total não restaurada. O espaço de busca escolhido consiste no conjunto de permutações das chaves de manobra, e é proposto um mecanismo de avaliação que leva sempre a configurações factíveis. O processo de otimização torna-se mais eficiente pela proposta de mecanismos de redução, que impedem a criação de algumas permutações redundantes. Além disso, uma etapa inicial é incluída que faz uso apenas de chaves telecomandadas, as quais são capazes de recuperar energia dentro de um certo limite de tempo que previne impactos nos índices de confiabilidade. O problema resultante é resolvido usando um Simulated Annealing seguido de uma busca local. Um mecanismo de decodificação também é proposto que converte um vetor de permutação em uma sequência de manobras que, além de respeitar um conjunto predefinido de regras de precedência, também estima o tempo total de manobras e a energia não suprida na presença de múltiplas equipes de manobra. O método completo retornou as mesmas soluções que um processo exato em instâncias pequenas e, em cenários mais complexos, foi capaz de obter soluções significativamente melhores quando comparado com um método Branch and Bound com uma heurística de poda.

Palavras-chave: otimização multiobjetivo, sistemas elétricos de distribuição, problema de restauração de carga, tempo de manobras, energia não suprida.

Published works

The two following papers were published as results from this work (transcribed from references [1, 2]):

- Fillipe Goulart, André L. Maravilha, Eduardo G. Carrano and Felipe Campelo. "Permutation-based optimization for the load restoration problem with improved time estimation of maneuvers". International Journal of Electrical Power & Energy Systems, 101:339–355, 2018.
- André L. Maravilha, Fillipe Goulart, Eduardo G. Carrano and Felipe Campelo. 'Scheduling Maneuvers for the Restoration of Electric Power Distribution Networks: Formulation and Heuristics". Electric Power Systems Research, 163:301–309, 2018

Acknowledgements

To begin with, I would like to thank my advisor, Professor Felipe Campelo, for helping (enduring?) me through all these eight to nine years. It would take me a text with half the size of this one to scratch the surface on how your help contributed not only with my academical but personal growth (which would not be an issue for me to write, but probably would be for you to read!). Thus, I will limit myself with the superficial "thank you for all the assistance", of course aware of all important details that lie underneath it. Also, I thank my co-advisor, Professor Eduardo Carrano, for all supplemental support. This work would lose most of its relevance if it were not for you help. Thanks to you both I am even closer for becoming a "true engineer".

To my dear ORCS co-workers and friends, most specifically, the twins Fernanda Takahashi and Carla Tahakashi, who simply *love* to be hugged; Letícia Mayra (Maíra) and Bel, who helped me share appropriate commentary on most relevant news happening in the world, ranging from the importance of scientific research to the importance of Selena Gomes' break-up; the sweet Moniquita, who taught me the relevance of brushing your vegetables and for helping me stay in (a circular) shape because of all the sweets and cakes; André Maravilha, who assisted me with all questions I had about combinatorial optimization, linear programming and investments, and who was kind enough to not laugh at most of them; the prodigies André Batista (with ten published papers, five presentations, two Oscar's and one Nobel Prize before even graduating) and Ana Cristina (who's finishing her simultaneous master/Ph.D. program whilst working at a company located at the other side of the city); the newcomers Ister, Lara, Natália, Bruna and André Élcio, who were strong enough to persist with their studies despite how weird we, their co-workers, are; and finally, Áthila, Jean and Reginaldinho, who *were accepted for public offices*, and in no way is this information relevant for the current acknowledgments. All of you helped making this process go way smoother.

Finally, I thank my family for all of their unconditional support. Andrezza, the one and only queen of the my life who in no way asked me to write this, thank you for knowing how to handle (enduring as well) my most difficult and annoying moments; Brunna, who is the son of the devil, I am grateful that, despite all of the universe's randomness, you were the one who happened to be my sister; and my parents, thank you for being supportive, lovely and weird you are. I know that even without understanding a single sentence, you all would still read this entire text from front to cover.

Contents

List of Figures

List of Tables

Symbols

$\mathcal{G}_D = (\mathcal{B}, \mathcal{L}^{CC})$	Complete graph of a distribution system
$\mathcal{B}$	Set of buses (sources, loads and any shunt element)
$\mathcal{L}$	Set of connections between two buses (lines, transformers, phase shifters)
$\mathcal{L}^{CC}$ and $\mathcal{L}^{CO}$	Subsets of currently closed (active) and open (inactive) connections
$\mathcal{G} = (\mathcal{N}, \mathcal{E}^{CC})$	Contracted version of $\mathcal{G}_D$
$\mathcal{N}$	Set of load sectors, with no maneuverable connection among them
$\mathcal{E}$	Set of maneuverable connections (switches) between two sectors
$\mathcal{E}^{CC}$ and $\mathcal{E}^{CO}$	Subsets of currently closed and open switches
$\mathcal{G}_R = (\mathcal{N}_R, \mathcal{E}_R)$	Restoration sub-graph
$\mathcal{M}$	Set of maneuvers: an unordered subset of edges in $\mathcal{E}$
M	Sequence of maneuvers: an ordered subset of edges in $\mathcal{E}$
$\mathcal{G}(M)$ or $\mathcal{G}(\mathcal{M})$	Resulting network after the maneuvers in M (or $\mathcal{M}$)
S_{NR}	(weighted) Power not restored
N_m	Number of maneuvers
T_m	Time of maneuvers
E_{NS}	Energy not supplied

Chapter 1

Introduction

The distribution system is the portion of the power system structure that delivers electrical energy to most customers [3]. Because of this direct relationship, it is the main responsible for maintaining a continuous supply to these loads, which can be understood as a reliability requirement. Unfortunately, the system is constantly subject to external factors [4], such as weather events, human errors, malfunction of protection devices etc., which makes it impractical - if not impossible - to design a system that will never fail.

The impacts of power outages, also known as *faults*, can range from simple nuisances to disasters. Moreover, the distribution company may be penalized and will incur in fines to clients and government depending on the effects of the interruptions. Assuming that faults will inevitably occur, the best course of action is to devise plans that minimize their consequences. The first type of approaches belongs to the *protection of power systems* field [5], which basically consists in isolating the faulty element(s) from the rest of the system and thus limiting the disturbances to a smaller area. The second is to employ a *load restoration* method, which is the topic of this work.

1.1 Brief overview of Load Restoration and its current treatment in the literature

To get a first grasp at what load restoration consists of, consider the network at the left of Figure 1.1. It shows a simplified distribution network depicted as a graph[1], in which each node indicates load sectors, connected by switches represented by the edges. The solid lines indicate active (closed) connections, while the dashed edges are inactive

[1] Appendix A provides a quick introduction to graphs for readers who may not be familiar with these concepts.

(open). There are three different circuits, called *feeders*, and initially the loads in each one of them are independently energized by the sources, represented by blue nodes.

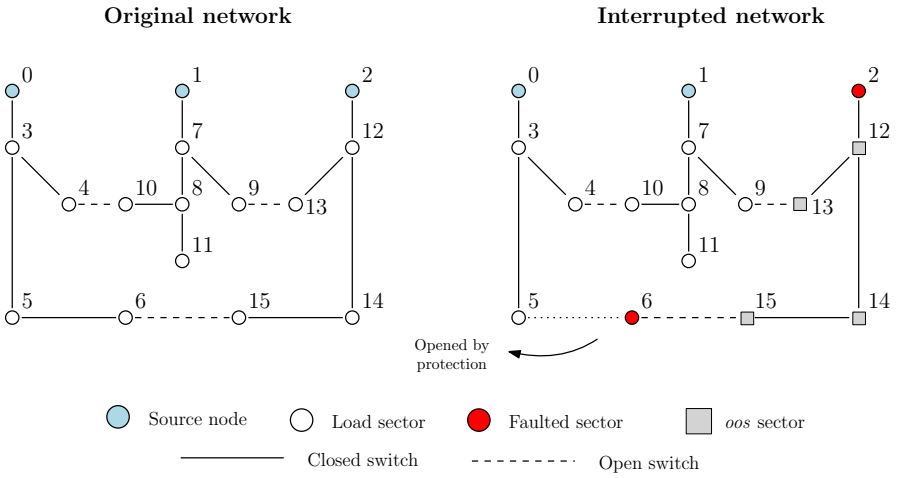


FIGURE 1.1: Graph representation of a distribution network.

Most distribution systems are operated in a radial configuration as shown in the example, which presents a number of advantages, such as easier fault current protection, voltage control, and overall lower fault currents and installation costs [3]. However, an interruption at any point – be it programmed, as in case of maintenance, or not, as in case of faults – leaves the whole downstream portion *out of service* (*oos*), thus extending its effects to a larger degree. This is shown in the right panel of Figure 1.1. For faults at nodes 2 and 6, sectors 12, 13, 14 and 15 become out of service. While it makes sense to leave the faulted nodes without energy, disrupting the other healthy loads leads to unnecessarily worse reliability indices [3]. Fortunately, even if the feeders are operated independently in a radial topology, the system is still interconnected and can be reconfigured by opening and closing switches, and some de-energized loads can be recovered by neighboring feeders. Three examples of this procedure are exemplified in Figure 1.2:

1. In the top left, by opening switches $(2, 12)$ and $(12, 14)$ and closing $(9, 13)$, nodes 12 and 13 were energized by the middle feeder. Sectors 14 and 15 were left out of service to respect operational limits of voltage, current and feeder capacity.

2. In the top right, by discarding the initially energized sector 11, the middle feeder has more capacity to also recover 14. Depending on the distribution utility, this load shedding of healthy *in service* (*ins*) nodes may be forbidden altogether or may be accepted in case the *oos* ones have higher priority (14 can be a hospital, for example);

3. At the bottom, nodes 8, 10 and 11 were transferred to the left feeder by opening $(7, 8)$ and closing $(4, 10)$, leaving enough capacity to the middle feeder to recover

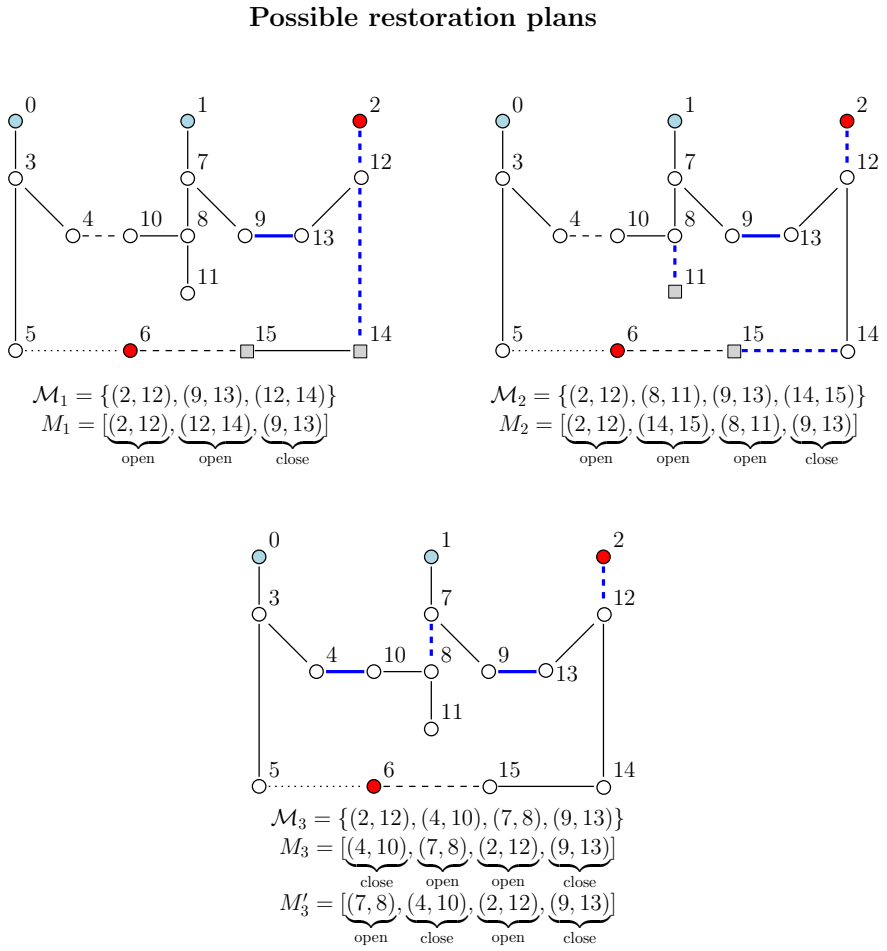


FIGURE 1.2: Examples of restoration plans. Below each plan are shown the sets of operated switches and possible sequences in which each maneuver is performed.

the whole *oos* region. However, notice a peculiarity in this *load transfer*: if $(7,8)$ is opened first, nodes 8, 10 and 11 are temporarily disconnected until $(4,10)$ is closed, and the reliability indices can suffer during this time. On the other hand, if $(4,10)$ is maneuvered before, no sector becomes *oos*, but the system is momentarily not radial, and some practical concerns may have to be taken care, such as possible loss of coordination in the protection scheme and the impacts of connecting two or more possibly out of sync generators.

The act of disconnecting loads is generally referred to *load shedding* in the literature. In the first case, this term may seem odd because we are not exactly "cutting any load", but rather "not energizing them". In any case, to simultaneously keep the usual terminology and prevent any confusions, this text will adopt *oos load shedding* to indicate the act of letting *oos* sectors de-energized (as in the first example), and *ins load shedding* for disconnecting customers in service (as in the second example).

The three examples just described constitute plans of a *load (or service) restoration* process. Its goal can be loosely stated as [6] to determine a *sequence of maneuvers* that restores the most *oos* loads in the shortest time possible without violating constraints such as voltage and current limits in buses and lines, feeder capacity and system radiality. This is a somewhat classic problem and has been widely studied in the literature. In spite of that, there are basically three issues that are currently either ignored or only partially addressed by the state of the art:

- What types of maneuvers are allowed? Even in the simple example of Figure 1.2 we encountered different strategies such as load shedding of both *oos* and *ins* nodes and load transfers with closing before opening or vice-versa. As argued, allowing or forbidding some maneuvers depends on the distribution utility and should be explicit when developing a new algorithm;

- From Figure 1.2, the new configurations after implementing each plan are evident. Unfortunately, this is commonly the only output of many studies. This is problematic because dispatchers expect a proper *sequence* of operations (see section 2.2.2) to be followed, which is supposed to never violate the operating constraints after each step, that is, they should respect some *rules of precedence*. In simple terms, these rules are responsible for allowing a neighboring feeder to energize *oos* loads only after a fault was isolated; dictating whether load transfer should be executed first with a closing or an opening; etc. Thus, apart from the *set* of operated switches, we also show at the bottom of each panel in Figure 1.2 some possible *sequences* of maneuvers;

- The "restore the most *oos* loads in the shortest time possible" condition of a good restoration plan requires determining a combination of *quality indices* (section 2.2.3) so that the problem can receive a proper mathematical formulation. While the "restore the most *oos* loads" portion tend to be straightforwardly represented by the power not restored $S_{NR}(\cdot)$, the "in the shortest time possible" is widely modeled by the number of maneuvers $N_m(\cdot)$, which is a poor representation due to its inability of (i) modeling the time delay taken by a dispatch team to move from its current location to a switch's position; (ii) incorporating the availability of multiple teams [1]; and (iii) distinguishing between remotely-controlled (which tend to become more common in the context of Smart Grids) and manually-operated switches. As will be seen, there are more realistic indices to be used instead, such as the time of maneuvers $T_m(\cdot)$ and the energy not supplied $E_{NS}(\cdot)$.

The first issue is one of the reasons why it is not easy to compare different works in a fair way. Moreover, it can be argued that not making the allowed rules explicit is probably an even bigger obstacle when determining advances in the field. Regarding the second, there are actually some works that recognize the importance of the sequence of maneuvers and propose a posterior step to output such parameter from the final configuration, and a representative class of them will be discussed later in the literature review (Chapter 3). Unfortunately, while this helps to alleviate the second issue, the objective problem is still solved in terms of the less realistic index $N_m(\cdot)$ instead of $T_m(\cdot)$ or $E_{NS}(\cdot)$, so the third point is still an open issue.

1.2 Goals of this work

The main contributions of this work are:

- The formalization of the concept of rules of precedences, and its complete analysis for a specific case (the one exemplified at the top left panel of Figure 1.2) of restoration plans involving maneuvers only in the out of service region;
- The proposition of a decoding heuristic that converts a new configuration into a proper sequence of maneuvers that not only respects the aforementioned rules but also provides an estimation of $T_m(\cdot)$ and $E_{NS}(\cdot)$ even in the presence of multiple dispatch teams;
- An optimization algorithm similar to a previous one in the literature but enhanced in terms of adopting the proposed time estimation;
- A perturbation scheme that tries to limit the exploration of the algorithm in redundant regions of the search space and thus improve its efficiency;
- A pre-processing step running the complete algorithm with remote switches only, which, if applicable, may be able to prevent unnecessary impacts on the reliability indices.

1.3 Structure of the text

This work is structured as follows:

- This first chapter just scratched the surface of the load restoration problem in distribution systems. The problem is set forth in Chapter 2. After a brief review

of the concepts of distribution systems and Fault Detection, Location, Isolation and Service Restoration (FDLISR), the load restoration problem is detailed and a mathematical formulation is presented. All of the concepts introduced in this chapter will be adopted in the rest of the text;

- With the primary concepts presented, Chapter 3 provides a literature review of studies that try to solve the load restoration. As will become evident, the problem can be approached in many different ways: with distinct modeling (i.e., which quality indices are used to compare different solutions), diverse encoding, various optimization techniques etc. The disparity is such that it is not straightforward to provide direct comparisons among these works. Therefore, this chapter tries to present each study in such detailed manner to make evident its weak and strong points;

- Chapter 4 discusses the first contribution of this work. First, the precedence rules are formalized with some examples for typical restoration plans. Then, a decoding process is presented with the aim of returning, from a new configuration, a sequence of maneuvers with the characteristics mentioned before in section 1.2. The chapter closes by comparing the performance of the proposed indices against the usual approach of using the number of maneuvers, in terms of providing good estimates of the time taken to perform a given restoration plan;

- Chapter 5 presents the proposed optimization algorithm, which, thanks to the method introduced in Chapter 4, is able to work directly in the set of maneuvers and still perform the optimization in terms of more realistic indices. The method presented here aims at improving an existing algorithm with – apart from the upgraded estimation of time of maneuvers – efficient reduction mechanisms and an initial step considering only remote switches;

- To evaluate the effectiveness of the proposed approach in a real network, Chapter 6 presents computational experiments performed using a real network model. First, it compares the described perturbation schemes to generate new solutions, and then, it shows how the proposed algorithm fares when solving the restoration problem in a large network. As will become evident, the proposed method is able to approximate the optimal solutions in small instances (when the optimum can be computed with an enumeration technique), while being able to return good solutions in reasonable time even in more complex scenarios when compared to a mathematical-based approach.

- Finally, Chapter 7 concludes this work and presents proposals for the continuity of the present work.

To allow for a more natural flow for the reader, as well as for a more self-contained thesis, a great portion of background knowledge for this work was left to appendices, which are mentioned whenever their contents are required as prerequisite for understanding particular concepts introduced in this thesis.

Chapter 2

Theoretical Foundations

2.1 Electrical Power Systems

A power system is composed of power sources, also called *generators*, and power users or customers, also referred to as *loads*. Generators, as the name implies, are the ones responsible for providing the electrical energy, including [7] fossil-fuel plants, nuclear power plants, hydroelectric power plants etc. The loads are the portion of the power system that consumes the electricity, like industries, houses, hospitals etc.

As the generation and the consumption are usually geographically removed from each other, it is necessary to interconnect them. This is achieved by the *transmission* and *distribution* systems, which are the other two main blocks of the power system. Figure 2.1 shows a one-line diagram of a typical power system and how its components are connected. As shown, although the transmission-distribution part is actually a single interconnected system, it is usually convenient to consider them separately, which simplifies the analysis.

Starting from the top of Figure 2.1, the energy produced at the generators is typically in the order of 5 kV to 30 kV, and this value is limited by insulation difficulties in higher voltages. Since the distance from the generators to the loads can be several hundred or even thousands of kilometers, transformers are employed to step up the voltage to the order of 230 kV to 750 kV, which reduces the resistive losses in the transmission.

The transmission system is composed of a network of three-phase transmission lines and transmission substations, and it is used to [8] (i) deliver energy from the generators to the system; (ii) provide energy interchange among utilities; and (iii) supply energy to the distribution systems. Sometimes there is also a sub-transmission system, which works

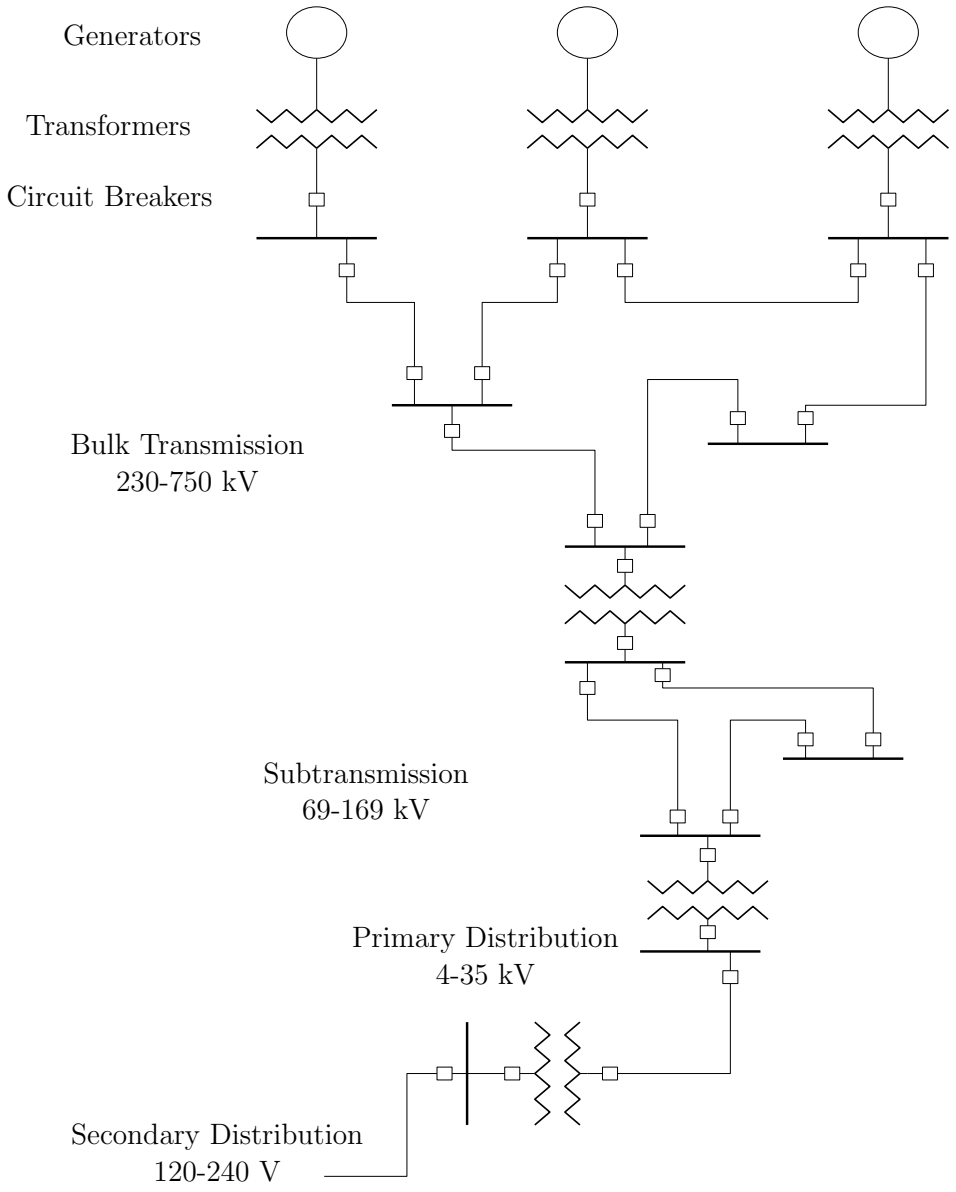


FIGURE 2.1: Overview of a power system structure. The energy produced at the generators is transported through the transmission and distribution systems and is delivered to the buses, represented by the thicker lines. The buses can be connected to generators, substations or even loads, depending on which portion of the power system they are.

with lower voltages, typically in the order of 69 to 138 kV, and is used to connect the bulk power substations to the distribution system.

Then, at the bottom of the figure, there is the *distribution system*, which is the portion of the delivery structure that takes the energy from the highly meshed, high voltage transmission and delivers it to customers [3]. It is normally divided into *primary* and *secondary* systems. The first one takes the higher voltage from the transmission or sub-transmission circuits and uses step down transformers to send it through several primary circuits, which fan out from the substations and have voltages usually at the

order of 600 V to 35 kV. When close to each end user, a distribution transformer lowers the voltage even further to the well-known values of 120/240 V (or 127 V/220 V as more common in Brazil). In this work, we focus on the primary distribution system.

2.1.1 Description of Distribution Systems

A power distribution system is typically composed of [9] (i) HV/MV power transformers; (ii) a MV substation; (iii) MV power cables; and (iv) MV/LV power transforms, wherein HV, MV and LV stands for "high", "medium" and "low" voltages, respectively. This is illustrated in Figure 2.2. The HV/MV transformer steps down the energy that comes from the transmission (or sub-transmission) system, and it goes out from the substation along circuits called *feeders* to energize the loads in the primary distribution system. Most industries and large condominiums are fed directly by this medium voltage[1], while, for others customers (like residences), the voltage is stepped down even further to LV before the energy is distributed.

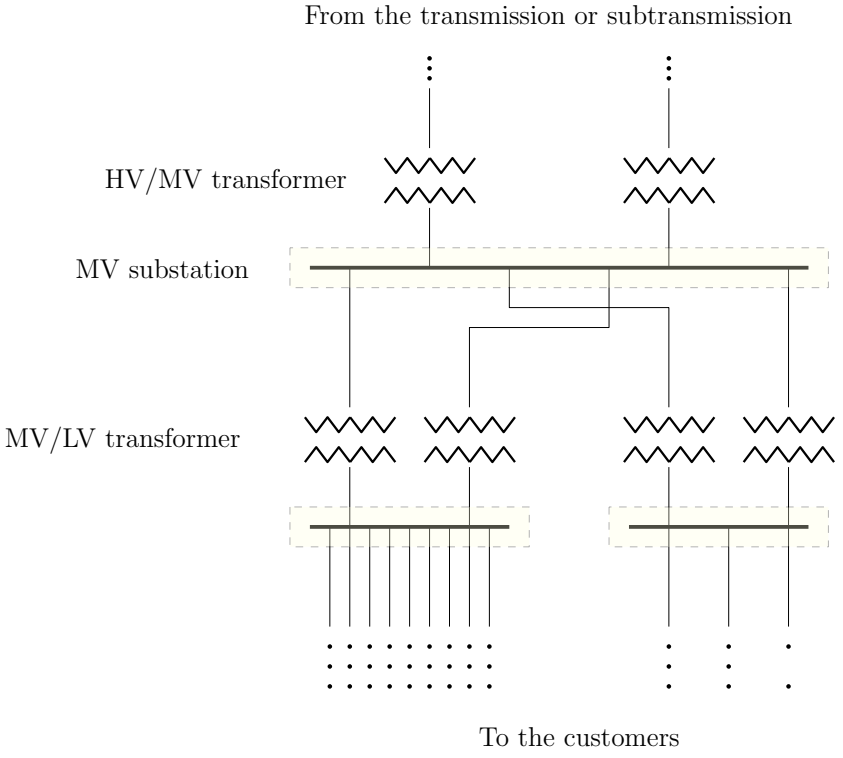


FIGURE 2.2: Distribution system overview. The high voltage coming from the (sub)transmission system is stepped down in the substations, from which the energy fans out in the primary system. When closer to the customers, the voltage is stepped down even further in the secondary system.

[1]Some industries are even energized by the sub-transmission system, which shows us how fuzzy the border between transmission and distribution can be in real systems.

The distribution systems can come in many different configurations and circuit lengths, but they are fundamentally classified into [10] *radial* and *meshed*. In short, radial systems have a single path from the source (the substation) to each load, while meshed systems may have more than one. This is shown in Figure 2.3. There are more complex configurations, like primary network and dual-service network [10], but they can be seen simply as variations of these two.

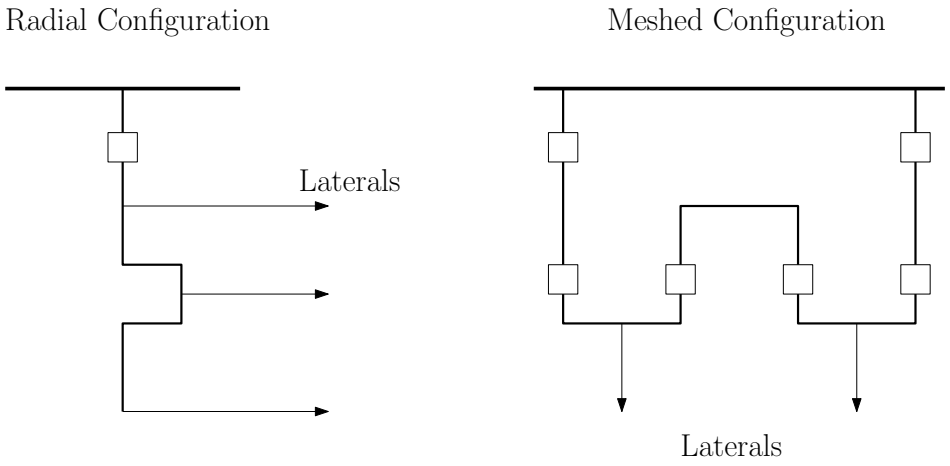


FIGURE 2.3: Simple examples of radial and meshed configurations of distribution systems. All laterals in the left configuration are energized by only one feeder, while the ones on the right can be fed by two different circuits.

This work deals exclusively with the radial topology. The top left panel of Figure 2.4 shows a portion of a radial distribution network with some of its many components, such as circuit breakers, fuses, transformers, sectionalizing switches etc. Despite their importance, it is actually cumbersome to analyze examples with too much information[2], so in this work a more abstract version based on graphs (see appendix A for a short overview on graphs) will be preferred, as shown at the top right panel of the figure. The complete network is now abstracted into a graph $\mathcal{G}_D = (\mathcal{B}, \mathcal{L})$, with the nodes $\mathcal{B}$ indicating buses (sources, loads or virtually any shunt element) and $\mathcal{L} = \mathcal{L}^{CC} \cup \mathcal{L}^{CO}$ the complete set of branches (comprehending distribution lines, transformers, phase-shifters, or any element connecting two buses). In this notation, $\mathcal{L}^{CC}$ is the set of all elements that are currently in operation, that is, currently closed (CC), and $\mathcal{L}^{CO}$ contains the set of components that are disconnected, i.e., currently open (CO), mainly maneuverable switches. Notice that the present configuration is completely described by the closed connections, but the open ones are also illustrated in a dashed version for completeness. Also, the source nodes merely indicate a bus connecting the output of a substation, which were abstracted into the so-called "sources" for simplicity. It does not mean that there is one substation (or transformer) per feeder – in fact, there can be many feeders radiating from a single substation.

[2] And I guarantee they are even more cumbersome to draw.

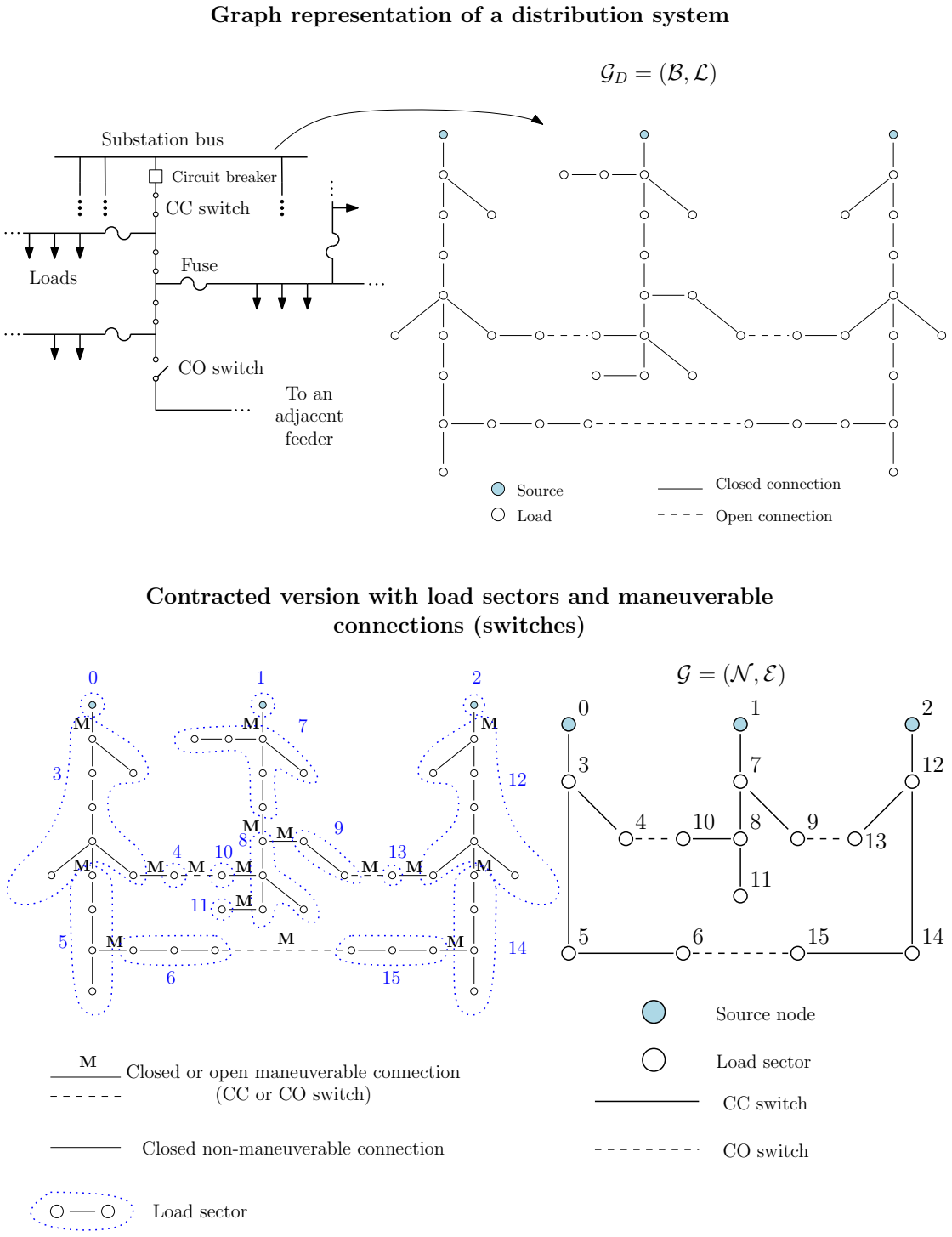


FIGURE 2.4: **Top:** A typical distribution network and its abstraction into a graph, where each node represents a bus and each edge indicates a connection. **Bottom:** Network contraction obtained by allowing only connections that can be opened or closed (containing an **M** in the bottom left graph). Each node now represents a sector, and the edges indicate maneuverable switches.

Even if this representation is already simpler, as will be evident later, the problem handled in this work requires the knowledge of which connections can be opened or closed, i.e., which lines contain any kind of maneuverable *switch* (such as sectionalizers

and sometimes even circuit breakers and reclosers). Thus, we can go even further in the abstraction and prefer a *reduced* or *contracted* version as shown in the bottom of Figure 2.4. The contracted graph is given by $\mathcal{G} = (\mathcal{N}, \mathcal{E})$, where the edges $\mathcal{E} = \mathcal{E}^{CC} \cup \mathcal{E}^{CO}$ indicate only branches with switches, and the terminology is similar as before, and the loads without any maneuverable connection among themselves were grouped (contracted) into a single node called sector, with $\mathcal{N}$ the corresponding set of all sectors. The graphs $\mathcal{G}$ and $\mathcal{G}_D$ will be often called "abstraction of the *complete graph*". However, keep in mind that a given configuration is described by its closed components, so we may occasionally talk about the subgraph $\mathcal{G}^{CC} = (\mathcal{N}, \mathcal{E}^{CC})$ (and similarly for $\mathcal{G}_D^{CC}$) in these situations.

The information contained in the reduced graph $\mathcal{G}$ is relevant for the bulk of this work, so we will naturally refer to it as the representation of a distribution system. The complete version $\mathcal{G}_D$, however, is still relevant for processes working "under the covers", such as power flow calculations (for that reason Appendix C, which presents a short introduction on power flow calculations, follows this notation).

Radial configurations often have many advantages over the other types, such as [3]:

- Easier and cheaper fault current protection;
- Lower fault currents over most of the circuit;
- Easier voltage control;
- Lower cost.

In spite of the benefits of radial topology, because each customer is fed by a single path, whenever there is an outage (either caused by a fault or maintenance) that interrupts a portion of the feeder, all of the loads downstream to this point become *out of service* (*oos*) .[3] Hence, the effects of an interruption are spread to a greater extent. This is less severe in the case of meshed systems because, even if a path from a load to the source is blocked, there may be other functioning feeders to energize it. Of course, this comes with the cost of more difficult analyses and protection coordination.

2.1.2 Reliability indices

Distribution reliability can be understood as the ability of the distribution system to perform its given task under stated conditions for a stated period of time without failure

[3]This assumes that there is no Distributed Generation or it is equipped with a passive protection that disconnects it from the system in case of faults. See the discussion later in section 2.1.3.2.

[11], or, in simple terms, the ability to supply energy continuously. This somewhat abstract definition is usually quantified by means of *reliability indices*. Regulation agencies are moving to a trend in which utilities are penalized or sometimes rewarded depending on these indices [11], so there is a great incentive to keep them at reasonable values.

Service interruptions are usually quantified by two reliability indices [3, 11]:

SAIFI *System Average Interruption Frequency Index*, which provide information about the average number of interruptions:

$$\text{SAIFI} = \frac{\text{Total number of interruptions}}{\text{Total number of affected customers}} \tag{2.1}$$

SAIDI *System Average Interruption Duration Index*, which describe the total time (in minutes or hours) that loads became out of service due to interruptions:

$$\text{SAIDI} = \frac{\sum \text{Customer interruption duration (hours or minutes)}}{\text{Total number of affected customers}} \tag{2.2}$$

Notice that these indices are valid for sustained outages only (i.e., the ones that last more than a given time, normally five minutes), and are measured for a given time window (one month, trimester, year etc.). Also, there are other indices that may be useful in other situations, such as CAIDI and ASAI [3, 11], but the described ones are enough for the purposes of this work.

In Brazil, we have FEC and DEC as equivalent indices to SAIFI and SAIDI, respectively. These are roughly translated to "Equivalent Frequency of Interruption per Constumer Unit" and "Equivalent Duration of Interruption per Constumer Unit" [12]. While comprehending the same idea, their formulation usually incorporates the number of affected customers, and can be stated as[4]

$$\text{FEC} = \frac{\sum_{i=1}^{\text{Number of interruptions}} \underset{\text{in the } i\text{-th interruption}}{\text{Number of affected customers}}}{\text{Total number of affected customers}} \tag{2.3}$$

and

$$\text{DEC} = \frac{\sum_{i=1}^{\text{Number of interruptions}} \underset{\text{in the } i\text{-th interruption}}{\text{Number of affected customers}} \times \underset{i\text{-th interruption}}{\text{Duration of the}}}{\text{Total number of affected customers}} \tag{2.4}$$

[4]Notice that, while DEC and FEC represent average indices among all constumers, there are the corresponding individual indices, FIC and DIC, which have the same formulation of equations (2.3) and (2.4) without the denominator, but valid for one customer unit only.

wherein the total number of interruptions again varies depending on the adopted time window (one month, one trimester, one year etc.). Once more, there are other indices that may be useful in other situations, but these are the most important in the context of this work.

Improving reliability indices means reducing the probability of interruptions and their duration. This can normally be accomplished by [11] reducing faults (tree trimming, arresters etc.); finding and repairing faults faster (e.g., adding automated equipment, as mentioned later in section 2.1.3); limiting the number of loads interrupted (with more fuses, reclosers and switches); and proposing an efficient *load restoration plan*. Of course, most of these tips are easier said than done, but this work provides a technique for helping with the last item. At least in terms of the duration index (SAIDI or DEC), the longer a node is out of service, the greater its contribution to this indicator, which provides a strong motivation for utilities to attempt to restore service as fast as possible.[5] In order to understand how this restoration is performed, let us consider what normally happens when a fault takes place.

2.1.3 Fault Detection, Location, Isolation and Service Restoration

Let us contemplate in this section an example of the complete process taking place in a distribution system when a fault happens until it is corrected. For that, consider the top of Figure 2.5, which shows a portion of a large system wherein a fault occurred at sector 2.[6] Assuming the protection is coordinated[7], the first protective device (edges with a **P**) to feel the failure and trip is the one in switch $(0, 1)$, which can be a recloser or a circuit breaker [13]. This action is normally performed automatically by means of relays, and it leaves the loads at nodes 1 to 11 out of service. What happens after that?

If the failure is temporary, the brief interruption of energy supply is enough to correct it, and protective devices are equipped with automatic reclosing features for these circumstances. If that is the case, after closing $(0, 1)$ the service is naturally restored for the whole feeder. In these situations there is no need for any of the remaining work of this text, so we may as well assume all faults considered are permanent.

[5] In principle, there is not much to be done in regards to frequency indices since, if a fault occurs, its contribution is accounted regardless of the quality of a load restoration plan. However, because of a time tolerance that starts to count a load as interrupted only after a given time ($\sim$ 3 minutes), Chapter 5 presents an approach for helping reducing even these indicators.

[6] Faults normally take place in distribution lines, usually connecting one or more conductor phases with the ground and/or with themselves. But since a sector is a collection of buses that are not connected by a maneuverable switch, understand "fault at a given sector" as "fault happening *somewhere* in a given sector".

[7] See Appendix B for a short introduction on the concepts of coordination of protection in distribution networks.

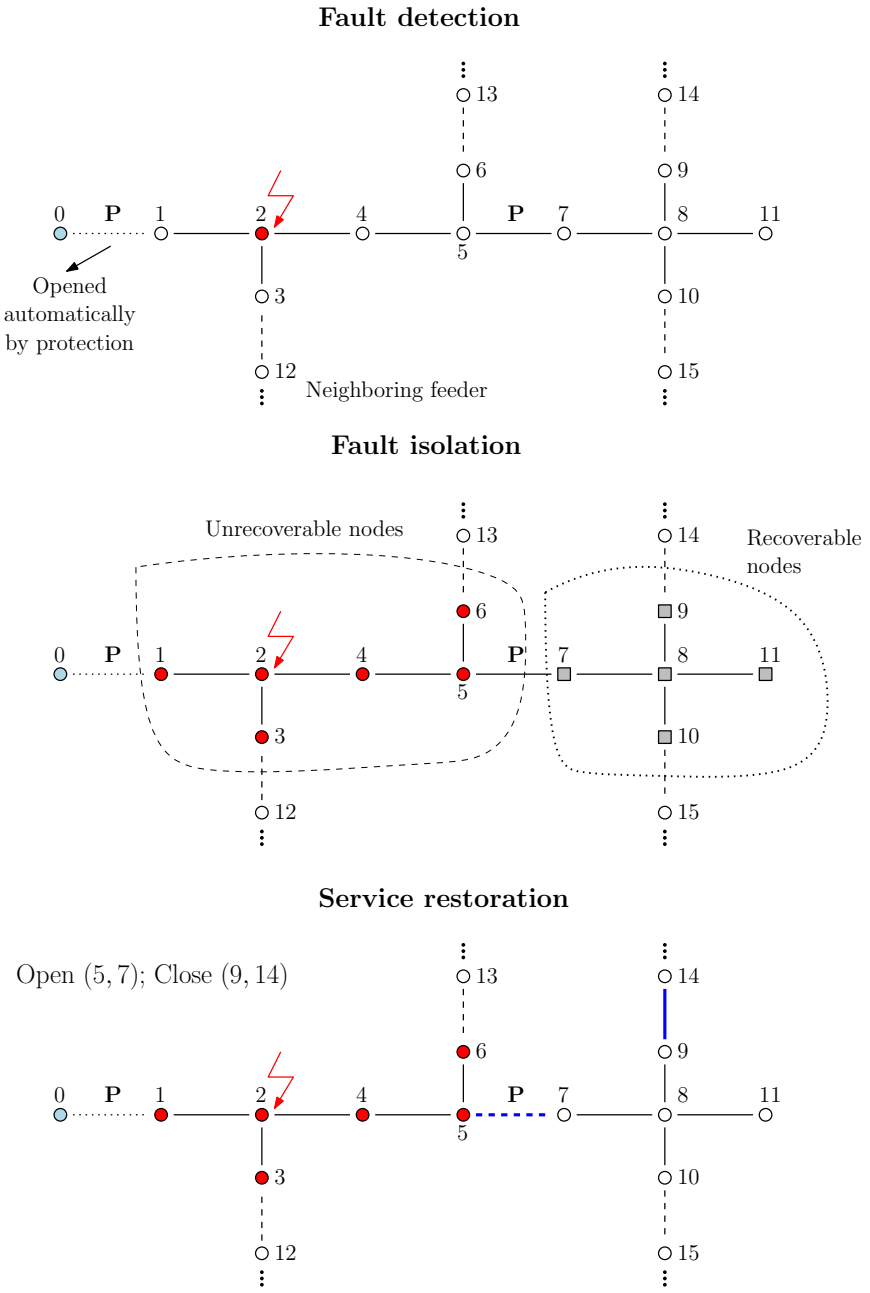


FIGURE 2.5: Example of Fault Detection, Location, Isolation and Service Restoration (FDLISR) in case of a fault with unknown exact location. **Top:** Fault detection by opening $(0, 1)$ and possible customers' complaints. **Middle:** Fault isolation dividing the *out of service* region into unrecoverable and recoverable portions. **Bottom:** Service restoration. In the figure, edges with a **P** indicate switches with a protective function.

Assume, then, that the fault persists after the predetermined number of reclosings of the protection switch. If the system is sufficiently automated, measurement devices provide on-line information about voltage and current in different parts, and there is a wide range of methods [14] that can be employed to detect fault symptoms. However, if that is not the case (which is not in most developing countries), the utility engineers are much probably not aware of the interruption. In fact, even the information that switch

$(0, 1)$ was opened may be lost in the middle of the numerous other alarm messages that arrive during the system's operation. Thus, the arguably most employed approach for detecting an outage is the "trouble call system" [15], which is simply a fancy name for the act of listening to customers' reports and complaints. This step will be referred to as *fault detection*, and, in this unfortunate case, the distribution utility will probably receive calls from customers among all sectors from 1 to 11 and will probably have to activate the automatic response feature in its call center.

Notice that we now know that a fault happened, but not exactly where. Then, unless someone provided the company with a very useful information such as "there is a tree on fire near a wooden pole in front of my house", in principle, a search would have to be conducted along the whole system to properly locate and repair it, and this may take a very long time depending on the feeder size and the on the characteristics of the fault. Fortunately, we can be smart in this case and remove some portions of the feeder from consideration. Revisiting the assumption that the protection scheme is coordinated, we can narrow the fault location to sectors 1 to 6 only. This is because, if the failure were to happen on any node from 7 to 11, then switch $(5, 7)$ would have been activated instead. With this, we can divide the whole out of service region into *recoverable* and *unrecoverable* portions (also referred to as *faulted* and *oos* nodes), as illustrated in the middle of Figure 2.5. In this case, until the fault is properly located, all nodes from 1 to 6 must be considered faulted, and the ones from 7 to 11 can be restored by neighboring feeders. This step will be denominated *fault isolation*, and in non-automated systems it can be performed before the outage is located, even with a lack of information like in this case.

With the proper isolation, the last step is the *Service or Load Restoration*, in which the *oos* recoverable sectors can be energized with neighboring feeders by opening and closing switches. The bottom of Figure 2.5 shows a restoration plan obtained by opening $(5, 7)$ to isolate the two regions and closing $(9, 14)$ to effectively transfer sectors 7 to 11 to a neighboring one. Notice that, unless the switches have a remote feature, dispatch teams must move and manually perform the openings and closings. Thus, this last step is normally not as simple as it seemed in this example and involves the job of deciding for a restoration plan and a proper coordination of these teams.

Keep in mind that, while this restoration step is being executed, another crew is searching for the fault in order to begin repairs. And remember: the complete process is not over until the problem is corrected. As a matter of fact, suppose the fault was properly located at sector 2 (this step will be known, for obvious reasons, as *fault location*). Now nodes 1, 3, 4, 5 and 6 can be moved to the recoverable zone and, depending on how

long the correction can take, the utility may find interesting to perform another Load Restoration step to restore more loads.

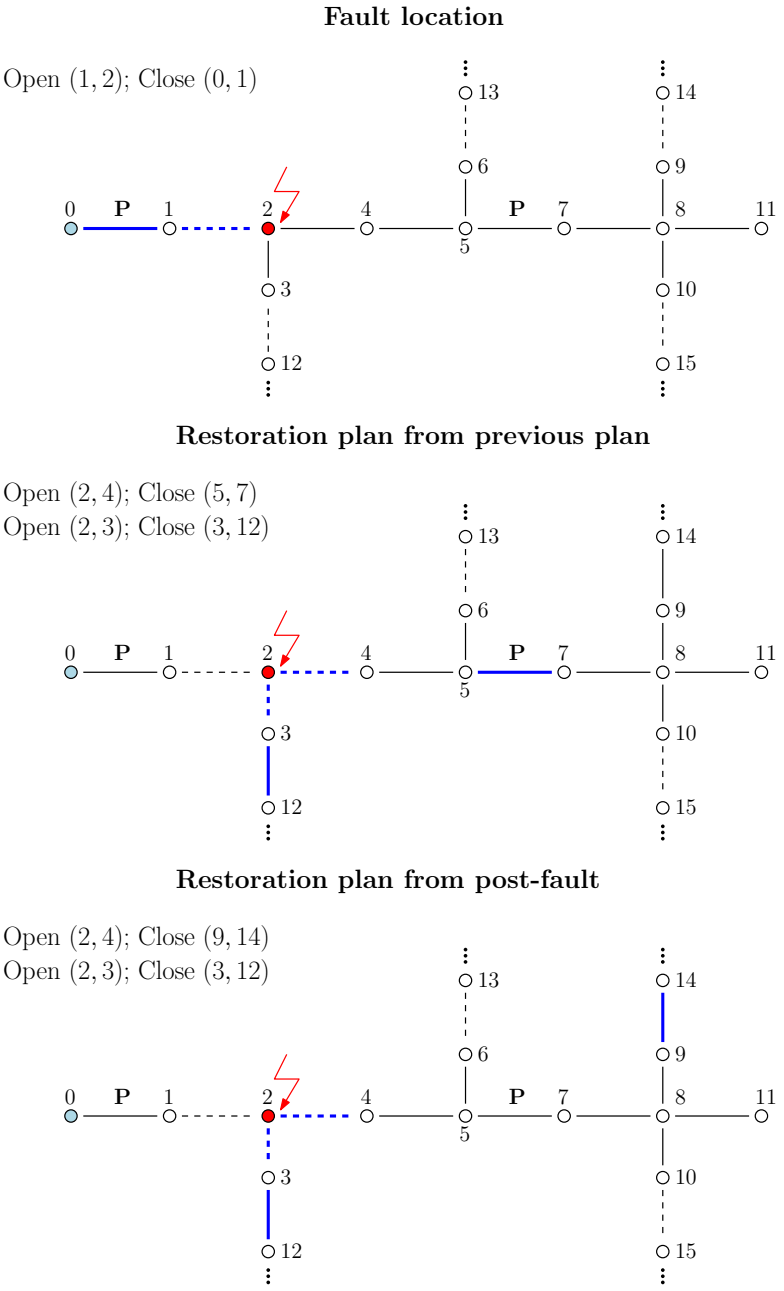


FIGURE 2.6: Example of FDLISR in case of a fault with known exact location. **Top:** Recovery of upstream *oos* nodes by reclosing $(0, 1)$ and isolating the fault by opening $(1, 2)$. **Middle:** Service restoration starting from previous plan of Figure 2.5. **Bottom:** Service restoration starting from the post-fault.

As shown at the top of Figure 2.6, by opening $(1, 2)$ (which was probably already done to better isolate the fault after its location), one can reclose $(0, 1)$ to restore all loads upstream to the fault belonging to sector 1. Now, determining the new restoration plan will depend on the current configuration of the system, that is, if a sequence of maneuvers was performed or not before the fault location. The middle and bottom panels

of Figure 2.6 shows examples assuming we start from the previous plan at Figure 2.5 and starting from the post-fault, respectively. Notice that, even if the overall operated switches are different, the final configuration is the same in both cases, but this was just a coincidence in this example. In any case, the main take-away from this discussion is the importance of a proper fault location in improving the reliability of the system and providing the Service Restoration step to recover more loads.

The complete process just described can be referred to as Fault Detection, Location, Isolation and Service Restoration[8] (FDLISR), and it is schematically depicted in Figure 2.7. It covers the possibility of outages with initially known (e.g., in case of programmed interruptions for maintenance) and unknown location (e.g., in case of faults). Each individual step deserves its own research field, and the interested reader can find further information in references such as [14, 16]. In the figure, the "Stretch Isolation" step corresponds to isolating the smallest stretch of the distribution system thanks to a proper protection coordination (middle image in Figure 2.5), which is possible even if the fault is not exactly located. We use this denomination to prevent confusions, but it can be seen as a form of "fault isolation". Lastly, once the fault is properly corrected, an additional sequence of maneuvers should be performed to bring the system back to the normal configuration. This is shown as a last step in Figure 2.7, but only for completeness, as it is not treated in this text (although the method proposed in Chapter 4 can be helpful with that, which something left for future studies).

The current work deals exclusively with the Load or Service Restoration portion, so the only required information is the knowledge of which nodes are recoverable and which are unrecoverable. Notice that only this information can handle all the instances where SR is required in Figure 2.7. The reason for this detailed example is to provide a (hopefully) useful context to the reader in helping locating where the main contribution of the current work lies.

Finally, in terms of notation, the terminology "Service" seems more common in automated distribution systems or when dealing with the complete FDLISR process. Therefore, we will adopt "Load Restoration" in this work, which seems a more popular term in works similar to this one.

[8] This abbreviation is not universal, tough. Some works ignore the "detection" part, others lump "location" together with "isolation" etc. Given the previous discussion, I decided to explicit all of the steps in the name.

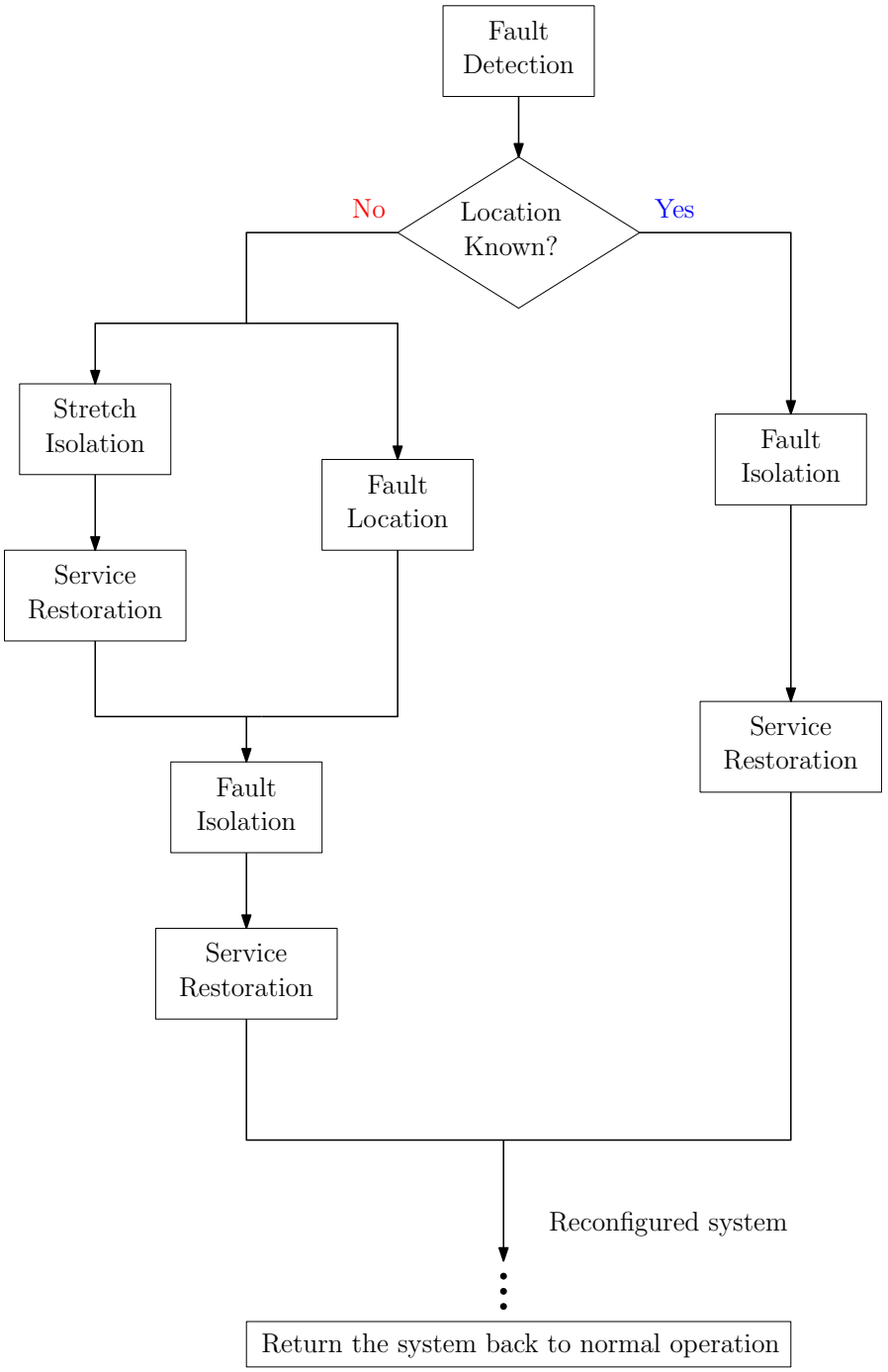


FIGURE 2.7: Fault Detection, Location, Isolation and Service Restoration scheme.

2.1.3.1 A case for automated systems

Before moving on, it should be instructive to compare how the previous example would go in a more automated system. Automated switches are equipped with three remote-controlled components [17]: an actuator to change its open/closed status; a position detector to output its current open/closed status; and a fault detector to sense faults. With this, let us see how the previous steps of FDLISR can be improved.

Fault detection tends to be easier with sensors (unless they fail, as discussed next), so usually there is no need for the trouble-call approach. In normal operation, these components constantly indicate whether or not there is a fault downstream to their respective switch. In Figure 2.5, assuming all switches are automated, then $(0, 1)$ and $(1, 2)$ would sense a fault, but not the remaining ones. Thus, for fault location, the problematic sector will be below the most downstream edge displaying an issue – here $(1, 2)$ – and above the most upstream switches indicating nothing is wrong – here $(2, 3)$ and $(2, 4)$.[9] Fault isolation then would follow just as naturally: open the switches immediately downstream the pinpointed faulted location – in this case, $(2, 3)$ and $(2, 4)$. Finally, service restoration would be executed with the same set of rules, trying to restore as much recoverable *oos* sectors without violating operating constraints. The difference is that coordinating a dispatch crew would be unnecessary, and the process would be much faster.

This quick discussion shows the benefits of system automation, which is why most developed countries invest in technologies such as smart grids. However, do not think that automated systems are always flawless and do not have their own set of issues. In fact, the three remote-controlled components of a switch may work under abnormal conditions and "lie" about their statuses [17]. Some topics to consider in these cases are:

- The correct detection of a fault depends on the underlying method implemented to properly diagnose its presence (see [14] for a review of this subject);
- A switch may not open/close properly after ordered to do so, and may even signal incorrectly that the operation was performed, which can hinder the fault isolation and service restoration steps;
- The fault location process can be compromised by incorrect information.

As an example of the last issue, consider Figure 2.8 with the same previous feeder but now with everything automated. Assume both detectors at $(2, 3)$ and $(2, 4)$ are signaling the presence of a fault. Under normal conditions, there should be faults at sectors 3 and 4 (as shown in the bottom right panel), and both switches would need to be opened for proper isolation. However, under uncertain conditions, there are actually four possibilities as illustrated in the figure, each one with their own set of decisions and outcomes. And what are the implications of a bad decision? For instance, if there

[9] And what about multiple failures at sectors 1 and 2, when both switches $(0, 1)$ and $(1, 2)$ would detect a fault in the same way? Well, we need compromises. We can be conservative and open $(0, 1)$ for safety until a dispatch team confirms the presence or absence of the failure; we can assume the probability of multiple faults is too small and take the risk (which is normally o.k. with underground systems); and so on. There is no general rule to be applied in every system every time, which is what makes this problem interesting (and hard).

were only a fault at 3 and we believed in both sensors, then $(2, 3)$ would be correctly opened, but $(2, 4)$ would unnecessarily leave sectors 4 to 11 out of service. The reader can work out many other possible outcomes in this simple example. Now imagine how such possibilities grow in larger systems during a fault with much more sensors under uncertainty.

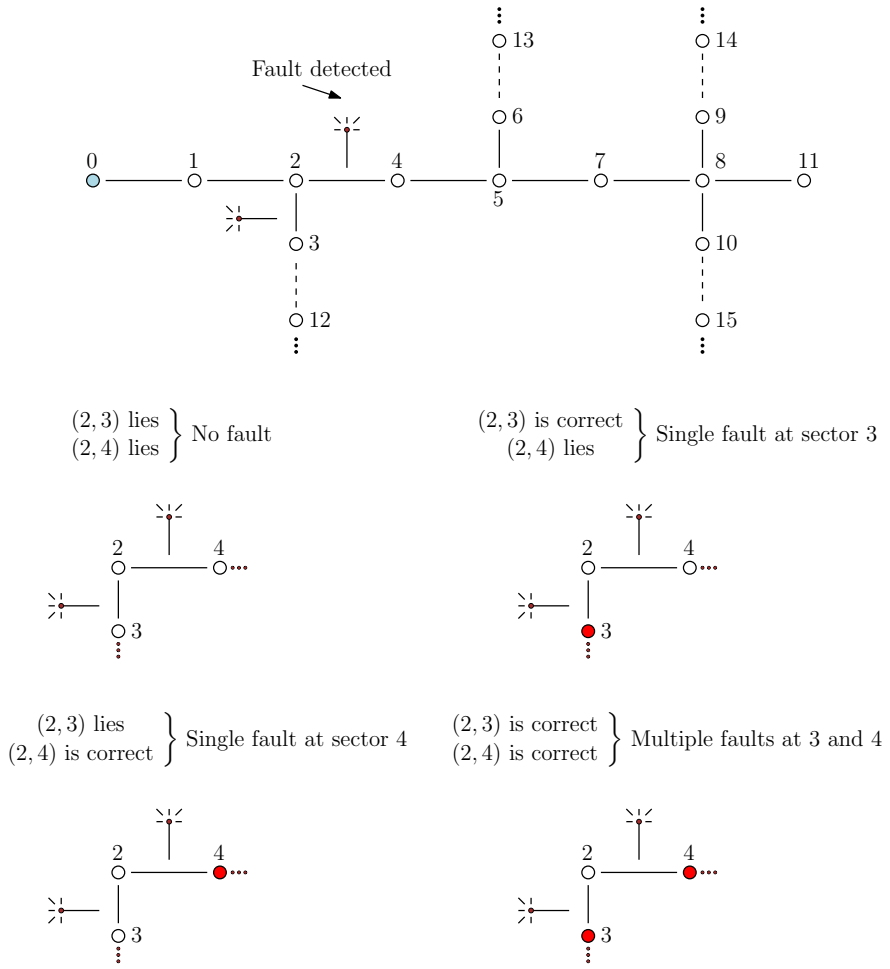


FIGURE 2.8: Difficulties when handling abnormal conditions of fault detectors in automated systems.

While interesting, this problem and the whole FDLISR topic in automated systems are outside the scope of this text, but the interested reader can check some studies such as [15, 17–20]. In this work we try to address distribution systems with low levels of automation, where the discussion of the previous section applies at most (with exception of some remote-controlled switches).

2.1.3.2 Another case, considering Distributed Generation

Distributed Generation (DG), also known as embedded generation or even co-generation, refers to any kind of power generation at distribution level [3]. In contrast with conventional power stations, which are centralized and often require a transmission system due to their geographical distance to customers, DG units are decentralized, modular and located relatively close to the loads they serve.

DG systems often adopt renewable sources, such as solar power, wind power, geothermal power, biomass, biogas etc. Albeit normally referring to units with small capacity (typically up to 10 MW), there are no universally accepted definition of sizes and voltages, and sometimes even subtransmission-level generation can be considered as distributed generation [3]. For simplicity, this work considers as DG any kind of generating unit outside the main utility substation.

Distributed generation continues to advance in terms of performance and low cost, and they provide a number of benefits to both the utility and customers, such as reduced power losses and voltage drops. Also, with the proper configuration, they improve power quality and reliability, as some loads can still be energized for some time after being disconnected from the substation due to an outage. However, despite all of these benefits, DGs are still not very friendly in the restoration context, which is probably due to the fact that radial distribution systems were designed for one-way energy flow [3]. For instance, protection coordination and power flow analyses become more complicated (as explained in Appendices C.2.1.1 and B.1.1), but the arguably most relevant complication occurs in the presence of unintentional *islandings* [3].

An island happens when a portion of a system with one or more DGs operate separately from the rest of the utility system. At first, it may look like a desirable situation: loads are still being energized despite the disconnection, which does not impact the reliability indices and makes customers and utility happy. However, when these islands are not intentional (think the act of a relay due to a fault, for example), this phenomenon introduces the following concerns [3]:

- Safety risks, as line crew may work in energized portions of the system thinking they are safe due to being disconnected from the main source. This danger also extends to general public;
- Voltage problems as a result of neutral shifts and ferroresonance;
- Once in island forms, DG units tend to drift out of phase with the main utility. When reconnected, this phase difference can damage equipment and cause severe power quality disturbances;

- The fault may not be cleared properly and continue to be energized due to downstream DG. For example, a fault in sector 7 in Figure 2.5 may open switch (5,7) to disconnect to the main power source. However, if a possible DG unit in, say, sector 11 keeps running, it will keep feeding the fault and may even cause problems for these loads due to new overcurrents.

Therefore, in these cases, the arguably most prudent act is to prevent unintentional islandings. The most common way is called passive protection, which uses voltage and frequency relays to trip a DG unit whenever either of these parameters migrate outside a specified window. The curious reader can see more information in [3].

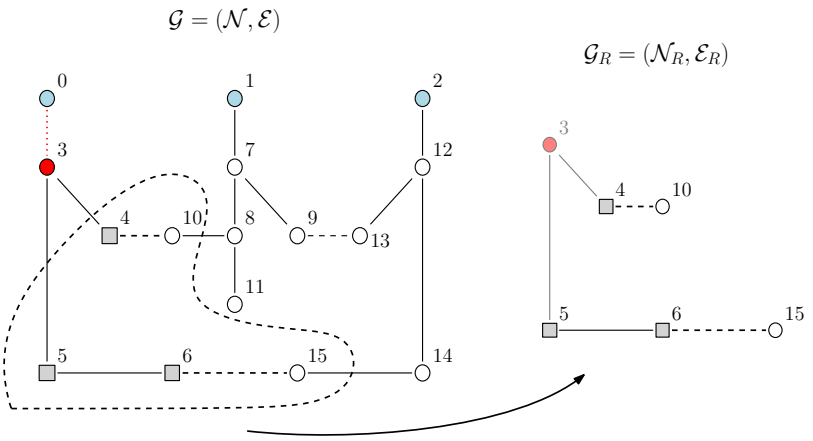
Even with unintentional islands being prevented, it is still an open issue how to handle the load restoration problem appropriately in the presence of distributed generation. Indeed, how to model the load point of DGs *during* and *after* the service restoration step if we only have their measures *before* the fault? With that said, the safest route I can take – at least for now – is to consider a DG unit as turned off until the fault is corrected and the system is back to normal operation.

2.2 Load Restoration Problem

After the quick introduction in Chapter 1 and the general description in the previous section, the reader might now have a better understand of what the load restoration problem is and where it stands in an industry application. In any case, let us recapitulate: it consists in determining a *sequence of maneuvers* that restore the most *oos* loads in the shortest time possible without violating constraints such as voltage and current limits in buses and lines, feeder capacity and system radiality. The purpose of this section is to present the required terms and abstractions used throughout the text in order to provide an appropriate mathematical formulation for this problem.

2.2.1 Restoration sub-graph

The *restoration sub-graph* $\mathcal{G}_R = (\mathcal{N}_R, \mathcal{E}_R)$ is, as the name implies, a sub-graph of $\mathcal{G}$ formed by (i) all nodes which are out-of-service; (ii) all nodes from neighboring feeders that are able to re-energize *oos* nodes; and (iii) all edges connecting those nodes. These concepts are illustrated in Figure 2.9. Notice that the faulted nodes and the edges adjacent to it do not belong to this sub-graph, but in most of the examples they will still be shown in transparency so the reader does not forget that these switches have the important task of isolating the fault in case a neighboring node is energized.



$$\mathcal{E} = \{(0,3),(1,7),(2,12),(3,4),(3,5),(4,10),(5,6),(6,15),(7,8),(7,9),(8,10),(8,11),(9,13),(12,13),(14,15)\}$$
$$\mathcal{E}^{CC} = \{(0,3),(1,7),(2,12),(3,4),(3,5),(5,6),(7,8),(7,9),(8,10),(8,11),(12,13),(14,15)\} \subset \mathcal{E}$$

$$\mathcal{E}_R = \{(4,10),(5,6),(6,15)\}$$
$$\mathcal{E}_R^{CC} = \{(5,6)\} \subset \mathcal{E}_R$$

FIGURE 2.9: A network $\mathcal{G} = (\mathcal{N}, \mathcal{E}^{CC})$ together with its restoration subgraph, $\mathcal{G}_R = (\mathcal{N}_R, \mathcal{E}_R)$.

This graph is useful in methods handling only the *oos* nodes directly, as exemplified in the top left example of Figure 2.10. It will be useful later when presenting the proposed method and when reviewing some works in Chapter 3.

2.2.2 Set and Sequence of maneuvers

Remember that the distribution system is abstracted here as a graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$, with a set of sectors $\mathcal{N}$ which is fixed during the process, and a set of connections $\mathcal{E}$ indicating maneuverable switches. If we remember that a given configuration is provided by the currently closed connections, that is, by the sub-graph $\mathcal{G}^{CC} = (\mathcal{N}, \mathcal{E}^{CC})$, reconfiguring a network then means opening and closing switches, which, by its turn, mean adding edges to $\mathcal{E}^{CC}$ or removing from it.

Given a restoration plan, the switches that are effectively operated can be enclosed in a subset $\mathcal{M} \in 2^{\mathcal{E}}$, referred here to as a *set of maneuvers*, as shown in the examples of Figure 2.10, repeated here from Chapter 1 for convenience. The notation $2^{\mathcal{E}}$ indicates the power set of $\mathcal{E}$, i.e., the set of all subsets of all elements of $\mathcal{E}$ including the empty one. This is simply a mathematical way of saying that we normally do not operate all available switches during a restoration plan, and the empty set represents the "do nothing" solution, which sometimes may be the only option.

The set of maneuvers provides the correspondence between the post-fault and the reconfigured networks. The notation $\mathcal{G}^{CC}(\mathcal{M}) = (\mathcal{N}, \mathcal{E}^{CC}(\mathcal{M}))$ will be used to represent the new configuration [notice that it makes no sense to use the complete graph $\mathcal{G}(\mathcal{M})$

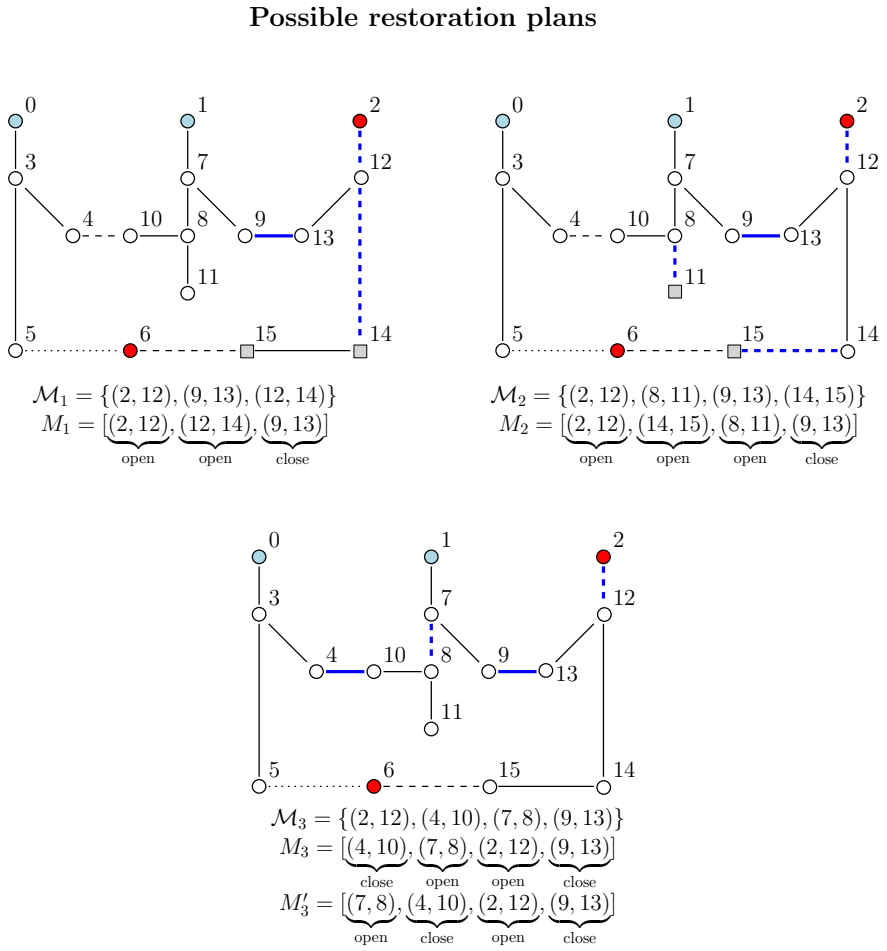


FIGURE 2.10: Examples of restoration plans (repeated from Chapter 1). Below each plan are shown the sets of operated switches and possible sequences in which each maneuver is performed.

as it already contains all maneuverable edges]. The nodes are fixed for a given network, but the new set of currently closed edges is given by

$$\mathcal{E}^{CC}(\mathcal{M}) = \{e \in \mathcal{E} : e \notin (\mathcal{E}^{CC}(\emptyset) \cap \mathcal{M})\} \tag{2.5}$$

that is, the switches in $\mathcal{M}$ should be removed/added if initially present/absent. By extension, $\mathcal{G}^{CC}(\emptyset) = (\mathcal{N}, \mathcal{E}^{CC}(\emptyset))$ indicates the post-fault configuration.

Another very important parameter is the *sequence of operations or maneuvers* $M \in \mathfrak{S}_{2\mathcal{E}}$, which gives a proper order in which each maneuver is executed. Notice that if a set $\mathcal{M}$ has $|\mathcal{M}|$ operations, then a sequence may consist of any permutation of these elements. Then, the notation $\mathfrak{S}_{2\mathcal{E}}$ is used with the purpose of indicating the set of possible permutations [21] of all switches in each set of maneuvers.

Figure 2.10 shows examples of sequences for each of the plans. Regardless of the order, if M and $\mathcal{M}$ have the same elements, the final configuration will be the same, so the notation $\mathcal{G}^{CC}(M) = (\mathcal{N}, \mathcal{E}^{CC}(M))$ has the same meaning as before. Nevertheless, there are operational differences between these two entities. To understand that, we first define a given configuration $\mathcal{G}^{CC}(\mathcal{M})$ [or $\mathcal{G}^{CC}(M)$ if $\mathcal{M}$ and M have the same elements] as a *feasible configuration* if it is radial and respect the voltage, current and feeder overload constraints [see a mathematical formulation in equation (2.11)]. With that established, differently from the set of maneuvers – which is by definition an unordered collection of switches to be operated –, determining a proper sequence of operations is important for the following reasons:

- In general, even if the final configuration is feasible, switches cannot be maneuvered in any order because some *precedence rules* [1] should be respected. For instance, $(2, 12)$ and $(12, 14)$ in the top left plan of Figure 2.10 must be opened before $(9, 13)$ is closed, otherwise the fault would be energized or more load than expected would be fed momentarily, causing overloads in the supporting feeder. Similarly, in the bottom plan, whether to close $(4, 10)$ or open $(7, 8)$ first during the load transfer is a decision provided by the utility and should also be respected in the final sequence. In summary, we call a *feasible sequence* one that generates feasible configurations in each incremental step. Thus, given a set $\mathcal{M}$, not all $|\mathcal{M}|!$ possible sequences induced by it can be employed as a restoration plan because in practice the operations are usually performed incrementally.
- The total time or energy not supplied (explained later in section 2.2.3) depends on the order in which each maneuver is performed. This has a direct impact on the reliability indices (specially SAIDI), which may change depending on which loads are kept unserved for longer time.
- Finally, engineers need to issue commands to dispatch teams to perform the operations. Hence they require not only the final configuration, but the steps to be followed in order to achieve it.

According to the first property, a useful sequence M must be one that never violates the network constraints after each maneuver. This leads to the conclusion that the dispatch engineer has the choice of not implementing the complete sequence M, but maybe only the first i steps, which we refer here to as $M^{(1:i)}$. Thus, a sequence with $|M|$ maneuvers can be broken down into $|M|$ sub-sequences.

A number of works deal directly with openings and closings, particularly the ones based on rule-based heuristics, as mentioned later in Chapter 3. However, other approaches

based on more complex methods (such as some metaheuristics and mathematical programming) employ an abstraction of the network into a vector $\mathbf{x} \in \mathbb{X}$, such as a binary codification where each switch is assigned to a value zero/one indicating the absence/presence of a switch in the final configuration [22]. Other possible codifications include forest-based [23, 24] and permutation of switches [2, 25]. The main benefit of working with $\mathbb{X}$ instead of handling the maneuvers directly is the availability of specific properties or neighborhood structures that the optimization algorithm can use to perform a more efficient search. On the other hand, while the correspondence between a vector $\mathbf{x}$ and a set of maneuvers $\mathcal{M}$ is usually evident in many cases (e.g., in a binary or forest-based codification), the mapping to a proper sequence M tends to be neglected, or postponed to the end after a "good" configuration is obtained. It can be argued that this may be a bad practice because if we perform the optimization in terms of $\mathcal{M}$, we cannot employ quality indices requiring M, which tend to be more realistic. Then, one of the purposes of this work is to propose a decoding process to allow for the search to be in terms of $\mathbb{X}$ while optimizing in terms of such indices. We will go further into this topic later in Chapter 4.

2.2.3 Quality indices

In order to characterize a restoration plan M as good or bad, we need a proper set of quality indices to be employed as objective functions. Despite the common understanding that a good restoration plan should recover the most *oos* loads in the shortest time possible, there is no universal agreement on which functions better model such requirements. In this section we analyze the arguably most popular indices in the literature.

2.2.3.1 (Weighted) Power not restored

This is possibly the most straightforward index to represent the "recover the most *oos* loads" requirement of a restoration plan. Assume each node $n \in \mathcal{N}$ has a demand power S_n^D associated to its load value[10] and a priority w_n, which can be, e.g., 1 for regular loads and a large number, such as 100, for priority consumers, such as hospitals, commercial centers etc. If, after implementing a solution M, the resulting network $\mathcal{G}^{CC}(M)$ has a set $\mathcal{N}_{oos}(M) \subseteq \mathcal{N}$ of *oos* nodes (which include the faulted [not recoverable] ones), then the (weighted) power not restored is given by

[10] See Appendix C for a detailed presentation of load modeling. In case another model apart from constant power is applicable, we can use the equivalent polynomial load with nominal voltage for estimating S_n^D.

$$S_{NR}(M) \triangleq \sum_{n \in \mathcal{N}_{oos}(M)} w_n |S_n^D| \tag{2.6}$$

To get a better picture of this index, it may be more interesting to check the $S_{NR}(\cdot)$ profile as a sequence of maneuvers is implemented incrementally. For that, see Figure 2.11, which shows at the top a restoration plan to recover a faulted network. Node 11 could not be recovered due to overload constraints. For simplicity, suppose all nodes demand the same complex power and have the same priority.

Two possible feasible sequences are provided to arrive at the new configuration. Observing the $S_{NR}(\cdot)$ profiles for each one, we can see that this value decreases when we recover *oos* nodes and it remains the same in case of *oos* load shedding. However, as also shown in the figure, the final $S_{NR}(\cdot)$ value is the same after a given set of maneuvers were performed, irrespective of the order. Thus, in this case, we can write $S_{NR}(M_1) = S_{NR}(M_2) = S_{NR}(\mathcal{M})$, with $\mathcal{M}$ a set with all elements of M_1 or M_2, and this is true for any other feasible sequence with the same elements in $\mathcal{M}$. This makes this index dependent only on the final configuration, and given its wide use, it may suggest an explanation for the lack of attention towards providing a proper sequence of operations in the literature.

2.2.3.2 Number of maneuvers

The "in the shortest time possible" portion of a "good" restoration plan is usually translated as the total number of maneuvers in a sequence, defined simply as

$$N_m(M) \triangleq |M| \tag{2.7}$$

wherein $|\cdot|$ gives the cardinality (number of elements) of a set (extended to a sequence in this case). It should be evident from the definition that this index also depends only on the final configuration, i.e., $N_m(\mathcal{M}) = N_m(M)$ if both have the same elements, regardless of the order in M.

The main reasons for using this index are probably its simplicity and the common sense that a plan with less operations tend to be executed faster than one with more maneuvers. Unfortunately, this is usually not true for two main reasons. First, it ignores the possible availability of *remote controlled* switches, which demands a negligible time when compared to *manual* maneuvers, these ones requiring a dispatch team to physically operate in a given location.

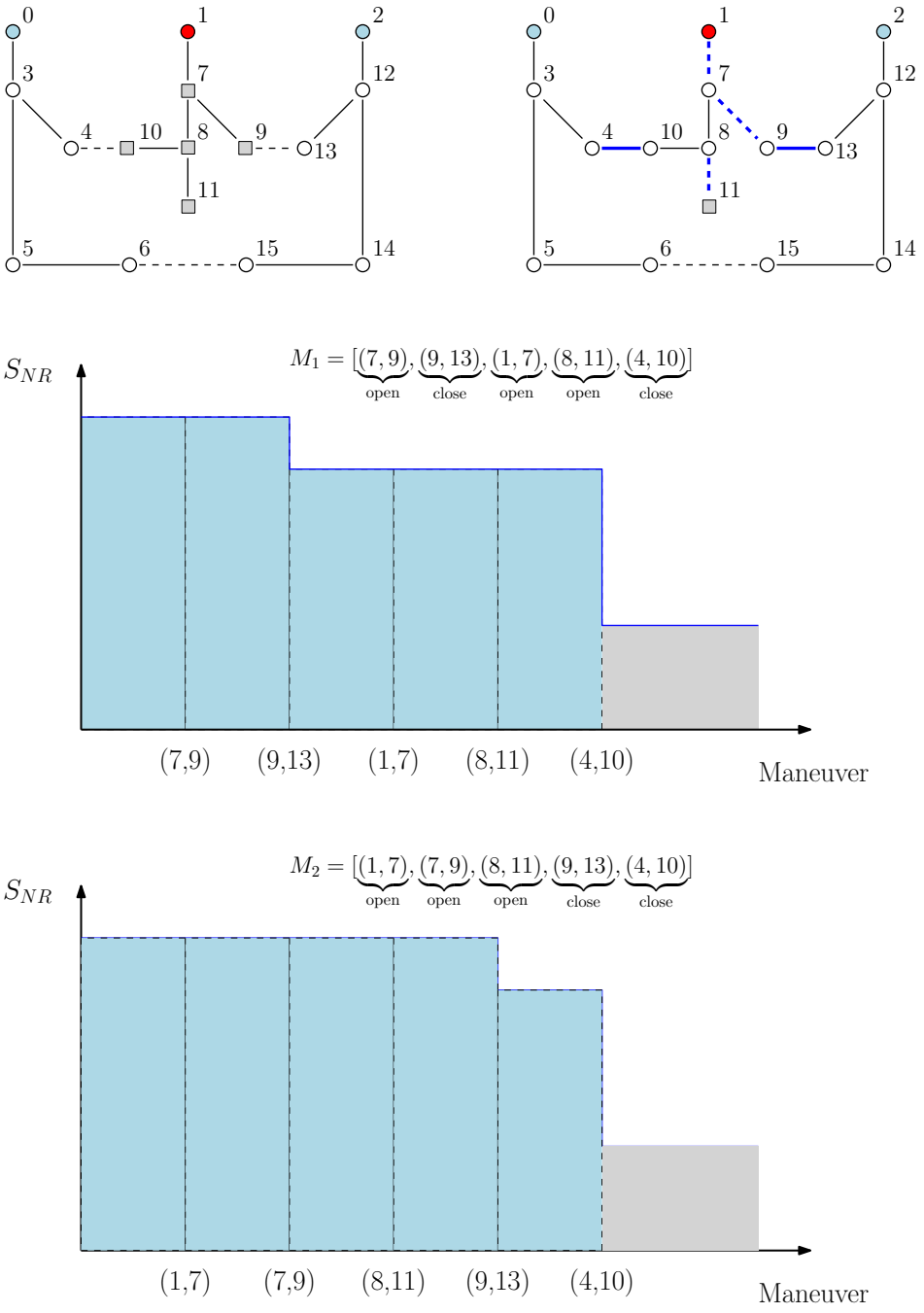


FIGURE 2.11: Profile of the (weighted) Power not Supplied for two sequences of maneuvers. Even if the final $S_{NR}(\cdot)$ value is the same for sequences with the same maneuvers, its curve may be different.

Even by recognizing and handling the first matter, the second complication is failing to realize the actual differences of time between the remaining manual switches. In fact, it is possible that two or more switches can be operated faster than a single one if they happen to be close by or through a proper coordination of more than one dispatch team, when many operations can be performed almost simultaneously. We have shown in [1] (and will perform a similar analysis in Chapter 4) that using $N_m(\cdot)$ to model

the "restoration speed" may lead to deceptive or sub-optimal solutions, so we suggest avoiding this index in general unless the system is completely automated.[11]

2.2.3.3 Time of maneuvers

Assuming switch $e \in M$ is operated at instant $T_m(e)$, the total time of maneuvers can be simply written as

$$T_m(M) \triangleq \max_{e \in M} \{T_m(e)\} \tag{2.8}$$

that is, the total time is the instant when the last switch in M was operated. This index tends to be more realistic for the reasons explained earlier, but a proper estimation of $T_m(e)$ is supposedly uncommon in the literature. Even if a number of works do realize the benefits of considering the time over the number of maneuvers, apparently only a few studies [2, 25–27] try to incorporate it as an actual objective function. Three of them [25–27] attribute a fixed number t_e to each switch e, which is not a realistic approach because (i) it ignores the dependency on a given team's current position, that is, as a team moves to operate a switch, the time taken to perform other maneuvers will also change and a fixed value is not capable of modeling this; and (ii) the total time is the sum of the individual values t_e, which disregards the presence of more than one dispatch team and assumes a sequential operation. Our work in [2] is apparently the first to propose a constructive heuristic to properly model $T_m(M)$ for a given sequence M which takes both concerns into account, and this method will be better detailed in Chapter 4.

2.2.3.4 Energy not supplied

As the name implies, the energy not supplied can be understood as the amount of load demand that is not being served over time. If the horizontal axis of Figure 2.11 represents the time (and thus, each maneuver takes the same time interval to be operated in this example), then the area below the $S_{NR}(\cdot)$ curve is a good measure of this index. It can be mathematically described as

$$E_{NS}(M) \triangleq \sum_{i=1}^{|M|} S_{NR}(M^{(1:i-1)}) \times [T_m(M^{(1:i)}) - T_m(M^{(1:i-1)})] \tag{2.9}$$

[11] As a matter of fact, if the system had only remote switches, then $S_{NR}(\cdot)$ alone would probably be enough (assuming it returns a *feasible sequence*).

in which we split a sequence M into its $|M|$ sub-sequences. We define $M^{(1:0)} \triangleq \emptyset$, which represents the post-fault configuration and, thus, $S_{NR}(M^{(1:0)})$ is the power not restored when no maneuver is executed, and $T_m(M^{(1:0)}) = 0$.

In equation (2.9), the i-th term is represented graphically as a rectangle, which is the power not restored *before* the i-th maneuver was completed [hence the $(1 : i-1)$ super-script] times the time taken to perform the i-th operation. Notice that, unlike $S_{NR}(\cdot)$, the order of the maneuvers is relevant. In fact, the area of M_1 is smaller than that of M_2, which means that, even if it generates the same final configuration, same power not restored and requires the same number of maneuvers, it recovers energy more efficiently.

It can be argued that this index is the one that best complies with the restoration requirements, since it combines the information of power not restored and maneuvering time [2]; it considers the order in which the switches are operated, which is interesting for handling priorities and preventing unnecessary deteriorations in the reliability indices; and, with a few modifications, it can be adopted to estimate costs for the company [28]. However, if only partial restoration is possible without violating constraints, there remains an unrecoverable portion (referent to nodes 2 and 6 in Figure 2.11, but still applicable if the faulted ones are not counted) which lasts until the fault is corrected and the network is back to the usual configuration, and since this time is usually unknown, it is not straightforward to add this contribution to $E_{NS}(M)$. Some feasible ways for circumventing this setback are to (i) estimate a time T_{fault} to clear the fault and add this share like usual; (ii) include a penalty factor if complete restoration is not obtained [26]; and (iii) minimize simultaneously the recoverable portion and the remaining power not restored as two quality indices [2].

Relation with the Reliability Indices

At first, we may wonder why the reliability indices are not being minimized directly if that is the end goal anyway. The main reason is because they are normally averaged by the total number of customers that were affected in a given time frame, and this information is not available at the moment of a given interruption.[12]

Thus, if we overlook this information, the energy not supplied can be related to duration-based reliability indices. In fact, using the formulation of equation (2.9), it is very similar to a load based reliability index called ASIDI (Average System Interruption Duration Index) [3, 11], which is equivalent to SAIDI but scaled by the load of each customer.

[12] We may overcome this limitation by using the information from previous time frames, but this is a topic for a different work.

Unfortunately, as argued by [11], it may be very hard to measure total load interrupted (although we may estimate it with load curves as mentioned in Appendix C) when compared to number of customers in a given sector, which is why very few utilities track load-based indices such as ASIDI. If that is the case, we can replace $S_{NR}(M)$ to return the total number of customers that remain out of service after solution M instead of providing the total (weighted) power not restored. Thus, we end up minimizing the DEC index of equation (2.4) (without the averaging).

I decided to keep the formulation (2.9) with the power not restored for coherence with other works in the literature. However, keep in mind that the whole method described here can be equally effective if we replace the energy not supplied with a "customers $\times$ duration" based index.

2.2.3.5 Other indices

While the previous indices may suffice for a proper modeling of the load restoration problem, some works may employ diverse objectives to accomplish different goals. I anticipate that they may not be well suited for the load restoration problem, but even so this section presents quick descriptions to help the reader understanding previous some works of the literature.

Power losses

For a given configuration $\mathcal{G}^{CC}(M)$, two important parameters for assuring that the network constraints are satisfied are the voltages in buses $\tilde{V}_b$, $b \in \mathcal{B}$ and the current on the branches $\tilde{I}_\ell$, $\ell \in \mathcal{L}^{CC}(M)$, which are obtained by a power flow algorithm (Appendix C). Notice here that these measures only make sense in the non-contracted network $\mathcal{G}_D^{CC}(M) = (\mathcal{B}, \mathcal{L}^{CC}(M))$, so remember to make the correspondence.

With this information, the power losses index will be defined as

$$P_{loss}(M) = \sum_{\ell \in \mathcal{L}^{CC}(M)} R_\ell |\tilde{I}_\ell|^2 \tag{2.10}$$

wherein R_ℓ is the resistance of line ℓ. Note that this equation is simply the sum of the ohmic losses in each distribution line.[13]

While the amount of losses can be relevant in an economic point of view for a utility, its importance can be significantly outweighed by the urgency of keeping the reliability

[13] It is possible to derive this equation in terms of the bus voltages, but this notation is simpler.

indices on check during an emergency state such as during a load restoration [6], and the previous indices provide a better insight in this regard. While $P_{loss}(\cdot)$ may be a good choice during regular reconfiguration problems [29], for the current topic it is suggested to be either kept as a secondary index – such as in [24] where it is used to help with diversity management of solutions – or disregarded altogether – as done in this work.

Feeder balancing

Some utilities may find useful to operate with feeders maintaining similar amounts of spare capacity instead of some being in the edge of overloading and others with almost no loads. There is not a commonly adopted mathematical formulation for this index, but equations (3.4) and (3.5) in Chapter 3 provide examples of its usage.

While this feeder balancing may be useful to be maintained in normal operation, it may actually lead to an unnecessary increase in maneuvers. Also, notice that we can effectively balance two or more feeders by cutting loads, so optimizing this index may actually produce the opposite of what we are trying to accomplish! For these reasons, we suggest adopting the same approach we recommended for $P_{loss}(\cdot)$, that is, ignoring this index in the context of power restoration and using it in situations where it is actually relevant, e.g., for the reconfiguration of regularly-operating networks.

2.2.4 Problem statement

Given the definitions provided so far, suppose that the quality of a restored network is modeled using m indices, combined in a vector function $\mathbf{f}(\cdot) : \mathfrak{S}_{2\mathcal{E}} \mapsto \mathbb{R}^m$, and let the distribution network be represented by an undirected graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$, together with its complete non-contracted version $\mathcal{G}_D = (\mathcal{B}, \mathcal{L})$. Given these definitions, the restoration problem can be stated as

$$\underset{M \in \mathfrak{S}_{2\mathcal{E}}}{\text{minimize}} \quad \mathbf{f}(M); \tag{2.11}$$

$$\text{subject to} \quad V_{b,min} \leq |\tilde{V}_b| \leq V_{b,max} \qquad \forall b \in \mathcal{B} \tag{2.12}$$

$$J_{b,min} \leq |\tilde{J}_b| \leq J_{b,max} \qquad \forall b \in \mathcal{B} \tag{2.13}$$

$$|\Delta\tilde{V}_\ell| \leq \Delta V_{\ell,max} \qquad \forall \ell \in \mathcal{L}^{CC} \tag{2.14}$$

$$|\tilde{I}_\ell| \leq I_{\ell,max} \qquad \forall \ell \in \mathcal{L}^{CC} \tag{2.15}$$

$$\mathbf{j}(M) = \mathbf{A}^*_\tau(M)\mathbf{i}(M) \tag{2.16}$$

$$\Delta\mathbf{v}(M) = \mathbf{A}^T_\tau(M)\mathbf{v}(M) \tag{2.17}$$

$$\mathbf{j}(M) = \mathbf{Y}_{bus}(M)\mathbf{v}(M) \tag{2.18}$$

$$\mathcal{G}^{CC}(M) = (\mathcal{N}, \mathcal{E}^{CC}(M)) \text{ is radial} \tag{2.19}$$

These parameters and constraints come from power flow considerations, and a complete derivation of these equations can be obtained from Appendix (C). For completeness, here is the meaning of each constraint:

- Equation (2.12) provides limits for the bus voltages $\tilde{V}_b$. Notice the upper bound takes into account possible Distributed Generation increasing too much the voltage;
- Equation (2.13) restricts the injection of currents $\tilde{J}_b$ in buses. In general only source nodes are limited to represent feeder capacity, so if a bus has no such constraint, its bounds can be set to $\pm\infty$;
- Equation (2.14) guarantees the voltage drops $\Delta\tilde{V}_\ell$ are not too large. Normally only the bus voltages are limited (in which case one can set $\Delta V_{\ell,max} \to \infty$), but the formulation is extended for completeness;
- Equation (2.15) confines the branch currents to the line capacity.
- Equations (2.16), (2.17) and (2.18) are the usual Kirchhoff's Laws for a given configuration obtained from a sequence of maneuvers M, which were derived in Appendix C.
- Equation (2.19) assures the final configuration is radial. The mathematical definition of radiality is provided in equation (A.4).

Recall that the reason why this derivation was left to an appendix is to provide a more natural flow of the text. Following this reasoning, if these constraints are encapsulated

into some kind of black-box that returns True of False depending on whether they are satisfied or not, we do not necessarily need to be reminded of them every time. Thus, we can simplify this notation to

$$\begin{aligned} \text{minimize} \quad & \mathbf{f}(M); \\ \text{subject to} \quad & M \in \text{feas}(\mathfrak{S}_{2^\varepsilon}) \subseteq \mathfrak{S}_{2^\varepsilon}; \end{aligned} \tag{2.20}$$

in which, not surprisingly, $\text{feas}(\mathfrak{S}_{2^\varepsilon})$ is the set of feasible sequences of maneuvers, which are the ones satisfying the previous constraints.

With this new notation, the load restoration problem was converted into a regular multi-objective optimization one, and all notions of Pareto-optimality and decision making apply. Appendix D presents an overview of all Multi-objective Optimization concepts required for this work.

2.3 Summary

The distribution system is the portion of the electric supply system that directly provides energy to the customers. They are usually operated radially, such that there is only one path from a source to each load. This presents a number of advantages, but, in the occurrence of faults, all loads that are downstream to the failed sectors become out of service (*oos*). These interruptions in the supply impact negatively the reliability of the system, which are measured by duration (SAIDI) and frequency (SAIFI) indices, and they can implicate in fines to be paid by the electrical company to the customers and government. In order to reduce the contribution of the healthy but out of service loads while a fault is still being recovered, we can use a *load restoration* procedure to recover at least a portion of them.

Load restoration (or "service restoration", a more common terminology in works handling automated systems), is defined as recovering the most *oos* loads in the shortest time possible. To accomplish that, we perform closings and openings of switches, keeping in mind that the reconfigured network should not violate the voltage, current, feeder capacity and radiality constraints. The resulting restoration plan will be composed of a *sequence of maneuvers*, which will be referred to as a "solution" of the load restoration problem in this work. In order to qualify each solution, we need to employ a combination of quality indices (described in section 2.2.3), which generally should be minimized.

With this, the restoration problem can be formulated in general as a multi-objective problem, given in equation (2.11), which can be solved using optimization techniques.

There is already a great deal of works trying to solve the load restoration problem in many different ways. Before presenting the proposed method, it is instructive to take a time analyzing the approach taken in some of these studies. This is the topic of the next chapter.

Chapter 3

Literature Review

The restoration problem is probably as old as the distribution systems themselves. There is a good reason to believe that most load restoration approaches of the past (and possibly a bigger portion of current systems than companies would like to admit) relied on a human expert making use of well-educated trial and error processes. So, whenever there was a fault, after the isolation part of the Fault, Detection, Location, Isolation and Service Restoration (FDLISR) procedure, an engineer would analyze the recoverable and unrecoverable portions of the system and decide for a maneuvering scheme that he/she may find reasonable at the moment.

It does not take too much time to think of some flaws with this approach: a miscalculation may overload a supporting feeder and the maneuvers must be reversed; it is hard to keep up with growing systems; and, of course, experts someday retire, and incorporating their insight into new engineers may be too costly. Fortunately, we now have a multitude of computer-based techniques trying to solve the load restoration problem, and some of them are reviewed in this chapter.

Before beginning, recall that there is no universally agreed definition of what is the "best restoration plan". More specifically:

- Utilities and authors may disagree on the best combination of quality indices to characterize a good restoration plan. This depends on a number of factors, such as how automated the system is, how long does it usually take to locate and repair a fault, how often does outages occur etc.;
- What operations are allowed? For instance, is load shedding permitted only for *oos* nodes, or can it be extended to *ins* but less important loads? Should we close first and open later (or vice-versa) during a load transfer, or is it a forbidden

task? This knowledge is important for developing more efficient methods for each situation in comparison to techniques that try to handle a general case;

- In the same direction as the previous point, other characteristics of the network can lead to a method in favor of another. For example, if the system is relatively small, a method that promises an optimal solution (for a given set of quality indices) may be preferred, while in larger networks a heuristic technique should be a better choice.

Insofar as there is no agreement among the authors (or distribution utilities), it is not possible to make a fair comparison among the studies in order to select the best one, as this would be another (hard) decision problem. However, it is still feasible to analyze the published literature and indicate some strong (like guarantee of optimality, returning a proper sequence of maneuvers etc.) and weak points (like not dealing with voltage and current constraints, leaving some important topics such as load shedding too vague etc.). This chapter reviews a number of relevant studies from the literature. For better presentation, they are grouped depending on the optimization algorithm adopted, classified into mathematical programming, heuristics and metaheuristics.

3.1 Mathematical Programming methods

The mathematical formulation of the load restoration task as presented in section 2.2.4 models the problem as a non-linear mixed combinatorial optimization problem. There are efficient commercial solvers available to handle it, but its $\mathcal{NP}$-hardness character precludes a solution from being returned in feasible time, at least in terms of scalability for larger systems. It is also possible to try a brute force or an enumeration approach that tests all (or most) of the configurations and outputs the best one(s), but these are equally affected by the curse of dimensionality [30]. Thus, to be functional in large systems, they are normally coupled with a mechanism to prune the search space, or the non-linear constraints are relaxed, as will be discussed in this section.

The first work, by Sarma *et al.* [31], is based on trees. Given a faulted network, consider the restoration sub-graph as illustrated at the top of Figure 3.1. The idea is to generate trees starting at nodes that are *ins* endpoints in supporting feeders (referred to as "interesting nodes" in the paper) and try to include as many *oos* nodes as possible without violating the constraints (shown at the bottom of the same figure). Notice that these trees, called "interesting trees", are composed of all spanning forests in this restoration sub-graph, and also of subforests, which may exclude some nodes. Therefore, the computation of all of these possibilities is impractical for big networks.

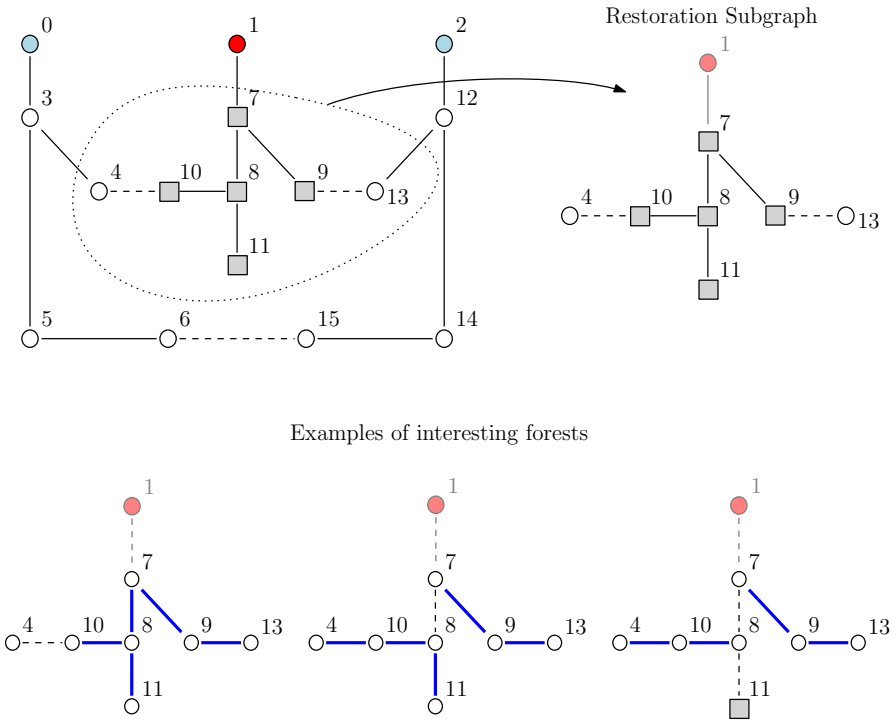


FIGURE 3.1: Method of Sarma *et al.* in [31]. The idea is to generate subtrees in the restoration region (shown on the top). The edges of the trees are shown in solid blue.

The authors propose techniques for computing these trees, along with network reductions to try to make the method more applicable. Since the goal is to cover as much nodes without constraint violations, we conclude that they aim to purely minimize $S_{NR}(\cdot)$, even if not explicitly mentioned, thus neglecting the "as fast as possible" portion of a good restoration plan. Moreover, since the final output is just a new configuration, a feasible sequence of maneuvers may not be always evident. Finally, there is no mention of how long the method takes or how well it can fare in a practical system.

Nagata *et al.* [32] adopt a binary codification for the search space, mapping each switch to 0/1 depending on its open/closed status. The objective function is a weighted aggregation of $S_{NR}(\cdot)$ and $N_m(\cdot)$, which converts the problem into a single-objective version that can be solved with a Branch and Bound method. In order to cope with the huge search space in large systems, the authors propose a set of rules to collapse some nodes with similar voltages, and artificially open or close selected switches. Depending on the degree of simplification, the problem may become smaller and the solution process is accelerated. At the end, a post-processing scheme is proposed to convert the final set into a proper sequence of maneuvers.

The main issue of the method is the weighted aggregation of the quality indices without any mention of function scaling and choice of weights, which are relevant in the decision making step. Also, since the codification encompasses the entire system, we can expect

some load transfer in the final solution in some cases. However, there is no mention on how to handle the practical aspects of such an operation (open first and close later or vice-versa, how to handle possible phase differences among feeders etc.). Finally, notice that due to the simplifications the final solution may lose quality in the original problem, thus making the method resemble more a heuristic than an exact approach.

Instead of reducing the number of alternatives, the work by Hijazi *et al.* [33] attacks the non-linearity of the model. The authors propose convex relaxations of the load flow equations, which can be solved by commercial algorithms to proven optimality much more efficiently than with the original non-linear equations. The resulting adaptation was used to solve problems of power losses, load balancing and load restoration. In the latter case, the objective function was the same weighted sum of $S_{NR}(\cdot)$ and $N_m(\cdot)$ used before, resulting in the same issues mentioned for Nagata *et al.*'s work [32].

According to the authors, the relaxations allowed them to handle networks with up to 880 buses in a very short time. However, apart from the objective function setbacks and the output of only a set of maneuvers, there is no guarantee that solutions that did not violate the relaxed constraints will retain their feasibility in the original model. In such cases, a non-linear integer programming optimizer is employed to try to fix them, but even so there is no guarantee of a successful correction, hence wasting their worth as actual solutions.

The final work reviewed in this class is the one by Romero *et al.* [6], which, following the same relaxation idea of the previous one, converts the load restoration problem into a mixed-integer second-order cone programming one, hence solved by commercial solvers. The difference this time is that the level of relaxation is much lower, thus maintaining feasibility in both models. Nonetheless, this comes at the cost of a higher processing time, which can hinder its applicability in a large system. Then, the method can either be stopped prematurely when the first feasible configuration is found (as suggested by the authors), or be employed as baseline for other approaches.

The objective function is again a weighted aggregation of $S_{NR}(\cdot)$ and $N_m(\cdot)$, but now prioritizing important loads and remote switches. Even so, this weighted sum retains the same problems as the previous works. Moreover, given its high processing time, an *a posteriori* approach experimenting with some combinations of weights is out of consideration. Finally, no proper sequence of maneuvers is returned.

3.1.1 Discussion

Methods employing mathematical programming approaches have as main benefit the optimality guarantee of the final solution. However, they seem to be applicable only in relatively small systems or as comparison baselines for other techniques.

To be employed in large networks, either some kind of truncation of the search space or a relaxation of the non-linear constraints should be adopted. In that case, the algorithms face a trade-off between feasibility of the final solution in the original problem and processing time. Finding the sweet spot is apparently an open problem in the literature.

Apart from this dilemma, the reviewed studies still have some issues when applied to practical scenario. To begin with, the objective function is essentially single-objective, considering only a single quality index, or a weighted combination on indices. In the latter case there is usually no consideration of function scaling or choice of weights, thus complicating the decision-making process. Also, the codification of the search space allows for a direct mapping of a set but not a sequence of maneuvers *during* the optimization process, which precludes the adoption of more realistic indices to measure the restoration time. Even with a post-processing step (as proposed in [32]), the optimization can only be made in terms of $N_m(\cdot)$ and, as showed in [1] and later in this work, minimizing $N_m(\cdot)$ is not as realistic as minimizing $T_m(\cdot)$.

3.2 Heuristics

A heuristic can be understood as a mental shortcut that allows people to solve problems and make judgments quickly and efficiently. In the computer science context [30], a heuristic is a method designed to solve a given problem more quickly, normally trading optimality, completeness, accuracy or precision for speed.[1] In this work, it is convenient to further divide these methods into (i) rule-based approaches, (ii) constructive heuristics, and (iii) local search.

3.2.1 Rule-based methods

Remember the senior engineer who used to (or maybe still does) solve the load restoration problem manually? What he/she is doing is essentially following a set of rules that may come from the utility's guidelines, years of experience or even a user manual. So in his/her mind there could be an algorithm such as "for a fault near this region we used

[1] Sometimes they can actually compute the optimal solution, like the Kruskal method for minimum spanning trees. But other times, like in the load restoration problem, they often cannot.

to close this switch and recover these loads, so we should open these other switches to prevent violations, and if it does not work then try closing this other switch and...".

This first class of methods can be seen as "automated" versions of the expert engineer.[2] In this case, we have an actual computer following a predefined set of "if then/else" instructions that may come from interviews with human experts, from proposed ranking indices etc. These rules are the main factor that distinguishes the works in the literature among themselves.

This quick introduction may have sounded a little too abstract, so let us present some studies. Aoki *et al.* [34] proposed an algorithm with the following structure: (i) try to recover all loads with a single closing; (ii) if there is violation, connect the supporting feeder to a higher order one to relieve its capacity; (iii) if the constraints are still not satisfied, perform *oos* load shedding. Elaborating on these steps:

i If there is more than one switch connecting an adjacent feeder to the *oos* region, rank the switches according to indices of violation, and prefer the maneuver that causes the smallest norm of voltage drop and current rise. If the constraints are satisfied, stop;

ii If all switches cause a violation, the load transfer operations are chosen according to the solution of an integer optimization problem, solved by the effective dual gradient method (see the original paper [34] for details);

iii The *oos* load shedding occurs by cutting loads that provide the "biggest constraint violation" (measured by another index proposed in the paper) or have the smallest priority.

This method returns solutions in short time, and since the operations are determined constructively, it outputs a proper sequence of maneuver. However, being a rule-based heuristic, it does not try to explicitly optimize a quality index, even thought we can argue that it minimizes $S_{NR}(\cdot)$ – so it is effectively single-objective. Besides, it has some practical flaws, such as handling only single faults and not discussing how load transfer should be conducted.

Devi *et al.* [35] start with an exhaustive search trying to recover all *oos* region with only a single closing (and possibly an isolation opening, not mentioned but assumed). If this step fails, then try closing two CO switches and opening a CC one in the path to prevent breaking radiality. The number of possibilities here may grow too fast in large systems, so the authors try to prevent some operations that may seem useless beforehand (such

[2] In fact, sometimes the denomination "expert systems" is used for them.

as closing switches from too loaded supporting feeders). Finally, if this part also fails, then try load transfer, which unfortunately is not well detailed in the paper.

This method has the same advantages of the previous paper, and the same issue with respect to choice of objective function. Also, from the algorithm, we can deduce that it can only handle single outages, and given the second step, partial restoration is not supported (*oos* load shedding is mentioned but never executed). Finally, since the authors employ a DC flux to perform the power flow calculations, there is no guarantee that the final network will be feasible.

The work of Miu *et al.* [36] is similar to the two former steps of the previous one. However, instead of simply trying all options, they assign two indices to CO switches estimating (1) spare capacity of the supporting feeder and (2) voltage drops once the corresponding CO edge is closed. Similarly, CC switches receive another index (3) providing an idea of the amount of current flowing in the lines, and so which edges should be opened for being "too loaded". With this in mind, the process goes as follows:

1. Close a CO switch with biggest spare capacity. If there are no violations, go to the next *oos* region in case of multiple faults;
2. If a voltage constraint only occurs, sort the CO edges by the second index and attempt to close them in order until the violation is gone or all maneuvers are tried;
3. If the second step fails or a current violation happened, start to open CC switches in order with respect to the third index, but leaving the edges that disconnect priority loads to the end. When the constraints are satisfied, go back to the first step.

The same benefits of low processing time and returning a proper sequence of maneuvers are applicable here. However, even thought the authors claim that the method minimizes $S_{NR}(\cdot)$ (divided between priority and common loads) and $N_m(\cdot)$, as argued before, there is no "true multiobjective" optimization as there is no way to control preferences over the objectives.

A follow-up study by the same authors was proposed in [37], which adopts the same idea and structure of the previous one but including load transfers of various levels in step 2. The same up and downsides can be made about this work, with the difference that the higher level restoration may help to restored more loads. However, this comes with the cost that a higher number of maneuvers in possibly far away places may need to be performed. This may greatly increase the time of maneuvers when compared to

a restoration plan involving only the *oos* region that recovers less loads but takes far less time. The proposed algorithm is not able to compute such compromises and only finds the first option. Finally, the practical aspects of load transfer (open or close first, connecting two or more out of phase generators etc.) are not discussed.

Shirmohammadi [38] follows a distinct procedure from the previous works. His method begins with all switches in the restoration region closed. Then run a power flow method for meshed grids, followed by a "secure flow pattern" algorithm. The details of this technique are in the paper, but, in short, it provides an index for each switch in the network such that the one with the smallest value should be opened. If an initially CC switch is chosen and some CO edges were opened before, then try to reclose the one with biggest value of this index. The process goes on until the network becomes radial and respects the constraints or a maximum number of maneuvers was reached. Notice that partial restoration is not obtainable.

As evident from the description, there is no guarantee that a feasible configuration will always be returned, but even in the affirmative case, it does not construct a proper sequence of maneuvers from it. Also, it requires a power flow for meshed networks, which may not be as efficient as the ones prepared for radial configurations. Finally, it seems to be applicable only for single outages.

Singh *et al.* [39] use the same approach of starting with all switches closed and opening one at a time. After a power flow method for meshed grids is executed, the switch with the smallest apparent power should be opened. The process keeps going until the network is radial. If voltage and current constraints are not satisfied, *oos* load shedding should be performed. However, the authors leave this stage for the engineer, which in the end places most of the burden in the human expert. Apart from this issue in concept and the requirement of a generic power flow algorithm, a sequence of maneuvers is not returned, and it is not clear whether multiple outages can be handled.

Borges *et al.* [40] restrict their attention to the restoration sub-graph, and the rules proposed to choose which operations to perform come from a more intricate method. The authors propose an Optimum Power Flow (OPF) formulation, in which the state 0/1 (indicating open/closed) of the switches is modeled by sigmoid functions, thus making the problem continuous and allowing its resolution by an interior points method. In order to truncate back the states to 0/1, a sensitivity index is also proposed which depends on the sigmoid value output by the algorithm, the current flowing and its spare capacity.

With that established, the method goes like:

1. Starting from the post-fault configuration, run the OPF and close the CO switch with biggest sensitivity index;
2. In case of violations, run again the OPF with the new configuration now considering the CC switches to determine which ones should be opened to guarantee feasibility;
3. If two CO switches form a loop, open the CC edge in this path with the smallest current;
4. The process is repeated until there is no CO switch to be closed or all loads are restored.

The authors mention that the algorithm is fast and priorities can be modeled in the OPF formulation. Nevertheless, it is not explicit which quality index the method tries to optimize, but given its structure, it seems to simply try to recover the most *oos* loads possible. Also, despite the arguments for its quick processing time, it is not evident whether that many executions of an OPF can slow down or not the algorithm. Lastly, only the final configuration is evident, and a sequence of maneuvers is not returned.

3.2.1.1 Discussion

Rule-based methods have as main advantage that, as the operations are performed iteratively, a sequence of maneuvers can usually be returned directly. Also, even if this sequence is not optimal with respect to time or energy not supplied, at least the precedence rules become available, which is already an important information for a proper restoration plan. Furthermore, these algorithms are normally very fast, which is a strong point when dealing with an emergency situation such as the load restoration problem.

The rules followed by older works were devised in great part by actual interviews with experts, which confer some characteristics to the resulting algorithms such as a subjective degree of confidence to the utility (assuming it trusted the work of its previous engineer) and the lack of generality since distribution systems around the world tend to differ in many aspects. Maybe this is a reason why more recent methods are preferring rules based on parameters of the network instead, which was seen in [40] and in graph-based methods, presented next.

The possibly most important issue with these heuristics is the lack of a "true multiobjective optimization", that is, it is not possible to prefer one objective over another. This is mostly because the set of rules is already fixed, thus turning these methods essentially

into a single-objective one. Even works aggregating quality indices into a single function still suffer from the absence of proper objective scaling and the subjective shortcomings of a weighted sum. Because of this, I believe these algorithms can be useful either as a starting point for more complex methods, or in extreme situations where a good enough solution is required as fast as possible.

3.2.2 Graph-based constructive heuristics

The methods in this category work in the same fashion as the previous one by starting with an empty solution and constructing a complete one from a set of rules. The main difference is that these rules are guided by graph-based approaches, hence the denomination and the reason why I thought it would be instructive to separate them in another class. Thus, the reader can expect the mention of some popular techniques such as minimum spanning trees and shortest paths.

The first work is by Sudhakar [41], and his heuristic can be briefly structured as:

1. Assign to each switch its corresponding line impedance as weight. Then, create minimum spanning trees starting from the source for each feeder. Any algorithm, like Kruskal, Prim, Dijkstra or Reverse Delete (all described in the paper) can be used to compute this tree;
2. If a feeder is overloaded, then try to transfer some load from a neighboring feeder. If not possible, perform load shedding;
3. Repeat for each feeder until the configuration is feasible.

This rather vague procedure is actually just as vague in the original paper. It does not detail the number of nodes each feeder should encompass, and when to stop step 1; what is the motivation for choosing the line impedance as graph weight; and finally the load transfer/shedding mechanisms are not described in detail, and the author just mention to start pruning the nodes with least priority. Until these steps are more detailed, this method does not seem appropriate for practical use.

Nahman and Strbac [42] propose a less obscure heuristic with the following stages:

1. Choose an *oos* sector $n \in \mathcal{N}_{oos}$ according to some criterion, such as priority;
2. Choose a CO switch $e^{CO} \in \mathcal{E}^{CO}$ in the intersection of the *oos* and *ins* region, that is, one that can recover loads when closed. This edge seems to be chosen at random;

3. Compute the shortest path from n to one of the endpoints of e^{CO}. Recover this node by closing this switch and, to keep radiality, open all CC edges adjacent to this path;

4. If step 3 fails to respect constraints, perform load transfer. This process is somewhat vague in the paper, tough;

5. If step 4 also fails, go back to step 2 and choose a different CO switch;

6. If all CO switches fail, give up this node and try a different one.

The algorithm stops when all nodes were considered. At the end, a post-processing phase is performed to output a proper sequence from the final configuration. This paper is actually one of the few which mention the importance of rules of precedences among maneuvers to create a feasible sequence. Moreover, being a heuristic, it returns solutions in a quick time.

Some relevant topics are still lacking, tough. For starters, some steps are not very well detailed, such as the load transfer. Similarly, the choice of CO switch in step 2 could be further researched as a proper choice could improve the final solution. Finally, the authors use a simplified power flow algorithm which discards the phase differences of the bus voltages, which can be inadequate in some cases (such as unbalanced networks) and output unfeasible restoration plans.

Dimitrijevic and Rajakovic [43] extend Prim's algorithm to compute Minimum Spanning Forests (MSF) instead of Trees. To understand it, assume a graph in which we know beforehand which nodes will be the roots in each tree. Also, there is a list containing the order in which we include an edge in each tree. The top of Figure 3.2 shows an example of creating a MSF with three trees. If the order {III, I, II} is assumed and the initial roots of each tree are as indicated, the first iteration adds to III the adjacent edge with the smallest weight as long as it does not create a path to any other tree. The second iteration augments an edge to tree I, then II, and the process is repeated until there is no edge available to be included without connecting two trees.

In the load restoration context, start with all switches in the restoration region opened, and assign each *ins* node connecting a *oos* one to the root of a tree, as shown at the bottom of Figure 3.2. The weights of the switches are chosen as the length of its corresponding line, plus a factor whose value depends on how costly it would be to operate this edge (higher for manual than for remote operations, higher for changes in edge status than for no changes). With this, the order in which each tree will be constructed depends on the available capacity of each feeder. For instance, if the total current supplied by the feeder of node 13 is farther from its limit, an edge is added

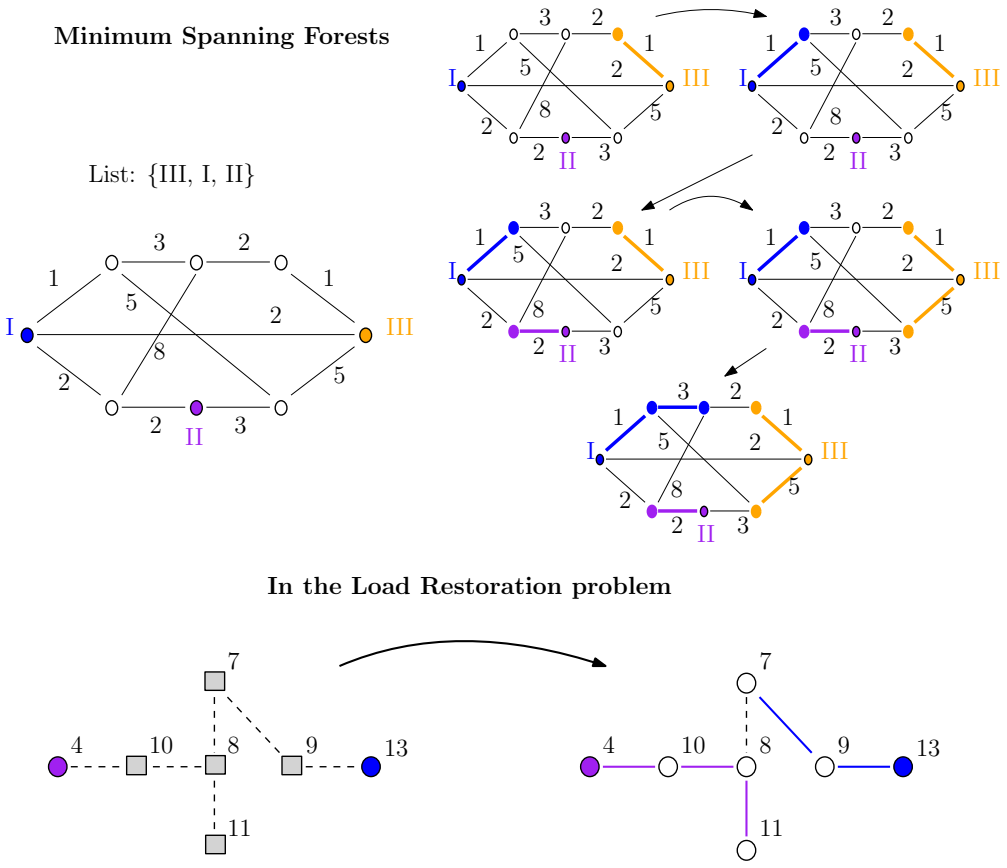


FIGURE 3.2: Extension of Prim's algorithm to generate Minimum Spanning Forests (on the top). In each turn, an edge is added to a tree according to a pre-specified list. In the restoration problem (bottom), the immediate node in the supporting feeder connecting to the *oos* region becomes the root of a tree, and the process goes like before.

to this tree. By following this procedure, we can end up, for instance, with the final configuration of Figure 3.2. It is not clear in the paper what to do if two roots belong to the same feeder so, apparently, any one of them can be chosen.

The method seems very simple and returns solutions quickly. Yet, since it is a constructive heuristic, there is no guarantee that the final solution will be good enough. Therefore, it should be a good option for simple faults or as a starting point for more complex algorithms. Also, it is not clear if it can handle more than one fault. Finally, the actual sequence of maneuvers is not directly obtainable from the final configuration.

The final work reviewed in this category is the so called "graph theoretic-based" method by Ibrahim *et al.* [44]. This technique starts by creating a new complete graph with the nodes of the restoration region. Each edge of this graph $e = (n_1, n_2)$ will receive a weight w_e with a value depending on a number of factors such as (i) the existence of a CO switch between n_1 and n_2 in the original network, (ii) whether one of the nodes is *oos*; (iii) and the physical characteristics of the distribution lines between these nodes,

such as length, material etc. The idea is to run a shortest path algorithm (Dijkstra in the paper) between *ins* and *oos* nodes to determine which switches will have to be open or closed in the original network. They argue that this choice of weights will return a configuration with the smallest number of maneuvers, most recovered loads and smaller losses. In the end, their method is compared to a Genetic Algorithm (GA) approach with a binary codification minimizing a weighted sum of $S_{NR}(\cdot)$, $N_m(\cdot)$ and $P_{loss}(\cdot)$.

As with most heuristics, the method is fast and in this case it can handle multiple outages. However, it is not evident whether constraints are satisfied, since there is no mention of power flow algorithms, and it is also not clear if load shedding is performed, which is required for partial restoration. Furthermore, the biggest problem in my opinion is the (possibly unfair) comparison between a heuristic with fixed rules with a metaheuristic such as the GA with no consideration of its control parameters (variation mechanisms in this codification) and minimizing a random aggregation of objectives.

3.2.2.1 Discussion

Since the works in this category are very similar in essence to rule-based methods, the same benefits and drawbacks also apply. A pertinent observation is that, possibly due to the specific character of rule-based approaches, the scientific community seems to be moving to graph-based techniques in order to make its works more applicable to general cases.

3.2.3 Local Search

Local search algorithms [45], differently from the previous heuristics which create a full solution from scratch, start from an already available solution[3] $\mathbf{x}$ and search its surroundings for a possibly better $\mathbf{x}'$. We accept this improvement $\mathbf{x} \leftarrow \mathbf{x}'$ and repeat the process until there is no better solution nearby, when we have arrived at a *local optimum*. These surroundings are represented by a set $\mathcal{V}(\mathbf{x})$ called *neighborhood*, which can be, for instance, a closed ball of size δ in continuous spaces, the k-opt in case of the Traveling Salesperson problem [45], the set of all binary vectors obtained by flipping one bit of $\mathbf{x}$ etc. Figure 3.3 shows an example of the local search process using a closed ball of size δ as neighborhood.

On the bottom of the same figure, we show that, for multi-modal functions, the chances of arriving at the global optimum from local search movements are reduced, and there

[3] I am using a vector $\mathbf{x}$ here to indicate any kind of solution for simplicity. In reality, the alternatives in each algorithm can be very generic, representing not only vectors, but also graphs, sets of operations of openings and closings etc.

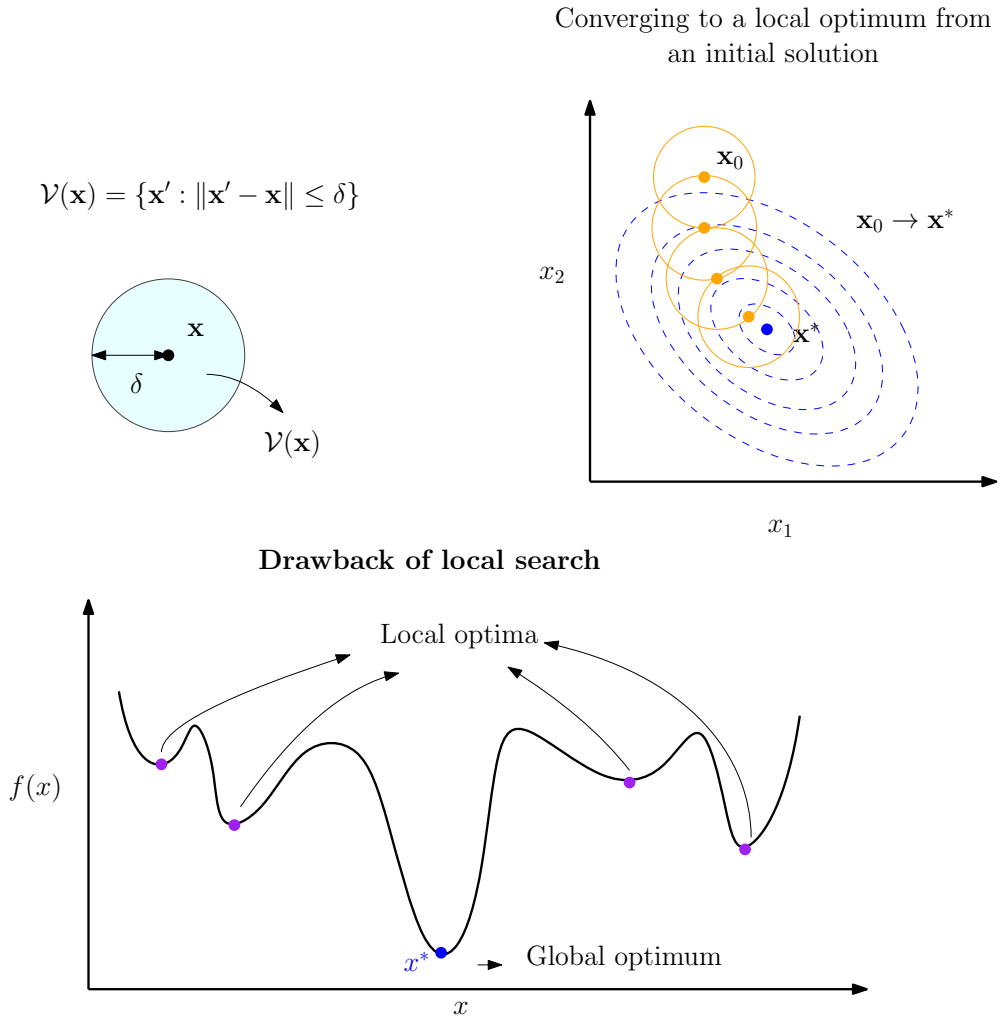


FIGURE 3.3: On the top, an example of obtaining a local optimum (here, also global) from a given point $\mathbf{x}_0$ using a closed ball of size δ as neighborhood. On the bottom, how the local search can have problems if the objective function is highly multi-modal. For many starting points, the local optima obtained may be way worse than the global optimum.

is no guarantee that this local optimum will be significantly better than the starting point. One way of alleviating this is by using larger neighborhoods, at the price of increased computational costs. Another way is to restart the search with different initial solutions $\mathbf{x}_0$, increasing the chance that one of them will be initialized in the attraction basin of the global minimum. However, this does not seem very useful in practice [46]. Therefore (and as a spoiler to the next section), the most recent works prefer to adopt metaheuristics, and keep the local search for possible improvements in the solution returned by those methods.

The work of Kalinowski *et al.* [47] can be somewhat viewed as a local search with an increasing neighborhood. Once a fault was properly isolated by opening all switches connected to it, start the process by closing a single CO switch connected to the *oos*

region. In case of violations, "increase" the neighborhood by closing two CO edges and opening a CC one in the path to keep radiality. If complete restoration was still not obtainable without violations, go to an even "larger" neighborhood by closing three CO and opening two CC switches, and so on. The details of how to determine the appropriate CC edges are given in the paper, but it is not evident how or when to stop if complete restoration is not possible.

The solutions in each neighborhood are compared using an aggregation

$$f_{Kalinowski}(\mathbf{x}) = w_1 S_{NR}(\mathbf{x}) + w_2 N_m^{weighted}(\mathbf{x}) \tag{3.1}$$

wherein w_1 and w_2 are weighting factors, and $N_m^{weighted}(\cdot)$ is the number of maneuvers with each switch weighted by a priority. This seems somewhat unusual as priorities are normally assigned to loads, not to switches.

There are some practical issues with this work. To begin with, some important information is not clear in the paper. For instance, it is not evident how or when to stop the algorithm if complete restoration is not achievable. Also, from the description is seems that partial restoration is not possible, but load shedding is actually performed during the tests, even if the process is not detailed.

Next, two different quality indices are aggregated with no discussion about objective scaling neither choice of weights. Also, once they are fixed, instead of returning only the best solution, the authors keep some worse alternatives in case the decision maker should have different preferences. Since the algorithm is fast, it would be interesting to re-execute it with distinct combinations of weights in order to present a better representation of the efficient front and facilitate the decision making process. Finally, it is only tested for single faults, and it does not return a proper sequence of maneuvers.

The work by Garcia and França [48] proposes to encode the solutions into a Forest graph representation [49]. There is a constructive phase, in which they use a random Prim algorithm to generate a set of initial different solutions, which are improved next by a multi-objective local search heuristic. The quality of the networks is modeled by $S_{NR}(\cdot)$ and $N_m(\cdot)$, and the algorithm aims to approximate the Pareto-optimal front, following an *a posteriori* approach.

The generation of a neighbor in the local search is performed by getting an initial *oos* node and connecting it to a different *ins* node. For example, at the bottom of Figure 3.2, node 9 is energized by 13. A perturbation would consist of changing the status of the switches so that this node is now fed by 4, which is accomplished by opening (9,13) and closing (7,8). In each iteration, many of these neighbors are created, and they are

stored in an archive. Since the archive size is limited, only non-dominated solutions are included. If the number of candidates is larger than the size of the archive, the method uses an Euclidean metric to keep only the most diverse ones (it is not clear however how this process works).

The method is relatively straightforward when compared to most previous heuristics, and it is the only one in this category to actually perform a true multiobjective optimization with possibility of trade-off analysis and inclusion of preferences. Regarding its issues: first, it only considers the number of maneuvers, and there is no distinction between the time for operation of remote or manual switches. Second, there are no guidelines on how to return a proper sequence of maneuvers. Lastly, because of the setback mentioned in Figure 3.3 and the lack of a diversification mechanism, the algorithm gets easily trapped in local minima.

3.2.3.1 Discussion

Local search methods may be viewed as the first attempt in improving on the multi-objective issue of heuristics discussed in section 3.2.1.1, since they are able to actually return a set of solutions approximating the Pareto-optimal front. Unfortunately, they are prone to getting stuck in local optima and thus not enhancing significantly the quality of the initial solution, and simply increasing the neighborhood may slow down the process. According to the tendency of the literature, local search heuristics are now either deprecated by metaheuristics or employed as post-processing phase in more complex methods.

3.3 Metaheuristics

Metaheuristics are a class of algorithms which try to combine different types of heuristics in order to explore more effectively the search space [30]. Although there is no universally accepted definition for metaheuristics, they usually satisfy the following principles:

- They are normally not problem dependent, that is, they do not require any problem-specific information, being able to be applied in black-box optimization;
- Their goal is to explore efficiently the search space in order to compute solutions as close as possible from the optimal;
- The methods are approximate and normally non-deterministic;

- They incorporate mechanisms to prevent getting trapped in local optima. For that, some bad solutions can be momentarily accepted;
- Some techniques use a history of the previous movements in order to guide the search to (hopefully) promising regions.

The first property is the one frequently used to differ metaheuristics from basic heuristics, since the latter are usually created with a specific problem in mind. Because of that, heuristics are readily available to solve an instance of the problem they were designed for, while we would need to first model the candidate solutions, a neighborhood structure etc. before a metaheuristic could be employed. Now, if we think hard enough, after we specialized this metaheuristic for this given problem, it has now become a heuristic! Therefore, the definition of metaheuristic as a "heuristic of heuristics", once rejected by [50], can make some sense when trying to define this class of algorithms. For the purposes of this work, we can consider a metaheuristic anything that has the properties already presented, and so I will go no further with this reverie.

Before applying any metaheuristic to a restoration problem, we need to at least provide the representation of a candidate solution $\mathbf{x}$ (that is, an encoding of the search space $\mathbb{X}$), and a neighborhood structure $\mathcal{V}(\mathbf{x})$ of each solution. Because of that, I decided to group the studies according to the representation used.

3.3.1 Binary-based codification

The binary representation is possibly the easiest and most straightforward to grasp: associate to each switch $e \in \mathcal{E}$ a number 0/1 indicating its status as open/closed in the reconfigured network (or absence/presence in the new graph, depending on the level of abstraction employed), respectively. A given solution $\mathbf{x}$ can then be modified by common operators: swap a bit from 0 to 1 or vice-versa, shift a piece of the vector to another position etc. The arguably biggest disadvantage of this representation is the possibility of wasting time with many non-radial solutions along with other problems, as exemplified in Figure 3.4. This may unnecessarily hinder the search and increase processing time.

The work of Augugliaro *et al.* [51] proposes to model only the CO switches as binary numbers and keep the CC edges unchanged, which immediately leads to the conclusion that there is no load shedding mechanism. The quality of each new configuration is measured by the global power margin (total spare capacity of each feeder) $P_{margin}(\cdot)$ and the power losses $P_{loss}(\cdot)$, a choice that is more common in reconfiguration problems than in the load restoration. These objectives are combined into a single objective by

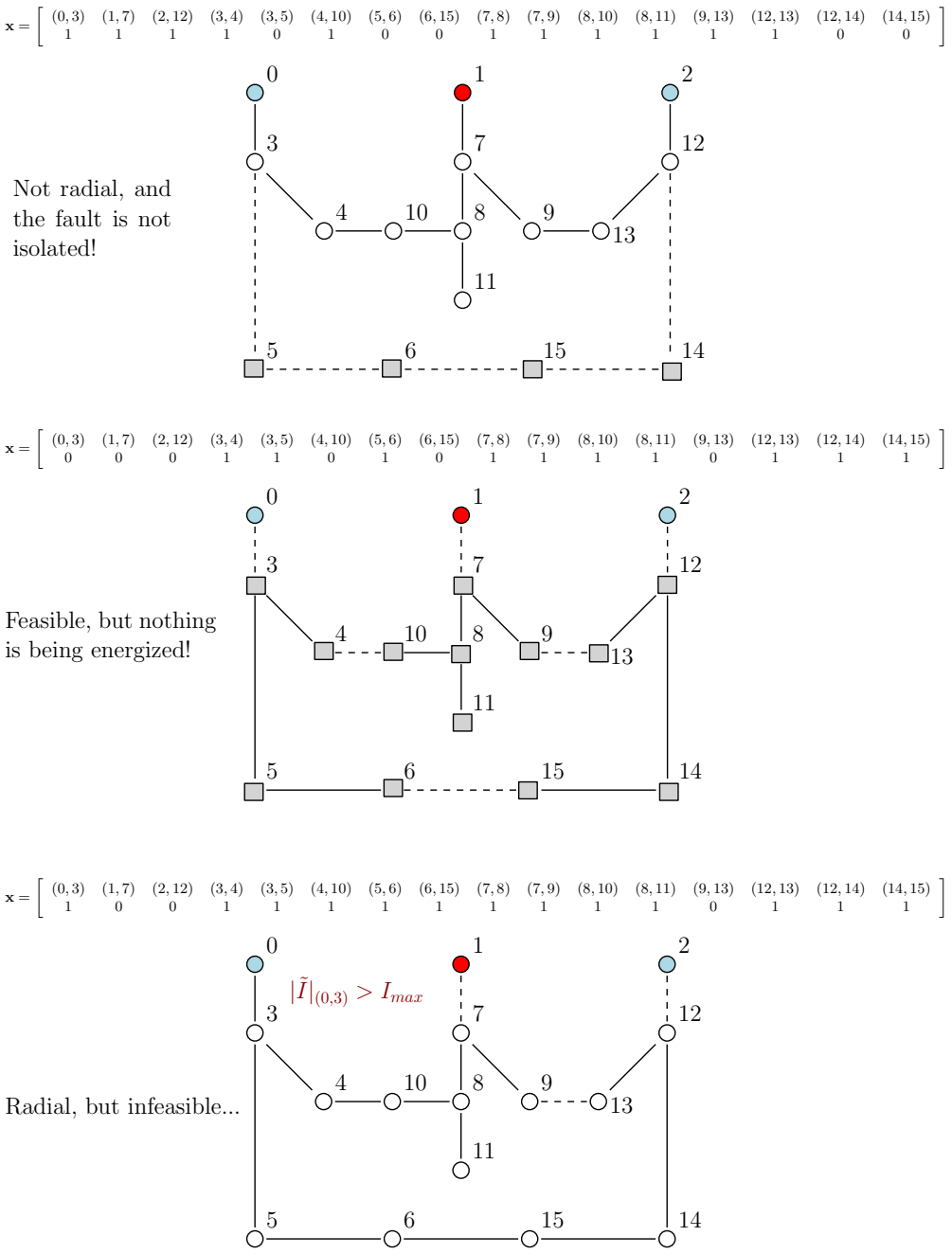


FIGURE 3.4: The binary representation wastes some time with configurations that are either unfeasible or do not make sense.

means of fuzzy sets, and the resulting function is minimized with an Evolution Strategies algorithm. The variation operators used to produce new solutions are not detailed.

This first study has probably more concerns than good points: lack of load shedding, an odd choice of quality indices, an algorithm with obscure structure, and also the unanswered question of how to handle the breaking of radiality if two CO switches create a loop, since these are the only edges operated.

Kumar *et al.* [52] model the entire network (including the CC switches) with a binary structure. In order to prevent radiality violations, they adopt before each evaluation a breadth first search starting from each source node, which artificially opens all edges that are visited twice (Figure 3.5). Then, a regular single-objective Genetic Algorithm is employed to minimize the following combination of known quality indices:

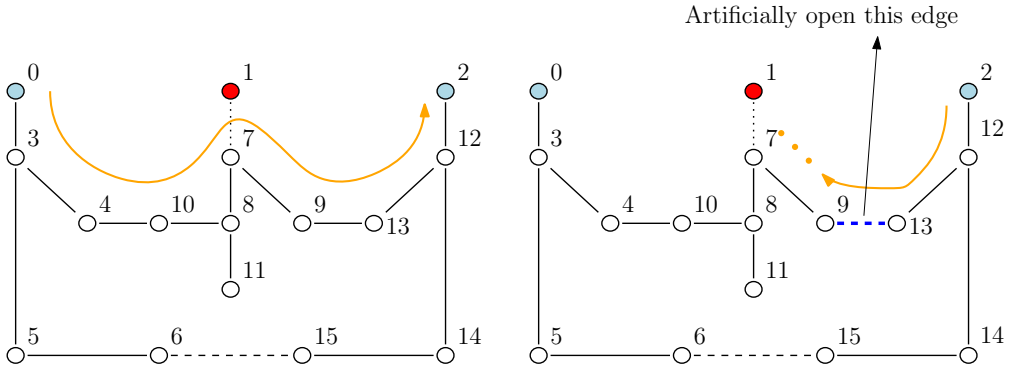


FIGURE 3.5: Mechanism to ensure radiality in the works [52] and [22]. After the first breadth first search at root 0, the second one starting in root 2 would see the edge (9,13) counted twice, so it would be artificially opened.

$$f_{Kumar}(\mathbf{x}) = w_1 S_{NR}(\mathbf{x}) + w_2 N_{m,man}(\mathbf{x}) + w_3 N_{m,rem}(\mathbf{x}) + w_4 P_{loss}(\mathbf{x}) + w_5 C_V + w_6 C_I + w_7 C_P \quad (3.2)$$

with w_i, $i = 1, \ldots, 7$ weighting factors, $N_{m,man}(\cdot)$ and $N_{m,rem}(\cdot)$ the number of manual and remote maneuvers, respectively, and C_V, C_I and C_P voltage, current and priority customer constraints violations. Their exact range or values are not informed in the paper.

The main issue with this technique is the lack of consideration for weighting factors, specially with respect to objective scaling. Additionally, including the constraints into the objective function without a proper treatment of penalty factors is a good recipe for an unfeasible final solution. Finally, the binary codification only returns a final configuration, and there is no post-processing scheme for outputing a proper sequence of maneuvers.

A follow-up study in [22] adopts the same codification and radiality treatment, and also minimizes the same quality indices but without splitting $N_m(\cdot)$ into manual and remote portions and without including the constraints into the objective functions. These indices are minimized separately by a Non-dominated Sorting Genetic Algorithm II (NSGA-II), which returns an approximation to the Pareto front. Despite mentioning the voltage and

current limits during the power flow computation, it is not clear how to handle them in the algorithm.

Apart from the lack of information on how to handle constraints, there is the odd selection of the final solution. Suppose $\mathcal{P}$ is the final population approximating the efficient front. The authors propose a lexicographic approach to choose the final alternative, following the order $S_{NR}(\cdot)$, $N_m(\cdot)$ and then $P_{losses}(\cdot)$. The problem with this is that $\mathcal{P}$ should have only non-dominated points (otherwise, there is a great chance the algorithm did not finish properly), so the point with smallest value of $S_{NR}(\cdot)$ is unique, and the other objectives are never considered. If that is the case, why not just minimize this index? This procedure seems like a lot of wasted computational resource. Also, the order of the sequences is never returned in any of these works.

The same authors propose yet another algorithm with similar structure and radiality handling in [53]. The quality indices are the same, but $N_m(\cdot)$ is again split into its remote and manual portions. The apparent enhancements over the last work are the management of constraints, in which feasible configurations are always better than unfeasible ones regardless of their quality indices, and the inclusion of another constraint requiring all priority loads to be recovered. The optimization algorithm is once more the NSGA-II, and the final solution follows the same peculiar lexicographic selection.

Apart from the same issues of the previous work, the treatment of priorities does not seem very effective because, since all unfeasible solutions are equally bad, how to proceed in a case when not all priority loads can be restored?[4] Also, partial restoration is not possible as the authors assume all feeders always have enough capacity to recover all nodes. Lastly, the algorithm was tested only in small networks.

The last work in this category is the one by Mohammadi and Afrakhteh [27], which has an interesting proposal: make use of distributed generation (DG) in intentional islandings to recover more loads if the supporting feeders are not enough. In this case, both switches and circuit breakers in front of DG units are modeled in a binary codification. Then, a Genetic Algorithm is employed to minimize the aggregated sum:

$$f_{Mohammadi}(\mathbf{x}) = w_1 E_{NS}(\mathbf{x}) + w_2 S_{NR}(\mathbf{x}) + w_3 P_{loss}(\mathbf{x}) \tag{3.3}$$

Configurations that violate constraints – voltage, current, radiality and DG units connected to other feeders – cannot be evaluated in terms of energy not supplied, so they receive a fixed value of 10^9.

[4] For the record, they were always recovered in the test cases, in which case we may also inquire about their validity in real world scenarios.

This is possibly the first work to mention the importance of restoration time [built-in in $E_{NS}(\cdot)$], even by using a not so realistic approach by setting a fixed value to each switch. Also, a proper sequence of maneuvers is constructed for each configuration in order to evaluate the solutions. However, it has its shortcomings. First, the Genetic Algorithm operators are not well detailed. Second, assigning a fixed big value to unfeasible configurations creates flat portions in the fitness landscape, which may hinder the algorithm's performance as a great number of solutions will be evaluated as being equally bad. Third, we have the usual problems of aggregating objectives with different scalings with no consideration on the choice of weights for the final decision making. Finally, employing Distributed Generation in island operation may be a risky move because of the practical concerns enumerated in section 2.1.3.2, and the authors do not explain how to handle them. These drawbacks may prevent the use of this algorithm in real world cases.

3.3.2 Forest-based codification

Since a radial distribution system can be represented as a forest with trees rooted at the sources (except for the portions that are out of service), another way of representing the solutions is by using a forest-based structure. The search space $\mathbb{X}$ in this case is such that every solution $\mathbf{x} \in \mathbb{X}$ is a forest, and all solutions in its neighborhood $\mathcal{V}(\mathbf{x})$ are also forests. With a small care of preventing two or more sources to be connected, perturbation mechanisms always generate radial systems. This is an improvement over the binary structure, which wastes a lot of effort with non-radial candidates. However, as will be shown, many works that follow such representation make the overly optimistic assumption that the whole out of service region is always restored, which is frequently not possible regardless of the number of maneuvers.

The first work is by Toune *et al.* [23], which uses a representation that I decided to call *upstream load*, very similar to a reverse star representation [49]. Given the restoration region, each *oos* node is described by its first upstream node, as exemplified in Figure 3.6. With this representation, the neighbors of a solution can be created, for instance, by changing the upstream load of a node. A method for creating initial solutions is also proposed.

With this encoding, the authors compare the performance of four distinct metaheuristics: genetic algorithm, simulated annealing, tabu search and reactive tabu search (see the implementations in the paper). Apparently, since the representation already assumes that all loads are restored, the authors opted for counterbalancing the spare capacity of each supporting feeder, so that they would "lend the same amount of power", and

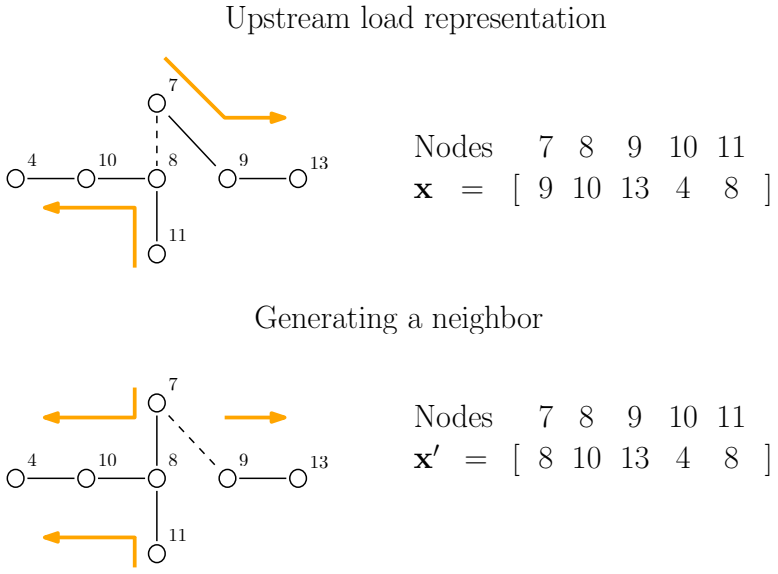


FIGURE 3.6: Upstream load representation used in [23] for a fault in node 1 (top). A solution vector $\mathbf{x}$ is composed of the node that is upstream a given *oos* load. A neighbor (bottom) is generated by changing the upstream load of node 7 from 9 to 8.

to maximize the minimum voltage of the network. These objectives are aggregated in a linear weighted sum like

$$f_{Toune}(\mathbf{x}) = w_1 \sum_{i=1}^{n_f} [SP_i(\mathbf{x}) - \overline{SP}]^2 + w_2 \frac{1}{\min_{n \in \mathcal{N}} |\tilde{V}_n|} \tag{3.4}$$

wherein SP_i is the spare power of the i-th supporting feeder (which seems to be given, since the authors do not mention how to compute or estimate it) and $\overline{SP}$ is the average of all n_f feeders. The first term tries to balance the amount of provided power by minimizing a variance measure of the spare capacity of the feeders, and the second tries to minimize the voltage constraint. According to the results, the reactive tabu search is the one which returns the best configurations.

While this representation is nice for always returning radial networks, even with the minimization of the second objective we cannot guarantee a feasible configuration because of the assumption that all loads are being restored. Also, in a somewhat "improvised" situation that is the load restoration, it may be more interesting to optimize the time taken to get the new configuration instead of assuring equality of "energy borrowed" from each feeder. In fact, this may require even *more* maneuvers, because the restoration from a single feeder would be a bad solution according to the objective (3.4). Moreover, the criteria in this equation are summed without turning them into dimensionless quantities, and the algorithm never returns a proper sequence of maneuvers.

Delbem *et al.* [54] propose a similar representation to the upstream load. The authors use a graph chain representation, in which the whole system (without the faulted nodes) is represented by *chains*. Each chain can be computed by starting at a leaf node, and writing down the upstream nodes until we arrive at a source (Figure 3.7). This representation makes the perturbations computationally more efficient, and they are performed with two steps: (i) open a CC switch and turn all loads downstream to it out of service; and (ii) if possible, close a CO edge that restores this load.

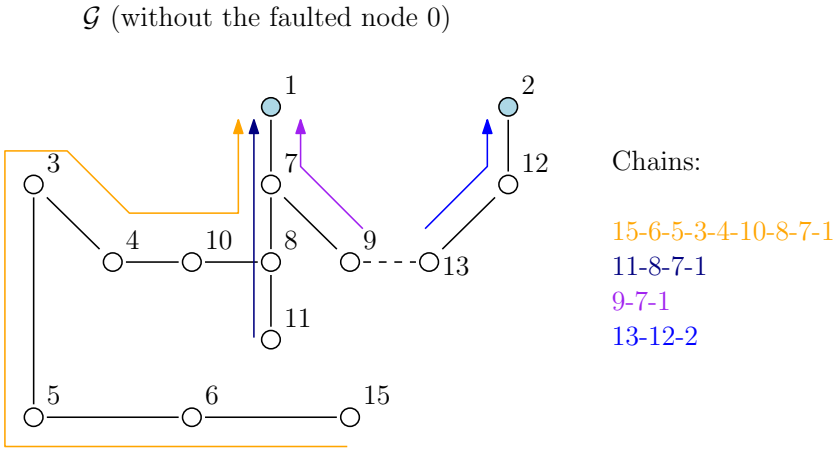


FIGURE 3.7: Graph chain representation of [54]: starting from each leaf node, go upstream until a source is reached. This will be a chain.

The objective-function is very similar to the one of the previous work, but, this time, they minimize also the number of maneuvers using the following function:

$$f_{Delbem}(\mathbf{x}) = \sigma(\mathbf{x}) + \beta \frac{N_m(\mathbf{x})}{N_{m,max}} \tag{3.5}$$

in which $\sigma(\cdot)$ is the standard deviation of the total power furnished by each feeder (its expression is not explicitly stated in the paper), $N_{m,max}$ is a desired limit of the number of maneuvers, and β is a user-defined parameter, making the equation very similar to a weighted sum.

The authors adopt a not very conventional genetic algorithm to minimize equation (3.5), in which the population size is variable and there is no recombination, only the perturbation described before. The criticisms directed at the previous work are also applicable here: the balancing of feeders seems to unnecessarily increase the number of maneuvers; the objective-function is an aggregation of two quantities with different scales [even if $N_m(\cdot)$ is dimensionless with the division by $N_{m,max}$, the standard deviation is not]; and the method does not return a proper sequence of maneuvers.

Carvalho *et al.* [55] use the same upstream load representation as shown in Figure 3.6, with the same perturbation method of changing the parent of each node, and include

this encoding in a genetic algorithm. The objective function is a little complicated for a short description, but, since they also assume that all loads are always recovered, it can be understood as an ϵ-constraint which minimizes the time of maneuvers for different conditions of overload.

This work is also one of the few that, after computing a final solution, executes a post-processing phase to generate a proper sequence of maneuvers; in this case, a dynamic programming approach. However, even though they recognize that it is not possible to restore the whole *oos* region every time, they prefer to provide the decision maker with a set of solutions with different numbers of maneuvers and their respective overloads. Since this Pareto-front approximation is performed point by point, the method can take too long, so it may not practical.

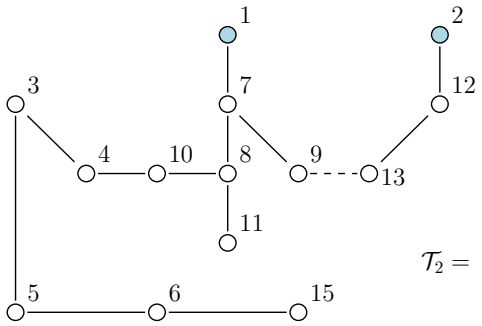
3.3.3 Node depth encoding-based codification

The node-depth encoding (NDE) was proposed in [56] and is illustrated in Figure 3.8. The NDE is actually another forest-based codification, but since it seems to be more popular than the previous representations, I decided to dedicate a separate section for it.

Each solution in this representation is a forest, and it has a separate description for each tree. This description shows the nodes starting at the root and traversed in a breadth-first search manner, together with their depths, which is simply their distances to the root. Together with this representation, two perturbation mechanisms, Preserve-Ancestor Operator (PAO) and Change Ancestor Operator (CAO), which can be seen in the original paper [56], were developed, and they can be performed very efficiently, requiring only $\mathcal{O}(\sqrt{|\mathcal{N}|})$ for the construction of a new forest. Also, the popular Forward/Backward sweep power flow algorithm can be executed very quickly because parent-child relations in the graph are already available in the codification. With this encoding and perturbation mechanisms, any metaheuristic can be employed to solve the problem.

There is a huge array of works using this codification, and they employ a similar optimization algorithm and objective functions. Thus, I will review only two I find representative of this class.

The first one discussed is by Sanches *et al.* [57]. The authors model the load restoration problem as the minimization of the number of maneuvers, power losses and the overload constraints:



Node-depth encoding

$$\mathcal{T}_1 = \begin{bmatrix} \text{depth} \\ \text{node} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 2 \\ 2 & 12 & 13 \end{bmatrix}$$

$$\mathcal{T}_2 = \begin{bmatrix} \text{depth} \\ \text{node} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 2 & 2 & 3 & 3 & 4 & 5 & 6 & 7 & 8 \\ 1 & 7 & 9 & 8 & 11 & 10 & 4 & 3 & 5 & 6 & 15 \end{bmatrix}$$

FIGURE 3.8: Node-depth encoding example. Each tree is represented by its nodes in a breadth-first search manner, together with their depths (distance from the root).

$$\mathbf{f}_{Sanches}(\mathbf{x}) = \left[\; N_m(\mathbf{x}) \quad P_{loss}(\mathbf{x}) \quad \max_{e \in \mathcal{E}^{CC}(\mathbf{x})} \left| \frac{\tilde{I}_e}{I_{e,max}} \right| \quad \max_{n \in \mathcal{N}_s} \left| \frac{\tilde{J}_n}{J_{n,max}} \right| \quad \max_{n \in \mathcal{N}} \frac{|\tilde{V}_s - \tilde{V}_n|}{\delta_V} \; \right] \tag{3.6}$$

in which $\mathbf{x}$ indicates the NDE representation of a reconfigured network, and the objectives are (i) number of maneuvers; (ii) power losses; (iii) maximum excess current in the edges; (iv) maximum excess injected current $\tilde{J}_n$ from each source, with $\mathcal{N}_s \subset \mathcal{N}$ the set containing them; and (v) maximum percentual voltage difference in each node with regards to its source (usually $\tilde{V}_s = 1\angle 0°$ pu), and $\delta_V = 0.10$ the maximum acceptable voltage drop. Just like the previous ones, this formulation assumes that the power not supplied is always zero, as the NDE does not create trees without the roots of the original network.

The authors then optimize (3.6) using a multi-objective evolutionary algorithm, such as NSGA-II and MEA**N** (which stands for "Multi-Objective Evolutionary Algorithm based on subpopulation tables and NDE"). In each study, a new modification is proposed for the algorithm to handle the high number of objectives. The version of the reviewed paper is the MEA**N** combined with a Differential Evolution, but later works use a MEA**N**-MH, where the 'M' stands for "Multiple criteria tables", and they included some kind of decomposition of the objectives using subpopulation tables; and the 'H' goes for "alarming Heuristic", focusing on the perturbation of solutions such that the restoration region is more emphasized.

Just like the previous works, the biggest flaw here is to assume that we can always restore the whole *oos* region. As their results show, in order to comply with the constraints, sometimes we need up to 19 maneuvers, which can be too high in practical situations. In other cases, even infinite maneuvers would not feasibly recover everything. Also, they suggest that we could obtain less maneuvers by preferring solutions that minimize the other objectives, which can be hard to check, since the Pareto-front can become too big

for the DM in this many-objectives problem; or by performing load shedding, although there is never a mention on how to exactly execute it.

The latest work in this category (at least at the time this thesis was written) is by Marques [24], which improves on the previous studies in the following subjects:

- An exhaustive local search is employed at the beginning which tries to restore the entirety of the *oos* nodes with a single closing and an isolation opening. The version proposed here is slightly modified to handle priorities;
- Partial restoration is now available by adding a new dummy source to the network to represent an "*oos* tree";
- A new operator called *Load Reconnector Operator* (LRO) is added to the PAO and CAO ones that prefers the energizing of *oos* and priority nodes;
- The codification of solutions include a description of a possible sequence of maneuvers, and so the more realistic index $E_{NS}(\cdot)$ can be used to evaluate the restoration plans;
- A final step is proposed to return only three out of the possibly huge number of non-dominated alternatives of the final archive (after all, this is an *a posteriori* approach with a great number of objectives), and thus helping reducing the burden on the decision maker;
- This is possibly the first work in this category to take special care in the load transfer operation, and the $E_{NS}(\cdot)$ formulation is prepared to handle both the open/close or close/open strategies.

As can be seen, a number of practical issues of previous works were addressed in this new version. Yet, some of these enhancements seem to come at a cost. For instance, the proposed load shedding for now seems to be tested only for single faults. Also, using the $E_{NS}(\cdot)$ required the knowledge of a time to repair the fault, which in real systems may vary a lot, maybe from two to eight hours, for example. Thus, the expected quality of the final restoration plan may greatly differ if this estimate is not properly adjusted.

Moreover, there are some topics that may be further improved for better adoption in practical scenarios. First, the time computation assumes a fixed value to operate each switch, thus ignoring the current position of a dispatch team and the availability of more than one crew. Second, the optimization algorithm seems to grow in complexity at each new study; as comparison, while the work of Sanches *et al.* [57] had about six tables, this one has twenty six, each one storing the best solution in a given index or a

combination of indices. It may be a better idea to employ an algorithm that is simpler but better adapted to many-objectives, such as IBEA, or even use frameworks such as the inclusion of preferences [58]. Still on the topic of preference incorporation, adopting such approaches (like PBEA and PAR [59]) to limit the final archive may be a suggestion for improving the somewhat simplistic "reduction of the final archive", whose enhancement was mentioned as a future work by the author. Finally, the sequence of maneuvers considered is only meant to be feasible (which is already a significant improvement over many works in the literature), so another suggestion for the future is to choose the order of operations to actually try to minimize a quality index [such as $T_m(\cdot)$ or $E_{NS}(\cdot)$], just like proposed in this work.

3.3.4 Ant Colony Optimization-based codification

I do not plan on providing a review on Ant Colony Optimization (ACO) methods in this text, so the curious reader can find adequate references in [60]. Anyway, here is a quick summary of such algorithms: similarly to constructive heuristics, they create a solution from scratch, but specific parts of this solution can be constructed differently each time depending on a parameter called *pheromone*, which is updated based on the quality of the created solution and other properties that can vary from different versions of the method. Even for readers with less familiarity, I hope that this synopsis alone will make the works of this class easier to understand.

Watanabe [26] was possibly the first author to employ the energy not supplied as a quality index. His objective function is the recoverable portion of $E_{NS}(\cdot)$ plus a penalty term if some *oos* loads were not re-energized and if the power not restored increases during some operations. His ACO algorithm constructs solutions in the following way: first close a CO switch and evaluate it. Then create new solutions by closing other CO edge and opening a CC one. The switches to be operated are biased depending on the pheromone parameter. In order to compute $E_{NS}(\cdot)$, the author adopted a fixed value of 1 as operating time to all switches.

A good point about this method is that a proper sequence of maneuvers becomes available at the end. Nevertheless, if you read this chapter carefully up to this point, the main drawbacks of this work are probably noticeable already: the proposed operations may cause radiality violations, and there is no mention on how to handle such cases; also, using a fixed penalty term may over or underestimate the constraints, and there is no discussion on the effect of this parameter.

A posterior study by Lambert-Torres *et al.* [61] adopts the same algorithm with some modifications in the computation of pheromone. The method follows the same structure,

but the objective function is not explicitly stated, even though it seems to minimize $S_{NR}(\cdot)$. Unfortunately there does not seem to exist improvements on the topics stated before, so the same strong and weak points also apply.

In general, ACO methods seem to have been underestimated by the works so far, as they have a high potential of satisfying many practical concerns. Given their constructive nature, they are able to work directly in the space of sequences of maneuvers and, being metaheuristics, have higher chances of minimizing the arguably more realistic indices $T_m(\cdot)$ and $E_{NS}(\cdot)$ with more quality when compared to pure heuristics. In any case, this is left as suggestion for future works in the field.

3.3.5 Permutation-based codification

This codification is exactly what it says: considering $\mathcal{E} = \{e_1, e_2, \ldots, e_n\}$ the set of all maneuverable switches, each alternative $\mathbf{x}$ will have the form

$$\mathbf{x} = [e_{\pi(1)}, e_{\pi(2)}, \ldots, e_{\pi(n-1)}, e_{\pi(n)}] \tag{3.7}$$

wherein [21] $\pi(\cdot) : \{1, 2, \ldots, n\} \mapsto \{1, 2, \ldots, n\}$ is a *permutation function*, which associates, for every integer $i \in \{1, 2, \ldots, n\}$, another unique integer $j \in \{1, 2, \ldots, n\}$ such that $\pi(i) = j$. Therefore, the search space $\mathbb{X}$ is the set of all permutations of the elements of $\mathcal{E}$ (or $\mathcal{E}_R$), usually called *symmetric group* and represented by $\mathbb{X} = \mathfrak{S}_{\mathcal{E}}$ (appropriately changed if only the restoration sub-graph is of interest). With this definition, the goal is to find the permutation $\mathbf{x}^* \in \mathfrak{S}_{\mathcal{E}}$ which generates the best restoration plan. Also, any perturbation mechanism that works on permutations (like swapping, reversing slices of the vector etc.) can be adopted to generate neighbors of a solution.

Converting a permutation vector $\mathbf{x}$ into a set or sequence of maneuvers requires an evaluation mechanism, which varies across different works. For instance, in Luan *et al.* [62], the evaluation procedure starts with all switches in the restoration region with the "open" state. Then, given a vector $\mathbf{x}$ of the form of equation (3.7), the method proceeds to close each edge in the order given by $\mathbf{x}$, ignoring the ones that would cause a radiality violation. To allow for load shedding, the authors propose adding a dummy edge with the label '0' such that, whenever it is encountered, the evaluation is interrupted. Thus, a complete new configuration is obtained when either all switches were tested or the dummy edge '0' is visited. After that, each new configuration is evaluated according to a linear aggregation of $S_{NR}(\cdot)$, $P_{loss}(\cdot)$, voltage, current and switch costs (whose meaning is not exactly clear in the paper). This function is minimized by a genetic algorithm.

The main issues of this work are: the usual troubles of aggregating objectives with different scales; in this case, the weights of the constraints should not be set by the decision maker as they are responsible for ensuring feasibility in the final configuration, so this is not guaranteed by the method; no proper sequence of maneuvers is returned; and finally, the space of permutations is much bigger than the binary one[5], and there is no comment on possibilities for reducing the search space and prevent equivalent configurations from different permutations, for example.

A later work is by Carrano *et al.* [25], who propose a more intricate evaluation process. Given a permutation vector $\mathbf{x}$, their procedure follows the steps:

1. Find the first CO switch e^{CO} in $\mathbf{x}$ that is capable of recovering *oos* loads and close it;
2. If the resulting network is feasible, remove e^{CO} from $\mathbf{x}$ and go back to the first step. Otherwise:
   (a) Open CC switches that are downstream to e^{CO} in the order that they appear in $\mathbf{x}$ until the violations are relieved. Call $\mathcal{E}_{sub}^{CC} \subset \mathcal{E}$ such a subset of switches. If there are no more such CC switches to open, revert the changes, drop e^{CO} and go back to the first step;
   (b) If required, open an isolation switch to isolate a fault and include it into $\mathcal{E}_{sub}^{CC}$.
3. Remove possible redundant maneuvers from $\mathcal{E}_{sub}^{CC}$ and go back to the first step.

The evaluation stops when there are no more CO switches to recover any load. Notice that in each iteration we have a stage of maneuvers composed of the closing of e^{CO} and the opening of all edges in $\mathcal{E}_{sub}^{CC}$, and thus, in the end, a proper sequence of maneuvers is available. Moreover, a stage is only finished if voltage, current and overload constraints are satisfied. Also, loops with two CO switches are prevented because such an edge is only closed if it can restore loads, thus preventing radiality violations. Therefore, the evaluation mechanism makes all permutation vectors feasible, and thus the optimization algorithm can perform an unconstrained minimization.

The authors employ a SPEA-2 to minimize $S_{NR}(\cdot)$ and $T_m(\cdot)$, and new permutation vectors are obtained by swapping the position of edges in a vector. Despite modeling all switches of a network, in order to reduce the search space, only the ones directly connected to the restoration region are considered. At the end, they compare their method with a Branch and Bound in the case of single faults, and both return similar solutions, but the proposed one takes much less processing time. One drawback of this

[5]Compare $n!$ with 2^n for $n > 4$.

method, tough, is that the time of maneuvers is computed by assigning a fixed value to each switch, which results in the problems mentioned a few times before and essentially turn this index into a weighted number of maneuvers.

The last study reviewed in this chapter is a follow-up to the previous one by Goulart *et al.* [2], which is essentially the work presented in this thesis. The same evaluation mechanism is employed, and since we decided to focus the attention only in the restoration region, only these edges are modeled in the vector. The main enhancements in this study compared to the previous one in [25] are:

- The energy not supplied is directly used as an objective, together with the unrecoverable power not restored, which prevents the need for setting penalty parameters or estimating fault correction times;
- A constructive heuristic is proposed to estimate the actual time to perform a sequence of maneuvers, which takes into consideration the possible existence of multiple dispatch teams which can operate maneuvers in parallel;
- New perturbation mechanisms are presented that may prevent the creation of permutation vectors inducing virtually the same sequences of maneuvers, and thus reducing the search space;
- An initial step considering only remote switches is proposed, which, if applicable, may recover some loads with nearly zero time. This is a simple yet very effective idea that can be employed in some networks that are starting to become more developed.

Since this is the study treated in this thesis, I will leave most of the details to be further explained in the next chapter.

3.3.6 Discussion

As may have become evident, works adopting metaheuristics are arguably the most diverse, even inside the same codification category. Indeed, we have approaches employing diverse groups or combinations of quality indices, trying to satisfy the constraints with different concepts, and possessing their unique sets of strengths and weaknesses. I feel that this wide range of adaptability, coupled with the capacity of returning good enough solutions in reasonable time turn metaheuristics into the best tool for solving the load restoration problem. Hence, we may as well see a growing number of publications employing such techniques in the future.

3.4 General Discussion

I hope that this broad outline of the literature is enough to introduce the reader to the main challenges involved in solving the load restoration problem. There is too much to consider. There are many practical concerns, lack of uniformity between utilities, time constraints in generally large problems etc. When comparing the three main classes considered in this chapter, we can say that:

1. Mathematical programming-based approaches still require important improvements to handle feasibility/solution quality/time efficiency compromises, and so far they seem applicable only in relatively small networks or as baseline comparison for other algorithms;
2. Heuristics are useful if a quick solution is sought or as starting point for other complex methods;
3. Metaheuristics have possibly the greatest potential among all approaches, and we are able to produce algorithms ranging from extremely simple and only able to deal with toy problems to complex tools capable of handling most of the practical concerns of real world scenarios.

In summary, if we need to devise a method that is able to cover most of the necessities of virtually any system while still returning high quality solutions in a quick time, I believe metaheuristics are the most appropriate for that. Therefore, the proposed algorithm of this work will belong to this class.

3.5 Summary

I started this chapter by mentioning that, thanks to the lack of agreement between authors and utilities on quantifying what a good load restoration plan is, it is extremely difficult to compare works in the literature in a fair way. By now this point of view has probably been reinforced. Even so, this chapter tried to present previous works grouped in categories such that their strengths and weaknesses became evident more easily.

While it is not easy to provide direct comparisons among these works, since they employ different optimization techniques, diverse quality indices to model the problem and distinct encodings for the solutions, it is possible to write down some desirable features the proposed methods should possess. Following the same approach as [25], this information is laid down in Table 3.1, where the features considered are:

Reference	F1	F2	F3	F4	F5	F6	F7	F8	F9
[31]	N	**Y**	N	**Y**	-	N	N	N	N
[32]	*P*	**Y**	N	**Y**	**Y**	N	N	**Y**	N
[33]	*P*	*P*	-	**Y**	**Y**	N	N	N	N
[6]	*P*	**Y**	-	**Y**	N	N	N	N	N
[34]	N	**Y**	N	**Y**	**Y**	**Y**	N	**Y**	N
[35]	N	N	N	N	-	N	N	**Y**	N
[38]	N	*P*	N	N	N	N	N	N	N
[36]	N	**Y**	**Y**	**Y**	**Y**	**Y**	N	**Y**	N
[37]	N	**Y**	**Y**	**Y**	**Y**	**Y**	N	**Y**	N
[39]	N	*P*	-	N	**Y**	N	N	N	N
[40]	N	**Y**	**Y**	**Y**	**Y**	**Y**	N	N	N
[42]	N	*P*	-	**Y**	**Y**	**Y**	N	**Y**	N
[43]	N	**Y**	-	**Y**	**Y**	N	**Y**	N	N
[41]	N	-	-	-	**Y**	-	N	N	N
[47]	*P*	-	-	-	**Y**	N	N	N	N
[48]	**Y**	**Y**	-	N	**Y**	N	N	N	N
[51]	*P*	N	-	N	**Y**	N	N	N	N
[52]	*P*	*P*	**Y**	**Y**	**Y**	N	**Y**	N	N
[22]	*P*	-	**Y**	**Y**	**Y**	N	N	N	N
[53]	*P*	**Y**	-	N	-	**Y**	**Y**	N	N
[27]	*P*	*P*	**Y**	**Y**	**Y**	N	N	**Y**	N
[23]	*P*	-	-	N	**Y**	N	N	N	N
[54]	*P*	*P*	-	N	**Y**	N	N	N	N
[55]	*P*	*P*	-	N	-	N	N	**Y**	N
[57]	*P*	*P*	**Y**	N	**Y**	N	N	N	N
[24]	**Y**	**Y**	N	**Y**	**Y**	**Y**	**Y**	**Y**	*P*
[26]	*P*	N	-	**Y**	**Y**	N	N	**Y**	*P*
[61]	N	N	-	**Y**	**Y**	N	N	**Y**	N
[62]	*P*	N	-	**Y**	**Y**	N	N	N	N
[25]	**Y**	**Y**	**Y**	**Y**	**Y**	**Y**	**Y**	**Y**	*P*
[2]	**Y**	**Y**	**Y**	**Y**	**Y**	**Y**	**Y**	**Y**	**Y**

TABLE 3.1: A summary with some desirable features (F1-9, described in the text) the studies cited in this work should possess. Inside each cell, **Y** means 'yes', *P* means 'partially', N stands for 'no' and - is used when the information is not provided or not evident from the paper.

F1: Is the formulation multiobjective to try to take both "restore most *oos* loads" and "in the shortest time possible" conditions into account? (Y) Yes, and there is some kind of decision support system to help choosing a final solution[6]; (P) yes, but the DM has either no control over the final solution or the decision support system is unsatisfactory; (N) no, it is single-objective.

[6]Examples of this decision support system are an *a priori* approach with a suitable initial choice of weights or other parameters, or an *a posteriori* technique that returns a sample of the Pareto-optimal front. However, notice that for more than four objectives the problem belongs to the many-objective field, and simply returning this sample is not helpful anymore (see discussion in [58, 59]).

F2: Are the radiality, voltage, current and feeder capacity constraints considered? (Y) the final solution is always feasible; (P) yes, but there is no guarantee that the final configuration will satisfy the constraints (e.g., they are included as penalties but the penalty factor is not properly adjusted); (N) they are either neglected or not even mentioned in the development of the method.

F3: Can the method handle simultaneous failures?

F4: Is the technique able to perform partial restoration, that is, does it work if not all *oos* loads are recoverable without violating constraints?

F5: Is the method guaranteed to output a solution in reasonable time for real world systems (up to ten minutes, for instance)?

F6: Can the method handle priority loads, that is, is it able to give preference to maneuvers that restore them in the initial stages or perform load shedding mostly on less important nodes?

F7: Does the method distinguishes manual from remote switches, specially their time differences when operated?

F8: Is a feasible sequence of maneuvers returned or evident from the final solution?

F9: Can the method compute an appropriate time to execute the maneuvers? (Y): Yes; (P) It is mentioned, but a fixed value is assigned to each switch, which ignores the current position of a dispatch team; (N) No, the number of maneuvers is considered at best.

Please notice that this table is not intended as the ultimate comparison of the works reviewed. First, we are not supposed to simply count the number of "yes" columns as the features are not equally desirable/relevant for all utilities. Also, it is not meant to be as simple as "work A has more **Y**s and partials than B, so it is better". In fact, it can be said that this table is somewhat biased, and thanks to the lack of universal agreement of a good restoration plan, almost anyone can construct a similar table favoring his/her work. For instance, the proposed method does not take power losses into account for the reasons mentioned in [6], and it also prevents load transfer because it depends on a number of practical factors that must be well-enumerated by the utility. Thus, it would receive two nos if these features were enumerated.

So, what is the purpose of building such a table? It is basically twofold: (i) to serve as a quick reference for picking an appropriate method depending on a given necessity, and (ii) to provide suggestions on how we can progress with new studies. In that regard, the

inclusion of new features is welcome as the published methods improve and most of the previous issues start to be commonly satisfied.

I personally believe that, in the current state of the literature, the most important feature that should be well satisfied are the handling of sequences of maneuvers in order to optimize a *realistic* estimation of the time of maneuvers (or even energy not supplied). This work was developed as a first step towards filling this gap (along with other minor contributions), and I hope to see improved studies also attacking such priorities.

Chapter 4

Proposed method: Time and Energy not supplied Estimation

4.1 Introduction: Sets *versus* Sequences

The difference between a set $\mathcal{M}$ and a sequence M of maneuvers was already explained in section 2.2.2, but let us make a quick review. Assume a distribution system is represented by a graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$, with the edges $\mathcal{E} = \mathcal{E}^{CO} \cup \mathcal{E}^{CC}$ indicating the set of all maneuverable switches, containing both the currently open (CO) and currently closed (CC) ones, and $\mathcal{N}$ the set of sectors, which are groups of loads with no maneuverable switch connecting them. A set of maneuvers $\mathcal{M} \in 2^{\mathcal{E}}$ is a subset of edges indicating switches with different state from the post-fault configuration – and they come in any arrangement –, while a sequence of maneuvers $M \in \mathfrak{S}_{2^{\mathcal{E}}}$ is a similar subset, but providing an order in which each operation is performed.

Sets of maneuvers have the benefit for being generally evident just by looking at the post-fault state and a new configuration. Thus, they play very nicely with modern methods that abstract the graph into a different encoding (binary, forest, permutation-based etc.) for better efficiency in the search. However, apart from very simple cases with very few maneuvers, this gain in efficiency is essentially wasted if the dispatcher is not able to properly lead the initial configuration to a better one. This is where the importance of M comes into play.

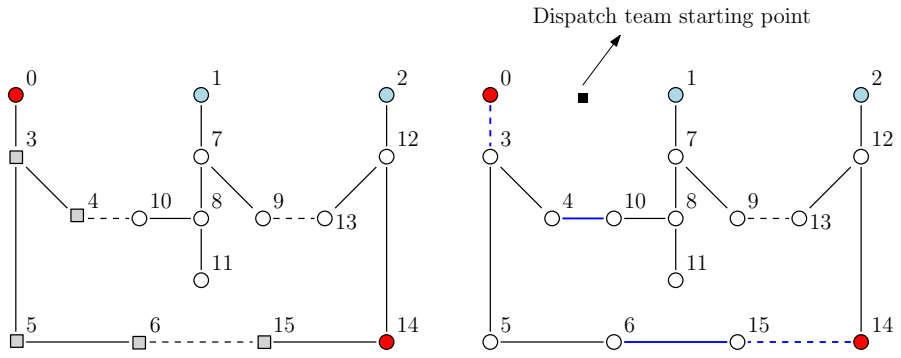
A proper sequence of maneuvers M should be able to:

- Prevent some operations that may cause constraint violations or physical impairments in the system. For instance, we cannot energize *oos* loads if the region was

not appropriately isolated from the faulted sector. Also, if a CO switch is closed before the load shedding maneuvers were executed, the supporting feeder would energize more loads than expected, causing problems to the healthy *ins* loads. In summary, this first condition can be satisfied by ensuring that some *precedence rules* [1] among the switches are respected;

- Select the order of operations not only for complying with the previous rules, but also to minimize another criterion, such as operating time. For instance, assume in Figure 4.1 that the geographical positions of the sectors are just as depicted, and each switch is operated at the middle point of an edge. Also, suppose for simplicity that a dispatch team moves with constant speed in straight lines.[1] Under these circumstances, between the two *feasible* sequences of maneuvers, the first one may require less time to complete than the second;

- In case of more than one dispatch team, the switches to be operated should be properly assigned to each crew, and the possible time savings by making more than one maneuvers in parallel should be taken into account.



$\mathcal{M} = \{(0,3),(4,10),(6,15),(14,15)\}$

Two feasible sequences of maneuvers:
$M_1 = [(0,3),(4,10),(14,15),(6,15)] \rightarrow$ probably faster
$M_2 = [(14,15),(0,3),(4,10),(6,15)] \rightarrow$ probably slower

FIGURE 4.1: Generating two feasible sequences of maneuvers from a set. Depending on the starting point of the dispatch team(s), even by respecting precedence rules it is possible to prefer some sequences for demanding less time to implement, for instance.

According to the literature review presented in the previous chapter, recent studies are starting to recognize the importance of returning a sequence instead of just a new configuration. However, the effort has been limited to respecting the first condition only, and so far (at least to the extent of my knowledge) no other study has tried to

[1] Do not worry, even by assuming here that dispatch teams have their own dedicated helicopter or are able to walk through walls, the main concept of the approach proposed in this chapter remains valid when the crew has to face traffic and other obstacles.

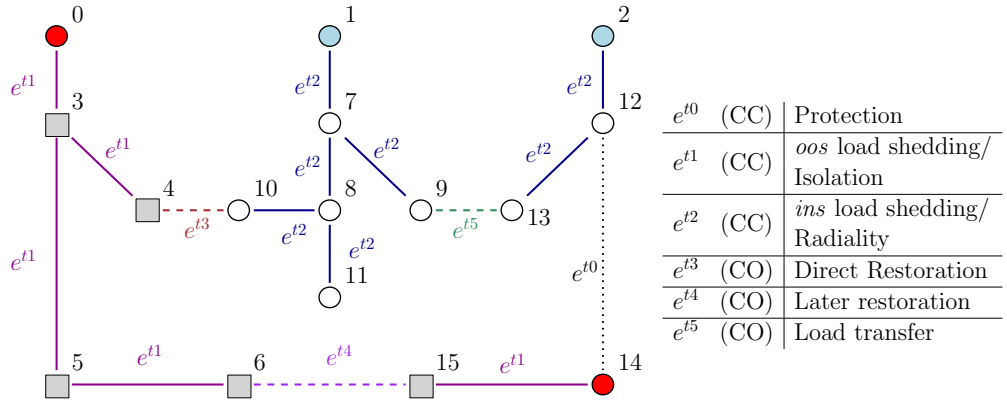
provide an adequate estimation of $T_m(\cdot)$ [or $E_{NS}(\cdot)$], much less considering the presence of more than one dispatch team. There may actually be a good reason for that, since guaranteeing both of these conditions may be very hard. Think about it: we employ an optimization algorithm handling a codification $\mathbb{X}$, and each alternative $\mathbf{x} \in \mathbb{X}$ should be mapped to a set $\mathcal{M}$, which, in turn, can be mapped into a number of sequences, some not respecting the precedence rules and, among the ones that do, we are still supposed to find the one that takes the smallest time (or energy not supplied). No wonder the previous works (probably intentionally) skipped this last laborious step and simply returned any feasible sequence (or none at all).

In this chapter I present a method for computing a sequence M that aims at minimizing $T_m(\cdot)$ and $E_{NS}(\cdot)$ from a given set $\mathcal{M}$, assuming the appropriate precedence rules are given. The main reason for this method is to allow the optimization to be performed in terms of these arguably more realistic indices (as we shall see later) instead of the number of maneuvers.

4.2 Preliminary definitions

4.2.1 Switch classification

The load restoration process consists in opening and closing switches. However, as mentioned in the beginning, in order to prevent risks such as energizing faults and to respect the utility's requirements, these maneuvers cannot be executed in any order given the precedence rules among the switches. To help with the determination of these rules, it is instructive to classify the switches (for a given scenario of fault) as shown in Figure 4.2:



e^{t0}	(CC)	Protection
e^{t1}	(CC)	*oos* load shedding/ Isolation
e^{t2}	(CC)	*ins* load shedding/ Radiality
e^{t3}	(CO)	Direct Restoration
e^{t4}	(CO)	Later restoration
e^{t5}	(CO)	Load transfer

FIGURE 4.2: Classification of switches.

Type 0 (Protection) CC switches opened to clear the fault, normally being the first upstream switches to interrupted sectors, assuming a coordinated protection (Appendix B). They are automatically activated by relays or by blowing fuses (in case of faults) or manually by a dispatch team (in case of maintenance), so we can assume them as opened until the fault is cleared and ignore them in this work. Edge $(12, 14)$ is such an example in Figure 4.2.

Type 1 (*oos* Load shedding/Isolation) CC switches connecting two *oos* nodes. They either (i) isolate the interruption when a neighboring node is energized [such as edges $(0, 3)$ and $(14, 15)$ in Figure 4.2] or (ii) are used for load shedding when the supporting feeder is not able to recover all *oos* loads without violating constraints.

Type 2 (*ins* Load shedding/Radiality) CC switches connecting two energized nodes. They are used either for (i) load shedding of in service (*ins*) sectors with less priority to provide more capacity to the supporting feeder or (ii) to guarantee radiality during a load transfer operation.

Type 3 (Direct restoration) CO switches connecting an energized node with an *oos* one [e.g., $(4, 10)$ in Figure 4.2]. They are the ones responsible for restoring energy to disconnected loads, providing what is sometimes known as Level 1 restoration [20].

Type 4 (Later restoration) CO switches connecting two *oos* nodes [e.g., $(6, 15)$ in Figure 4.2]. They can be neglected initially, but may become useful later and turn into a Type 3 if one of the endpoints becomes energized.

Type 5 (Load transfer) CO switches connecting two energized nodes [e.g., $(9, 13)$ in Figure 4.2]. They are used for load transfer, shifting some nodes of the supporting feeder to a neighboring one and thus increasing its capacity. In this case they will be mentioned as Level 2 restoration. Notice that Levels 3 or higher are also possible.

We shall refer to the sets of these switches as $\mathcal{E}^{ti}$, $i = 1, \ldots, 5$. Keep in mind that such classification depends on a given fault scenario and on the current state of the switches. Every time an operation is executed, elements of a set can change.

4.2.2 General procedure for a restoration plan

With the previous classification, a general procedure for obtaining a restoration plan, outlined in [2], can be described here as follows:

1. Attempt to close a $e^{t3} \in \mathcal{E}^{t3}$ switch to recover loads;
2. If this causes a violation, then:
   - Perform load transfer from the supporting feeder to a neighboring one by closing some Type 5 switches $\mathcal{E}^{t5}_{sub} \subseteq \mathcal{E}^{t5}$ and opening appropriate Type 2 edges $\mathcal{E}^{t2}_{sub} \subseteq \mathcal{E}^{t2}$ to keep radiality; and/or
   - Perform load shedding of in service nodes by opening a subset of Type 2 switches $\mathcal{E}^{t2}_{sub} \subseteq \mathcal{E}^{t2}$; and/or
   - Perform load shedding of out of service nodes by opening a subset of Type 1 switches $\mathcal{E}^{t1}_{sub} \subseteq \mathcal{E}^{t1}$.
3. If a feasible configuration is obtained, open an adequate $e^{t1} \in \mathcal{E}^{t1}$ switch to isolate an interrupted node, if required.

The use of subsets $\mathcal{E}^{t1}_{sub}$, $\mathcal{E}^{t2}_{sub}$ and $\mathcal{E}^{t5}_{sub}$ in the second step covers the necessity in some cases of opening various CC switches to prevent constraint violations in load shedding, and the possibility of higher orders of load transfers (which may or may not be desirable, depending on the utility). Notice that Type 4 switches are only considered when they become a Type 3 one, hence their absence in the algorithm.

This completes a stage, and the process should be repeated until there is no $e^{t3} \in \mathcal{E}^{t3}$ switch available that can restore loads without violations. The choice of which edges to operate in each step is one of the factors that differentiate most methods, as became evident from the literature review chapter.

4.3 Proposed method

4.3.1 Determination of precedence rules

The precedence rules dictate which switches should not be operated before some others. This is required to prevent maneuvers that may feed faulted sectors, overloads in supporting feeders, and to respect company's general rules. In general, they may vary from utility to utility, and this is just another source of difficulties when comparing different methods. Even so, here is a sample of guidelines that may be valid for most radial distribution systems:

Case 1 For load shedding of *oos* nodes only:

- Assuming the fault was properly isolated, Type 1 switches can be operated at any moment since they do not involve energizing loads. Also, there is no precedence among them. So, for instance, in the top left restoration plan of Figure 4.3 (shown in chapters 1 and 2, but repeated once more for convenience), $(2, 12)$ and $(2, 14)$ can be opened at any moment in any order;
- Type 3 edges can only be closed after the appropriate Type 1 switches were opened, or else it may cause overloads or even feed a faulted node. For instance, again in the top left panel of Figure 4.3, $(9, 13)$ can only be closed after $(2, 12)$ and $(12, 14)$ were opened;
- Unless there are additional rules provided by the underlying optimization algorithm, there are no precedences among Type 3 edges themselves, and they can be closed at any time as long as their Type 1 precedences are satisfied;
- Type 4 edges have the same kind of precedences as Type 3 after one of its endpoints becomes energized. Thus, they require adequate Type 1 edges and a Type 3 switch to be maneuvered. For instance, in Figure 4.2, to recover node 15, switch $(6, 15)$ could only be closed after $(4, 10)$ and $(14, 15)$ were operated.

Case 2 For load shedding of both *ins* and *oos* nodes, all previous rules still apply, together with:

- In this case, Type 2 switches only disconnect loads (which is assumed to be an acceptable outcome from the utility), so there is no risk of overload or energizing faults. Thus, like Type 1 edges, they have no precedence among any other switches or among themselves. For example, $(8, 11)$ in Figure 4.3 can be opened at any moment;
- To prevent possible overloads, Type 3 edges belonging to feeders with *ins* load shedding must wait for the appropriate Type 2 operations besides the usual Type 1 edges. Then, $(9, 13)$ in Figure 4.3 must wait until $(8, 11)$ is opened, apart from $(2, 12)$ and $(14, 15)$.

Case 3 For load transfer and all types of load shedding, aside from the previous rules in **Case 1**, we have:

- As exemplified at the bottom of Figure 4.3, depending on the dispatcher's decision we can set Type 2 switches to be opened only after the corresponding Type 5 were closed, or vice-versa. Thus, the decision to first close $(4, 10)$ to open $(7, 8)$ in the bottom plan of Figure 4.3 or the opposite should be provided *a priori*, and the implications of each one were already mentioned before;

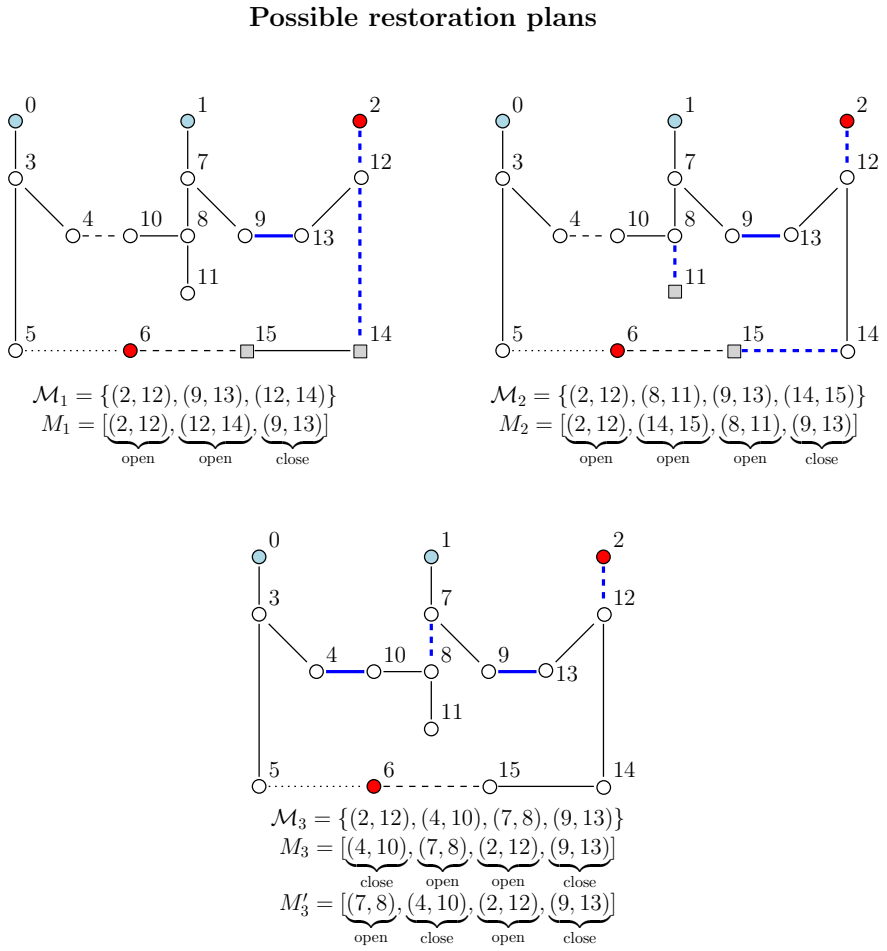
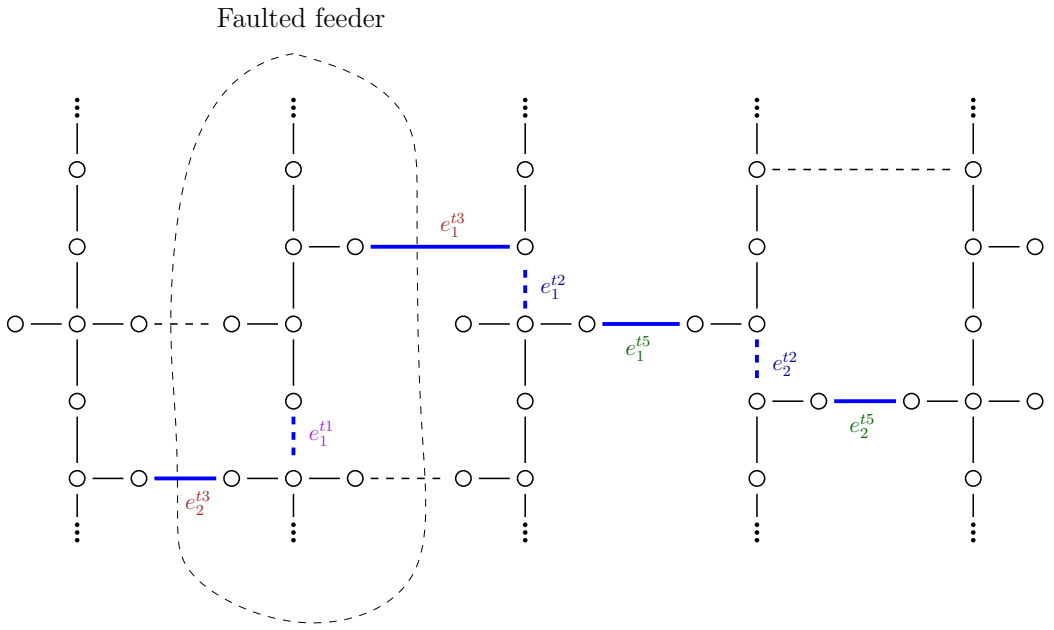


FIGURE 4.3: Examples of restoration plans for each case. Below each plan are shown the sets of operated switches and possible sequences in which each maneuver is performed.

- For higher levels of restoration, we should start with the highest level of load transfer to relieve the back-up feeders until a direct restoration with a Type 3 switch is performed. This is illustrated in Figure 4.4, which shows the portion of a larger distribution system. A fault (not shown) at the second feeder has its *oos* region recovered with a Level 3 restoration. The load transfers must start from the rightmost feeder and work up to the closest one which effectively energizes loads. The precedence rules are shown in the figure. In this case, closing e_1^{t3} requires, apart from the previous precedences, all the load transfers. In contrast, e_2^{t3} performs only a Level 1 restoration, so it only demands e_1^{t1} to be open.

Keep in mind that it is not always necessary to enumerate all precedences for a switch. For instance, in Figure 4.4, even if e^{t3} requires all $\{e_1^{t1}, e_1^{t2}, e_2^{t2}, e_1^{t5}, e_2^{t5}\}$ to be operated, we can write only $\{e_1^{t1}, e_1^{t2}, e_1^{t5}\}$ as we know that the pair $\{e_1^{t2}, e_1^{t5}\}$ is only maneuvered after $\{e_2^{t2}, e_2^{t5}\}$. Moreover, these guidelines can be overwritten or appended by any specific rule



Assuming closings before openings during load transfer (for openings before closings, reverse e_i^{t5} with e_i^{t2}):

- e_2^{t5} can be closed at any time, and e_2^{t2} only after;
- e_1^{t5} can be closed after the pair (e_2^{t5}, e_2^{t2}) was operated, and e_1^{t2} only after;
- e_1^{t3} can be closed after the pair (e_1^{t5}, e_1^{t2}) was operated, apart from e_1^{t1} and possible isolation switches.

Finally, e_2^{t3} depends only on e_1^{t1} to be opened before.

FIGURE 4.4: Precedence rules for higher level restoration. For a fault in the second feeder, the load transfers must start from right to left.

provided by the utility or the underlying optimization algorithm. In any case, we stress that these rules should be available *before* any algorithm for generating a restoration plan can be executed, together with the types of acceptable maneuvers (whether load shedding of *ins* loads is available, if load transfer is possible, and if yes, whether to close or open before etc.).

In this work, we focus only on **Case 1**, which handles switches in the *oos* region. The main reason for this choice is to maintain compatibility with some Brazilian distribution utilities, which have a policy of "no messing with healthy loads". Also, the specific rules of the other cases (for instance, only some particular *ins* loads can be shed, or only some feeder may be connected during load transfer) may prevent proposing an approach applicable to a large number of systems, so they are left for future studies. In any case, it is instructive to mention that the proposed algorithms for estimating $T_m(\cdot)$ and $E_{NS}(\cdot)$ are still valid for the other cases, as long as the precedences are appropriately determined.

The precedence rules for a given set of maneuvers may be evident in simple examples, but for more general cases, Figure 4.5 presents a pseudo-code for determining such precedences according to the aforementioned guidelines for **Case 1**. This algorithm

respects the previous instructions, and requires no additional load flow executions nor violation tests apart from the ones used to obtain the new feasible configuration, only some simple graph operations (such as determining subsets of nodes connected to a given node). Hence, the method runs very quickly. Also, any additional set of rules can be included as well.

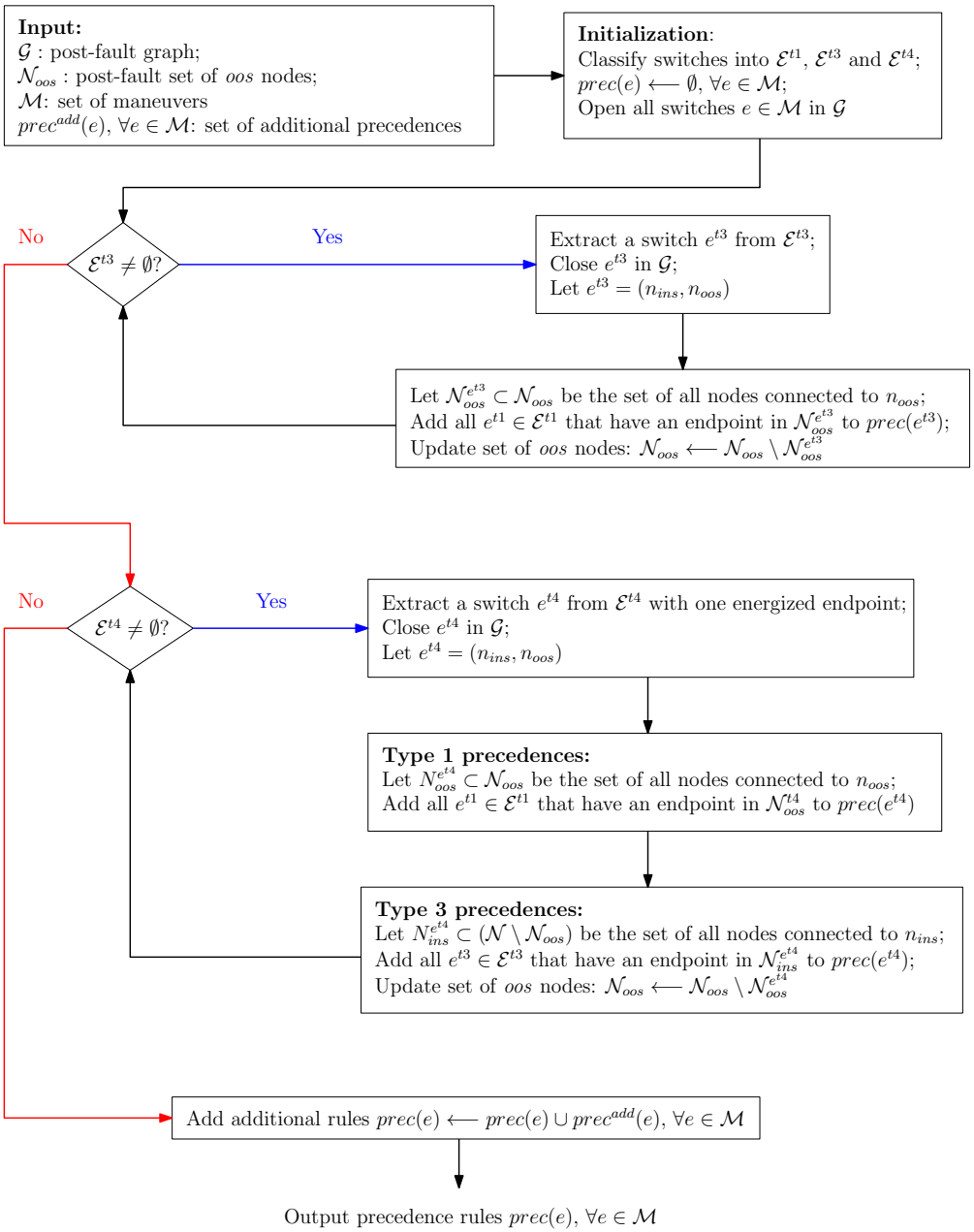


FIGURE 4.5: Pseudo-code for determining precedence rules for **Case 1**: load shedding of only *oos* loads. Notice here that only Types 1, 3 and 4 are used in this case.

4.3.2 A constructive heuristic for the Time of Maneuvers

Even by respecting the rules of precedence, there can be a number of sequences induced by a set of maneuvers. As argued before, they are not necessarily the same solution since they energize different loads at distinct instants, and thus would possess diverse values of $T_m(\cdot)$ and $E_{NS}(\cdot)$. Furthermore, the availability of more than one dispatch team can also influence these numbers depending on how the crew is coordinated. Regarding $T_m(\cdot)$, we proposed in [1] a mathematical formulation based on the scheduling problem for determining a sequence that minimizes the total time of maneuvers and can handle multiple dispatch teams, and in [2] a constructive heuristic was presented to handle **Case 1** of restoration, which was also considered in that work. Despite treating the same case, we extend this heuristic here to any restoration situation as long as the rules of precedence are available.

Assume there is a set $\mathcal{T} = \{t_1, \ldots, t_{|\mathcal{T}|}\}$ with $|\mathcal{T}|$ available dispatch teams with known initial locations. The time taken for a team $t \in \mathcal{T}$ to operate (open or close) a switch $e \in \mathcal{M}$ is given by a cost

$$c_{t,e} = c^{disp}_{p_t,p_e} + c^{op}_{t,e} \tag{4.1}$$

with p_e indicating the geographical position of switch $e \in \mathcal{M}$, and p_t the current position of team $t \in \mathcal{T}$. In equation (4.1), $c^{disp}_{p_t,p_e}$ is a cost indicating the time taken for team t to move from its current position p_t to p_e, which usually depends on traffic conditions and can be estimated on real-life scenarios with mapping or GPS software such as Google Maps, Openstreetmaps etc.; and $c^{op}_{t,e}$ is a time cost to actually operate e once t reached its location, which accounts for practical delays such as checking if a team is cleared to execute the maneuver.

With this, a constructive heuristic to coordinate the dispatch teams and estimate the total time of maneuvers is provided in Figure 4.6. First, all remote controlled switches with no precedences are operated, as detailed at the top of the figure. The algorithm uses a somewhat loose notation when including the elements of an unordered set $\mathcal{M}^{\emptyset}_{rem}$ into an ordered one. This is not a problem in this situation because we assumed that automatic switches are activated with negligible time, so the order in which they are included does not matter (assuming their precedences are before them). Notice that this process should be iterative, since processing some maneuvers now may satisfy the precedences of a posterior remote switch, which can thus be operated right after. Thus, the execution

is repeated until there is no remote controlled switch with empty precedences to be maneuvered.[2]

After this, the main loop consists in operating manual switches with all precedences satisfied, and for that we choose the team that operates the switch with smallest time cost (added with possible additional costs of previous maneuvers). Then, we go over the automated process again in case some remote-controlled switches had their precedences satisfied. This complete process is repeated until all edges are assigned to a team at specific instants, and a proper sequence of maneuvers M and the operating time of each maneuver $T_m(e)$, $\forall e \in M$, are returned. The algorithm generalizes the constructive heuristic used in [2] to manage **Cases 2** and **3** and to handle a complete set of maneuvers directly instead of one stage at a time. As a final remark, observe that the coordination of maneuvers, i.e., which team operates which switch at each time, is also returned. More specifically, by storing which t_{min} operates which e_{min}, we have a complete load restoration plan.

4.3.3 Estimation of the Energy not Supplied

The previous heuristic is able to return a proper sequence M together with the operating instant of each maneuver $T_m(e)$, $e \in M$. With this, equation (2.9) can be applied directly as the heuristic of Figure 4.6 is executed: just perform the operations in M in sequence, computing or updating the set of *oos* nodes (and thus the current [weighted] power not supplied) after each operation, and using the available time to compute the ENS contribution of each maneuver. At the end, just add each individual portion to return the overall index value.[3]

4.4 Results

To get a quick grasp on how the proposed method can improve on practical aspects of an optimization algorithm, consider the usual test system with 16 buses used in most of the examples of this thesis. The nodes and edges data (complex power demand, nodes geographical positions, line impedances, maximum current etc.) can be found in [29, 63], and in this work we assume a minimum voltage of 0.95 pu in all loads. Notice that, in this case, all branches contain a maneuverable switch, so the complete and contracted graphs ($\mathcal{G}_D$ and $\mathcal{G}$, respectively) are identical.

[2] Observe that the case of a complete automated system can also be handled here, and the algorithm ends after this initial loop with a sequence M satisfying all precedence rules and with a time $T_m(M) = 0$.

[3] Just remember that the the unrecoverable portion still needs to be handled separately as discussed in section 2.2.3.

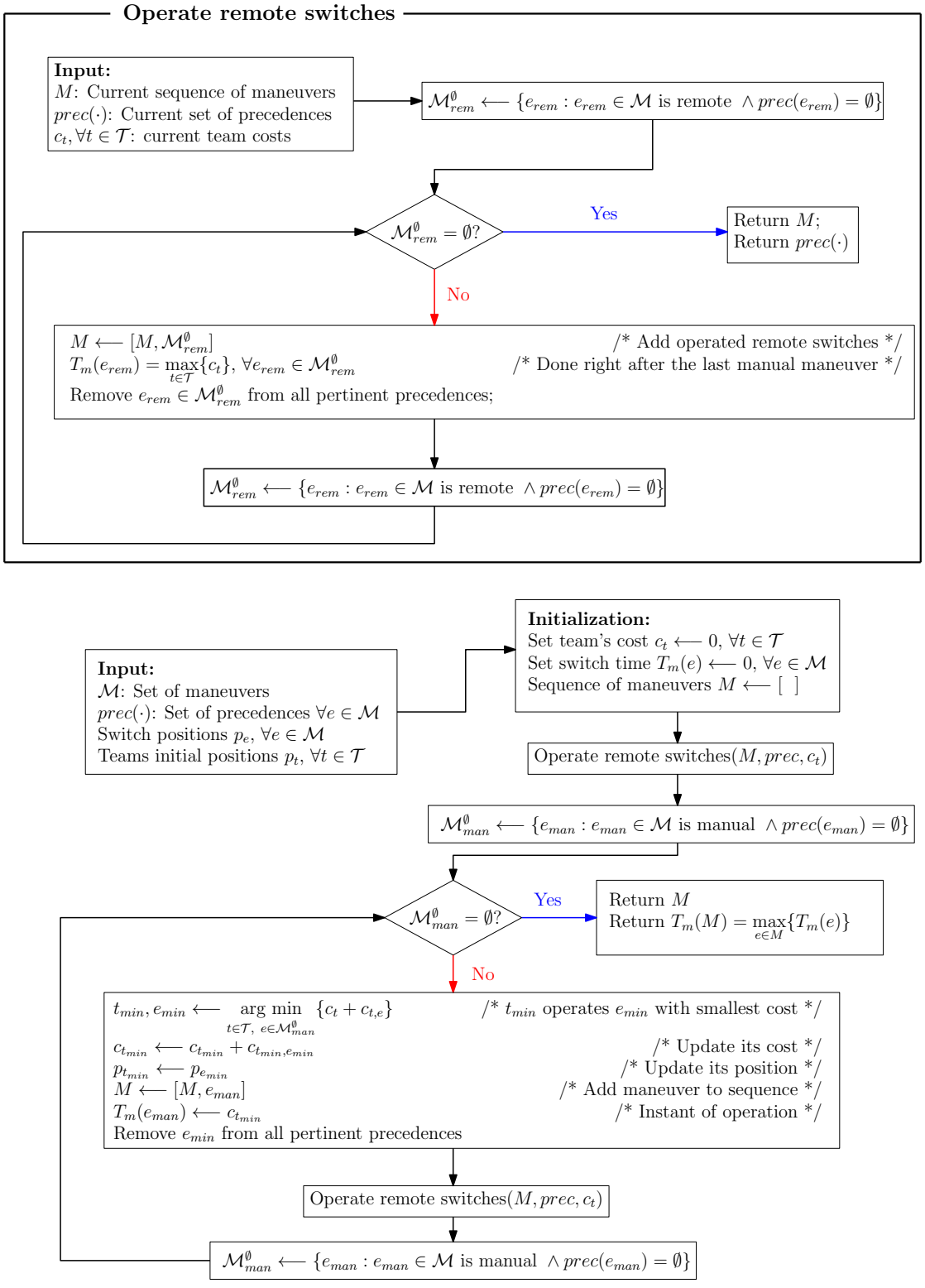


FIGURE 4.6: Constructive heuristic to output a sequence M estimating the best time of maneuvers.

Suppose there were two simultaneous faults at sources 0 and 1 as shown in Figure 4.7, which leaves two whole feeders out of service. For simplicity, let us model the edges into a binary codification, such that the new configured network is represented by a vector

$\mathbf{x} \in \{0,1\}^{|\mathcal{E}_R|}$. Since in this work we focus on **Case 1**, only the edges in $\mathcal{E}_R$ of the restoration subgraph $\mathcal{G}_R$ will be mapped, thus leaving switches (2,12) and (12,14) fixed.

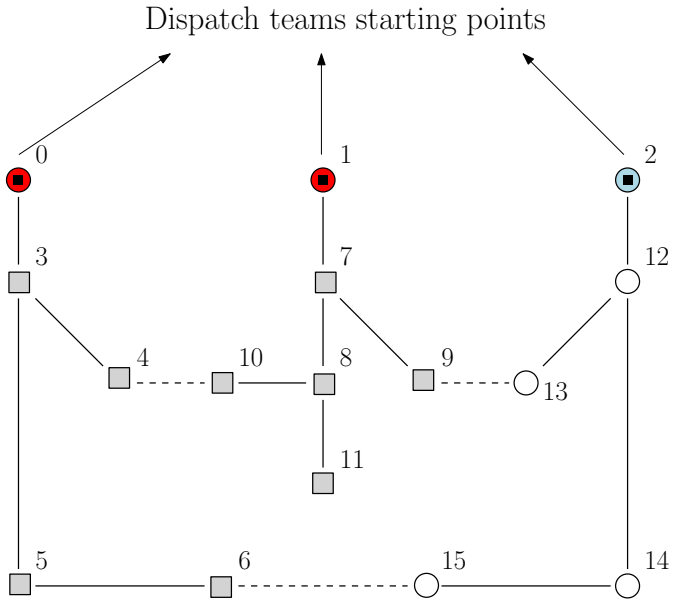


FIGURE 4.7: Simple test system with 16 nodes from [29, 63].

The assumptions for this test are:

- There are three dispatch teams, each one starting at a source node;
- Since the actual position of the switches is not provided, they will be placed at the mean position between its two endpoints in the graph [e.g., switch (0,3) is operated at the center of the geographical positions of nodes 0 and 3];
- The time cost to move between two positions is proportional to the Euclidean distance among them;
- The teams are cleared to operate a switch as soon as they get to their location, so $c^{op}_{t,e} = 0$, $\forall e \in \mathcal{M}, \forall t \in \mathcal{T}$.

Note that these assumptions do not take away the practical character of the method. In practice, this information can be assembled quickly before the optimization begins. Regarding the third assumption, we only require a matrix of time costs (that can be assembled with mapping APIs as mentioned in section 4.3.2) which is queried every time we require the cost between the team's current location and a given switch. In this case we just happened to use a simple distance matrix to cover for the present needs.

Because this is a small network, we can try all different combinations ($2^{12} = 4\,096$ in this case) and return the Pareto-optimal ones according to a given combination of quality indices. More specifically, we perform the following exhaustive search:

1. Initialize an archive $\mathcal{P}_A \longleftarrow \emptyset$ of non-dominated solutions;
2. For each binary string $\mathbf{x} \in \{0,1\}^{|\mathcal{E}_R|}$:
   (a) Extract the corresponding set of maneuvers $\mathcal{M}$;
   (b) Check if the resulting network $\mathcal{G}^{CC}(\mathcal{M})$ is feasible (i.e., it is radial, respect the voltage, current and feeder capacity constraints and the faults are properly isolated). If it violates constraints, discard this configuration;
   (c) If feasible, extract the precedence rules $prec(e), \forall e \in \mathcal{M}$, using the algorithm outlined in Figure 4.5;
   (d) Return a sequence M with an estimation of the time of maneuvers and the recoverable portion of the energy not supplied, as well as the (weighted) power not restored and the number of maneuvers;
   (e) Add each subsequence of M into $\mathcal{P}_A$ and remove the dominated solutions according to a predefined combination of quality indices.
3. Return $\mathcal{P}_A$.

In step 2e, remember that, if we consider a sequence M as a solution, then each subsequence consisting of the first i maneuvers, $M^{(1:i)}$, is also a different solution with its own (possibly distinct) values of $N_m(\cdot)$, $T_m(\cdot)$, $E_{NS}(\cdot)$ and $S_{NR}(\cdot)$. Then, each configuration $\mathbf{x}$ induces a sequence M which adds $|M|$ solutions to the archive at each iteration.[4]

Table 4.1 shows the Pareto-optimal set when we minimize the combination of $S_{NR}(\cdot)$ and $T_m(\cdot)$. For completeness, the values of $N_m(\cdot)$ and $E_{NS}(\cdot)$ are also shown. The units of time and energy not supplied are given in a standardized time and energy units, but the power not supplied is provided in percent of the total *oos* loads for better comparison. Also, as a baseline, the results of the "do nothing" solution (which is simply the post-fault configuration) are also presented.

There are some interesting points to be made about this result. First, notice how all precedence rules are satisfied in all sequences: the appropriate opening is always performed before, and in the last solution the later restoration switch $(4, 10)$ is only closed after $(9, 13)$. Next, see how all sequences generating partial restoration are considered equally good, despite the difference in numbers of maneuvers. This example shows how a proper coordination of multiple teams can efficiently speed up the process. In this case, solutions with four maneuvers could be executed as quickly as one with only two, thus providing an evidence against the common assumption that "a solution with less

[4] Of course, some sub-sequences are very likely dominated, but this is appropriately dealt with the non-dominated sorting step.

i	**Solution**	N_m	T_m (pu)	E_{NS} (pu)	S_{NR} (%)
0	post-fault	0	0	0	100%
1	$[\underbrace{(1,7)}_{open}, \underbrace{(9,13)}_{close}]$	2	2.5	6.88×10^{-4}	36.30%
2	$[\underbrace{(1,7)}_{open}, \underbrace{(3,5)}_{open}, \underbrace{(9,13)}_{close}]$	3	2.5	6.88×10^{-4}	36.30%
3	$[\underbrace{(1,7)}_{open}, \underbrace{(3,4)}_{open}, \underbrace{(9,13)}_{close}]$	3	2.5	6.88×10^{-4}	36.30%
4	$[\underbrace{(0,3)}_{open}, \underbrace{(1,7)}_{open}, \underbrace{(3,4)}_{open}, \underbrace{(9,13)}_{close}]$	4	2.5	6.88×10^{-4}	36.30%
5	$[\underbrace{(0,3)}_{open}, \underbrace{(1,7)}_{open}, \underbrace{(3,5)}_{open}, \underbrace{(9,13)}_{close}]$	4	2.5	6.88×10^{-4}	36.30%
6	$[\underbrace{(0,3)}_{open}, \underbrace{(1,7)}_{open}, \underbrace{(9,13)}_{close}]$	3	2.5	6.88×10^{-4}	36.30%
7	$[\underbrace{(0,3)}_{open}, \underbrace{(1,7)}_{open}, \underbrace{(9,13)}_{close}, \underbrace{(4,10)}_{close}]$	4	2.62	7.00×10^{-4}	0.00%

TABLE 4.1: Pareto-optimal solutions when minimizing $T_m(\cdot)$ and $S_{NR}(\cdot)$. $S_{NR}(\cdot)$ is given as a percentage of the total.

maneuvers takes less time". Of course, with a little attention the reader can see that some maneuvers in these solutions are definitely useless – why open $(0,3)$ or $(3,4)$ if their feeder is not even energized in these cases? Well, the reason for this is because of the overly simplistic approach used to compute new solutions, which performs an exhaustive search with no discrimination. With more sophisticated methods these solutions would not even be present in the final solution set. In any case, the most important point here is to show the power and importance of considering the time and the presence of multiple teams in the formulation.

If we replace $T_m(\cdot)$ with $E_{NS}(\cdot)$ we get a similar set of Pareto-optimal solutions. Then, let us now try the usual approach in the literature of minimizing the combination $[S_{NR}(\cdot), N_m(\cdot)]$, which culminates in the Pareto-optimal set shown in Table 4.2. In this case the solutions were stored according to the number of maneuvers, but the time and energy not supplied were estimated with the proposed algorithm. Thus, for each of the sequences 2, 3 and 4 there are four more combinations with the same values of $T_m(\cdot)$ and $E_{NS}(\cdot)$ but which may be considered different for some optimization algorithms.

In this case, the partial restoration was obtained only with the smallest number of maneuvers, as expected, but we received many options for the full restoration case. Comparing solutions 2 and 3 we can see the pitfalls of neglecting the relative positions of the teams, which results in different operating times for each switch at each stage. In fact, we can see in Figure 4.7 that switch $(6,15)$ is further from the teams' initial positions when compared to $(4,10)$, thus option 3 should actually be discarded from the list. Furthermore, compare sequences 3 and 4. Besides taking the same time, the

i	**Solution**	N_m	T_m (pu)	E_{NS} (pu)	S_{NR} (%)
0	post-fault	0	0	0	100%
1	$[\underbrace{(1,7)}_{open}, \underbrace{(9,13)}_{close}]$	2	2.5	6.88×10^{-4}	36.30%
2	$[\underbrace{(0,3)}_{open}, \underbrace{(1,7)}_{open}, \underbrace{(9,13)}_{close}, \underbrace{(4,10)}_{close}]$	4	2.62	7.00×10^{-4}	0.00%
3	$[\underbrace{(0,3)}_{open}, \underbrace{(1,7)}_{open}, \underbrace{(9,13)}_{close}, \underbrace{(6,15)}_{close}]$	4	4.0	8.38×10^{-4}	0.00%
4	$[\underbrace{(0,3)}_{open}, \underbrace{(1,7)}_{open}, \underbrace{(6,15)}_{close}, \underbrace{(4,10)}_{close}]$	4	4.0	11.01×10^{-4}	0.00%

TABLE 4.2: Pareto-optimal solutions when minimizing $N_m(\cdot)$ and $S_{NR}(\cdot)$. For each solution in the second to the fourth rows, there are four more with different orders of switches but with the same $T_m(\cdot)$ when computed with the proposed heuristic.

first closing maneuver $(6, 15)$ recovers less load in the first stage when compared to the other solutions, and this is reflected in a greater (worse) energy not supplied. Finally, add to this discussion the fact that some algorithms may consider all of the last three sequences equal in terms of performance, and thus the best one may only be presented to the decision maker by chance.

In summary, by assuming that $N_m(\cdot)$ is a good representation of the "in the shortest time possible" portion of a good restoration plan, we (i) ignore the time savings obtained by considering multiple dispatch teams, and (ii) we may incorrectly return solutions thinking they are efficient when in reality they are not. And even if this conclusion were derived in a simple and small network, our work in [1] extended the analysis to a larger and real system, obtaining essentially the same conclusions.

4.5 Summary

Sets of maneuvers indicate which switches must be operated to achieve a new configuration. Sequences of maneuvers provide the incremental steps to do this. In short, sequences should guarantee that the resulting network remains feasible after each operation, and that the order of the maneuvers aims at taking the shortest time possible to be implemented [a job for $T_m(\cdot)$], or that the impacts on the reliability index SAIDI is minimum [a task for $E_{NS}(\cdot)$]. To achieve both goals, this chapter formalized the concept of *rules of precedence* – which state that some switches should be first operated before others in order to maintain feasibility – and also presented a constructive heuristic for estimating the time of maneuvers and energy not supplied respecting these rules and taking into account the availability of more than one dispatch team.

The main message of the present discussion is that, despite the evolution in modern methods for actually returning a feasible sequence of operations, it is not enough to handle sets during the optimization method and only provide a post-processing step for the best new configuration. The problem is that this practice does not allow one to perform the optimization in terms of the more realistic indices $T_m(\cdot)$ and $E_{NS}(\cdot)$. And why have I been calling these objectives "more realistic" since the beginning of this text? As showed here (and detailed in [1]), adopting $N_m(\cdot)$ to estimate the actual time taken to execute a restoration plan can be deceptive: it ignores the compensation in time obtained by the availability of multiple dispatch teams, and may unintentionally output bad solutions in practice.

Given this discussion, the first impression is a slight advantage in methods that create solutions constructively – such as heuristics and some metaheuristics based on ACO and permutations – over techniques that deal directly with sets – such as mathematical programming approaches and metaheuristics with binary and forest-based codifications. But that is not a correct conclusion. The proposed algorithm should be used as a decoding mechanism (Figure 4.8) for enabling any kind of technique not only to output a proper sequence of operations but also to use (again) more realistic indices of quality. Thus, I hope that this first contribution may help with the development of better-performing algorithms in regards to these practical topics, and finally let the human experts receive their deserved rest after being implemented in real distribution utilities.

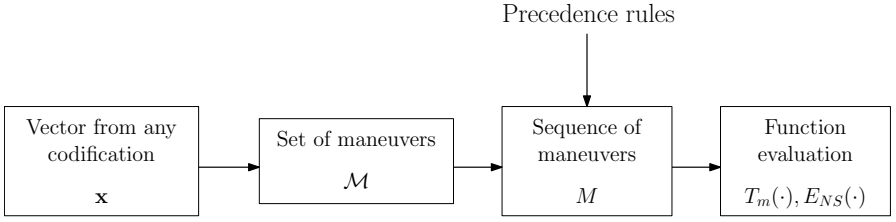


FIGURE 4.8: The proposed heuristic estimating $T_m(\cdot)$ and $E_{NS}(\cdot)$ should be viewed as a decoding mechanism.

Chapter 5

Proposed method: Permutation-based Algorithm for solving the Load Restoration Problem

Thanks to the mechanism proposed in the previous chapter, we are now capable of mapping any set of maneuvers $\mathcal{M}$ into a unique sequence M estimating time and energy not supplied. And since the mapping from a given codification $\mathbf{x} \in \mathbb{X}$ to a set $\mathcal{M}$ tends to be evident in most cases, we can essentially work in space $\mathbb{X}$ and at the same time optimize indices that require the space of sequences. Thus, our optimization technique can be both efficient – in terms of enjoying specific properties of an algorithm or the availability of neighborhood structures – *and* realistic – in terms of adopting $T_m(\cdot)$ and $E_{NS}(\cdot)$ as quality indices.

This chapter presents an optimization method following the same permutation-based structure of Carrano *et al.* [25], which already possesses several practical advantages as detailed in Table 3.1. The difference is that here we attempt to improve this approach with the following contributions:

- An efficient perturbation scheme that results in a reduction of neighborhood sizes, and can enhance the performance of an optimization algorithm;
- A more realistic estimation of the time taken to perform the maneuvers, which takes into consideration the existence of more than one dispatch team and improves the accuracy of performance indices that depend on time;

- A preliminary step of the problem considering only remote controlled switches.

The second contribution was already presented in the previous chapter, so here we focus on the other two points.

5.1 Modeling the Load Restoration Problem

In section 2.2.3 a number of quality indices were presented as possibilities for mathematically modeling what a "good" restoration plan may be. Given the importance of the energy not supplied in terms of economic impact, prioritizing important loads to be recovered first, and in also incorporating the total time in its formulation, this index is employed as part of the quality evaluation of solutions. However, remember that (Figure 5.1) a general load restoration plan M may induce a "quantifiable" portion of the energy not supplied $E_{NS}(M)$ and a power not restored $S_{NR}(M)$ that lasts until the fault is corrected. To account for this residual portion, we propose to minimize both quantities, so the load restoration problem is modeled as:

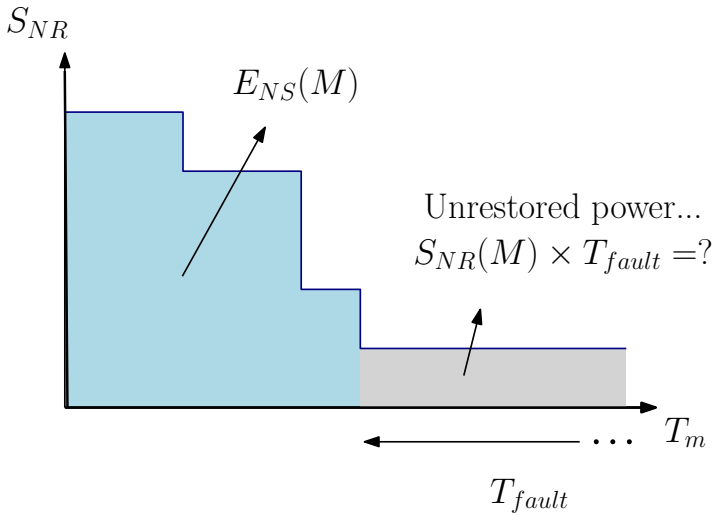


FIGURE 5.1: Energy not supplied and power not restored. When only partial restoration is possible, there is a residual portion that lasts until the fault is cleared.

$$\begin{aligned} \text{minimize} \quad & [S_{NR}(M) \;\; E_{NS}(M)]; \\ \text{subject to} \quad & M \in \text{feas}(\mathfrak{S}_{2^\varepsilon}) \subseteq \mathfrak{S}_{2^\varepsilon} \end{aligned} \tag{5.1}$$

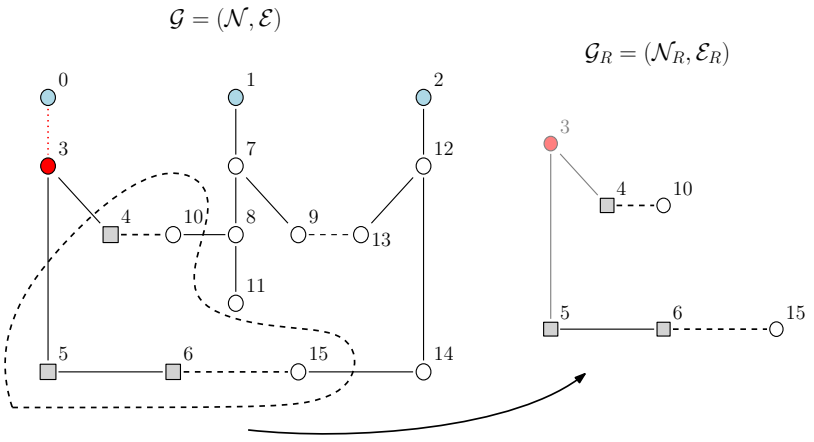
in which all overload and radiality constraints are enforced in $\text{feas}(\mathfrak{S}_{2^\varepsilon})$ as used in equation (2.20). Notice that this choice alleviates the need of estimating a time T_{fault}

to clear the fault or the adjustment of penalty factors, both of which can be very hard to properly set.

This multi-objective formulation has as main advantage over a single-objective minimization of, say, $E_{NS}(\cdot)$ with a penalty or some estimation of SAIDI, the fact that it handles trade-offs with regards to load restored and time required. Thanks to the dynamic nature of distribution systems, it is not possible to predict when quicker restoration plans (which may recover less customers) would be better then a slower one (but recovering more). Thus, while a single-objective approach returns only one solution with a fixed trade-off, the adopted multi-objective formulation is able to present such compromises to the decision maker so he/she can select the appropriate solution in each occasion.

5.2 Proposed structure and evaluation mechanism

Following the same concept in most modern studies, instead of directly handling all possible sequences of maneuvers, we abstract the distribution system into a codification $\mathbb{X}$. In this work $\mathbb{X}$ is modeled as $\mathfrak{S}_{\mathcal{E}_R}$, which is the set of possible permutations of all elements $e \in \mathcal{E}_R$ of the restoration sub-graph $\mathcal{G}_R$. $\mathcal{E}_R$ denotes all available switches as illustrated again in Figure 5.2. In general, if $\mathcal{E}_R = \{e_1, e_2, e_3, \ldots, e_n\}$, then $\mathbf{x}$ will have the form:



$$\mathcal{E} = \{(0,3),(1,7),(2,12),(3,4),(3,5),(4,10),(5,6),(6,15),(7,8),(7,9),(8,10),(8,11),(9,13),(12,13),(14,15)\}$$
$$\mathcal{E}^{CC} = \{(0,3),(1,7),(2,12),(3,4),(3,5),(5,6),(7,8),(7,9),(8,10),(8,11),(12,13),(14,15)\} \subset \mathcal{E}$$

$$\mathcal{E}_R = \{(4,10),(5,6),(6,15)\}$$
$$\mathcal{E}_R^{CC} = \{(5,6)\} \subset \mathcal{E}_R$$

FIGURE 5.2: A network $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ together with its restoration sub-graph, $\mathcal{G}_R = (\mathcal{N}_R, \mathcal{E}_R)$. In this work we handle only the switches belonging to $\mathcal{E}_R$.

$$\mathbf{x} = [e_{\pi(1)}, e_{\pi(2)}, \ldots, e_{\pi(n-1)}, e_{\pi(n)}], \tag{5.2}$$

in which [21] $\pi : \{1, 2, \ldots, n\} \mapsto \{1, 2, \ldots, n\}$ is a *permutation function*, which associates, for every integer $i \in \{1, 2, \ldots, n\}$, another (unique) integer $j \in \{1, 2, \ldots, n\}$ such that $\pi(i) = j$.

This structure is a simplification of the one proposed by Carrano *et al.* [25], ignoring isolating switches and load transfer from higher-order neighboring feeders.[1] The former ones can be artificially opened whenever a node neighboring a fault is energized, as shown in the proposed evaluation mechanism of Figure 5.3 presented next. The latter is a direct consequence of our choice of handling **Case 1** load restoration, defined previously in section 4.3.1. In short, it was our choice to follow the policy of "not disturbing healthy loads" when performing restoration, a common practice in Brazilian energy utilities.

Recall that a permutation vector $\mathbf{x} \in \mathbb{X}$ as it is tells us basically nothing about which switches to open or close, so we require an evaluation process to convert it into a proper set or sequence. For that, the procedure depicted in Figure 5.3 is proposed. The method is almost a literal implementation of the general restoration plan outlined in section 4.2.2 without considering the edges of types 2 and 5. It starts a stage trying to close the first CO switch e^{CO} that appears in $\mathbf{x}$ and is able to recover *oos* nodes (hence a Type 3). If the resulting network is unfeasible (i.e., if it violates voltage, current, or feeder capacity constraints), CC edges that are downstream to e^{CO} are opened in sequence in the same order as they appear in $\mathbf{x}$ until a feasible state is reached. If there is no available switch to open, the stage is discarded. If a feasible stage $\mathcal{S}_i$ is successfully assembled, the redundant maneuvers are discarded and the energized faults are isolated by opening isolation switches, if needed. The whole process is repeated until there is no CO edge available, at which point the constructed set $\mathcal{M}$ can be converted into a proper sequence M with its $S_{NR}(\cdot)$ and $E_{NS}(\cdot)$ estimations.

Figure 5.4 shows a detailed example of this evaluation process for our usual sample network with faults at nodes 1 and 14. Consider a permutation vector and its split into CO and CC partitions as given in the figure. The first CO switch that appears in $\mathbf{x}$ and is able to recover load is $(9, 13)$, so it is closed in the first iteration. Assume the resulting network violates a constraint. In that case, we should perform load shedding by searching for the first CC edge in $\mathbf{x}$ that is downstream to $(9, 13)$, which is $(8, 10)$. If its opening alone is not enough, open the next edge satisfying the same conditions, which is $(8, 11)$. It was still insufficient, so we settle by finally opening $(7, 8)$ and obtaining a feasible configuration. Notice that this final maneuver turned the previous operations useless, so we may as well discard them, as shown in the figure. Finally, because node 7, which is a neighbor of the faulted node 1, was energized, the isolation switch (not

[1] Which, as a matter of fact, are essentially neglected in their work as well as proposal for a more efficient search.

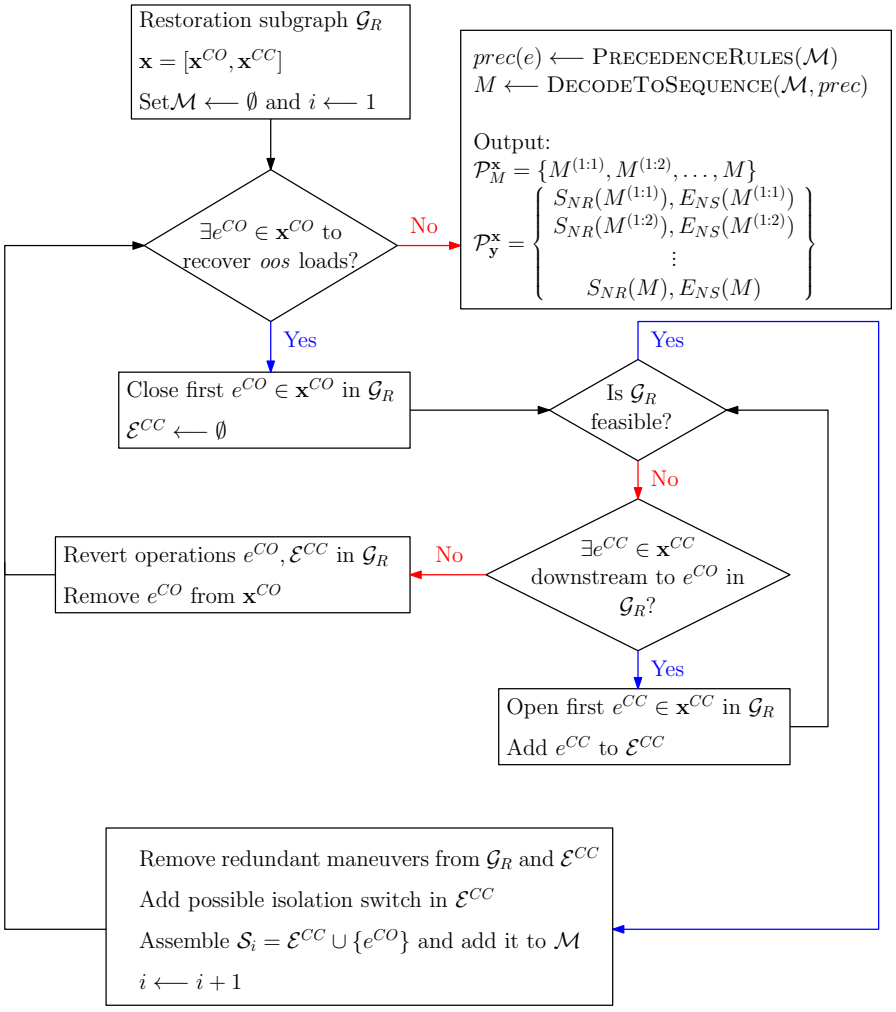


FIGURE 5.3: Proposed evaluation strategy to generate a sequence of maneuvers M with $S_{NR}(\cdot)$ and $E_{NS}(\cdot)$ estimations from a given permutation $\mathbf{x}$. Recall that each sub-sequence of a sequence is also a solution, so a permutation vector actually outputs multiple restoration plans.

shown) $(1, 7)$ should also be opened. This completes the first stage. The remaining stages are shown in Figure 5.4. After the process, we use the method proposed in the previous chapter to obtain the precedences and convert the returned set of maneuvers into a proper sequence with its associated estimations of the time, energy not supplied and power not restored.

There are a few characteristics of this method that must be highlighted. First, since we deal with **Case 1** restoration, only switches of Types 1, 3 and 4 are considered, but thanks to the constructive nature of the technique, Type 4 edges are only used when they become a Type 3. Thus, for simplicity of notation, we will refer to CC and CO switches only, and it should be implied that a CO edge is one that is able to recover *oos* loads in a given stage. This is reflected in Figure 5.3 where a permutation vector is split into its CC and CO partitions before the method begins.

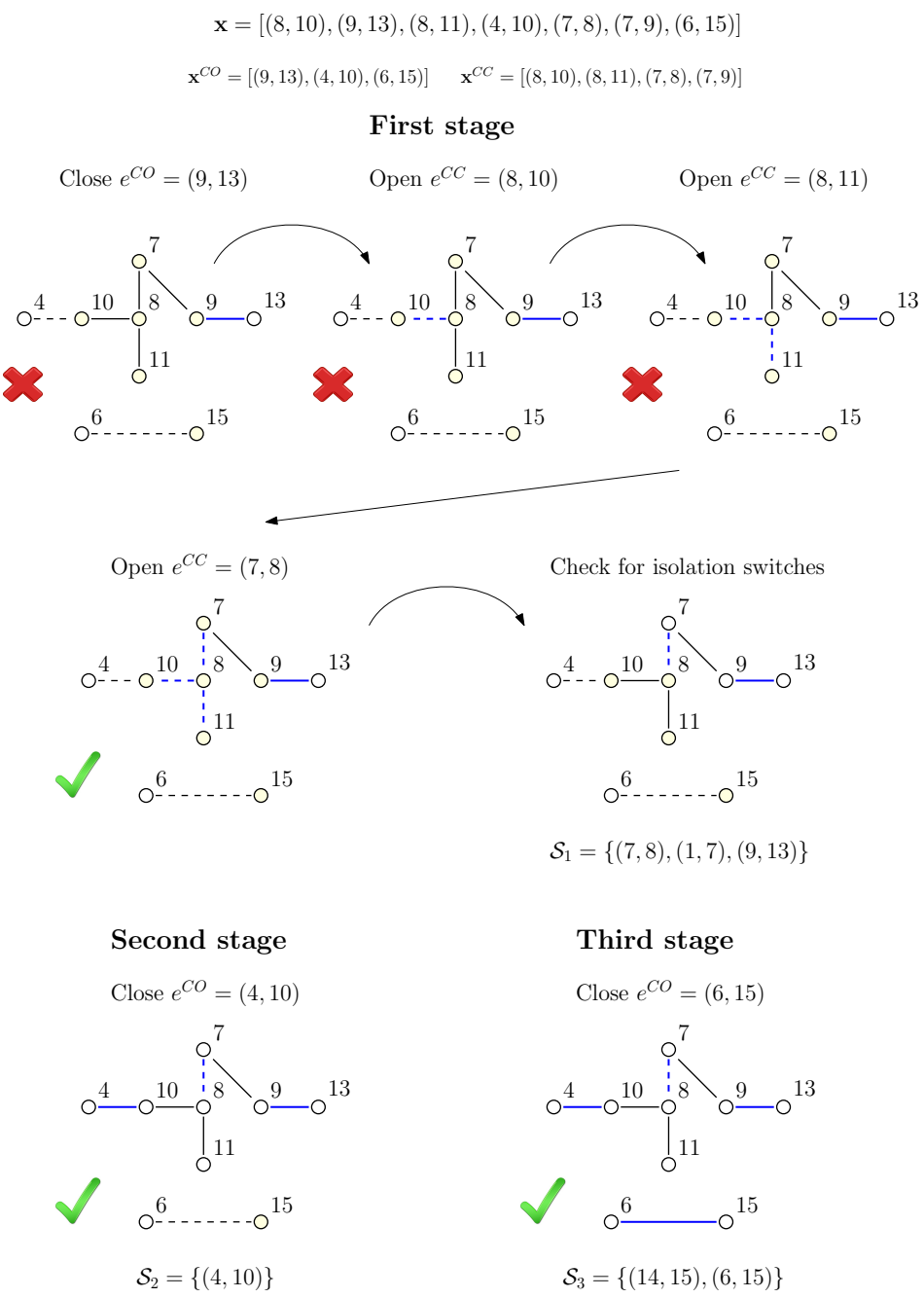


FIGURE 5.4: Example of the evaluation procedure assuming faults at nodes 1 and 14 (not shown for space constraints).

Next, it should be evident from the figure that a stage is only assembled if, in order to close a CO switch e^{CO}, there is a subset $\mathcal{E}^{CC}$ of CC switches such that, when opened, produce a feasible configuration. Moreover, a switch e^{CO} is only chosen if it is connected to an *oos* node, which ensures radiality. Therefore, we have the guarantee that the subset of maneuvers $\mathcal{M} \subset \mathcal{E}_R$ returned by the algorithm contains enough switches to create feasible sequences and thus respect constraints (2.12)-(2.19).

Lastly, remember that each sub-sequence $M^{(1:i)}$ composed of the first i maneuvers of a

sequence M is also a separate solution, so this evaluation mechanism actually outputs a total of $|M|$ different solutions for each permutation vector $\mathbf{x}$. For coherence, the induced sequences are returned in a set $\mathcal{P}_M^{\mathbf{x}}$, together with $\mathcal{P}_{\mathbf{y}}^{\mathbf{x}}$ containing their respective objective values. We will see next how to easily deal with this multi-solution aspect.

5.3 Proposed algorithm

The multi-solution scheme of the proposed decoding method only makes sense given the multi-objective character of the problem, in which the expected output of an algorithm is a set of non-dominated candidate solutions. For this work, archives $\mathcal{P}_{\mathbf{y}}^A$ and $\mathcal{P}_M^A$, with sizes limited to μ_A solutions, are employed to store the best sequences obtained so far. Each time a new vector $\mathbf{x}_p$ is encountered, the update procedure given in Algorithm 1 is executed. The process is simple: after decoding $\mathbf{x}_p$ into its sub-sequences, all solutions are included in the current archive. Then, remove all dominated solutions and, if the resulting size exceeds μ_A, truncate further using a crowding distance approach [64].

Data: A new solution $\mathbf{x}_p$, Archives $\mathcal{P}_{\mathbf{y}}^A$ and $\mathcal{P}_M^A$ limited to μ_A solutions;
Result: Updated archives $\mathcal{P}_{\mathbf{y}}^A$ and $\mathcal{P}_M^A$;
1 $\mathcal{P}_{\mathbf{y}}^{\mathbf{x}_p}, \mathcal{P}_M^{\mathbf{x}_p} \longleftarrow$ EvaluatePermutation($\mathbf{x}_p$) ; // Figure 5.3
2 $\mathcal{P}_{\mathbf{y}}^A \longleftarrow \mathcal{P}_{\mathbf{y}}^A \bigcup \mathcal{P}_{\mathbf{y}}^{\mathbf{x}_p}$;
3 $\mathcal{P}_M^A \longleftarrow \mathcal{P}_M^A \bigcup \mathcal{P}_M^{\mathbf{x}_p}$;
4 Truncate $\mathcal{P}_{\mathbf{y}}^A, \mathcal{P}_M^A$ with nondominated sorting and crowding distance if their size exceeds μ_A;

Algorithm 1: Pseudo-code for the update procedure of the archive of solutions.

With this, the optimization algorithm will follow the steps: (i) start with a given permutation and archive of solutions; (ii) search for different permutations in a systematic way; and (iii) update the archive. This process is repeated until a specific stopping criterion is satisfied. Notice that it is perfectly possible for the final archive to contain sub-sequences of different permutations. In this work the optimization approach consists of a Simulated Annealing (SA) followed by a Local Search (LS) refinement, which will be explained in the following.

5.3.1 Simulated Annealing Algorithm for the Load Restoration

The adopted Simulated Annealing is presented in Algorithm 2, which is very similar to the basic version of [65] but with some adaptations for the load restoration problem that will be detailed step by step.

Data: Initial permutation $\mathbf{x}_0$ with n maneuverable switches; Maximum archive size μ_A;
Result: Archives $\mathcal{P}_{\mathbf{y}}^A$ and $\mathcal{P}_M^A$ limited to μ_A solutions;

$\mathcal{P}_{\mathbf{y}}^A \longleftarrow \emptyset$; // Initialize archives
$\mathcal{P}_M^A \longleftarrow \emptyset$;
$\mathcal{P}_{\mathbf{y}}^A, \mathcal{P}_M^A \longleftarrow$ UpdateArchive$(\mathbf{x}_0, \mathcal{P}_{\mathbf{y}}^A, \mathcal{P}_M^A)$; // Algorithm 1
$T \longleftarrow$ InitialTemperature$(\mathbf{x}_0, \mathcal{P}_{\mathbf{y}}^A)$; // Algorithm 4
$\mathbf{x} \longleftarrow \mathbf{x}_0$;
$n_{eff} \longleftarrow$ NumberOfEffectiveManeuvers$(\mathbf{x})$; // Openings and closings in a full sequence
$t_{min} \longleftarrow n_{eff}$;
$t_{max} \longleftarrow n$;
$t_{in,nonimp} \longleftarrow 0$; // Inner loops without improvement
while $t_{in,nonimpr} < 3$ **do**
  $t_{accep} \longleftarrow 0$; // Number of accepted moves
  for $t \longleftarrow 0 : t_{max}$ **do**
    $\mathbf{x}_p \longleftarrow$ Perturbation$(\mathbf{x})$; // Section 5.4.3
    $\mathcal{P}_{\mathbf{y}}^A, \mathcal{P}_M^A \longleftarrow$ UpdateArchive$(\mathbf{x}_p, \mathcal{P}_{\mathbf{y}}^A, \mathcal{P}_M^A)$; // Algorithm 1
    $flag \longleftarrow$ CheckAcceptance$(\mathbf{x}_p, \mathcal{P}_{\mathbf{y}}^A, \mathcal{P}_M^A)$; // Algorithm 3
    if $flag$ **then**
      $\mathbf{x} \longleftarrow \mathbf{x}_p$;
      $t_{in,nonimp} \longleftarrow 0$; // Only stop for consecutive non-improvements
      $t_{accep} \longleftarrow t_{accep} + 1$;
    end
    if $t_{accep} \geq t_{min}$ **then**
      break ; // End inner loop earlier
    end
  end
  $T \longleftarrow \alpha T$; // Update the temperature
  $n_{eff} \longleftarrow$ NumberOfEffectiveManeuvers$(\mathbf{x})$;
  $t_{min} \longleftarrow n_{eff}$;
  if $t_{accep} = 0$ **then**
    $t_{in,nonimp} \longleftarrow t_{in,nonimp} + 1$;
  end
end

Algorithm 2: Pseudo-code for the proposed simulated annealing.

5.3.1.1 Perturbation of solutions

Thanks to the permutation-based character of the search space, any of the perturbation schemes proposed in [66] and illustrated later in Figure 5.5 can be adopted to generate a new permutation vector $\mathbf{x}_p$ from a current $\mathbf{x}$. However, this work proposes a modification of one of these schemes, and this discussion will be postponed to section 5.4.3.

5.3.1.2 Acceptance criterion

Originally, a new permutation $\mathbf{x}_p$ is accepted (i) deterministically, if it is better than the current one; or (ii) probabilistically (using the Metropolis criterion [65]), if it is worse. The first part of the condition can be adapted to the multi-objective case by trying to include $\mathbf{x}_p$ in the current archives: for a successful inclusion (if any of the sub-sequences induced by this permutation remains in the archive after the update) we set $\mathbf{x}_p$ as the current point. If the inclusion is a failure, we follow the proposition of [67] and convert a vector of objectives $\mathbf{y}$ into a sum of logarithms of each coordinate. This is valid only if the objective functions have a positive image. Fortunately, in our formulation, we known that $Im\{S_{NR}\} = [0, \infty)$ and $Im\{E_{NS}\} = [0, \infty)$. Thus, we suggest to convert each vector $\mathbf{y}$ into the scalar version

$$f_{log}(\mathbf{y}) \triangleq \sum_{i=1}^{m} \ln(y_i + 1) \tag{5.3}$$

in which the $+1$ prevents taking logarithms of zero values.

For a set $\mathcal{P}_{\mathbf{y}}^{A}$, we define this scalarization as the smallest one over all of its elements:

$$f_{log}(\mathcal{P}_{\mathbf{y}}^{A}) \triangleq \min_{\mathbf{y} \in \mathcal{P}_{\mathbf{y}}^{A}} f_{log}(\mathbf{y}) \tag{5.4}$$

and this is also valid after decoding a permutation $\mathbf{x}_p$ into a set $\mathcal{P}_{\mathbf{y}}^{\mathbf{x}_p}$ with a number of sub-sequences as output from the algorithm of Figure 5.3.

The issue with this scalarization is that it can be easily biased by different scales in the objectives. Thus, before computing it, it is suggested to normalize each $\mathbf{y}$ by the maximum and minimum of the solutions of the current archive. The complete acceptance criterion is shown in Algorithm 3.

5.3.1.3 Temperature update

This work employs the usual geometrical law of decrease, $T(t+1) = \alpha T(t)$, with $\alpha < 1$. Because of the reduction mechanisms proposed later and a posterior local refinement procedure, we opted for a fast cooling schedule and adopted $\alpha = 0.9$ without considerable performance losses.

Data: Perturbed permutation $\mathbf{x}_p$; Archives $\mathcal{P}_{\mathbf{y}}^A$ and $\mathcal{P}_M^A$;
Result: Binary acceptance flag;
$\mathcal{P}_{\mathbf{y}}^A, \mathcal{P}_M^A \longleftarrow$ UpdateArchive$(\mathbf{x}_p, \mathcal{P}_{\mathbf{y}}^A, \mathcal{P}_M^A)$; `// Algorithm 1`
$flag \longleftarrow$ False;
if *At least one sub-sequence of* $\mathbf{x}_p$ *was included in the archives* **then**
    $flag \longleftarrow$ True;
else
    $r \sim U(0,1)$; `// Uniform random number between 0 and 1`
    if $r < exp\{f_{log}(\mathcal{P}_{\mathbf{y}}^A) - f_{log}(\mathcal{P}_{\mathbf{y}}^{\mathbf{x}_p})\}$ **then**
        $flag \longleftarrow$ True;
    end
end

Algorithm 3: Adapted acceptance criterion for the proposed Simulated Annealing. Remember that, when computing $f_{log}(\cdot)$ for the sets, the elements should be scaled by the maxima and minima of the current archive to prevent biasing the algorithm.

5.3.1.4 Equilibrium condition

The inner loop of the Simulated Annealing is normally executed with the same temperature until "equilibrium is achieved", which is usually translated as [65] stopping after (i) t_{min} perturbations were accepted (either for improvement or acceptable deterioration); or (ii) t_{max} perturbations were attempted. In this work, suppose $\mathbf{x}$ was the current permutation at the beginning of the inner loop, and it induced a full sequence of maneuvers with n_{eff} effectively operated switches (closed or opened). If n is the total number of available switches, we propose to set $t_{min} = n_{eff}$ and $t_{max} = n$.

5.3.1.5 Stopping condition

The stopping condition here is set as the occurrence of 3 successive runs of the inner loop without a single accepted move [65]. Even if this stops the SA too early, it should not be problematic thanks to the local refinement presented later.

5.3.1.6 Initial value of the temperature

The initial value for parameter T must be [65] high enough so that most perturbations are accepted in the beginning, but not too high to place the algorithm in this "global exploration" for an unnecessarily long time. To accomplish that, Dreo *et al.* [65] suggest to sample a number of neighbors and adjust T_0 according to the average difference in objective value of the worse solutions. A similar procedure is adopted here as described in Algorithm 4.

Data: Initial solution $\mathbf{x}_0$; Archives $\mathcal{P}_{\mathbf{y}}^A$ and $\mathcal{P}_M^A$; Rate of acceptance τ_0;
Result: A suitable initial temperature T_0;

1 $n_{eff} \longleftarrow$ NumberOfEffectiveManeuvers($\mathbf{x}_0$) ; `// Openings and closings in a full sequence`
2 **for** $i = 1 : n_{eff}$ **do**
3 $\quad \mathbf{x}_p \longleftarrow$ Perturbation($\mathbf{x}_0$);
4 $\quad \mathcal{P}_{\mathbf{y}}^A, \mathcal{P}_M^A \longleftarrow$ UpdateArchive($\mathbf{x}_p, \mathcal{P}_{\mathbf{y}}^A, \mathcal{P}_M^A$) ; `// Algorithm 1`
5 **end**
6 $f_{log,min}, \mathbf{y}_{min} \longleftarrow \min_{\mathbf{y} \in \mathcal{P}_{\mathbf{y}}^A} \{f_{log}(\mathbf{y})\}$; `// Best scalarized value in the archive`
7 $\overline{\Delta f} \longleftarrow \operatorname{mean}_{\mathbf{y} \in \mathcal{P}_{\mathbf{y}}^A \setminus \{\mathbf{y}_{min}\}} \{f_{log}(\mathbf{y}) - f_{log,min}\}$; `// Mean value of degradations`
8 $T_0 \longleftarrow -\frac{\overline{\Delta f}}{\ln \tau_0}$;

Algorithm 4: Preliminary step for generating the initial temperature T_0.

In the algorithm, we sample n_{eff} permutations corresponding to the number of maneuvers in a full sequence induced by $\mathbf{x}_0$ and update the archive with their sub-sequences. After that, we use these solutions to determine an initial temperature.[2] The parameter τ_0 is dependent on the "quality" of the initial solution: using $\tau_0 = 0.2$ for good initial guesses and $\tau_0 = 0.5$ for bad ones is recommended in [65]. To allow for a better initial exploration in the search space, we suggest using $\tau_0 = 0.5$ regardless of the quality of $\mathbf{x}_0$.

5.3.2 Local Search Refinement

Due to the greedy characteristic of the SA with lower values for the temperature parameter, some studies suggest that the algorithm loses effectiveness once this state is reached [65]. A common technique for handling this problem is to stop the simulated annealing run, even if prematurely, and switch to a local search refinement. This strategy is adopted in this work, and the proposed local search is described in Algorithm 5, where $\mathcal{V}(\mathbf{x})$ denotes the neighborhood of permutation $\mathbf{x}$, as described at the end of Section 5.4. Note that we opted for storing the corresponding permutation $\mathbf{x}$ that generated a given sub-sequence in the archive (line 2 of the algorithm). This is a design choice that is justified by the possibility of reducing the number of local searches, since more than one solution in the archive can come from the same permutation.

[2] Notice that, unless the problem has no solution, there are always at least two points including the "do nothing" solution, so this algorithm is always valid.

Data: Archives $\mathcal{P}^A_{\mathbf{y}}$ and $\mathcal{P}^A_M$ limited to μ_A solutions;
Result: Archives $\mathcal{P}^A_{\mathbf{y}}$ and $\mathcal{P}^A_M$ with possibly improved solutions;

```
for M ∈ P^A_M do
    x ⟵ GetCorrespondingPermutation(M);
    improvement ⟵ True;
    while improvement == True do
        improvement ⟵ False;
        for x_p ∈ V(x) do
            P^A_y, P^A_M ⟵ UpdateArchive(x_p, P^A_y, P^A_M) ;        // Algorithm 1
            if At least one sub-sequence of x_p was included in the archive then
                x ⟵ x_p;
                improvement ⟵ True;
                break;
            end
        end
    end
end
```

Algorithm 5: Pseudo-code for the local search refinement.

5.4 Perturbation mechanisms

One of the main benefits of working with a permutation space $\mathbb{X}$ together with the evaluation method of Figure 5.3 is that, since all induced solutions are feasible, the optimization algorithm can perform an unrestricted search over $\mathbb{X}$. However, a drawback is that, in some cases, vectors that are distinct in the permutation space can map to the same sequences of maneuvers. This is mostly because (i) not all elements of $\mathbf{x}$ are used in its evaluation, i.e., $\mathbf{x}$ can contain *neutral components*; and (ii) the absolute order in which elements appear in the vector is less important than the relative position within their corresponding CC or CO partitions. The consequence of this characteristic is that the space of objectives may contain "flat" regions as many different permutations in the search space map to equivalent solutions in the objective one, which can hinder the performance of optimization algorithms.[3] To avoid convergence problems that may arise from this, we propose specific perturbation mechanisms to create neighboring solutions in a more efficient manner, preventing (or at least reducing) the occurrence of neutral moves during local search.

Given the structure of $\mathbb{X}$, it is possible to employ any perturbation scheme appropriate for permutations. In this work, schemes PS1 – PS6 presented by Tian *et al.* [66] and

[3] This is another reason for choosing the SA as main algorithm in this work, besides my relative familiarity with the algorithm. Accepting new solutions even if equal or worse than the current one may help leaving these flat regions and may make it a better choice *in this situation* when compared to other local search based metaheuristics such as Variable Neighborhood Search, Iterated Local Search, Guided Local Search etc.

illustrated in Figure 5.5 are considered. As neighborhood sizes can be very large, *reduction schemes* can be used to make the search more efficient, albeit sometimes at the cost of losing possibly useful perturbations. However, as will be shown later, this loss does not seem to be significant compared to the efficiency gains.

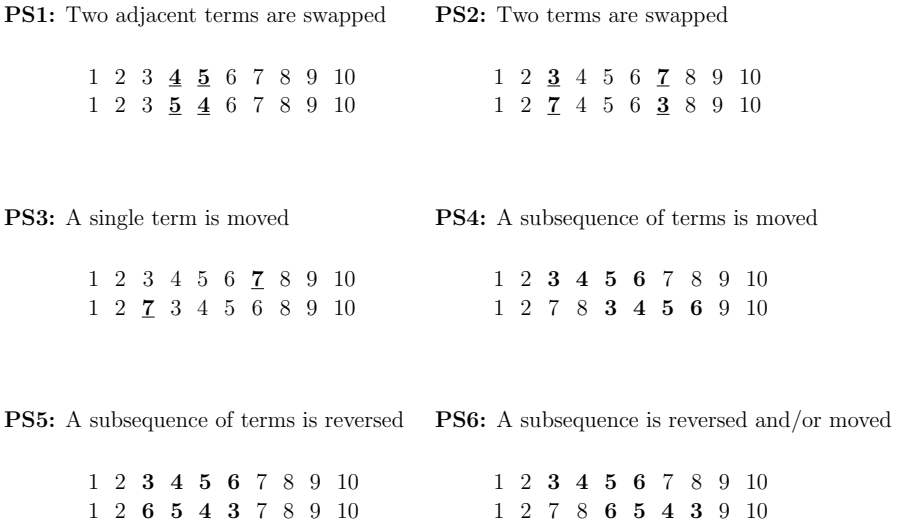


FIGURE 5.5: Perturbation schemes that can be used to modify permutation-based solutions.

A specific characteristic of the restoration problem as defined in this work is that the currently opened (CO) and currently closed (CC) components are independent – if elements in the whole vector are permuted but the elements of each partition retain their relative positions, then the modified candidate will most likely map to the same sequence of maneuvers as the original one. Therefore, we can justify our partition $\mathbf{x} = [\mathbf{x}^{CO}\ \mathbf{x}^{CC}]$ and perform perturbations independently, as discussed below.

5.4.1 Perturbation of CC switches

This is generally the partition containing most elements and, as it will be evidenced in the following discussion, there can be many permutations that lead to equivalent solutions. To overcome this problem, two *reduction schemes* are presented in this section – one already used in [25], and the other as a new proposal. Scheme PS2, the simple swap of any two elements in $\mathbf{x}^{CC}$, is used as the basic perturbation mechanism in this work.

5.4.1.1 Scheme 1: Prevent swaps between different groups

Depending on the severity of the fault scenario, it is possible that more than one feeder becomes out of service, which leads to a restoration sub-graph with disconnected portions, called *groups*, as illustrated in Figure 5.6.

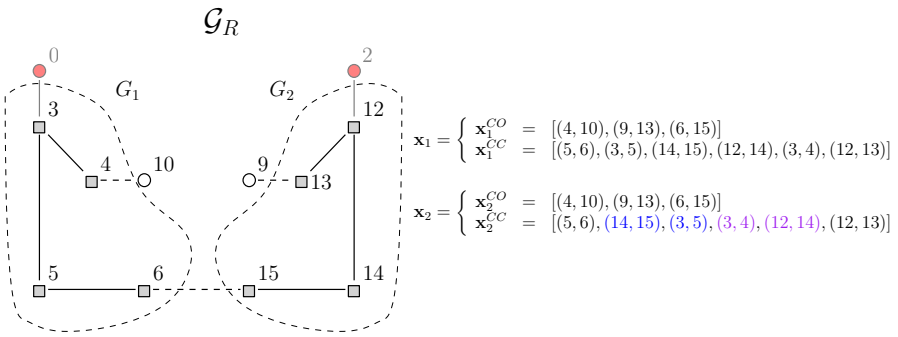


FIGURE 5.6: Scheme 1 - Swap CC components within each group and not between groups.

Consider two vectors $\mathbf{x}_1$ and $\mathbf{x}_2$ of possible permutations, illustrated in Figure 5.6, which have the same CO components but some swapped CC edges. I leave as a quick exercise to the reader to see that, if the evaluation mechanism of Figure 5.3 is followed, both permutation vectors will generate the same set (and consequently, same sequence) of maneuvers.

If the edges in each permutation are classified according to their groups, it is easy to notice that they come in exactly the same order, thus explaining the equivalence between both vectors. Based on this property, the first reduction mechanism consists in swapping only elements within the same group [25].

Notice that, in a few cases, some permutations cannot be achieved with a single move. For instance, in $\mathbf{x}_1^{CC}$, it is not possible to get the elements of the second group in the order $[(12,13),(14,15),(12,14)]$ with a single swap, while this is possible by allowing interchanges among any two elements [swap $(12,13)$ with either $(3,5)$ or $(5,6)$]. Such losses are to be expected, but they are usually not significant in practical cases.

5.4.1.2 Scheme 2: Prevent swaps of unused switches

Scheme 1 was already used in [25], but even preventing swaps among elements of different groups still allows perturbations leading to equivalent solutions. To understand the proposed reduction scheme, consider a permutation $\mathbf{x}$ with 10 CC switches, in which only three of them were effectively opened after the evaluation procedure, as shown in Figure 5.7.

It is easy to see that, because the last three switches were never considered in the evaluation, any permutation among these elements would change absolutely nothing in the resulting set or sequence, provided the CO components are unchanged. A similar result can be expected if only the first two switches, which ended up being redundant, are swapped. Therefore, perturbations that involve switches from e_3^{CC} to e_7^{CC}, which

include the effectively opened edges and a few others in between, have higher chances of being useful.

$$\mathcal{M}^{CC} = \{e_3^{CC}, e_6^{CC}, e_7^{CC}\}$$

$$\mathbf{x}^{CC} = [\underbrace{e_1^{CC}, e_2^{CC}}_{\text{Redundants}}, e_3^{CC}, e_4^{CC}, e_5^{CC}, e_6^{CC}, e_7^{CC}, \underbrace{e_8^{CC}, e_9^{CC}, e_{10}^{CC}}_{\text{Never considered}}]$$

FIGURE 5.7: Scheme 2 - allow permutations involving only the effectively opened edges.

The proposal of this scheme is to go further and ignore all other elements, allowing exchanges only among the effectively opened edges, that is, only switches that appear in $\mathcal{M}$ (or M). With this, if a permutation $\mathbf{x}$ induces a sequence where n_{eff}^{CC} edges are to be opened, each one of them can be interchanged with any of the other n^{CC} terms, provided they belong to the same group, of course.

5.4.2 Perturbation of CO switches

Recall that only a single CO switch is closed at each stage, and even if two of them belong to the same feeder, they usually energize different *oos* nodes by distinct lines. As a consequence, all perturbations in this partition generally lead to distinct solutions, and essentially any of the schemes depicted in Figure 5.5 can be used. In general, there are fewer CO than CC switches, so the neighborhood size is already smaller. However, it is possible to employ a reduction scheme similar to *Scheme 2*: if a permutation $\mathbf{x}^{CO}$ with n^{CO} CO switches induces a sequence which only tries to close the first n_{eff}^{CO} edges, then no distinct solution results from modifying only the last $n^{CO} - n_{eff}^{CO}$ switches (provided the CC components are kept fixed). Given these considerations, we suggest that any perturbation in $\mathbf{x}^{CO}$ should only be considered if there are changes in the first n_{eff}^{CO} elements.

5.4.3 Complete neighborhood search

I promised back in section 5.3.1.1 to detail the perturbation process in the Simulated Annealing and Local Search refinement. For the former algorithm, it is possible to create a neighboring permutation $\mathbf{x}_p$ from $\mathbf{x}$ by, for instance, choosing a partition (CC or CO) and performing random changes using the proposed schemes.[4] On the other hand, the Local Search requires the ability to perform a neighborhood search from a given permutation $\mathbf{x}$. Once the CC and CO perturbation schemes have been chosen,

[4]Since the schemes were developed assuming the other partition is kept fixed, we suggest performing random perturbations in one partition at a time.

its neighborhood $\mathcal{V}(\mathbf{x})$ can be searched by the following combined operations: (i) create a neighbor $\mathbf{x}_p^{CO}$ in the CO partition, and (ii) with $\mathbf{x}_p^{CO}$ fixed, perform a complete neighborhood search for all CC components. Repeat this process for each CO neighbor.

5.5 Handling remote-controlled switches

It was established in the beginning of the thesis that this work was not developed with completely automated distribution systems in mind. Despite that, relatively modern systems do contain a number of switches that can be operated remotely, even if (normally) in much smaller number when compared to manual ones. Remote switches require very little time to be opened or closed. In some faulted scenarios, it is possible to first coordinate only these switches, so that some fraction of the *oos* loads can be recovered very quickly. This is exemplified in Figure 5.8, where remotely-operated switches are indicated by the letter **R**. For a fault at source 1, a sequence that opens (7,8) and closes (4,10) is able to recover some of the out of service nodes.

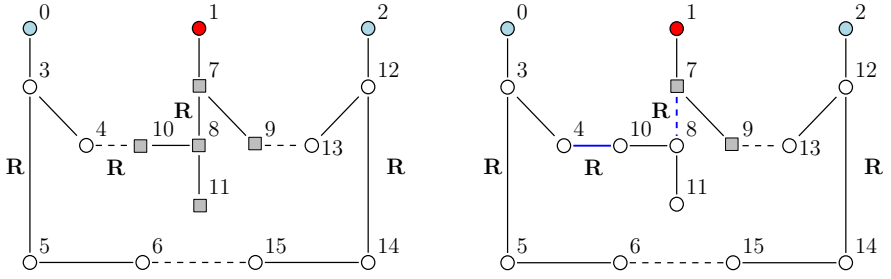


FIGURE 5.8: Handling remotely controlled switches (edges marked with an **R**). For a fault at source 1, a partial restoration can be performed by opening (7,8) and closing (4,10). Restoration plans involving only remotely controlled switches can be implemented very quickly, and loads recovered by these plans will have little to no impact on the reliability indices.

The motivation for a preliminary step of considering only remotely controlled switches is the existence of a time window that regulatory agencies provide such that, if a load is recovered within this interval, then it results in no impact to the reliability indices SAIDI and SAIFI.[5] These considerations lead to the complete procedure proposed in this chapter for obtaining restoration plans, shown in Figure 5.9. After an outage is detected and the faulted and out of service nodes (unrecoverable and recoverable, respectively) are identified, two instances of the algorithm are executed in parallel, one considering only remotely controlled switches, and the other all maneuverable ones. Since the search space of the first case tends to be much smaller, a solution becomes available very quickly and the dispatcher may choose to implement it or not. If affirmative, the remotely controlled

[5]Brazilian regulations define this time as three minutes for distribution utilities. Also recognize that only restorations within this time window are able to prevent increases in SAIFI.

switches are operated to implement the initial restoration plan, ideally within the no-penalties time window. The algorithm is then restarted, now with the updated network configuration as its starting condition and considering all maneuverable switches. In the end, the operator selects a final solution from the output of the algorithm. Keep in mind that solutions coming from the right track of Figure 5.9 do not become obsolete if a remote switches-only initial solution is implemented (left track), as the original state of the network can be recovered quickly by operating the same remote switches used for the initial solution.

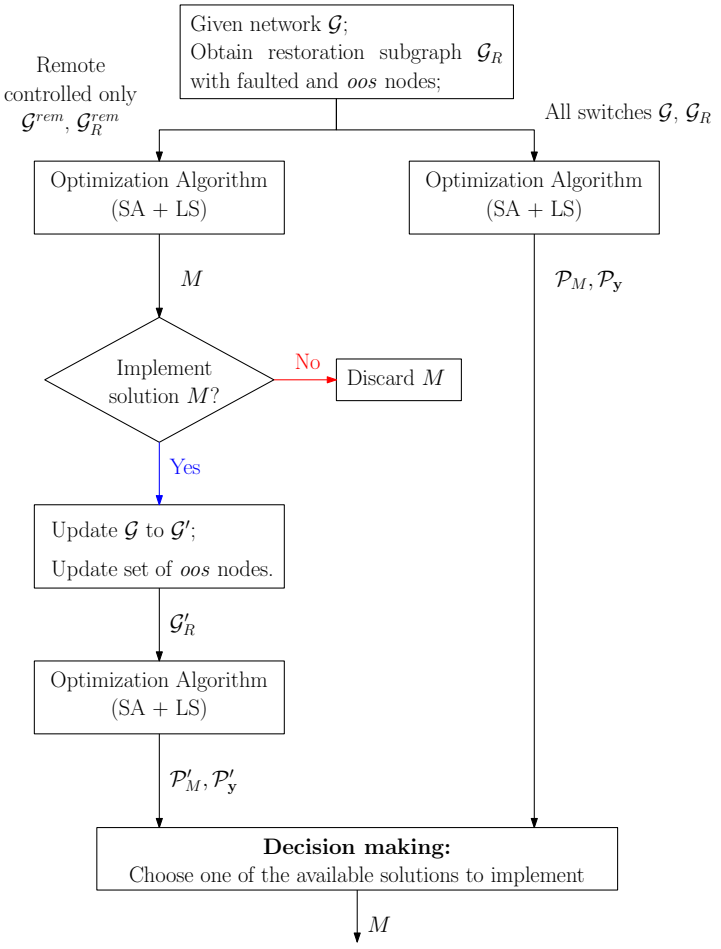


FIGURE 5.9: Complete solution flowchart for the load restoration problem.

Finally, notice that considerations regarding maneuver times become meaningless when handling only remote switches. Fortunately, thanks to the dedicated treatment they receive in the constructive heuristic estimating the time and the energy not supplied (Figure 4.6), the $E_{NS}(\cdot)$ value for any such solution is zero. Therefore, the proposed formulation (5.2) is automatically converted into a single-objective one that minimizes $S_{NR}(\cdot)$, and there is no need to make any adaptations in the algorithm. Moreover, the output in this case is a single sequence (which can even be the null, "do nothing" solution), which makes decision making completely straightforward.

5.6 Summary

This chapter presented the proposed approach for solving the restoration problem. It was formulated as a multi-objective optimization problem, in which the power not restored $S_{NR}(\cdot)$ and the recoverable energy not supplied $E_{NS}(\cdot)$ should be minimized simultaneously. The search space $\mathbb{X}$ is a permutation of all available switches $\mathcal{E}_R$ of the restoration sub-graph, and an evaluation procedure is proposed to convert each permutation vector $\mathbf{x}$ into a proper sequence of maneuvers M using the methods of the previous chapter.

The adopted codification together with the evaluation process have the nice property that all vectors always produce sequences satisfying radiality, voltage, current and overload constraints, so the underlying optimization algorithm performs essentially an unrestricted search in the space. This prevents the need for adjusting penalty factors and other parameters that may be too problem dependent. On the other hand, a number of permutations may induce essentially the same sequence because of the existence of neutral elements. Thus, this chapter also proposed *reduction* schemes to prevent the creation of some permutations that have high chances of generating identical solutions. The resulting multi-objective problem is solved by a Simulated Annealing followed by a Local Search refinement, both adapted to the load restoration problem and equipped with the presented perturbation schemes.

Lastly, a preliminary step considering only remote switches is suggested. This phase makes use of the ability of these maneuvers to be performed almost instantly such that, if able to recover even a small fraction of the *oos* loads, they can prevent the bad contribution of these nodes to both the SAIDI and SAIFI reliability indices. The complete algorithm is presented in Figure 5.9.

Chapter 6

Computational Tests

This chapter presents the results obtained when employing the proposed techniques of the previous chapter in a practical context. In the first part, the different perturbation schemes for creating new permutations described in chapter 5 are compared. In the second part, the proposed algorithm is employed to solve various scenarios of faults.

The simple network originally presented in [63] and used throughout the whole text is helpful for introducing concepts and examples, but, in order to validate the performance of the methods in this work, it is interesting to adopt a larger one, which is closer to real world systems. For that, the network shown in Figure 6.1 will be used as case study. It represents a system from a Brazilian distribution utility with five feeders, 703 buses and 132 switches – among which 10 are remote controlled. Loads were estimated according to the expected consumption at the peak hours on a weekday, and we employ an equivalent single-phase structure, with a constant power model for the loads to be used in the power flow algorithm. In Figure 6.1, the loads without maneuverable switches connecting them were condensed into sectors, so only 121 nodes are actually illustrated. Also, each sector took as label its highest load number, which explains the odd numbering. Finally, notice that the nodes are graphically depicted in a somewhat "artificial" manner to facilitate the presentation, not necessarily following their geographical position.

6.1 Comparison of perturbation schemes

After proposing reduction schemes to prevent some permutations from being generated, an obvious question is whether or not the efficiency gains compensate the possible loss in exploration capability. This first part investigates this topic.

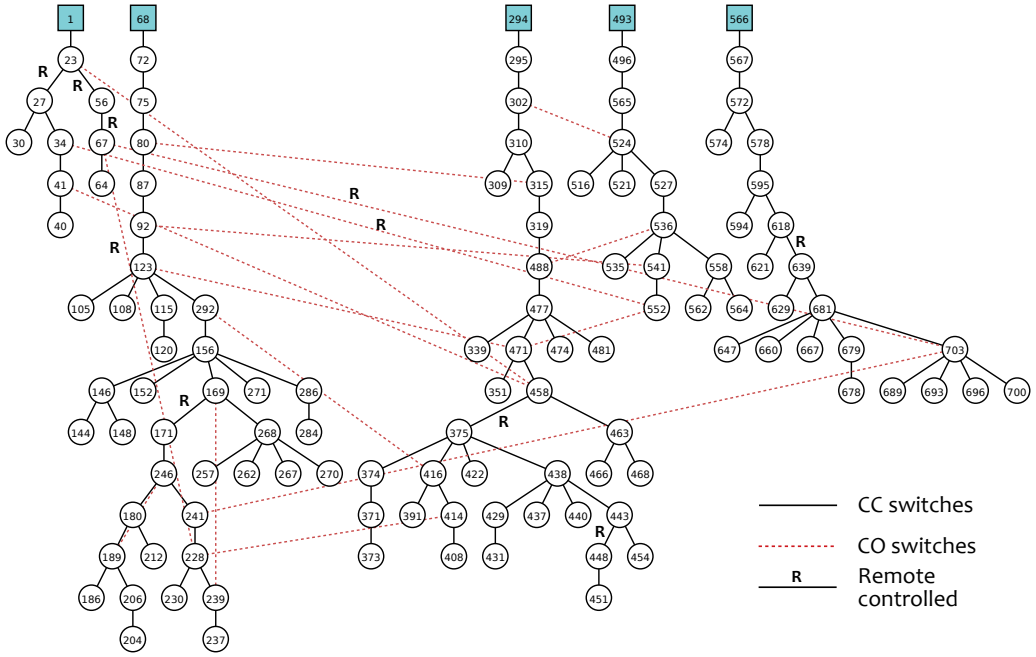


FIGURE 6.1: A real-world based distribution system used to validate the results of this chapter. Like the previous networks, solid and dashed lines indicate currently closed (CC) and currently open (CO) switches, respectively. The figure also emphasizes the remote ones with an **R** on top of the respective edge.

6.1.1 Comparing CC perturbation schemes

Given a permutation $\mathbf{x} = [\mathbf{x}^{CO}\ \mathbf{x}^{CC}]$, we compare the following schemes for creating disturbances in the $\mathbf{x}^{CC}$ switches:

Sch0: swaps any two elements of $\mathbf{x}^{CC}$ without repetition;

Sch1: swaps any two elements of $\mathbf{x}^{CC}$ that belong to the same feeder (Scheme 1), as used in [25];

Sch2: limits the possibilities of exchanges to only the effectively opened elements in a full set $\mathcal{M}$ (Scheme 2);

We consider a scenario of faults at nodes 68 and 294, interrupting two feeders completely. The resulting restoration subgraph has permutations with $n = 85$ switches, among which 14 are CO and 71 are CC. The search space then has $14!71! \approx 7.414327 \times 10^{112}$ different permutations, which is large enough to justify the need for the proposed reduction schemes.

In order to compare the different schemes, $N = 100$ distinct permutations, $\mathbf{x}_j \in \mathbb{X}$, $j = 1, \ldots, N$, were sampled in the search space. For each vector, a complete neighborhood search in the CC partition was executed, after which we stored:

N_{total} the total number of neighbors;

N_{diff} the number of perturbations that are different from the current vector. Consider two permutations $\mathbf{x}$ and $\mathbf{x}_p$ distinct if they differ in at least one sub-sequence induced in the evaluation procedure of Figure 5.3;

N_{ndom} the number of neighbors not dominated by the current permutation. A vector $\mathbf{x}_p$ is non-dominated with respect to $\mathbf{x}$ if, after decoding each one into sub-sequences and combining them, at least one solution from the former remains non-dominated with respect to all solutions.

The results are illustrated in Figure 6.2. The left panel shows how the proposed mechanism, **Sch2**, provides a substantial reduction in neighborhood size when compared to the other two. However, this by itself is not necessarily an indicative of a good scheme, as we can lose interesting perturbations. To verify that this is not an issue, the middle and right panels show N_{diff} and N_{ndom} for each perturbation scheme. As it can be observed, as the neighborhood size is reduced, some distinct and non-dominated solutions can be lost. Nevertheless, these differences seem less significant for the proposed method specially when compared to the actual N_{total}.

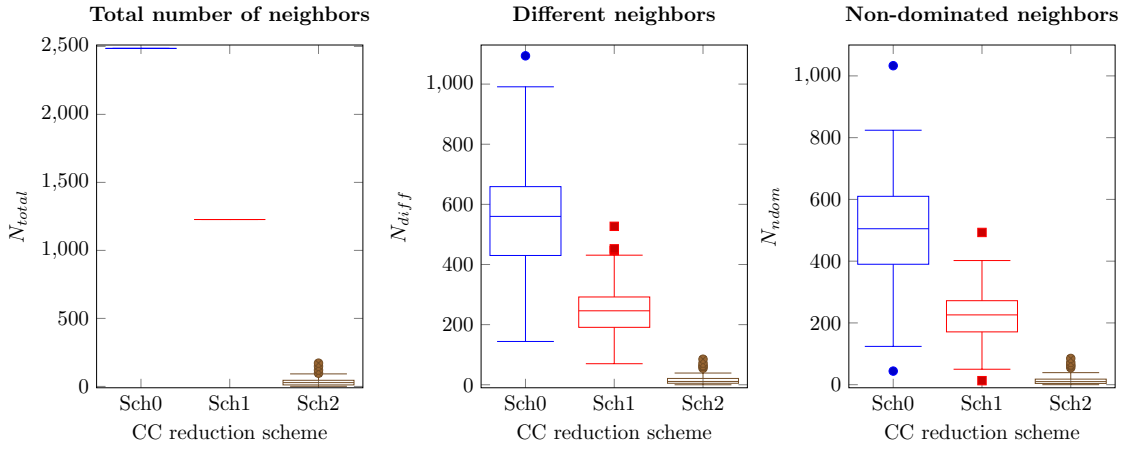


FIGURE 6.2: Comparing different CC perturbation schemes. Left: total number of neighbors. Middle: number of distinct neighbors. Right: number of non-dominated neighbors.

A relevant point to be perceived is: for **Sch0** and **Sch1** the number of perturbations that produce at least one distinct (and thus, interesting) solution can vary from less than 8% to less than 40% of the total. This means that, for these schemes, we may have to explore more than half of the neighborhood of a permutation to obtain any new information. With this discussion, we can argue that the *absolute* number of distinct solutions is not the most important factor. In a metaheuristic like SA, what is most effective is to increase the likelihood that a sampled neighbor will be different from the current vector, that is, the *relative* number of useful perturbations is more relevant.

Similarly for a local-search based method: the search will be more effective as it spends less time looking for potential inclusions in the archive, which is also related to the ratio of N_{diff} (or N_{ndom}) to N_{total}.

With this reasoning, we show in Figure 6.3 the proportions of different and improved perturbations, each normalized to the interval $[0, 1]$ for each starting permutation $\mathbf{x}_j$, $j = 1, \ldots, N$, wherein 0 is the worst proportion and 1 is the best. The data now seems to suggest a strong advantage of the proposed method Sch2 when compared to the others.

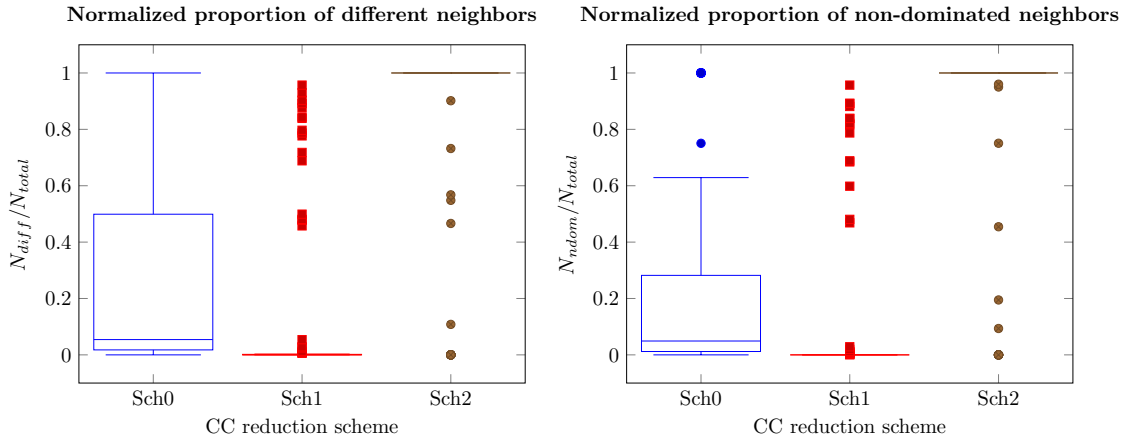


FIGURE 6.3: Comparing the proportion of different and non-dominated CC perturbation schemes. The data are normalized between 0 and 1 for each solution.

To prevent relying on visual and probably deceptive analysis, bootstrap confidence intervals for the mean of matched differences between these schemes were computed [68, 69]. Being non-parametric tests, bootstraps do not require the assumption of normality, and also have better accuracy and are more general than other non-parametric methods. The only necessary requirement is independence of the observations, which is already guaranteed thanks to the nature of the experiment.

Keep in mind that the tests require multiple comparisons (**Sch0** with **Sch1**, **Sch0** with **Sch2** and **Sch1** with **Sch2**), so we need some kind of correction to obtain the desired family-wise confidence level. Here we adopt the simple and conservative Bonferroni correction [68], which states that, if we wish for a simultaneous confidence level of α, then each individual interval should be constructed with the following adjusted significance level:

$$\alpha_{adj} = \frac{\alpha}{\text{number of comparisons}} \tag{6.1}$$

Thus, to provide a simultaneous $\alpha = 0.05\%$ confidence level (thus a 95% confidence interval), since we have three comparisons, the individual significance level of each interval should be set to $\alpha_{adj} \approx 0.017$.

The resulting intervals are shown in Figure 6.4. Intervals that do not touch the horizontal zero line imply a statistically significant difference at the 95% confidence level. As shown in the figure, **Sch0** and **Sch1** present mean values of both N_{diff}/N_{total} and N_{ndom}/N_{total} that are significantly inferior to **Sch2**, confirming the qualitative analysis from Figure 6.3. This suggests that **Sch2** is a superior method regarding the proportion of different and non-dominated solutions. Furthermore, we can also make a probably not so evident conclusion that the reduction scheme adopted in a previous work (**Sch1**) is actually worse than using no reduction scheme at all (**Sch0**) when considering these ratios. This also shows the relevance of this kind of analysis, as simply reducing the neighborhood size is not always enough for obtaining gains in efficiency.

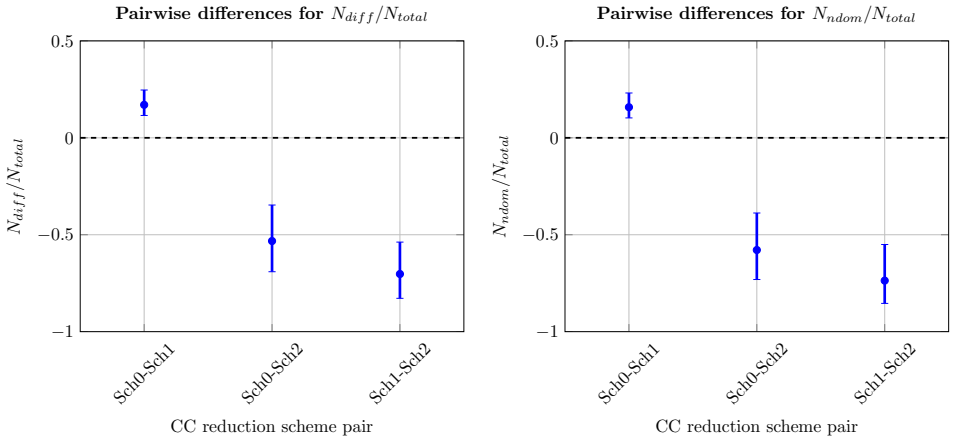


FIGURE 6.4: Simultaneous 95% confidence intervals for differences in the proportion of different (left) and improved (right) neighbors. Intervals that do not intercept the zero line indicate statistically significant differences.

Lastly, while this conclusion may seem valid only for the specific scenario of faults at feeders 68 and 292, further experiments using all combinations of interruptions in two different feeders in this network yielded very similar results to the ones shown in Figure 6.4. Hence, it is possible to suggest the adoption of **Sch2** as the perturbation method for CC switches in the proposed method.

6.1.2 Comparing CO perturbation schemes

For the CO partition, we compare schemes PS1 to PS6 (depicted in Figure 5.5), but always applying the simple reduction of disregarding permutations that involve only the last never considered switches in $\mathbf{x}^{CO}$ (section 5.4.2). We use the same scenario of faults at nodes 68 and 292 and the same procedure of generating $N = 100$ random vectors in the space and performing a neighborhood search in their CO components. Figure 6.5 shows the resulting (normalized) proportions of different and improved perturbations for each perturbation scheme.

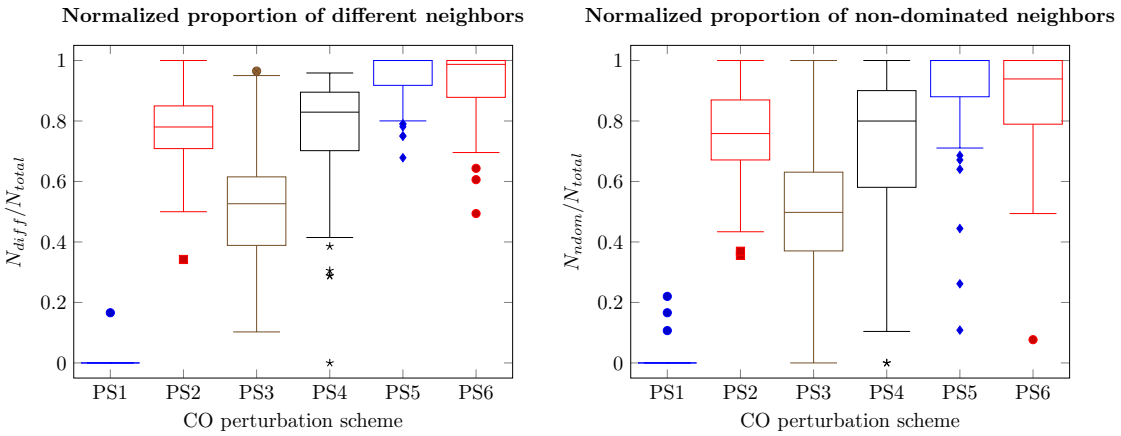


FIGURE 6.5: Comparing the normalized proportion of different and improved CO perturbation schemes.

It is harder here to draw any conclusions based only on the visual analysis, although it may suggest an advantage for PS5 or PS6. Then, for a more rigorous approach similarly to the previous one, confidence intervals on the mean of paired differences between all schemes were computed. This results in 15 matched comparisons, and, using equation (6.1), we adjust an individual significance level $\alpha_{adj} \approx 0.0033$ so that the family-wise error rate becomes the desired $\alpha = 0.05$. The resulting confidence intervals are shown in Figure 6.6. Even with the large number of intervals, it is evident that all comparisons involving PS5 and PS6 suggest statistically significant advantages for both of them, and there is no significant difference between the two. Even so, we choose PS5 over PS6 thanks to its smaller absolute number of neighbors, and so it is chosen as CO perturbation scheme for both the Simulated Annealing and the Local Search in the proposed algorithm. Once more, these results persist for different scenarios of two-feeder interruptions, so the conclusion remains valid.

6.1.3 Discussion

Thanks to the results of this section, we saw that the proposed reduction scheme **Sch2** is able to substantially decrease the required number of perturbations in order to obtain a different solution in the objective space, and this without a significant loss in possible alternatives given the relative high proportions shown in Figure 6.3. Also, for the CO partition, this analysis helped us settle with the PS5 scheme. Therefore, the proposed algorithm will be composed of a Simulated Annealing (SA) followed by a multi-objective Local Search refinement (LS) with **Sch2** and PS5 as CC and CO perturbation schemes, respectively.

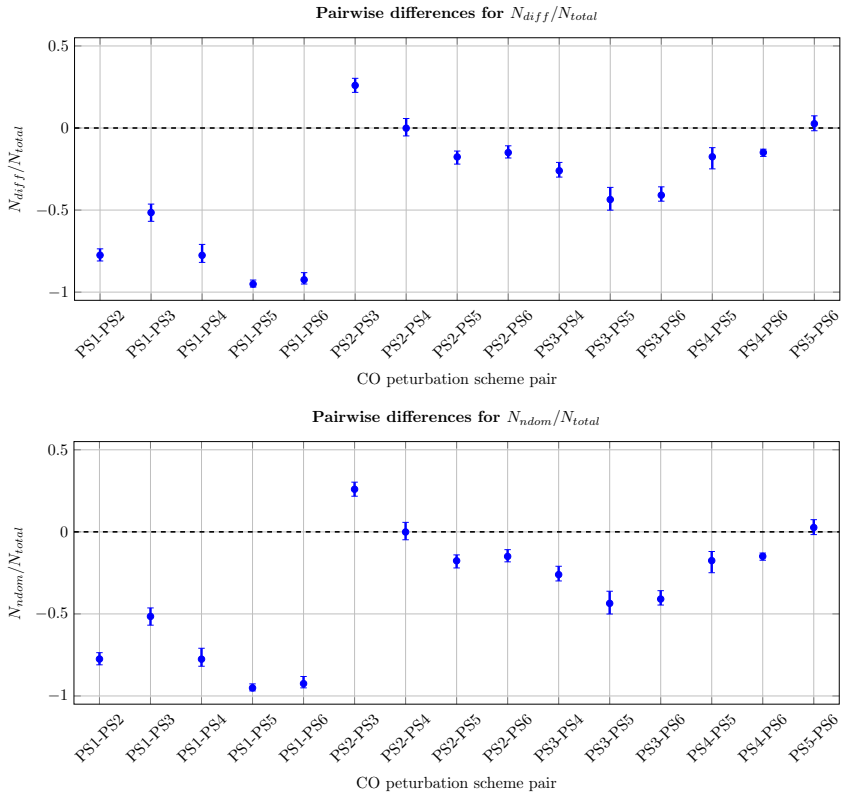


FIGURE 6.6: Simultaneous 95% confidence intervals for differences in the proportion of different (top) and improved (bottom) neighbors for CO schemes. Intervals that intercept the zero line indicate absence of statistically significant differences.

6.2 Assessing the performance of the proposed algorithm

In this section the performance of the proposed solution framework is assessed in different fault scenarios. The following characteristics have been set up for the experiments here:

- Internal archives are limited to $\mu_A = 15$ solutions;
- The initial vector for the SA is a random permutation of the edges of the restoration subgraph;
- Similarly to the experiments of Chapter 4, it is assumed that there is one dispatch team per feeder, which accounts for $|\mathcal{T}| = 5$ teams in this case. Also, for simplicity, it is assumed that the time taken for a team to arrive from its current position to a given switch is linearly related to the Euclidean distance between their geographical positions, and a switch is readily operated after this time, i.e., $c_e^{op} = 0 \ \forall e \in \mathcal{E}$. Notice that these assumptions are only pragmatic simplifications, and do not diminish the practical character of the proposed method.
- Regarding the teams' initial positions, two cases are considered:

Case a: it is possible that some teams are in the middle of a previous job or just finishing it so, when summoned, they are initially dispersed. This situation is reproduced by placing each team in a source node, specifically, at nodes 1, 68, 294, 493 and 566.

Case b: another typical circumstance has all teams concentrated at an operations center until a service is requested. To simulate this case the initial location of all teams is set to the mean point of all sources' positions.

To obtain a comparison baseline for the performance of the proposed algorithm, a Branch and Bound (BB) method was developed, which constructs sequences with stages containing one closing and up to three openings (i.e., it operates 0, 1, 2 or three CC switches), plus an isolation maneuver if required. To prevent interrupting the flow of this chapter, the interested reader can check its complete formulation in Appendix E.

6.2.1 Simple scenarios

In this first part, three scenarios for which the search space is small enough to allow for the determination of the true Pareto-optimal solutions by a brute force method such as the adopted Branch and Bound were used as test problems. Even if this approach limits the tests for scenarios with small dimensions, they are still representative of common real-world cases.

The results shown in this first part are limited to the right track of Figure 5.9, as no solution with only remote switches was obtained.

6.2.1.1 Scenario 1: Faults at nodes 56, 180 and 541

The faults in this scenario are not very serious since they do not have many downstream loads, but three simultaneous failures at distinct feeders is enough to hinder the performance of many methods from the literature, which assume only one fault occurs at a time (see Table 3.1, criterion F3). The corresponding restoration subgraph is shown in Figure 6.7. The search space here has $n = 9$ maneuverable edges, with $n^{CO} = 5$ and $n^{CC} = 4$. The total number of permutations is $5!4! = 2880$, which is small enough to be solved quickly even by an exhaustive search.

The results obtained for this scenario are illustrated in Figure 6.8 for each starting point of the teams (cases **a** and **b**, described earlier). In both panels, the rightmost point corresponds to the "do nothing" solution. It is evident from these results that the Simulated Annealing was able to reach the whole Pareto-optimal front in both cases.

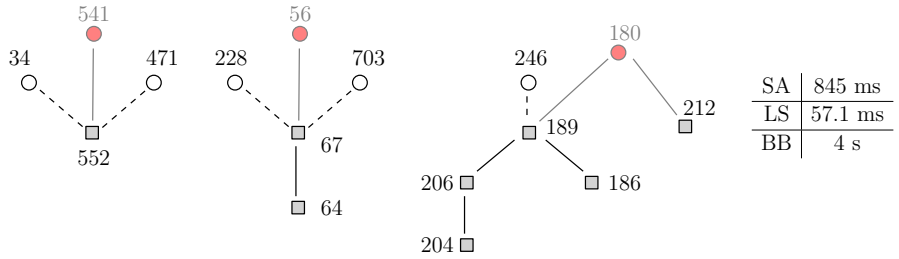


FIGURE 6.7: Restoration subgraph for the first simple scenario. The table on the right shows typical processing times of each component of the algorithm, as well as that of the Branch and Bound. These times are valid for both cases **a** and **b**.

Hence, a subsequent execution of the local search (not shown in the figure) resulted in no improvement.

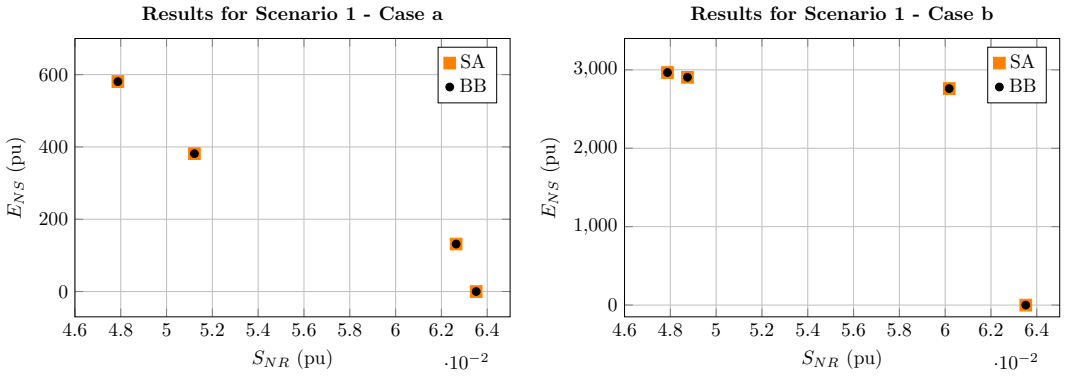


FIGURE 6.8: Results of the first scenario. **Left**: Case **a**, assuming each team starts from a different source. **Right**: Case **b**, where all teams begin at the same common point. SA indicates the results of the Simulated Annealing, and BB stands for Branch and Bound. In this case, the BB approach computed the true Pareto-optimal front.

The sequences of maneuvers induced by the solutions in the final archive are shown in Table 6.1 and, for easy reference, in Figure 6.9 for Case **a**. Apart from the adopted quality indices $S_{NR}(\cdot)$ and $E_{NS}(\cdot)$, two estimations of the total maneuver time are also shown for reference.[1] The first, T_m^{prop}, corresponds to the proposed heuristic (Figure 4.6), while T_m^{seq} is a sequential approximation assuming only one team is available – for fairness, the team that could perform all maneuvers in the smallest time was used for the comparison. This sequential approximation is closer to the approach adopted in existing works [24–26], but even in this case the suggested methodology of considering the actual displacement time from a team's current position to a switch is still more realistic than assigning a fixed weight t_e for each switch, as discussed many times in this text.

The absolute values of these indices are not necessarily important, so focus more on how they can be compared in the cases considered. Table 6.1 shows that, for Case **a**,

[1]Notice that, given that the transit times were artificially set as a linear function of the Euclidean distance between locations, no specific time units were used.

Case	Sequence of maneuvers	S_{NR}	E_{NS}	T_m^{prop}	T_m^{seq}
Post fault	**Open:** {}; **Close:** ()	100%	0.00	0.00	0.00
Case **a**	**Open:** {(67, 64), (56, 67)}; **Close:** (67, 228)	98.58%	1.31×10^2	2068	2068
	Open: {(67, 64), (56, 67)}; **Close:** (67, 228) **Open:** {(180, 189)}; **Close:** (189, 246)	80.63%	3.81×10^2	6059	6059
	Open: {(67, 64), (56, 67)}; **Close:** (67, 228) **Open:** {(180, 189)}; **Close:** (189, 246) **Open:** {(541, 552)}; **Close:** (34, 552)	75.43%	5.81×10^2	9951	9951
Case **b**	**Open:** {(541, 552)}; **Close:** (471, 552)	94.80%	2.76×10^3	43472	60090
	Open: {(541, 552)}; **Close:** (471, 552) **Open:** {(180, 189)}; **Close:** (189, 246)	76.69%	2.90×10^3	45858	75601
	Open: {(541, 552)}; **Close:** (471, 552) **Open:** {(180, 189)}; **Close:** (189, 246) **Open:** {(64, 67), (56, 67)}; **Close:** (67, 703)	75.43%	2.97×10^3	47108	76850

TABLE 6.1: Sequences of maneuvers induced by each solution in the final archive for Scenario 1. Each cell contains a different solution, with each row representing a stage. The first solution corresponds to the "do nothing" one. S_{NR} is given as percentage of the post-fault, E_{NS} is in pu, and the times are in a convenient time unit. Also, T_m^{prop} is obtained with the proposed heuristic and T_m^{seq} with a sequential approach.

both time estimations yielded the same value, meaning that in this case it was faster to operate with only a single team for all maneuvers, possibly due to the disperse initial disposition of the dispatch crews. However, as Case **b** shows, if parallel assignments are possible, the actual time can be greatly reduced. For instance, the third solution in Case **b** has five more operations than the first, but the total maneuver time increased roughly by 8% with the proposed heuristic, while this increment was about 28% with the sequential approximation. Also, the time required by a single maintenance crew to perform all operations, T_m^{seq}, is 63% greater than the total time for the case of multiple teams (T_m^{prop}). These results actually confirm the same conclusions obtained from section 4.4, indicating that the proposed approach can return solutions capable of preventing unnecessary degradations of the SAIDI index.

Once provided with these results, the dispatcher can analyze each solution and determine the one to be implemented. Typical processing times for each component of the algorithm, as well as for the BB approach, are provided in the summary table at the right of Figure 6.7.[2] As expected, due to the small size of this scenario all methods were fast enough for practical purposes.

6.2.1.2 Scenario 2: Fault at node 493

Even with a single fault, this scenario leaves a whole feeder offline, so it can be considered more severe than the previous one. The corresponding restoration subgraph is shown in

[2]For reference, these results and the following ones where obtained using a 3.10GHz Intel i3 processor machine with 4GB RAM, running Fedora Linux 27. The code is implemented in Python 3.5 with the `networkx` module.

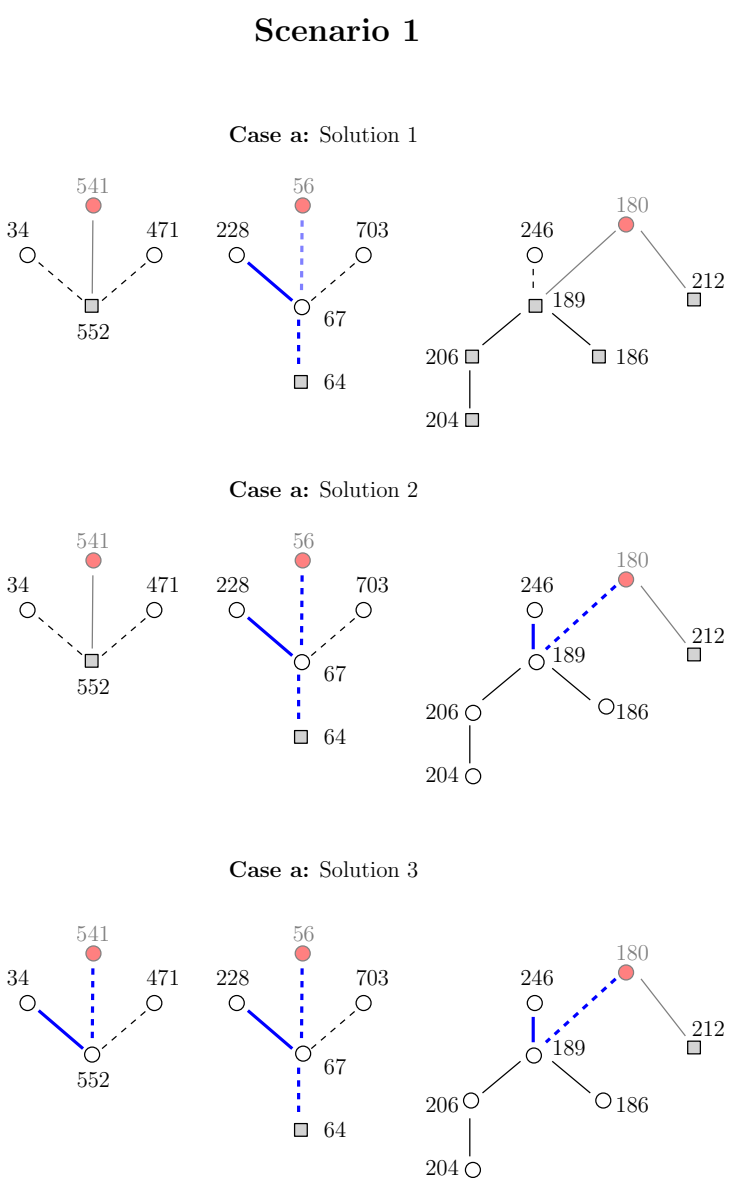


FIGURE 6.9: Solution implementations for the first simple scenario. Only Case **a** is shown since the other has a very similar idea.

Figure 6.10. Here, the search space has dimension $n = 17$, with $n^{CO} = 5$ and $n^{CC} = 12$, resulting in $5!12! \approx 5.75 \times 10^{10}$ permutations, and it requires a few days to be solved to proven optimality using the BB approach.

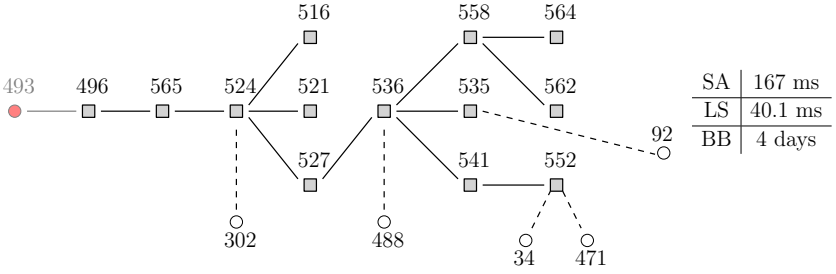


FIGURE 6.10: Restoration subgraph for the second simple scenario.

The results for this scenario are shown in Figure 6.11. Despite its severeness, the supporting feeders have enough capacity so that all *oos* loads can be recovered with a single pair of isolation opening and a closing. So, in both cases, apart from the "do nothing" solution, there is only one Pareto-optimal point. Again, the SA was able to solve it very quickly, specially when compared with the BB approach. In this case, since no closing requires load shedding, the proposed *Scheme 2* prevents any swaps among CC switches, and thus the algorithm automatically handles only the CO partition, which makes the search very efficient.

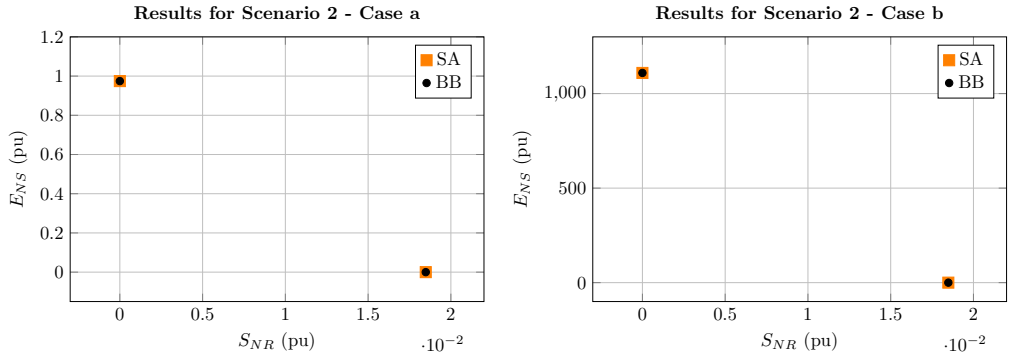


FIGURE 6.11: Results for the second scenario. **Left**: Case **a**. **Right**: Case **b**. The Local Search did not produce any improvement as the solutions were already optimal.

The resulting sequence of maneuvers is given in Table 6.2 and in Figure 6.12. It is again clear that, for the case where the teams are initially located at each source node (Case **a**), it was enough to use a single crew to execute the whole sequence, and thus both approaches resulted in the same value. Conversely, for Case **b** a proper coordination of the teams resulted in a total time that is almost half of that obtained by a single team.

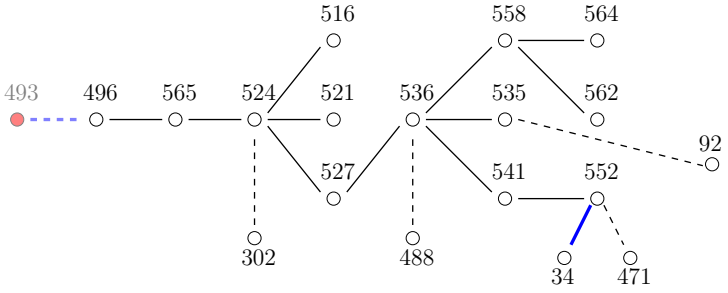
Case	**Sequence of maneuvers**	S_{NR}	E_{NS}	T_m^{prop}	T_m^{seq}
Post fault	**Open:** {}; **Close:** ()	100%	0.00	0.00	0.00
Case **a**	**Open:** {(493, 496)}; **Close:** (34, 552)	0.00	0.9744	52.72	52.72
Case **b**	**Open:** {(493, 496)}; **Close:** (302, 524)	0.00	1.11×10^3	60016	113757

TABLE 6.2: Optimal sequence of maneuvers for Scenario 2. In this case, $S_{NR}(\cdot)$ achieved a zero value because the faulted node is a source, which has no power demands. In general, this value can be positive if the faulted nodes have a demand, even if all *oos* nodes are recovered.

As a final observation, notice how the closing maneuver changes depending on the team's initial position. An approach based solely on the number of maneuvers would return all possible closings as equally good sequences and completely neglect the differences in displacement time between these switches. Thus, depending on the final solution picked by the decision maker, there could be a unnecessary worsening on the SAIDI index due to the lack of a proper modeling of the implementation time.

Scenario 2

Case a: Only solution



Case b: Only solution

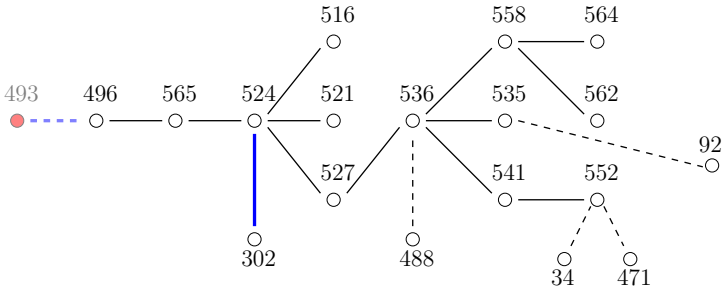


FIGURE 6.12: Solution implementations for the second simple scenario. Since there is only one solution per case, both cases were illustrated.

6.2.1.3 Scenario 3: Faults at nodes 536 and 681

This final "simple" scenario has two simultaneous faults, generating a search space with $n = 13$, $n^{CO} = 5$ and $n^{CC} = 8$, with a total of $5!8! \approx 4.84 \times 10^6$ different options. The BB procedure takes a few hours to find the true optimal front, which would also be impractical in real-world situations. The restoration subgraph is shown in Figure 6.13, along with typical processing times for each component in comparison with the exact procedure. Despite its similarity with the first scenario, the faults are more geographically apart from each other, and this should influence in the maneuver times required for a single team in comparison with the use of multiple maintenance crews.

The results for both cases are shown in Figure 6.14. Again the SA was able to compute the true Pareto-optimal front, precluding the necessity of any local search refinement. Table 6.3 provides the sequences induced by each solution in the final archive, and Figure 6.15 illustrates these solutions for Case **a**. As can be seen, the first solution in Case **a** can be implemented with a single team without losing efficacy (in terms of time required), but for the second one, which recovers both *oos* sub-regions, there is a

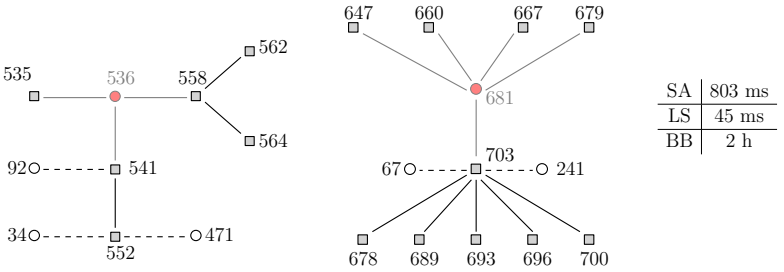


FIGURE 6.13: Restoration subgraph for the third simple scenario.

considerable time economy when adopting more than one team. In this case, the initial distance between the crews was used as an advantage, and so the proposed method was able to prevent a possibly large negative impact in the reliability indices. Case **b** again shows a great benefit when handling more teams in parallel, which reinforces the conclusions drawn from the previous scenarios.

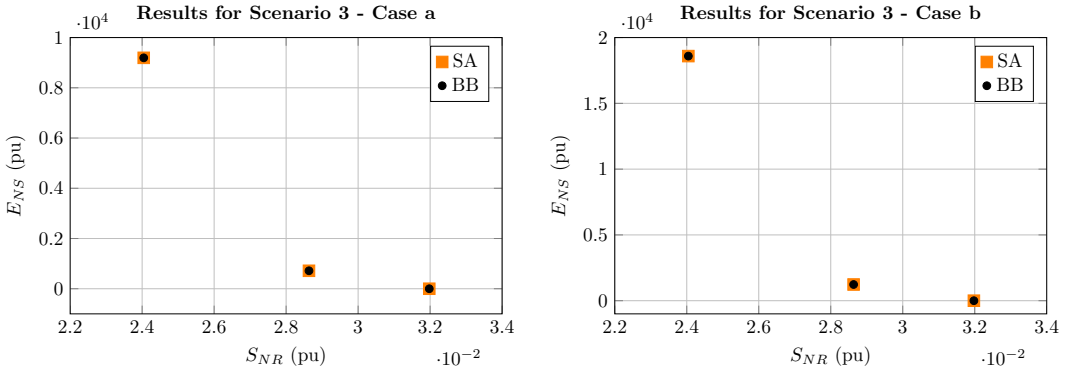


FIGURE 6.14: Results of the third simple scenario. **Left**: Case **a**. **Right**: Case **b**. Once again the Local Search was not required.

Case	**Sequence of maneuvers**	S_{NR}	E_{NS}	T_m^{prop}	T_m^{seq}
Post fault	**Open:** {}; **Close:** ()	100%	0.00	0.00	0.00
Case **a**	**Open:** {(536,541)}; **Close:** (34, 552)	89.38%	7.15×10^2	22 363	22 363
	Open: {(536, 541)}; **Close:** (34, 552) **Open:** {(681, 703)}; **Close:** (241, 703)	75.00%	9.19×10^3	318 433	1 866 229
Case **b**	**Open:** {(536,541)}; **Close:** (92, 541)	89.38%	1.23×10^3	38 496	45 227
	Open: {(536, 541)}; **Close:** (92, 541) **Open:** {(681, 703)}; **Close:** (67, 703)	75.00%	1.86×10^4	644 777	952 479

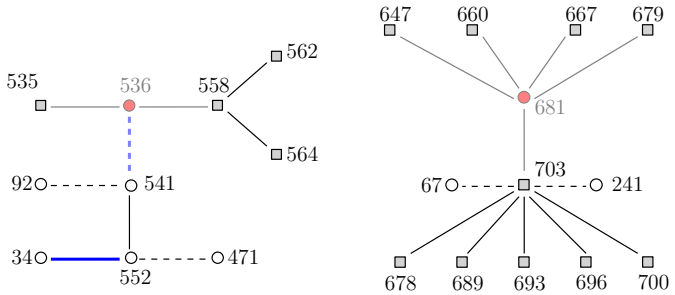
TABLE 6.3: Sequences of maneuvers induced by each solution in the final archive for Scenario 3. Each cell contains a different solution, for which each row represents a stage.

6.2.2 Complex scenario

After validating the algorithm in simple scenarios, let us assess its performance in a more complex case in this section. Consider the same very serious situation of section 6.1 with faults at feeders 68 and 294, with $14!71! \approx 7.41 \times 10^{112}$ permutations, which is virtually

Scenario 3

Case a: Solution 1



Case a: Solution 2

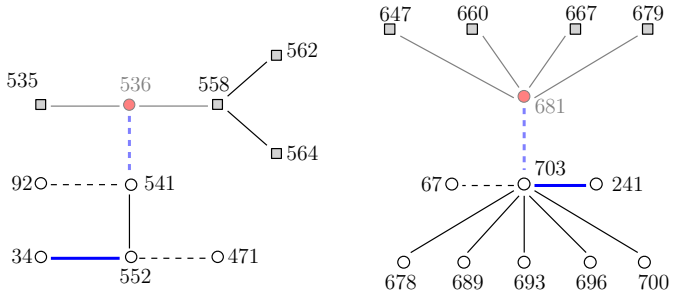


FIGURE 6.15: Solution implementations for the third simple scenario. As in Figure 6.9, only Case **a** is shown.

impossible to be solved by any exact approach. Therefore, for the comparisons in this case, a pruning heuristic was included in the BB method such that, after computing all possible maneuvers with only one stage, the non-dominated ones are retained, and the process is repeated for maneuvers with two stages, then three, and so on, as detailed in Appendix E. This approach resembles more closely the more recent mathematical programming methods, which usually employ heuristics to make the algorithms applicable as discussed in section 3.1 of the literature review. Also, considering that the benefits of a proper coordination of more than one dispatch team were already made clear in the first scenarios, this section handles only Case **a**, so that the focus can be in the solution process of the restoration problem using the complete procedure of Figure 5.9.

Unlike the simple scenarios, in this case a solution containing only remote switches was obtained and is shown in Figure 6.16 and Table 6.4. Given the very small number of these switches, the first execution of the algorithm (top left block in Figure 5.9) took less than one second. By implementing these maneuvers, the distribution utility was able to prevent a negative contribution of about 20% of the total non-energized load with no impact to its reliability indices. And keep in mind that, if the decision maker is

quick on pressing the buttons, even some negative impacts on SAIFI can be prevented in this case.

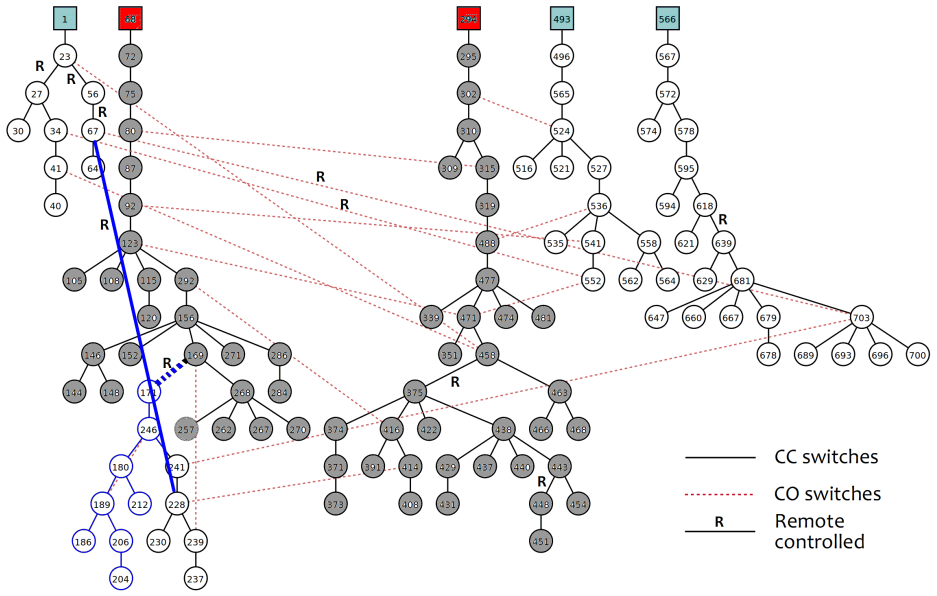


FIGURE 6.16: Graphical representation of a solution with only remote switches. Sectors downstream to 169 become energized before penalties start to count, so the implementation of this solution acts as if these nodes were never interrupted in the first place.

Sequence of maneuvers	S_{NR}	E_{NS}
Open: {}; **Close:** ()	100%	0.00
Open: {(169, 171)}; **Close:** (67, 228)	79.03%	0.00

TABLE 6.4: Initial results for the complex scenario, considering only remotely controlled switches. The total maneuver time is virtually zero.

Following the remaining guidelines of the complete algorithm, the method now considers all available switches and runs the proposed algorithm in parallel (i) starting from the post-fault configuration, and (ii) after the operation of the remote switches. The results of a typical execution are shown in Figure 6.17. In this case, there are no guarantees that the solutions are Pareto-optimal (even for the BB, given the use of the pruning heuristic).

A first observation for this scenario is that, differently from the previous ones, the local search refinement was able to improve the archive returned by the SA in both cases. Also, the final archive of the proposed algorithm dominates most of the solutions of the BB method, and, just as importantly, is not dominated by any BB solution. Regarding processing times, the BB with the pruning heuristic took over four hours to be concluded, while the total time of the proposed algorithm ranged from five to seven minutes, which is not only much quicker but also complies with the usual technical requisite of less than ten minutes. Also, it is important to highlight that, while the final archive tends to

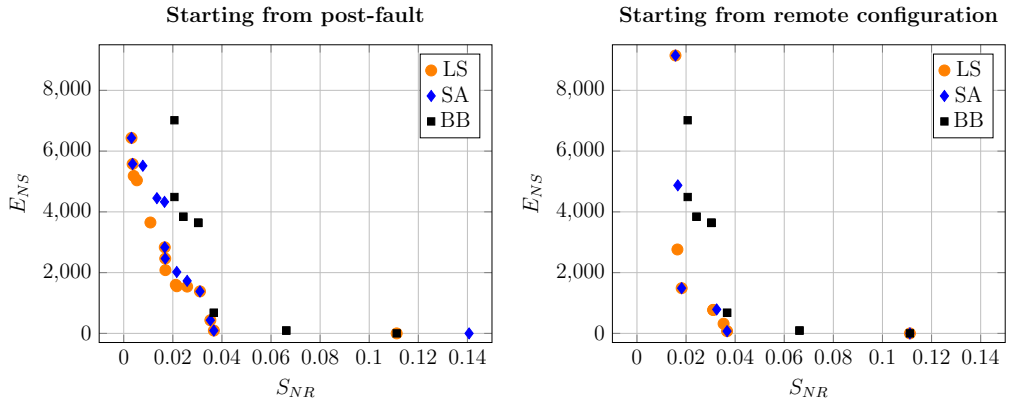


FIGURE 6.17: Typical results of the complete algorithm in the complex scenario compared to the Branch and Bound (with pruning heuristic). **Left**: results starting from the post-fault configuration. **Right**: starting after the remote switches were operated.

vary between different executions, this pattern of generating better solutions in a much shorter time persisted in all runs.

To finish the procedure, we can combine the sequences in both cases and remove the dominated ones, resulting in Figure 6.18. The solutions in this non-dominated front can then be presented to the dispatcher. Notice that, due to the small number of objectives, we follow here an *a posteriori* philosophy (section D.4.2). In this case, the total number of candidate sequences is $14 < \mu_A$, so no final truncation was required; otherwise, another run the non-dominated sorting with crowding distance would be necessary to prevent the archive from exceeding its limits. Given these solutions, the dispatcher would perform a final decision making step to select a final sequence to be implemented in the system. For sequences obtained with respect to the post-fault configuration, the operated remote switches could be brought back to their original state at basically no cost, as explained in Section 5.5.

6.2.2.1 Choosing a final solution

The idea of the *a posteriori* philosophy is to present a "good representation" (section D.4.2) of the Pareto-optimal solutions to the engineer. We adopted this approach in this section for simplicity, and its applicability depends on how many solutions are presented so that the decision maker is not overwhelmed. Here, we chose 15 as the archive size and assumed that this was an adequate limit. However, this depends on how stressful the moment may be for the engineer, which is very problem-dependent, and in some cases evaluating 15 sequences may be too much.

One option to overcome this issue is to output two or three solutions, such as the extremes and a middle one, or even something more intricate using a reference point

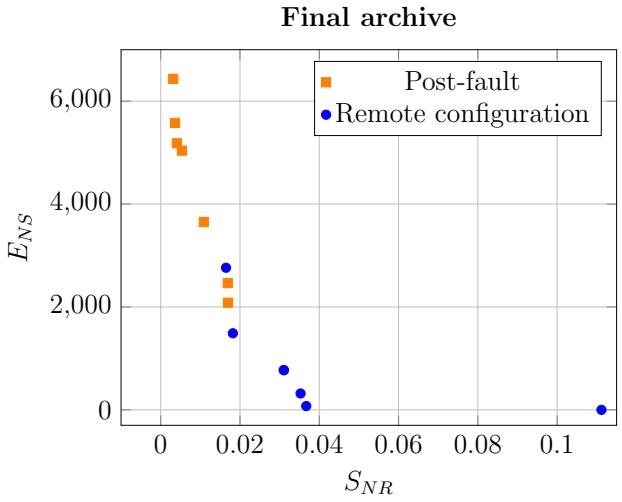


FIGURE 6.18: Final archive presented to the dispatch engineer. Square orange points indicate sequences obtained after starting from the post-fault configuration, so they would require bringing some previously operated remotely controlled switches to their initial states.

and returning the closest solution to it (section D.3.2). Unfortunately, depending on the Pareto-front size, if these solutions are too scattered we may lose the main benefit of the multi-objective formulation and prevent any useful trade-off analysis.[3] Another possibility is to employ a more "interactive" approach and return fewer solutions but concentrated in a more interesting region, using, for instance, preference-guided mechanisms (section D.3.2.3 and [58, 59]). We leave such possible adaptations for future works.

6.2.3 Discussion

The analyses of case studies presented in this section were able to elucidate some characteristics of the proposed method for the restoration problem. Specifically, for our choice of encoding and evaluation of permutations, (i) the "size" (i.e., dimension) of a problem does not depend on the size of the original network $\mathcal{G}$, but rather on the restoration subgraph $\mathcal{G}_R$, which is (somewhat loosely) related to the severeness of the fault; and (ii) the hardness of a problem does not depend on its size (as seen in the second simple scenario, which was large but easily solvable), but on other factors, like the spare capacity of the supporting feeders and load requirements of *oos* nodes.

The results suggest that, in simple scenarios, the proposed algorithm is able to return a set of Pareto-optimal solutions, validated with an enumeration procedure. Unlike the BB approach, our metaheuristic seems less sensitive to the size of the problem when it

[3] Even so, one may argue that this is still better than flooding the decision maker with too many options, so in some cases this may be the best option.

is relatively easy, and it tends to take very little time to solve it. This can be contrasted with the BB approach, which seems more susceptible to the actual size and may not be applicable in some situations, regardless of the difficulty.

Another point that was made clear by the experiments in this section are the potential gains in performance due to the ability of the proposed approach to distribute the sequences of maneuvers by multiple teams working in parallel. In the worst case the proposed coordination of multiple teams culminates in results that are as good as those obtained using the sequential maneuvering approach, typically proposed in the literature. This worst case can occur (i) if there is only one dispatch team available; or (ii) when one of the crews is much closer to the *oos* region than the others. Outside of these two specific situations, the appropriate coordination of multiple teams, as proposed in this work, can considerably reduce the total maneuver time, resulting in shorter blackout time for customers and better reliability indices for the distribution utility. Also, remember that even the sequential approach used as baseline here is still more realistic than providing a fixed weight to each switch, as more common in previous works.

The results obtained for the more complex scenario also corroborate these observations. The proposed method was able to return solutions that dominate the ones from the Branch and Bound with a pruning heuristic (the exact BB cannot solve this problem in any reasonable time). The processing times were in the order of a few minutes for the proposed method, contrasting with the few hours needed by the BB + heuristic. While this is not necessarily a concrete statement of efficiency, as processing time depends on implementation, language and hardware issues, it can be used as a baseline for possible adoption in distribution companies.

Finally, the complex scenario was also useful in illustrating the effectiveness of generating an initial partial solution considering only remotely controlled switches, which was able to recover 20% of the load in virtually no time. This, together with the team coordination, is useful in reducing negative contributions to the reliability indices as these operations can be performed within the no-penalties time window. In the end, the dispatcher can choose his or her preferred solution from the available ones, as illustrated in Figure 6.18.

6.3 Summary

The performance of the proposed methods was verified in this chapter. First, mechanisms of perturbation were compared among the currently closed (CC) and currently opened (CO) partitions. In the first case, we saw that the proposed scheme, **Sch2**,

was able to drastically reduce the possible perturbations while not losing a significant number of distinct and improved neighbors. Therefore, the *proportion* of these measures was shown to be significantly better than the current method in the literature (**Sch1**) and also better than no reduction scheme at all (**Sch0**). Regarding the CO partition, even though none of the schemes were proposed in this work, the analysis adopted was useful in determining which of these perturbations would be more interesting to be implemented. More specifically, we saw that PS5 and PS6 were the ones that generated the best ratio of distinct and improved disturbances among the available perturbation schemes, but PS5 the preferred thanks to its smaller number of absolute neighbors. Noticing that these conclusions remained true for a wide range of scenarios, it was a simple choice to adopt PS5 and **Sch2** as schemes for the proposed algorithm.

In the second part, we compared the performance of the complete algorithm in a real world based network, depicted in Figure 6.1. In simple scenarios, where it is possible to compute the true efficient front with an exact procedure, we saw that the algorithm was able to reach Pareto-optimal sequences. Another conclusion was the time savings obtained due to a proper coordination of dispatch teams when more than one crew can be used to execute the maneuvers in parallel when compared to a sequential approach.

Regarding a more complex scenario, the proposed method was compared to an exact approach with a pruning heuristic, which is more similar to usual techniques in the literature. Here, we saw that the adopted metaheuristic returned solutions on par – sometimes even better – with the (pruned) enumeration method, but at much smaller processing time, thus validating its performance like in the simple scenarios. Furthermore, we observed the complete proposed method in action, and also how the pre-processing step of considering only remote switches – when applicable – can help preventing bad contributions to the reliability indices, even if for a small percentage of the *oos* loads.

Chapter 7

Conclusions and Continuity Proposals

7.1 Quick summary

Distribution systems have the task of delivering energy to the customers, a.k.a. loads. Among the expectations we may have about them, it is imperative that they promote continuous supply, that is, we expect a reliable service. Unfortunately, we cannot design a practical system that will never fail; the best we can usually do under the occurrence of a fault is to minimize its impacts over the system. For that, we usually rely on protection devices and load restoration methods, the latter being the focus of this work.

We broadly defined load restoration as the act of "recovering the most out of service (*oos*) loads in the shortest time possible without violating constraints such as minimum voltages at buses, maximum current at branches, feeder overloads, and radiality of the final configuration". In practice, this is accomplished by implementing a sequence of closings and openings of switches. With a proper selection of quality indices to evaluate how good a restoration plan is, the problem can then be modeled in general as a multi-objective, non-linear, constrained and combinatorial problem as described in equation (2.11).

Being a very mature problem, there is a great number of studies in the literature that try to solve it. Unfortunately, there is no universal agreement among utilities on technical aspects such as best combination of quality indices to model a "good plan" and what types of operations are allowed or forbidden (load shedding of in service nodes? load transfer? etc.). This makes it hard to compare previous studies in a fair way and to determine advances in the literature.

After an extensive review of previous studies in the literature trying to highlight some of their strong and weak points, we concluded that one point in need for improvement is the fact that the more modern optimization approaches handle only sets instead of sequences of maneuvers. Also – and as a possible consequence of this first topic – there is no proper estimation of the time taken to execute a restoration plan. Sequences of maneuvers should be able to (i) dictate an order of operations that satisfy *precedence rules*, and (ii) allow the use of more realistic indices to better model the "in the shortest time possible" portion of a good restoration plan, such as time of maneuvers and energy not supplied. With the current approach in state of the art methods, only the first condition may be satisfied (if a post-processing step is employed to output a sequence from a set) or none at all, thus leading to deficiencies in their practical character.

7.1.1 Contributions

The first contribution of this work is a decoding heuristic that receives a set of maneuvers $\mathcal{M}$ and outputs a proper sequence M satisfying predefined rules of precedences $prec(e), \forall e \in \mathcal{M}$, and providing an estimate of $T_m(M)$ and $E_{NS}(M)$. This estimate is able to properly model the availability of more than one dispatch team, and also the costs for displacing in the distribution system as maneuvers are being executed. With this decoding heuristic, both of the previous issues are fundamentally solved, and any modern optimization algorithm can employ a codification $\mathbb{X}$ that allows for efficient search mechanisms, while performing the minimization in terms of $T_m(\cdot)$ or $E_{NS}(\cdot)$.

Then, the load restoration problem was modeled as the simultaneous minimization of the recoverable energy not supplied $E_{NS}(\cdot)$ and the remaining (weighted) power not restored $S_{NR}(\cdot)$. The second contribution are some significant additions to a previous algorithm of the literature [25], namely: a Simulated Annealing algorithm followed by a Local Search refinement used to deal with the proposed formulation of the load restoration problem; and an efficient perturbation mechanism which reduces the number of possible permutations without significant losses in the exploration ability of the optimization methods. This leads to a solution methodology that is capable of returning efficient restoration plans in a very short amount of time, a critical feature for adoption in operations centers of distribution utilities.

In addition to the previous items, we proposed a pre-processing framework following the complete optimization method but considering only remote controlled switches. Given that these can be operated at essentially no time (i.e., within the no-penalties time window provided by regulatory agencies), the rationale for attempting to find an initial solution for very quick implementation is to recover at least part of the loads before they can

negatively impact the reliability indices. In this case, given that $T_m(\cdot)$ [and thus $E_{NS}(\cdot)$] has a zero value for remote switches, the proposed modeling is automatically converted into the single-objective minimization of $S_{NR}(\cdot)$ and thus the decision making is virtually immediate, which is expected to for an implementation within the available time window. As network infrastructures become more automated and remotely-operated switches more widespread, the efficacy of this initial step is expected to increase, with positive effects for both consumers and distribution utilities.

7.2 Results and discussion

For the first contribution, we solved the load restoration problem in a test system comparing the minimization of $T_m(\cdot)$ [or $E_{NS}(\cdot)$] with $N_m(\cdot)$, the latter being the current approach in the literature. The results suggest that:

- Employing multiple teams is able to significantly reduce the total time of a sequence. In fact, we saw sequences of two and four maneuvers requiring virtually the same time;
- Considering $N_m(\cdot)$ to represent the "in the shortest time possible" portion of a good restoration plan may lead to deceptive results because the costs for arriving at a switch's location and operating it are neglected;
- $N_m(\cdot)$ is not able to prefer sequences that recover more important loads in the first stages, culminating in higher energy not supplied and, thus, unnecessary impacts on the reliability indices.

With this, we have our reasons for referring to $T_m(\cdot)$ and $E_{NS}(\cdot)$ as "more realistic indices", and with the proposed decoding approach, most of the methods in the literature can be employed to optimize them instead of the (arguably) misleading $N_m(\cdot)$.

In the second part, we first showed that the proposed reduction mechanism for CO switches, **Sch2**, is able to effectively cut the possible perturbations of a given permutation vector without losing a significant number of interesting permutations. A similar analysis allowed us to settle with PS5 as the preferred perturbation scheme for the CO components.

Next, results obtained using simulated fault scenarios in a real network with 5 feeders, 703 buses and 132 switches showed that, for relatively simple scenarios, the proposed approach was able to return the exact Pareto-optimal front (validated with a Branch and Bound approach) in less than one second. In a more complex case, the proposed method

was able to outperform the Branch and Bound coupled with a pruning heuristic, both in terms of result quality and processing time, returning a set of optimized solutions in under ten minutes. Also in this scenario, we observed that about 20% of the service could be recovered using only remotely-controlled switches, a solution that could be implemented under a minute if so desired.

Finally, as a complementation to the first contribution, we notice more evidence of the benefits of considering multiple dispatch teams. In the worst case, which happens if there is only one crew or if the geographical distances judge the use of more than one team of little gain, this method leads to the same solution as the usual sequential approach. However, in other circumstances, the time savings can considerably prevent increases in the reliability indices. In any case, all situations are effectively handled by the proposed heuristic.

With all of these considerations, it can be tentatively stated that the proposed method has been shown to be an effective technique, which satisfies all properties of Table 3.1 and still improves the good points of previous algorithms with the contributions outlined before.

7.3 Proposals of continuity

There are some topics that can be further explored in a future work:

1. We can extend the proposed algorithm to effectively handle **Cases 2** and **3**, involving load shedding of in service nodes and load transfer. Notice, however, that the proposed time estimation (Figure 4.6) is already capable of handling any case as long as the adequate rules of precedences are determined.

2. Once some practical concerns about Distributed Generation mentioned in section 2.1.3.2 (specially islandings) start to be thoroughly discussed, we may be able to also expand the method for using their contribution. Nevertheless, keep in mind that DGs without islandings are already able to be modeled in the current formulation.

3. Finally, if more quality indices (such as power losses) become required for a better modeling of the load restoration problem, the decision support system may be updated with appropriate mechanisms for handling many objectives, such as the inclusion of preferences as detailed in section D.3.2.3.

Some of these points are already under investigation, and I hope to present them as part of future studies.

Appendix A

Graphs, Trees and Forests

A.1 Graphs

A *graph* is a finite collection of *nodes*, some of which are joined by *edges* [49]. Typically, we write $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ to represent a graph, wherein $\mathcal{N}$ is a set containing the nodes, sometimes called vertices, and $\mathcal{E}$ the set of edges, which are also known as arcs.

Figure A.1 shows two examples of graphs. For the first one, the node set is $\mathcal{N} = \{1, 2, 3, 4, 5, 6\}$, while the edges are $\mathcal{E} = \{(1, 2), (1, 5), (2, 3), (2, 4), (3, 4), (5, 6)\}$. Notice that, in this case, once the node set is described, the edge set is also well defined by $\mathcal{E} = \{(n_1, n_2) : n_1 \text{ is connected to } n_2; n_1 \in \mathcal{N}, n_2 \in \mathcal{N}\}$. Also, the only important information is whether a node is connected to another, that is, the *direction* of the edge is not relevant. We then call call it a *undirected graph*. Similarly, the second graph has $\mathcal{N} = \{a, b, c, d, e\}$ and $\mathcal{E} = \{(a, b), (b, a), (b, d), (c, a), (d, c), (d, e)\}$. As it may be evident, because the direction of the edge is important, we call it a *directed graph*. In the literature, the terminology "edge" is usually reserved to undirected graphs, while "arc" is used in the directed case. The first term will be used most of the time in this work, but keep in mind that in other texts both terms may be consciously employed as synonyms irrespective of the graph type.

Graphs are a mathematical abstraction of things, and it would not be surprising if these definitions also sounded too abstract. However, they constitute a very powerful tool to solve a lot of combinatorial problems. Apart from offering a convenient way of picturing a given problem, there is a large body of mathematics behind graph theory. Many important applications in the literature, such as [49, 70] shortest paths, maximum flow, minimum spanning trees etc. can be easily solved with the adoption of graphs.

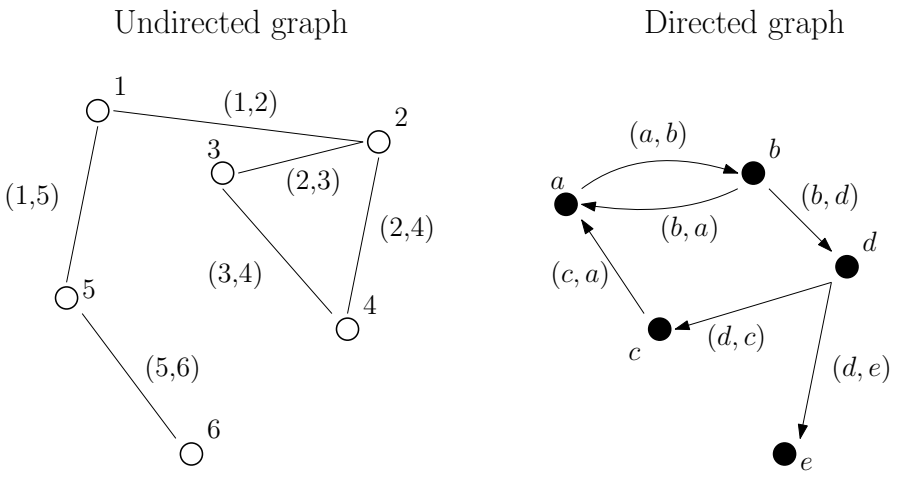


FIGURE A.1: Example of graphs. The left graph is undirected, while the right one is directed.

A.2 Trees and Forests

Graphs can be very generic. There are, however, some types of graphs that possess a structure which can be more useful for some problems. In this section the concept of *trees* and *forests* are briefly described.

Before that, here are two quick definitions:

- We call a graph $\mathcal{G}$ *connected* if we can start from any node and arrive at any other node by traversing its edges. A non-connected graph is referred as *disconnected* (compare the top panel in Figure A.2);
- A *cycle* exists in $\mathcal{G}$ if we can start at a node, walk along a subset of edges, and end up at the same node without repeating them. We call such a graph *cyclic*, and refer ot its version without cycles as *acyclic* (bottom panel of Figure A.2).

Therefore, a graph $\mathcal{G}$ is a *tree* if $\mathcal{G}$ is *connected* and *acyclic* (see equivalent definitions in [70]). The bottom right graph of Figure A.2 is a tree, as well as the top left one in Figure A.3. Essentially, the concept of tree is reserved to undirected graphs, but there is a directed concept called the *rooted tree*, in which the edges of $\mathcal{G}$ emanate from (or to) a node called *root*. This is illustrated at the top right panel of Figure A.3.

Together with the definition of a tree, we have the extension to a *forest*. It can be understood as a disjoint union of trees, and an example can be seen at the bottom of Figure A.3. Again in this case, forests are usually undirected, but it is possible that their connected components are rooted trees as seen before.

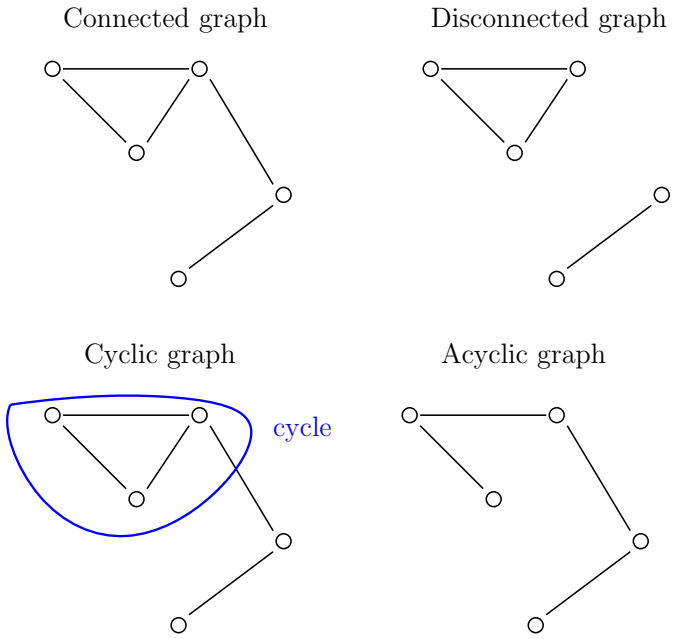


FIGURE A.2: **Top:** connected vs disconnected graphs. **Bottom:** cyclic vs acyclic graphs.

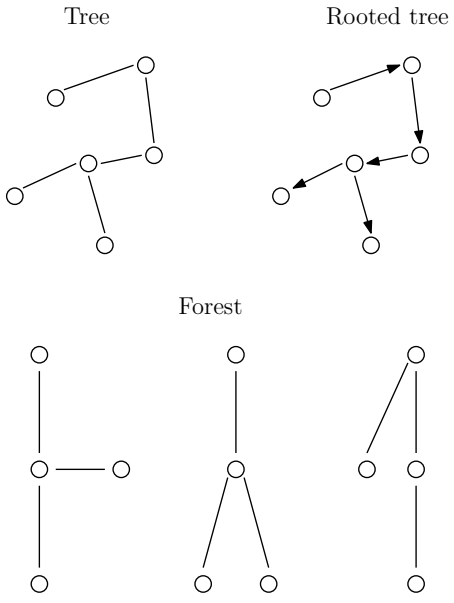


FIGURE A.3: **Top:** examples of an undirected tree (left) and a rooted tree (on the right). **Bottom:** example of a forest.

A.3 Distribution system representation

A distribution system contains a number of components, such as the substation bus, loads, transformers, voltage regulators, distribution lines, fuses, switches etc., each one with their respective intrinsic characteristics, like length, equivalent circuit, voltage, impedance, maximum current allowed etc. The required level of detail when modeling each component depends on the application [71]. For the bulk of this work, the relevant

points are the distinction between sources and loads in a feeder, and the inter-connections among them. The other constructive properties, such as line impedance/admittance, complex power of loads, bus voltages and branch currents will be relevant in the power flow solution, detailed in Appendix C.

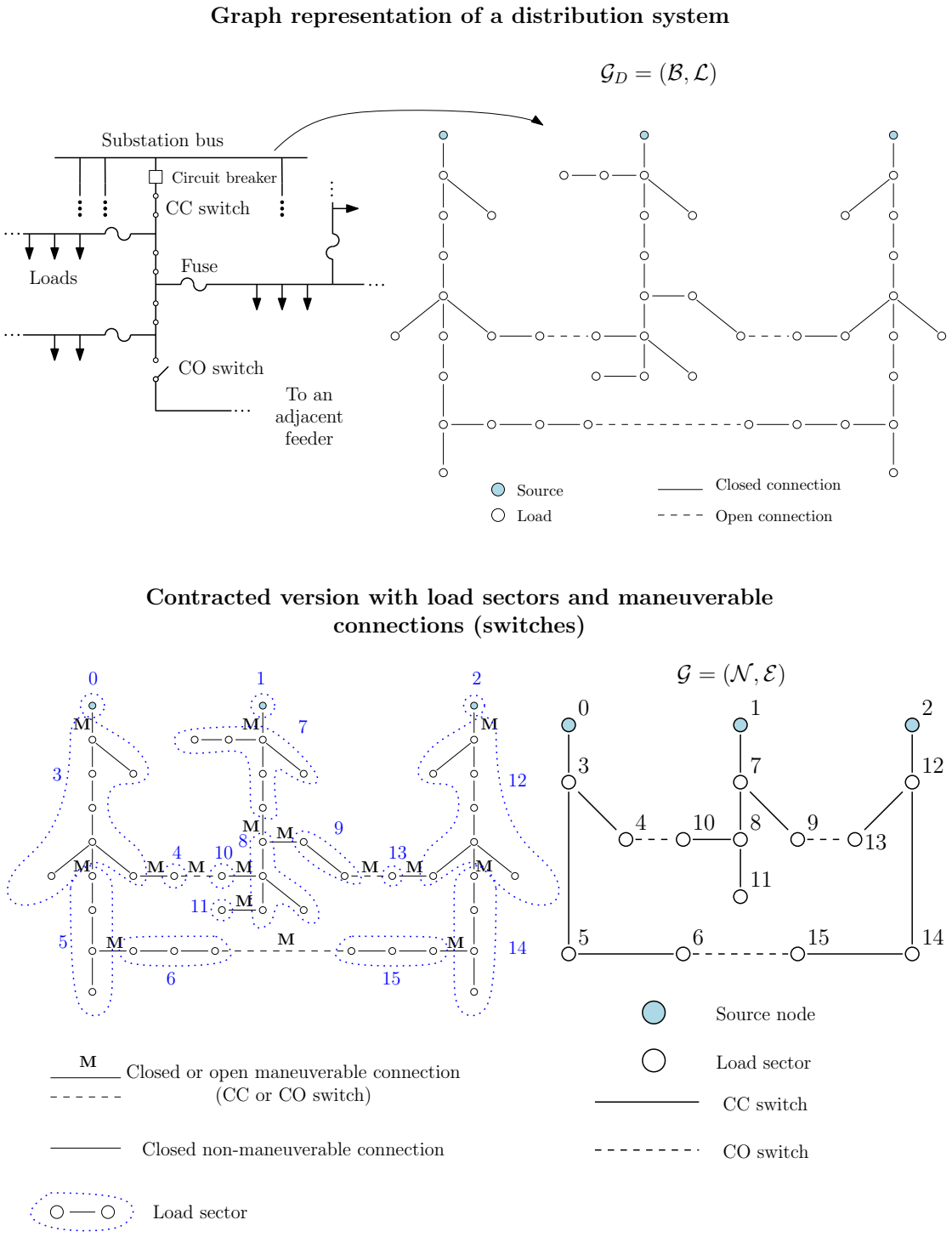


FIGURE A.4: **Top:** Graph representation of a radial distribution system. **Bottom:** A reduced (contracted) version considering only maneuverable connections (switches) and load sectors.

As shown in the top portion of Figure A.4 (repeated from the main text for convenience), the distribution network is represented by a graph $\mathcal{G}_D = (\mathcal{B}, \mathcal{L})$, with the set of nodes $\mathcal{B}$ representing system buses (which can be sources, loads or virtually any shunt element), and $\mathcal{L} = \mathcal{L}^{CC} \cup \mathcal{L}^{CO}$ the complete set of branches (comprehending distribution lines, transformers, phase-shifters, or any element connecting two buses). In this notation, $\mathcal{L}^{CC}$ is the set of all elements that are currently in operation, that is, currently closed (CC), and $\mathcal{L}^{CO}$ contains the set of components that are disconnected, i.e., currently open (CO), mainly maneuverable switches. Notice that the present configuration is completely described by the closed connections, but the open ones are also illustrated in a dashed version for completeness.

While this representation is already simpler and more useful than handling the original scheme of a distribution system, since the reconfiguration process used in the load restoration problem requires the knowledge of the connections that can be opened or closed, that is, lines with maneuverable switches, an even better representation will be the *reduced* or *contracted* one shown at the bottom of Figure A.4. The contracted graph is given by $\mathcal{G} = (\mathcal{N}, \mathcal{E})$, where the edges $\mathcal{E} = \mathcal{E}^{CC} \cup \mathcal{E}^{CO}$ indicate only branches with switches, and the terminology is similar as before, and the loads without any maneuverable connection among themselves were grouped (contracted) into a single node called sector, and $\mathcal{N}$ is the corresponding set of all sectors. Therefore, the meaning of a node and an edge will depend on which system representation we use. Fortunately, the reduced version will be useful during the bulk of the work, which is why I reserved the more familiar notation to it. However, for power flow considerations, the original non-reduced graph will be required, and so the corresponding graph notation will be adopted in Appendix C.

A.3.1 Radial constraint formulation

Oftentimes, during the restoration process, we reconfigure the original network by following a sequence of maneuvers M, which generates a new graph $\mathcal{G}^{CC}(M) = (\mathcal{N}, \mathcal{E}^{CC}(M))$, with $\mathcal{E}^{CC}(M) \subseteq \mathcal{E}$ indicating a (possibly) different set of closed switches. Notice the use of the subgraph $\mathcal{G}^{CC} \subset \mathcal{G}$, which contains only the CC switches as the complete graph $\mathcal{G}$ already contains all connections.

It is important, hence, to enforce the radiality constraint, or else M will generate a unfeasible configuration. The condition that $\mathcal{G}^{CC}(M)$ is a forest is not enough, as exemplified in Figure A.5. In this example, even thought $\mathcal{G}^{CC}(M)$ is a forest (actually, it is a tree, but, from the definition, a tree is also a forest), there is a path from each load to more than one source, which does not characterize a radial system.

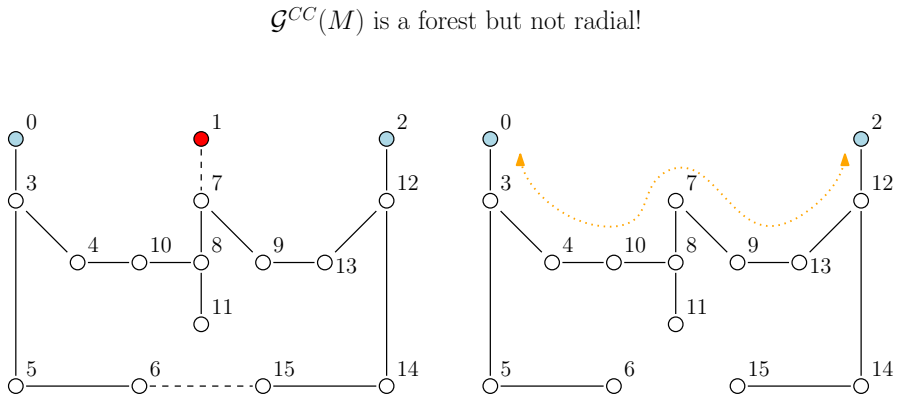


FIGURE A.5: Just enforcing that $\mathcal{G}^{CC}(M)$ is a forest is not enough. In this example (complete graph on the left and only the CC arcs on the right), the loads are energized by more than one source.

There can be various ways of formulating the radial constraint. In this work, it is proposed to write it as

$$\mathcal{G}^{CC}(M) \text{ is radial} = \begin{cases} \mathcal{G}^{CC}(M) \text{ is a forest} \\ \text{Each connected component of } \mathcal{G}^{CC}(M) \text{ has at most one source} \end{cases}$$

To a get a mathematical formulation, assume $\mathcal{G}^{CC}(M)$ has m connected components, such that $\mathcal{N} = \mathcal{N}_1 \cup \mathcal{N}_2 \cup \ldots \cup \mathcal{N}_m$. Also, let $\mathcal{N}_s \subset \mathcal{N}$ be the set with the source nodes. The second information is available beforehand, while the first can be obtained by simple algorithms like breadth first search or depth first search.

Once $\mathcal{G}^{CC}(M)$ is constructed, let

$$a_{(n_1,n_2)} = \begin{cases} 1, & \text{if } (n_1,n_2) \in \mathcal{E}^{CC}(M) \\ 0, & \text{otherwise} \end{cases}$$

According to [72], a forest can be described with the relations:

$$\sum_{(n_1,n_2)\in\mathcal{E}^{CC}(M)} a_{(n_1,n_2)} = |\mathcal{N}| - m \tag{A.1}$$

$$\sum_{\substack{(n_1,n_2)\in\mathcal{E}^{CC}(M) \\ n_1\in\mathcal{N}_{sub},n_2\in\mathcal{N}_{sub}}} a_{(n_1,n_2)} \leq |\mathcal{N}_{sub}| - 1, \quad \forall \mathcal{N}_{sub} \subset \mathcal{N}, \mathcal{N}_{sub} \neq \mathcal{N}, \mathcal{N}_{sub} \neq \emptyset \tag{A.2}$$

Each tree must have at most $n-1$ edges, with n its number of nodes. Extending this to a forest with $|\mathcal{N}|$ nodes and m connected trees, we get constraint (A.1). The second

constraint comes from the so called subtour elimination constraints, which prevents cycles in any subset $\mathcal{N}_{sub} \subset \mathcal{N}$. There are actually three of them for forests in general but, according to the discussion of [72], these can be expressed as the single constraint (A.2).

The second condition for radiality can be simply expressed as:

$$|\mathcal{N}_s \cap \mathcal{N}_i| \leq 1, \forall \mathcal{N}_i \in \mathcal{N} \tag{A.3}$$

which guarantees that the intersection of each connected component $\mathcal{N}_i$ with the set of sources $\mathcal{N}_s$ is at most one, that is, there is no more than one source per component.

Combining equations (A.1) to (A.3), we get the radiality constraint as:

$\mathcal{G}^{CC}(M)$ is radial if

$$\begin{cases} \sum\limits_{(n_1,n_2)\in\mathcal{E}^{CC}(M)} a_{(n_1,n_2)} = |\mathcal{N}| - m & \\ \sum\limits_{\substack{(n_1,n_2)\in\mathcal{E}^{CC}(M) \\ n_1\in\mathcal{N}_{sub}, n_2\in\mathcal{N}_{sub}}} a_{(n_1,n_2)} \leq |\mathcal{N}_{sub}| - 1, & \forall \mathcal{N}_{sub} \subset \mathcal{N}, \mathcal{N}_{sub} \neq \mathcal{N}, \mathcal{N}_{sub} \neq \emptyset \\ |\mathcal{N}_s \cap \mathcal{N}_i| \leq 1, & \forall \mathcal{N}_i \in \mathcal{N} \end{cases} \tag{A.4}$$

One last observation: since energy flows from the source to the loads, a radial system would probably be best represented by a directed graph instead of a undirected one as used here and in the rest of the text. However, some operations, like the closing of a CO switch, would require us to invert the direction of an edge[1], which, rigorously, change the complete graph. Fortunately, given the simplicity of the examples used in this text, the simple knowledge of the sources allows us to readily determine the correct direction of energy in basically any configuration. Besides, the addition of arrows in the graphs would probably offer more clutter than actual relevant information. Therefore, the undirected representation was adopted in this whole text.

[1] See Figure A.5. In edge (7,8), for instance, energy originally flowed from 7 to 8, but, with the closing of (4,10) [and ignoring (9,13)], the energy flux is not reversed.

Appendix B

Protection of Distribution Systems

Most readers are probably familiar with Murphy's law, usually stated as "Anything that can go wrong, will go wrong". There seems to be enough evidence to suggest an adaptation for power systems: "If the electrical system can fail, it will fail". The system is constantly subject to [4] natural effects, like storms, blizzards, hurricanes, tree branches; human errors; malfunctions of protection devices; and also the insulation is subject to failures, mainly because of ageing, temperature and chemical pollution. With all of these (generally unpredictable) factors, it is impractical (if not impossible) to devise a power system which will never fail [73].

Given that faults will inevitably occur, a more practical job of a protective system is to minimize their impacts. These can range from [7] overcurrents, which may overheat the equipment and reduce their useful life, to possibly fire and explosion, putting at risk the safety of the personnel. We mitigate the impacts of these faults by [5, 73] foreseeing any possible effects that may cause a long-term outage and quickly isolating the faulty element(s) from the rest of the system, limiting the disturbance to an area as small as possible.

Faults are normally classified into temporary (or transient) and permanent [73, 74]. The first type usually occurs when phase conductors are momentarily connected to other phase conductors or the ground, usually due to trees, animals, storms, lightning etc. Temporary faults can be cleared by briefly interrupting service in order to eliminate the power arc, and then the system can be re-energized normally. The second type involves permanent damage to the insulation, and thus require adequate repairs to replace burned-down conductors, blown fuses or other damaged equipment; remove tree limbs from the line; manually reclose a circuit breaker or recloser; etc. Notice that a

temporary fault that is not treated in time can evolve into a permanent one. The load restoration problem is applied when the second type of interruptions take place.

In order to operate correctly, the most important protective devices adopted in distribution systems are [13]:

Fuses: it acts on its own by self-destructing in case of overcurrent;

Circuit breakers: switching devices able to carry enormous currents, so unlike fuses they interrupt circuits based on feedback from relays;

Reclosers: an equipment that automatically trips and recloses a pre-determined number of times in order to clear temporary faults;

Line sectionalizers: a device that automatically isolates a faulted section of a distribution circuit once an upstream breaker or recloser has operated.

Complementing these equipment, there are:

Voltage and current transformers: they monitor and provide accurate feedback about the healthiness of a system;

Relays: convert the signals from the previous monitoring devices and provide instructions to breakers to trip under faulty conditions;

DC batteries: supply uninterrupted power to relays and breakers to operate independently on the main power system.

Figure B.1 illustrates a portion of a feeder with some of these components. It is common to place a fuse to protect a distribution transformer, either externally (as shown) or internally inside the transformer tank [13]. A recloser or a circuit breaker with reclosing relays is normally installed at the substation, and it can be used to clear temporary faults. More of these devices can be placed along the feeder for protection coordination (see next). Finally, for completeness, regular switches (with no protective function, used only for reconfiguration) are also illustrated. Notice that, if required, breakers and reclosers can also be employed as switches for the load restoration problem.

The interested reader can find operating details of each component in references such as [13, 73, 74]. For the purposes of this chapter, what matters is that the protection is appropriately *coordinated*, which will be described next.

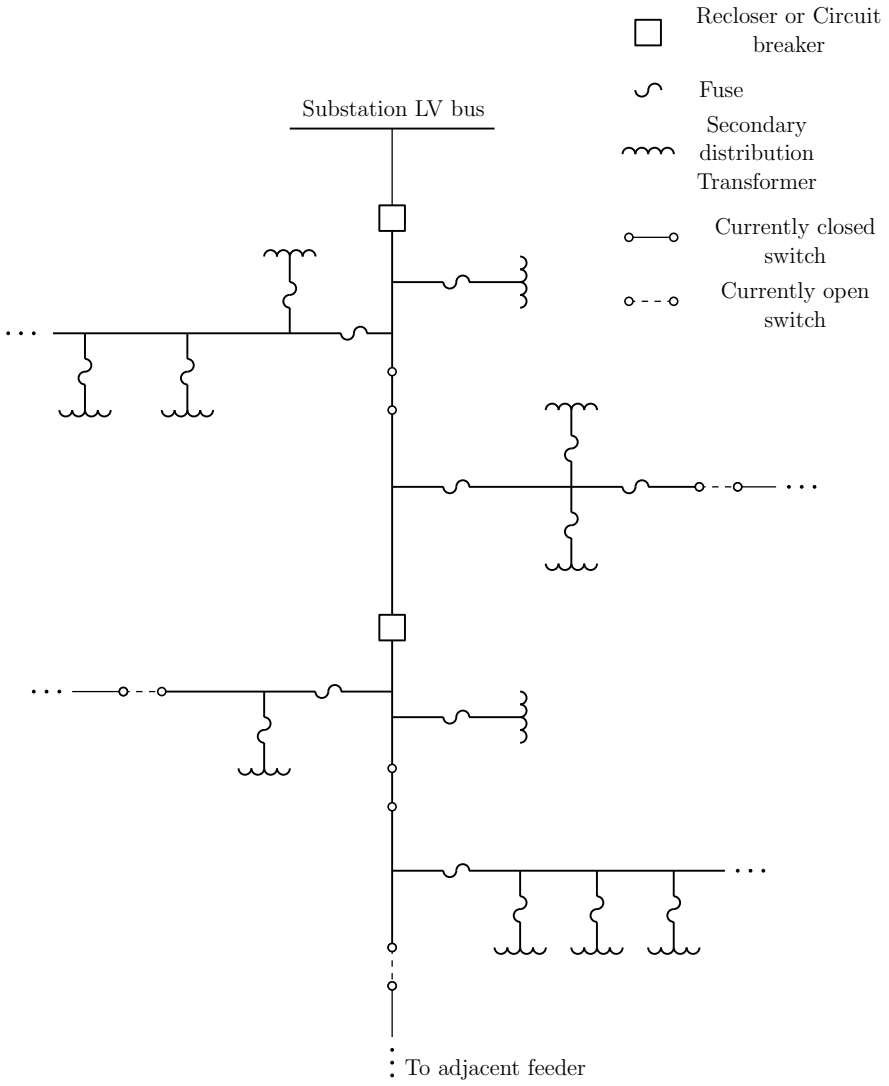


FIGURE B.1: Typical disposition of protection equipment in a distribution feeder. For completeness, regular switches with no protective function are also shown.

B.1 Protection coordination

Under normal load conditions, we require the protection equipment to not operate. Whenever there is a fault, in most situations, there is usually a build up of current which is felt in the whole system. In this case, it is desirable that [7] only the closest protection equipment is activated in order to remove the smallest number of customers from service. In case this device malfunctions, the next closest one should act.

This general description can be made more clear by looking at the radial system of Figure B.2, in which the protective equipment (which can be any of the previously discussed) P_a to P_d are able to sense overcurrents and trip if required. Suppose there is a fault in a point of the line protected by P_c. We require this device to be triggered to de-energize the line until the failure is corrected. We call this a *primary* protection and,

in this case, the load fed by bus 4 becomes out of service. If, for some reason, P_c fails to activate, we may have P_b to trip instead. This is called a *back-up* protection, and we usually need to make sure that it does not happen unless there is no other way. The reason for this is that, by triggering P_b, not only bus 4 is de-energized, but also 3 and 5, which have no initial connection to the fault.

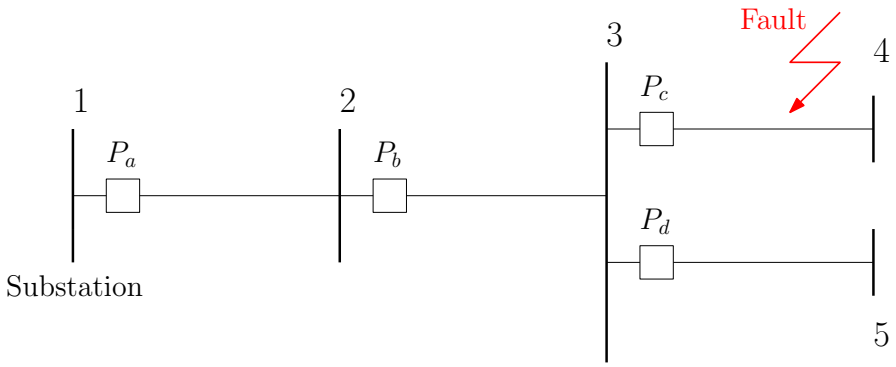


FIGURE B.2: Example of protection coordination. The protective equipment P_a through P_d should not operate under normal conditions. In case of a fault right to P_c, only it should operate. If it fails to do so, the back-up P_b should take its turn.

At first, achieving this condition may be hard because the effects of a fault can be felt by many other portions of the system, and thus any device may be operated virtually at random. Fortunately, overcurrent protective equipment usually follow an inverse relationship between time and current, meaning that the bigger the current magnitude, the faster it operates. Therefore, by either employing different fuses in distinct portions of the system or adjusting internal parameters of relays, it is possible to set proper activating times for each device. With this, it is possible to arrange the system such that back-up devices should trip *after* the primary ones. For instance, in Figure B.2, we can organize P_d and P_c to be fastest to trigger, then P_b followed by P_a. In the literature [75], this constraint is formulated by assuring that the time t_{backup} for the back-up protection is greater than the time $t_{primary}$ of the primary by at least an amount Δt, that is,

$$t_{backup} - t_{primary} \geq \Delta t \tag{B.1}$$

in which Δt is referred as *coordination time interval*, or CTI. We call this process *protection coordination* and, whenever equation (B.1) is satisfied for all pairs primary/back up, we say that the protection system is coordinated. A properly coordinated system is important for not only preventing disconnecting healthy portions of the system, but also in helping locating the fault faster. For instance, incorrectly tripping P_b instead of P_c in Figure B.2 requires an unnecessary search in lines $(2, 3)$ and $(3, 5)$ apart from $(3, 4)$.

Sure, this process is easier said than done. In general meshed networks (such as transmission systems), the primary/back-up classification are not necessarily evident, and may depend on a given fault. For example, if there was a source in bus 4 in Figure B.2, different faults may generate fault currents at distinct directions, and thus the roles of devices P_b and P_c may be reversed. This problem was and still is extensively studied in the literature, mostly with coordination of relays, and the interested reader can check some references such as [75–79].

Regarding radial distribution systems, the coordination requirement is generally easier to achieve because current flows in only one direction, and thus the primary/back-up designation is straightforward. Of course, it also demands a proper selection of fuses and relay settings[1] [13], but it still is a less complicated process.

B.1.1 Coordination in the presence of Distributed Generation

Despite the benefits of DGs as mentioned in the main text (section 2.1.3.2), the increase in their penetration is slowly making distribution systems more similar to transmission networks where generation and loads are mixed, and this reflects in the necessity of more contrived protection coordination designs [80]. Some drawbacks of DGs include [80–82]

- Increased fault currents, which becomes more alarming as units grow or more generators are connected to the system;
- The direction of power flow may be changed.

Both of these effects can cause miscoordinations to initially well-organized devices, requiring updates in the device settings every time a unit is connected or disconnected.

According to some works published about this topic [80, 81, 83], it seems that the best practice for obtaining a coordinated protection in the presence of DGs is to adopt directional relays and reclosers (or just activate this function in the already installed devices) and replace fuse sizes. In the end, the network is treated like regular meshed systems, and the pertinent approaches for obtaining appropriate relay, recloser and fuse settings are still being developed in the literature. This problem is completely outside the scope of this work, and thus we assume an adequately coordinated protection, regardless of the presence or absence of DGs.

[1] Also, some specific guidelines should also be followed. For example, since fuses do not have a reclosing option and need to be replaced after tripping, they are normally set as back-up for circuit breakers and reclosers.

Appendix C

Power flow in distribution systems

The load restoration process followed in this work requires the testing of many different configurations of a distribution network in order to return the non-dominated solutions with respect to the adopted quality indices. An important step in this process is to ensure that each trial configuration is feasible, which, apart from radiality, requires the knowledge of the voltages and currents in many different parts of the system. The procedure of computing these quantities is known as *power flow* or *load flow* problem [7, 84, 85], and for the bulk of this work it was adopted as a black box capable of receiving a configuration and returning a Yes/No regarding its ability to satisfy the operating constraints. In this appendix the complete power flow algorithm will be detailed.

There is a number of methods used to solve this problem, with the most famous probably being [7, 84] Gauss-Seidel, Newton-Raphson and the Decoupled Load flow. While they may work well in transmission systems, it has been repeatedly stated that they lose efficiency and may even diverge in distribution systems [71, 86, 87]. More specifically, these "classic" techniques may have trouble in cases when the system [88]:

- is multi-phase and has an unbalanced operation and unbalanced distributed load;
- has an extremely large number of edges and nodes;
- has a large resistance to reactance ratio R/X in the lines, preventing the use of a decoupled analysis.

As can be seen, we just described a distribution system. Fortunately, there are methods better suited for this case, notably for radial structures, such as the *ladder* or *Backward-Forward* [71] and the Y-matrix method [85]. One such algorithm will be proposed in this appendix.

Two important observations before proceeding:

- We are interested in the steady-state conditions of the systems, which means that any possible transients (due to faults, for instance) have already settled. This allows to represent the time-varying quantities of the system as phasors [89]. For example, the voltage $v_b(t)$ at a bus b will be indicated by the phasor $\tilde{V}_b$, which is independent on the time.
- A graph representation of the power system must indicate all buses and the branches connecting them, not only the maneuverable connections. Therefore, while the contracted graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ was employed for most of this work, the power flow problem needs the complete version $\mathcal{G}_D = (\mathcal{B}, \mathcal{L})$. In that case, the terminology "nodes" and "edges" are now synonyms of "buses" and "branches", respectively.

C.1 System modeling

Despite considering distribution networks simply as a set of nodes interconnected by edges, a real system is composed of [85] transmission/distribution lines, transformers, phase shifters, generators, loads, shunt elements etc. When analyzing an electrical distribution system, each individual component must receive a mathematical representation that approximates its physical behavior. Such representation is referred to as a *model*, and there can be multiple modelings for a given element [7, 71]. The choice of which component to model and how to perform it depends upon the analysis to be performed. For instance, the bulk of this work required basically the distinction between generation and consumption units (sources and loads); the available connections among them and their current states (open or closed); and a flag indicating whether the configuration satisfies the operating constraints or not. Therefore, considerations such as presence/absence of reactors or shunt capacitors, modeling of loads, core losses in transformers, line lengths or impedances etc. were completely abstracted into a black-box, allowing such a complex system to be reasonably described by the graph representation used throughout the text.

The components of an electrical system can be roughly classified into two groups [90]: the ones connected to a bus and the reference (ground) one, such as generators, loads

and shunt elements; and the ones connected between two buses and compose a branch, including transmission/distribution lines, transformers and phase-shifters. The pertinent models for these elements will be described next.

C.1.1 Node elements

For the purposes of this work, the important parameters of a node b are the voltage $\tilde{V}_b$ and the *net* injected current $\tilde{J}_b$, which is the difference between what is injected by a source and what is drained from a load or shunt element.[1] Since each quantity is a complex number which can be expressed by either its magnitude/phase or real/imaginary portions, this accounts for four variables in each node. Depending on which variable is given, nodes are usually classified into three types:

PQ The complete characteristics of a load are provided, usually its active P_b and reactive power Q_b. In this case, the complete voltage $\tilde{V}_b = |\tilde{V}_b|\angle\theta_b$ should be computed. In this work this denomination is extended even for other types of loads in which other variable (such as current or impedance) is given instead of the power demand;

PV It normally represents generators, in which the voltage magnitude $|\tilde{V}_b|$ and the injected real power P_b are known. The goal is to determine θ_b and the reactive power Q_b;

slack node It also represents a generator, but one to be chosen as a reference for the voltage angle, so that $\tilde{V}_b = |\tilde{V}_b|\angle\theta_b$ is completely known, usually with $\theta_b = 0°$. In this case, by the Tellegen's Theorem [91], its injected current (or power) can be computed such that the overall injected and demand balances to zero. Thus, this node takes up the slack (hence its denomination) for the other nodes.

In some applications, other classes of nodes can arise, such as PQV, P and V [90], but the presented classification is enough for most systems.

C.1.1.1 Loads

Loads are possibly the most important portion of the distribution system. During the main text, they were treated merely as nodes with an associated complex power which became a white circle or a gray square depending on whether or not they were connected

[1] In normal references, the injected complex power S_b^c is used instead of $\tilde{J}_b$. This is probably because loads are normally modeled as constant power demands (see section C.1.1.1), which is not assumed in this work. Keep in mind that the injected current can be easily computed even if only S_b^c is provided.

to a source. While this definition sufficed for those purposes, we need more background on where this complex number comes from and why it is justified.

First, what exactly is a load? Well, a toaster is a load, just as a computer, a shower, a light bulb and a cellphone charger. But this is not restricted to individual equipment, and so a house, a building, a military base, an oil refinery and an entire city can also be considered a load [92]. Therefore, the precise meaning of this term depends on what part of the power system we wish to focus,[2] which is important when deciding for an appropriate load model due to its time and voltage dependency [93].

Time dependency of loads

Every time a battery charger is unplugged, someone enters the shower or the street lights are turned on, the load seen by the distribution feeder changes. This ever-changing characteristic makes it difficult to settle for a specific load value to be used in the power flow problem.

Let us consider the toaster as a load. Plugging it into the power outlet means applying a voltage $\tilde{V}$ into its terminals. If we measure the current $\tilde{J}$ flowing, we have an associated complex power $S^c = \tilde{V}\tilde{J}^*$ that can be used as a measure for this load.[3] Thus, we can say that the toaster has a load equal to S^c. However, most people (at least the responsible ones) do not leave such electrical appliance turned on during the entire day; a typical scenario would be to turn it on twice or three times a day for a couple of minutes (depending on the size of the family or how many friends went for a surprise visit). Therefore, the *demand curve* for a toaster in a given day would be the one shown at the top panel of Figure C.1. The *demand* of a load can be defined as [71] the average of the instantaneous load during a specific time interval, such as 15 min, 30 min, 1 hour etc. To facilitate the analysis, the demand curves in Figure C.1 use a 2-hour interval, so each bar represents the instantaneous load averaged during 2 hours. This interval might by too long in applications, but it serves for the purposes of this discussion.

Of course, distribution systems are not constructed or planned so we could enjoy a crispy bread twice a day, and there are different equipment that would be used in diverse moments, during distinct time intervals and with varied complex powers. Hence, the demand curve for a house, for instance, would look like the middle panel in Figure C.1.[4]

[2] If we stretch this comment a little more, we can consider even distribution systems as loads when dealing with the transmission system.

[3] Other measures can also be used [71], such as the complex power magnitude, also called the apparent power $|S^c|$, the current magnitude $|\tilde{J}|$ or even the active power (the real part of S^c) in some cases. Conventional load units are kVA, kW or A, but this work deals only with *per unit* (pu) to simplify the analysis.

[4] Sure, customers with different habits, such as nocturnal workers, people who work at home etc. would present a distinct demand curve.

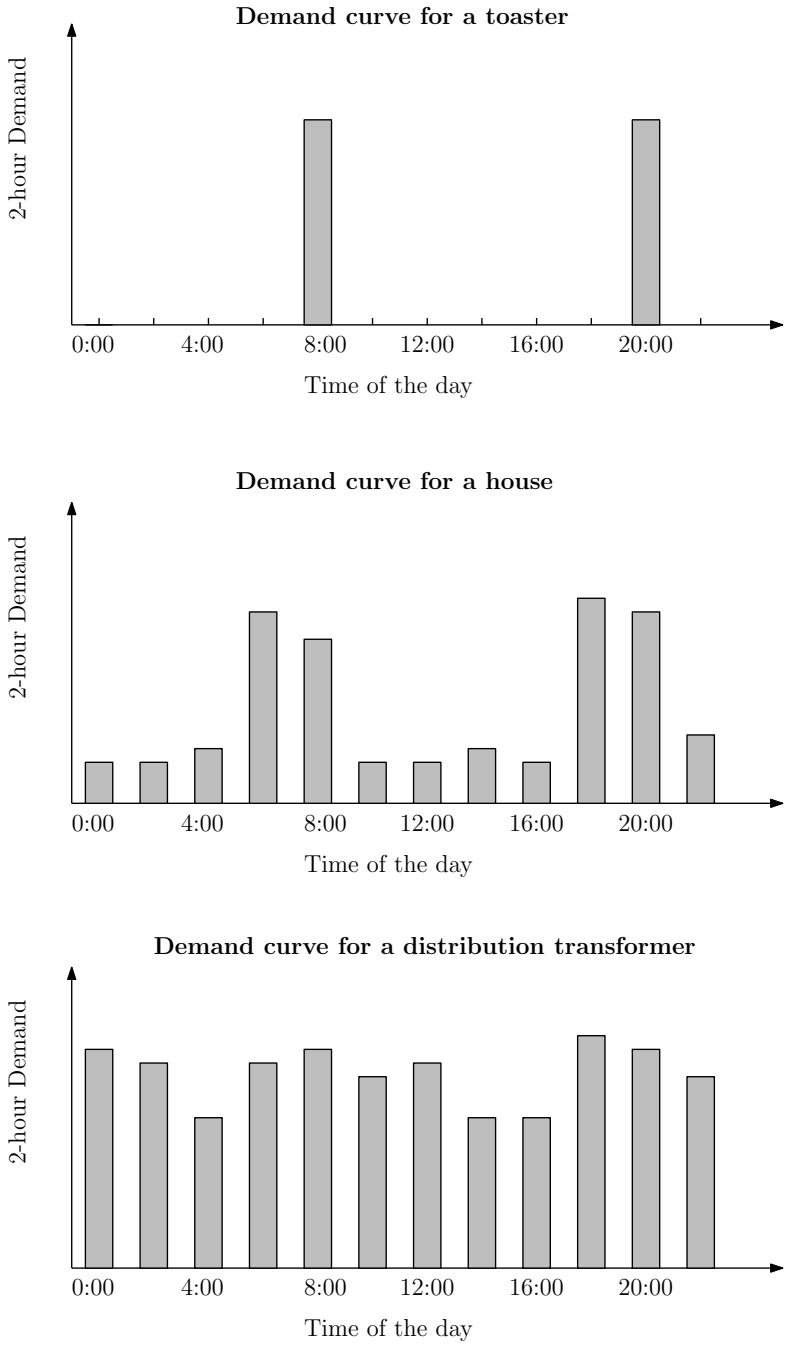


FIGURE C.1: 2-hour demand curves for some loads. **Top:** for a toaster being used twice a day. **Middle:** a house where the residents wake up around 6:00 am, go to work and come back around 6:00 pm. **Bottom:** a distribution transformer, which connects a lot of customers. Notice that these curves do not correspond to any specific real system, and they are only meant to be used as example.

Finally, when we consider a transformer which connects the primary distribution system with the secondary one and thus energizes a large number of customers, the demand curve may look like the bottom of Figure C.1.

Notice that as we get distant from individual equipment the demand curves tend to

smooth out [71]. The simple explanation for this is, by considering hundreds of customers, the odds are good that someone enters the shower as another just leaves it, or someone wakes up when another is going to sleep. Because this work deals with primary distribution, the loads are normally large, such as industries and secondary distribution system, which tend to follow the bottom curve of Figure C.1. Therefore, the approach followed here is to adopt as load number for a node the peak value of its demand curve in a given weekday and consider it fixed during the entire load restoration process. The smoother characteristic when compared to more residential loads prevents this "worst case scenario" from being too abnormal and conservative, and thus the approach is justified.

Voltage dependency of loads

For some loads it is possible to use metering equipment to determine its complex power requirements, such as in secondary distribution transformers. This modeling is referred to as *measurement-based* [93]. However, in other cases (specially for industrial loads), it is better to use a *component-based* modeling, in which each load type is tested in order to determine the relationship between its complex power *versus* the applied voltage[5] [93]. The following models are normally employed [94]:

Constant power: the load at a node b demands a fixed power S_b^D, which is independent on the applied voltage;

Constant current: the load drains a fixed current $\tilde{J}_b^D$ regardless of the voltage. The power varies linearly with the voltage;

Constant impedance: the load can be modeled by a fixed impedance Z_b^D or admittance Y_b^D. The power depends on the square of the voltage;

Exponential load: the power S_b^D at a node b varies with the voltage magnitude $|\tilde{V}_b|$ according to [95]

$$S_b^D(\tilde{V}_b) = P_{b,0}\left(\frac{|\tilde{V}_b|}{V_{b,0}}\right)^p + \jmath Q_{b,0}\left(\frac{|\tilde{V}_b|}{V_{b,0}}\right)^q \tag{C.1}$$

in which $P_{b,0}$ and $Q_{b,0}$ are, respectively, the active and reactive powers at the nominal voltage $V_{b,0}$, and p and q are exponents that depend on the equipment. Some values for typical loads (such as battery chargers, lamps etc.) are provided

[5] In the case of a *dynamic* category, which is required in, e.g., transient analysis, the frequency should considered as well [93]. For the power flow problem, which uses the *static* category and considers the steady state characteristics, the frequency can be assumed constant.

in Table 1 of [95]. Notice that this type of load generalizes the previous ones; specifically, $p = q = 2$ reduces to constant impedance, $p = q = 1$ to constant current, and $p = q = 0$ to constant power.

Polynomial load: sometimes a load is not well represented by a single model. A composition of the previous models leads to the polynomial one [95]:

$$S_b^D(\tilde{V}_b) = P_{b,0}(a_0 + a_1|\tilde{V}_b| + a_2|\tilde{V}_b|^2 + a_3|\tilde{V}_b|^p) + \jmath Q_{b,0}(b_0 + b_1|\tilde{V}_b| + b_2|\tilde{V}_b|^2 + b_3|\tilde{V}_b|^q) \quad \text{(C.2)}$$

wherein $P_{b,0}$ and $Q_{b,0}$ have the same meaning as before, $a_0 + a_1 + a_2 + a_3 = b_0 + b_1 + b_2 + b_3 = 1$, and

- a_0 and b_0 are parameters for the constant power component;
- a_1 and b_1 for the constant current portion;
- a_2 and b_2 for the constant impedance term;
- a_3 and b_3 refer to the exponential load part.

While in most cases the loads are modeled as constant power [71], specially with the measurement-based type, it is important for the power flow algorithm to be able to handle any of the presented models. This work employs an approach known as *current-injection* model [88], which converts the loads into equivalent current sources as shown in Table C.1. Notice that the constant power, exponential and polynomial loads have the complex power as a function of the voltage, so their representations are equal. These models can be extended to the three-phase case, and the interested reader can check [71] for details.

Model	**Equivalent** $\tilde{J}_b^D$
Constant impedance	$\tilde{J}_b^D = \tilde{V}_b / Z_b^D = \tilde{V}_b Y_b^D$
Constant current	$\tilde{J}_b^D$ given
Constant power Exponential load Polynomial load	$\tilde{J}_b^D = \left(\frac{S_b^D(\tilde{V}_b)}{\tilde{V}_b}\right)^*$

TABLE C.1: Equivalent current sources for the current-injection model for each load model for a given phasor voltage $\tilde{V}_b$. In case of constant power, $S_b^D(\tilde{V}_b)$ is a constant.

C.1.1.2 Shunt elements

Capacitors and reactors are normally employed to control reactive power in some loads[6], improving the power factor and thus alleviating the voltage drops [7]. An obvious appropriate model for such elements is as a shunt admittance, as shown in Figure C.2. In most cases, their contribution can be added as a constant impedance load.

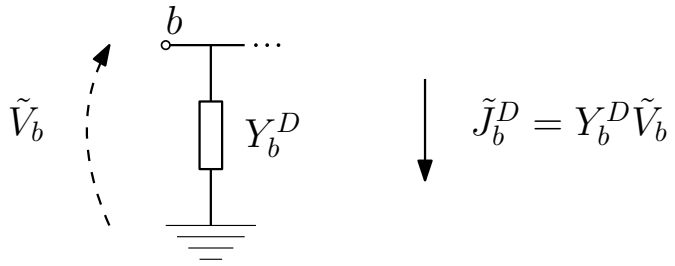


FIGURE C.2: A shunt element at node b can be modeled as a constant admittance connect between b and the reference node.

C.1.1.3 Generators

Assuming steady state, generators can be represented simply as voltage sources [85], and since their excitation system controls the magnitude of the terminal voltage, normally we can assume $|\tilde{V}_b|$ and the injected real power P_b provided for a node b with a generator. As explained before, one of the generators in a system is assumed to be a slack node (and its phase can be arbitrarily set to 0°), while the others receive the PV denomination. In the distribution context, the output buses from substation are usually set to be slacks. This holds even for multiple feeders because in normal operation they are disconnected among then, so it is possible to set multiple reference sources independently.

Regarding distributed generation (DG), depending on the control mode, a DG unit can be set to output power at constant power factor (normally for smaller generators) or constant voltage magnitude (for larger ones) [3, 96]. In the first case, since both the active and reactive power is known, a DG can be modeled as PQ node with a constant injected power S^G [3]. In the second case, the PV classification is more appropriate.

Notice that we assume the system prevents islanding with DGs because of the issues and dangers it can cause (section 2.1.3.2). The analysis in this case becomes much more complex as the frequency is no more constant, and the phasor manipulations may not be valid. Check [97] for an initial idea of how to handle such situations.

[6] In some cases capacitors are connected in series to lines to reduce their impedance, and the appropriate model in this case follows immediately.

C.1.1.4 Unified node model

With all appropriate components described, any node can be represented by the unified model of Figure C.3. It consists basically of all possible load types (a shunt admittance, a constant current demand, and a power demand, which can vary with the voltage as in the exponential and polynomial models) and a complex power injection to model generators. By using the current injection model, all of these components can be described by a *net current injection* $\tilde{J}_b$, computed as

$$\tilde{J}_b = \left(\frac{S_b^G - S_b^D(\tilde{V})}{\tilde{V}_b} \right)^* - \tilde{J}_b^D - Y_b^D \tilde{V}_b \tag{C.3}$$

in which the complex power demand is taken as a function of the voltage to generalize the loads of Table C.1. In regular systems with constant power load, S_b^D is also a constant. Notice that any of the previous elements can be modeled by making the appropriate parameters equal to zero.

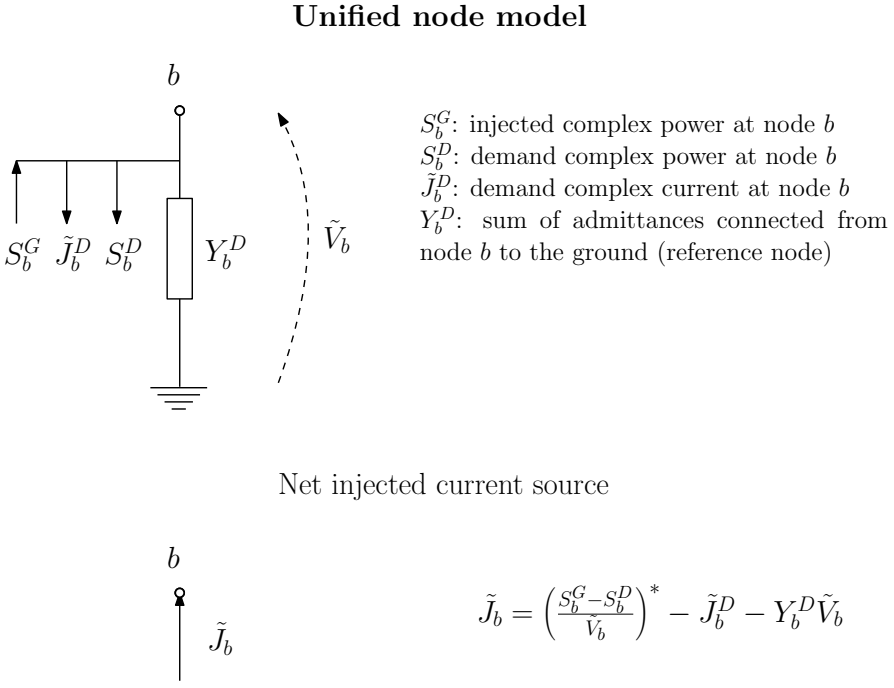


FIGURE C.3: Unified node model used in this work. All pertinent parameters are described here.

C.1.2 Edge elements

C.1.2.1 Distribution lines

A typical (transmission or distribution) line can be characterized by four distributed parameters, which are dependent on the line geometry [7, 89]: series resistance R', to account for the losses by Joule effect; series inductance L' and shunt capacitance C', to represent the effects of the electric and magnetic fields around the conductors; and a shunt conductance G' to account for leakage currents along insulators strings and ionized pathways in the air. Using these parameters, a power line can be modeled by differential sections of the equivalent circuit shown in Figure C.4, where all quantities are in a convenient unit per-length. This model leads to the well-known *telegraph equations* [7, 89].

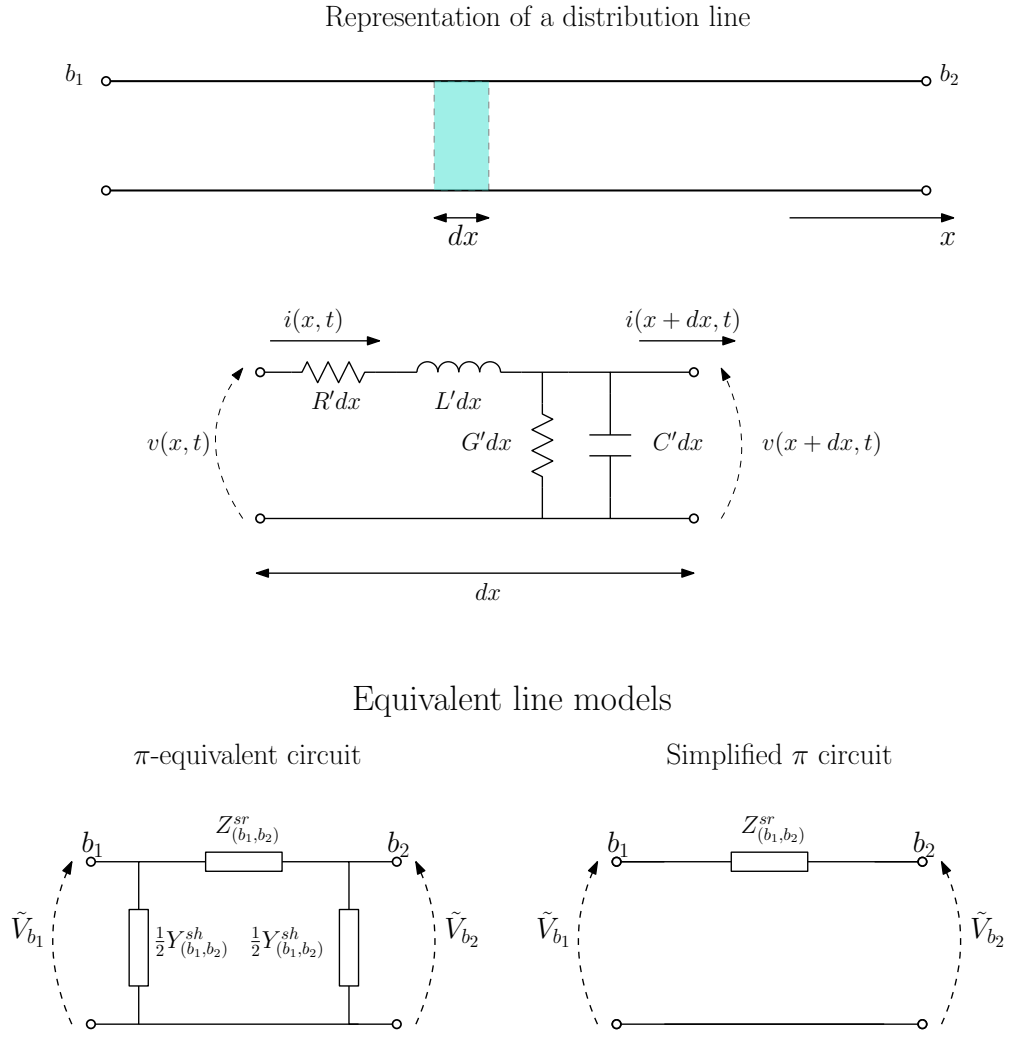


FIGURE C.4: Modeling of a transmission or distribution line by differential lumped-parameter circuits. These circuits can lead to a π-equivalent model, or even a simplified version which is frequently used in short distribution lines.

The telegraph equations are in general partial differential ones since the voltage and current are functions of the time t and the position x. But since we are interested in the steady state conditions, we adopt phasor quantities to replace $v(x,t)$ with $\tilde{V}(x)$ and $i(x,t)$ with $\tilde{I}(x)$ and remove the time dependency. The resulting equations provide the voltage and current for any position in the line.

For engineering purposes, it is more interesting to have a circuit-based equivalent model for the whole line relating the voltages and currents at each end. One such model is the π-equivalent circuit [7] shown at the bottom left of Figure C.4 where, for a typical branch $\ell = (b_1, b_2)$, it has a series impedance $Z^{sr}_{(b_1,b_2)}$ and shunt admittances $\frac{1}{2}Y^{sh}_{(b_1,b_2)}$. The complete model is normally used in long and medium lines in case of transmission [7], while for short lines a simplified model ignoring the shunt elements (bottom right of Figure C.4) is preferred. In the case of distribution systems, as most lines are considerably short, the simpler model is normally adopted. However, in specific situations, such as [71] long, rural and lightly loaded lines or underground cables, the shunt admittance should not be neglected.

C.1.2.2 Transformers and phase-shifters

Transformers are usually employed when stepping down voltage from the transmission to the primary distribution system, and from this to the secondary portion to feed the loads. Some studies seem to absorb them in the generators and loads [95, 98–102], and there is no mention on how they are modeled. While this approach is acceptable when handling quantities in per unit, it cannot deal with conflict of bases or phase-shifting.

A real transformer at an edge between nodes b_1 and b_2 can be represented by an ideal one with turns ratio $\tau_{(b_1,b_2)}$ in series with an impedance $Z^{sr}_{(b_1,b_2)}$ indicating the resistive losses and the leakage reactance [84], as shown in the top of Figure C.5. This first model is valid whether the transformer is in-phase or phase-shifting ($\tau_{(b_1,b_2)}$ real or complex, respectively). In the first situation, an equivalent π-model can be derived as shown in the bottom of the figure, where the elements $Z^{A}_{(b_1,b_2)}$, $Z^{B}_{(b_1,b_2)}$ and $Z^{C}_{(b_1,b_2)}$ take the turns ratio into account [84]. With this model, transformers can be analyzed in the same way as the distribution lines.

In case of phase-shifters, the π-model is not valid. Fortunately, if the system is radial, it is possible to neglect them in the analysis and then, once the bus voltages are computed, manually translate the phase of the voltage drops in the appropriate branches [103, 104]. While this is well established for these cases, the literature seems to be yet maturing for the more general case of meshed configurations.

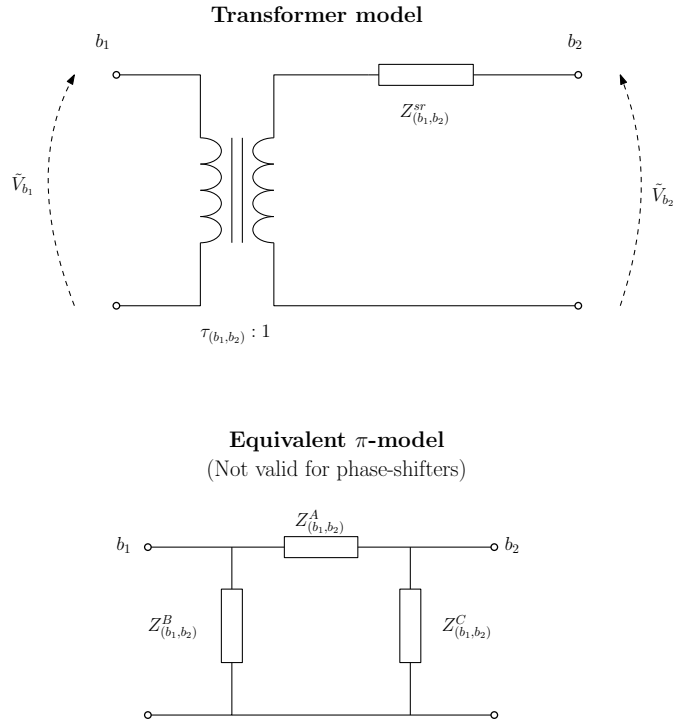


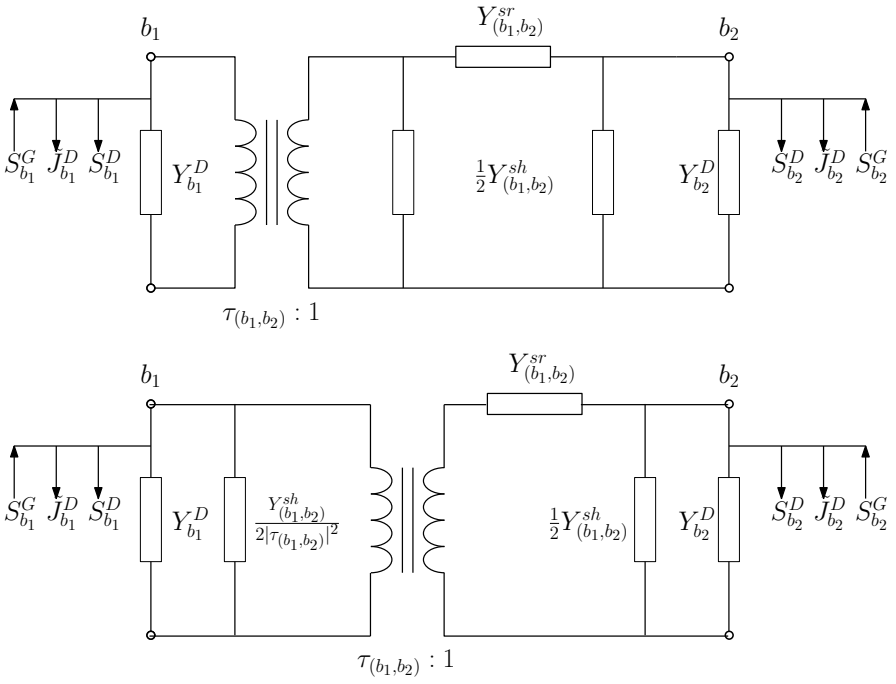
FIGURE C.5: **Top:** A realistic transformer can be modeled as the series combination of an ideal one and a series impedance. **Bottom:** equivalent π-model, valid only if the turns ratio is real.

C.1.2.3 Unified branch model

As more equipment needs to be modeled for wider operating conditions, the number of possible modelings can grow and the analysis of a system becomes more and more complicated. This can be seen for instance in the proposed method of [103], which requires a different and more complex model for phase-shifters (a pseudo-π circuit) when compared to distribution lines and regular transformers. On the other hand, the less complicated approach of considering all branches equal to the short line model employed in many other works (such as [88, 102]) can be too simplistic for some applications.

To cope with this trade-off, in this work the unified branch model shown in Figure C.6 and adapted from [105] will be used. The top circuit is composed of an ideal transformer with turns ratio $\tau_{(b_1,b_2)} : 1$ (which can be complex) in series with an equivalent π-circuit, in which all impedances were converted into admittance for later convenience. The nodes have the same model as given in Figure C.3. With this, any model of lines, transformers and phase-shifters can be represented. For instance, lines and regular transformers can be obtained by setting the turns ratio equal to 1.

The complete model can be simplified by incorporating the shunt admittances $\frac{1}{2}Y^{sh}_{(n1,n2)}$



Equivalent node-branch unified model

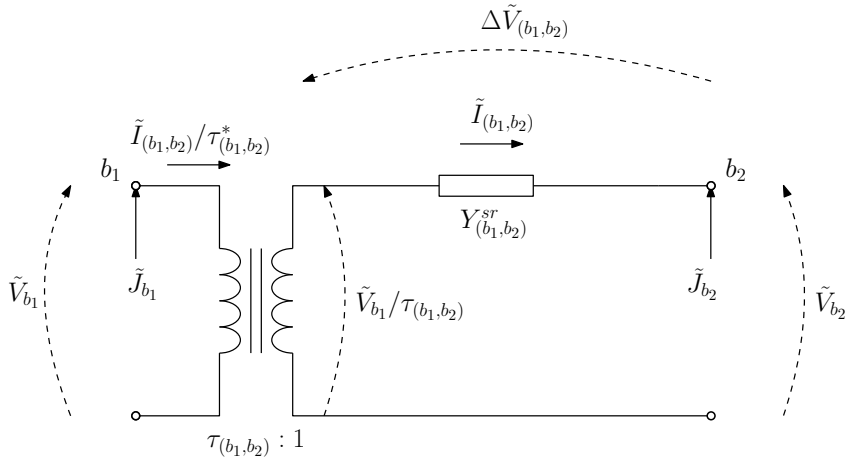


FIGURE C.6: Unified branch model used in this work. **Top:** A representation with all parameters used in any type of branch of the distribution systems of this work. **Bottom:** Equivalent model with all shunts combined into one admittance, all power injections combined into a single current source, and the appropriate parameters for the power flow method.

from the π-circuit into the other shunt nodes.[7] With this, we obtain the resulting node-branch model adopted in this work.

[7] The left admittance $\frac{1}{2}Y^{hs}_{(n1,n2)}$ was reflected to the primary of the transformer, becoming $\frac{1}{2|\tau_{(n1,n2)}|^2}Y^{sh}_{(n1,n2)}$.

C.1.3 Overall system modeling

Any distribution system represented by a graph $\mathcal{G}_D = (\mathcal{B}, \mathcal{L})$ now has a model with lumped components for nodes and edges. The parameters of interest for the power flow problem can be summarized as:

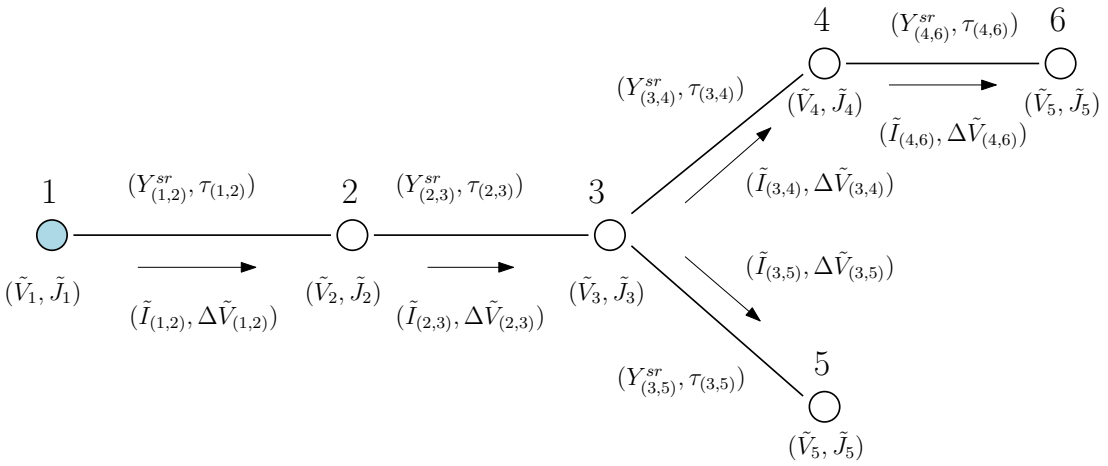
- A complex vector $\mathbf{v} \in \mathbb{C}^{|\mathcal{B}|}$ with the node/bus voltages $\tilde{V}_b$, $b \in \mathcal{B}$;
- Assuming an equivalent current-injection model, a complex vector $\mathbf{j} \in \mathbb{C}^{|\mathcal{B}|}$ with the net *injected* currents $\tilde{J}_b$ at node b, $b \in \mathcal{B}$, as shown in Figure C.3. In this text, we assume a positive active power when it is injected in the circuit (in case of sources), and negative when drained from it (in case of loads), so that its expression for a general node is given in equation (C.3);
- A complex vector $\mathbf{i} \in \mathbb{C}^{|\mathcal{L}|}$ with the edge/branch currents $\tilde{I}_\ell$, $\ell \in \mathcal{L}$. This corresponds to the to the secondary current at the edge model of Figure C.6;
- A complex vector $\Delta\mathbf{v} \in \mathbb{C}^{|\mathcal{L}|}$ with the voltage drops in each branch $\Delta\tilde{V}_\ell$, $\ell \in \mathcal{L}$. Notice that, for an edge $\ell = (b_1, b_2)$, these drops are defined from the secondary of node b_1 to b_2 in accordance with Figure C.6, so that $\Delta\tilde{V}_{(b_1,b_2)} = \tilde{V}_{b_1}/\tau_{(b_1,b_2)} - \tilde{V}_{b_2}$. This reduces to the usual node voltage difference in simple branches when $\tau_\ell = 1$;
- A complex vector $\boldsymbol{\tau} \in \mathbb{C}^{|\mathcal{L}|}$ with the turns ratio of the transformers. It is a complex number in general to model phase-shifters, but it can be an integer for regular transformers, and even 1 to represent distribution lines;
- The *branch admittance matrix* $\mathbf{Y}_b \in \mathbb{C}^{|\mathcal{L}|\times|\mathcal{L}|}$ [91], in which the diagonal elements $Y_{(\ell,\ell)}$ correspond to the series admittance Y_ℓ^{sr} of the branch ℓ, and the non-diagonal terms $Y_{(\ell,\ell')}$, $\ell \neq \ell'$, take mutual coupling (if any) between the branches ℓ and ℓ'. If there is no coupling, this matrix is diagonal.

Figure C.7 shows these parameters for a simple feeder, in which we assume that each edge $\ell = (b_1, b_2)$ is oriented such that b_1 is in the primary of the transformer as in Figure C.6 (this is not important if $\tau_\ell = 1$). This can be extended for unbalanced three-phase systems by modeling each phase separately, and thus the vectors and matrices would be at most three times bigger.[8] Also, couplings between the phases or between components of the same phase can be included in the non-diagonal elements of $\mathbf{Y}_b$.

With the important parameters described, we now need to find the relationship between them. To begin with, consider writing the Kirchhoff's Current Law (KCL) at each node:

[8] Some nodes or connections may not exist in all phases, so stating that these matrices would be three times bigger is not always valid.

$$\mathcal{G}_D = (\mathcal{B}, \mathcal{L}) \begin{cases} \mathcal{B} &= \{1,2,3,4,5,6\} \\ \mathcal{L} &= \{(1,2),(2,3),(3,4),(3,5),(4,6)\} \end{cases}$$



$$\mathbf{v} = \begin{bmatrix} \tilde{V}_1 \\ \tilde{V}_2 \\ \tilde{V}_3 \\ \tilde{V}_4 \\ \tilde{V}_5 \\ \tilde{V}_6 \end{bmatrix} \qquad \mathbf{j} = \begin{bmatrix} \tilde{J}_1 \\ \tilde{J}_2 \\ \tilde{J}_3 \\ \tilde{J}_4 \\ \tilde{J}_5 \\ \tilde{J}_6 \end{bmatrix} \qquad \mathbf{i} = \begin{bmatrix} \tilde{I}_{(1,2)} \\ \tilde{I}_{(2,3)} \\ \tilde{I}_{(3,4)} \\ \tilde{I}_{(3,5)} \\ \tilde{I}_{(4,6)} \end{bmatrix} \qquad \Delta\mathbf{v} = \begin{bmatrix} \Delta\tilde{V}_{(1,2)} \\ \Delta\tilde{V}_{(2,3)} \\ \Delta\tilde{V}_{(3,4)} \\ \Delta\tilde{V}_{(3,5)} \\ \Delta\tilde{V}_{(4,6)} \end{bmatrix}$$

$$\boldsymbol{\tau} = \begin{bmatrix} \tau_{(1,2)} \\ \tau_{(2,3)} \\ \tau_{(3,4)} \\ \tau_{(3,5)} \\ \tau_{(4,6)} \end{bmatrix} \qquad \mathbf{Y}_b = \begin{array}{c} \\ (1,2) \\ (2,3) \\ (3,4) \\ (3,5) \\ (4,6) \end{array} \begin{array}{c} \begin{array}{ccccc} (1,2) & (2,3) & (3,4) & (3,5) & (4,6) \end{array} \\ \begin{bmatrix} Y^{sr}_{(1,2)} & 0 & 0 & 0 & 0 \\ 0 & Y^{sr}_{(2,3)} & 0 & 0 & 0 \\ 0 & 0 & Y^{sr}_{(3,4)} & 0 & 0 \\ 0 & 0 & 0 & Y^{sr}_{(3,5)} & 0 \\ 0 & 0 & 0 & 0 & Y^{sr}_{(4,6)} \end{bmatrix} \end{array}$$

FIGURE C.7: Example of the overall description of power system. An unbalanced three-phase version can also be described by mapping each phase individually, and couplings can be accounted in the branch admittance matrix. The extension to multiple feeders is straightforward.

$$\begin{array}{rcccccccc} \tilde{J}_1 &=& +\tilde{I}_{(1,2)}/\tau^*_{(1,2)} & & & & & & \\ \tilde{J}_2 &=& -\tilde{I}_{(1,2)} & +\tilde{I}_{(2,3)}/\tau^*_{(2,3)} & & & & & \\ \tilde{J}_3 &=& & -\tilde{I}_{(2,3)} & +\tilde{I}_{(3,4)}/\tau^*_{(3,4)} & +\tilde{I}_{(3,5)}/\tau^*_{(3,5)} & & & \\ \tilde{J}_4 &=& & & -\tilde{I}_{(3,4)} & & & +\tilde{I}_{(4,6)}/\tau^*_{(4,6)} & \\ \tilde{J}_5 &=& & & & & -\tilde{I}_{(3,5)} & & \\ \tilde{J}_6 &=& & & & & & & -\tilde{I}_{(4,6)} \end{array}$$

in which the differences between the primary and secondary branch currents were taken into account. More specifically, if $\tilde{I}_\ell$ is the current at the secondary in branch ℓ, then $\tilde{I}_\ell/\tau_e^*$ is the current flowing in the primary, as shown in Figure C.6.

The previous system can be rewritten as

$$\begin{bmatrix} \tilde{J}_1 \\ \tilde{J}_2 \\ \tilde{J}_3 \\ \tilde{J}_4 \\ \tilde{J}_5 \\ \tilde{J}_6 \end{bmatrix} = \begin{bmatrix} 1/\tau^*_{(1,2)} & 0 & 0 & 0 & 0 \\ -1 & 1/\tau^*_{(2,3)} & 0 & 0 & 0 \\ 0 & -1 & 1/\tau^*_{(3,4)} & 1/\tau^*_{(3,5)} & 0 \\ 0 & 0 & -1 & 0 & 1/\tau^*_{(4,6)} \\ 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} \tilde{I}_{(1,2)} \\ \tilde{I}_{(2,3)} \\ \tilde{I}_{(3,4)} \\ \tilde{I}_{(3,5)} \\ \tilde{I}_{(4,6)} \end{bmatrix}$$

or even

$$\mathbf{j} = \mathbf{A}_\tau^* \mathbf{i} \tag{C.4}$$

in which $\mathbf{A}_\tau^* \in \mathbb{C}^{|\mathcal{B}|\times|\mathcal{L}|}$ is the complex conjugate of a *modified incidence matrix*, with elements $\mathbf{A}_\tau = (a_{b,\ell})$ given by [91]:

$$a_{b,\ell} = \begin{cases} 1/\tau_\ell & \text{if branch } \ell \text{ leaves node } b \\ -1 & \text{if branch } \ell \text{ enters node } b \\ 0 & \text{if branch } \ell \text{ is not incident with node } b \end{cases} \tag{C.5}$$

This modified incidence matrix generalizes the original definition by taking the complex turns ratio into account. Notice that it can be constructed incrementally with the same complexity as the one adopted in most works that cannot handle phase-shifters.

Now, let us apply the Kirchhoff's Voltage Law (KVL) to each branch:

$$\begin{array}{lcccccccc} \Delta\tilde{V}_{(1,2)} & = & \tilde{V}_1/\tau_{(1,2)} & & -\tilde{V}_2 & & & & \\ \Delta\tilde{V}_{(2,3)} & = & & \tilde{V}_2/\tau_{(2,3)} & & -\tilde{V}_3 & & & \\ \Delta\tilde{V}_{(3,4)} & = & & & \tilde{V}_3/\tau_{(3,4)} & & -\tilde{V}_4 & & \\ \Delta\tilde{V}_{(3,5)} & = & & & \tilde{V}_3/\tau_{(3,5)} & & & -\tilde{V}_5 & \\ \Delta\tilde{V}_{(4,6)} & = & & & & & \tilde{V}_4/\tau_{(4,6)} & & -\tilde{V}_6 \end{array}$$

which can be rewritten as

$$\begin{bmatrix} \Delta\tilde{V}_{(1,2)} \\ \Delta\tilde{V}_{(2,3)} \\ \Delta\tilde{V}_{(3,4)} \\ \Delta\tilde{V}_{(3,5)} \\ \Delta\tilde{V}_{(4,6)} \end{bmatrix} = \begin{bmatrix} 1/\tau_{(1,2)} & -1 & 0 & 0 & 0 & 0 \\ 0 & 1/\tau_{(2,3)} & -1 & 0 & 0 & 0 \\ 0 & 0 & 1/\tau_{(3,4)} & -1 & 0 & 0 \\ 0 & 0 & 1/\tau_{(3,5)} & 0 & -1 & 0 \\ 0 & 0 & 0 & 1/\tau_{(4,6)} & 0 & -1 \end{bmatrix} \begin{bmatrix} \tilde{V}_1 \\ \tilde{V}_2 \\ \tilde{V}_3 \\ \tilde{V}_4 \\ \tilde{V}_5 \\ \tilde{V}_6 \end{bmatrix}$$

or, more compactly,

$$\Delta\mathbf{v} = \mathbf{A}_\tau^T \mathbf{v} \tag{C.6}$$

in which $\mathbf{A}_\tau^T$ is the transpose of the modified incidence matrix defined at equation (C.5). Even if derived for the simple network of Figure C.7, equations (C.4) and (C.6) generalize the KCL and KVL for any network with nodes and edges modeled by Figure C.6.

In order to combine both equations, refer again to the unified branch model. The relationship between the branch current and voltage drops is

$$\tilde{I}_{(b_1,b_2)} = Y^{sr}_{(b_1,b_2)} \Delta\tilde{V}_{(b_1,b_2)}, \ \forall \ell = (b_1, b_2) \in \mathcal{L}$$

or, in matrix notation,

$$\mathbf{i} = \mathbf{Y}_b \Delta\mathbf{v} \tag{C.7}$$

Pre-multiplying each term of this equation by $\mathbf{A}_\tau^*$ and using (C.4), we get

$$\mathbf{j} = \mathbf{A}_\tau^* \mathbf{Y}_b \Delta\mathbf{v}$$

and now, using equation (C.6),

$$\mathbf{j} = \mathbf{A}_\tau^* \mathbf{Y}_b \mathbf{A}_\tau^T \mathbf{v}$$

which takes the familiar form

$$\mathbf{j} = \mathbf{Y}_{bus} \mathbf{v} \tag{C.8}$$

with $\mathbf{Y}_{bus} \triangleq \mathbf{A}_\tau^* \mathbf{Y}_b \mathbf{A}_\tau^T$ and $\mathbf{Y}_{bus} \in \mathbb{C}^{|\mathcal{B}|\times|\mathcal{B}|}$ a generalization of the well-known *bus admittance matrix* taking the possible complex turns ratio into account.[9] Notice that this matrix is in general not symmetric.

Despite its importance, the linear system (C.8) cannot be used directly in the power flow problem because it is not well defined. This is expected since the current injected at the slack node is the total injected at each other load, thus creating a linear dependency in the system. Fortunately, the voltage at some nodes is already known, and thus the system can be reduced and appropriately solved.

For that, partition the set of nodes into $\mathcal{B} = \mathcal{B}_{slack} \cup \mathcal{B}_{oos} \cup \mathcal{B}_{ins}$, with $\mathcal{B}_{slack}$ the set of slacks, in which the voltage is given; $\mathcal{B}_{oos}$ the set of *out of service* nodes, which are not connected to any source and thus have a zero voltage[10]; and $\mathcal{B}_{ins}$ the remaining *PV* and *PQ* nodes, referred here to as "in service". The system (C.8) can then be written as

$$\mathbf{j} = \mathbf{Y}_{bus}[:, \mathcal{B}_{slack}]\mathbf{v}[\mathcal{B}_{slack}] + \mathbf{Y}_{bus}[:, \mathcal{B}_{ins}]\mathbf{v}[\mathcal{B}_{ins}] + \mathbf{Y}_{bus}[:, \mathcal{B}_{oos}] \underbrace{\mathbf{v}[\mathcal{B}_{oos}]}_{\mathbf{0}}$$

with the notation $\mathbf{Y}_{bus}[:, \mathcal{B}_{slack}]$ representing a partition of the bus admittance matrix with all original rows but only the columns corresponding to slack nodes, $\mathbf{v}[\mathcal{B}_{slack}]$ a vector with the (known) voltages at $\mathcal{B}_{slack}$, and similar definitions for $\mathbf{Y}_{bus}[:, \mathcal{B}_{ins}]$, $\mathbf{v}[\mathcal{B}_{ins}]$, $\mathbf{Y}_{bus}[:, \mathcal{B}_{oos}]$ and $\mathbf{v}[\mathcal{B}_{oos}]$.

In this new system, we can temporarily ignore the injected currents at the slack nodes, and assume the *oos* ones are equal to zero. Thus, if we consider only the rows with nodes from $\mathcal{B}_{ins}$, it takes the form

$$\mathbf{Y}_{bus}[\mathcal{B}_{ins}, \mathcal{B}_{ins}]\mathbf{v}[\mathcal{B}_{ins}] = \mathbf{j}[\mathcal{B}_{ins}] - \mathbf{Y}_{bus}[\mathcal{B}_{ins}, \mathcal{B}_{slack}]\mathbf{v}[\mathcal{B}_{slack}] \tag{C.9}$$

which is in general well defined and can be solved for the unknown voltages if the node currents are determined. The simple system (C.8) [or (C.9)] provides the necessary relationship between all parameters of interest for the power flow. Notice that this formulation is valid for faulted and non-faulted networks.

[9] Notice this formulation takes only the series admittances into account because the shunts were incorporated into the loads. Should constant impedance loads be separated, each element in the diagonal of $\mathbf{Y}_{bus}$ would be added to the total shunts.

[10] The *oos* nodes can be either provided during the evaluation mechanism of a restoration plan (which is the case of this work, for example), or computed by (i) a BFS starting from each slack node and counting the unvisited nodes as *oos*, or (ii) by getting all connected components and outputting the ones with no slack.

C.2 A Y-matrix algorithm for the Power Flow

Equation (C.8) provides a starting point for most well-known power flow algorithms, such as most variants of Gauss-Seidel and Newton-Raphson [85]. In this work we make direct use of equation (C.9) to describe a very simple power flow algorithm.

Assume initially that the system has only slacks and PQ nodes [the case of multiple slacks, represented by a multi-feeder distribution network, is evident from the notation in (C.9)]. Begin by noticing that, thanks to the injected current model of Table C.1, if the loads were all of the constant impedance or current types, this system would be linear, and then equation (C.9) would provide a direct solution to the unknown load voltages. However, the other types of load require the node voltage at the denominator and thus introduce non-linearities.

To solve this non-linear equation, we employ the following iterative scheme:

1. Begin with an approximation for $\tilde{V}_b$, $b \in \mathcal{B}_{ins}$. In the lack of better options, usually a flat profile is a good choice, in which all voltages begin equal to the corresponding slack;
2. Compute the overall injected current $\tilde{J}_b$, $b \in \mathcal{B}_{ins}$ with equation (C.3) for the appropriate load models in this node;
3. Estimate new voltage values by solving the linear system (C.9);
4. Repeat the two previous steps until the difference in magnitude in all voltages between two consecutive iterations is smaller than a given tolerance.

At the end, the voltages at all nodes is known. Thus, we can use equation (C.8) to determine the node injected current at the remaining slack nodes [which were excluded in equation (C.9)], and the voltage drops and branch currents become available by equations (C.6) and (C.7). Notice these last operations require only simple matrix multiplications. A pseudo-code of the complete method is given in Algorithm 6. Keep in mind that $\mathbf{v}[\mathcal{B}_{slack}]$ in line 6 is actually given *a priori*, which explains the missing iteration index.

Probably because of a lack of uniformity when naming methods in the literature, a proper classification of the proposed method is not immediate. At first sight, it can be considered as a Y (or implicit Z)-matrix method [85]. Curiously, some works prefer to refer to it as a variant of Gauss-Seidel [106] and, if we let the rigor a little loose, it can be seen as a matrix-based Forward-Backward Sweep, given its similarity to [102] and

Data: Non-contracted network $\mathcal{G}_D = (\mathcal{B}, \mathcal{L})$; Modified incidence matrix $\mathbf{A}_\tau$; Branch admittance matrix $\mathbf{Y}_b$; Sets of slack $\mathcal{B}_{slack}$ and out of service $\mathcal{B}_{oos}$ nodes; voltage tolerance ϵ_v;
Result: Bus voltages $\tilde{V}_b$ and node injected currents $\tilde{J}_b$, $\forall b \in \mathcal{B}$; Branch currents $\tilde{I}_\ell$ and voltage drops $\Delta\tilde{V}_\ell$, $\forall \ell \in \mathcal{L}$;

Get in service nodes $\mathcal{B}_{ins} \longleftarrow \mathcal{B} \setminus (\mathcal{B}_{slack} \cup \mathcal{B}_{oos})$;
$\mathbf{Y}_{bus} = \mathbf{A}_\tau^* \mathbf{Y}_b \mathbf{A}_\tau^T$; `// Build bus admittance matrix`
$\tilde{V}_b^{(0)} \longleftarrow 1.0\angle 0°$, $\forall b \in \mathcal{B}_{ins}$; `// Initial voltage profile in p.u.`
$k \longleftarrow 1$;
do
  Compute $\tilde{J}_b^{(k)}$ for each node $b \in \mathcal{B}_{ins}$ using $\tilde{V}_b^{(k-1)}$ with equation (C.3) and Table C.1;
  Solve linear system for $\mathbf{v}^{(k)}[\mathcal{B}_{ins}]$
  $\mathbf{Y}_{bus}[\mathcal{B}_{ins}, \mathcal{B}_{ins}]\mathbf{v}^{(k)}[\mathcal{B}_{ins}] = \mathbf{j}^{(k)}[\mathcal{B}_{ins}] - \mathbf{Y}_{bus}[\mathcal{B}_{ins}, \mathcal{B}_{slack}]\mathbf{v}[\mathcal{B}_{slack}]$; `// Equation (C.9)`
  $k \longleftarrow k + 1$;
while $\max_{b \in \mathcal{B}_{ins}} \{|\tilde{V}_b^{(k)}| - |\tilde{V}_b^{(k-1)}|\} > \epsilon_v$;
$\tilde{V}_b^{(k)} \longleftarrow 0$, $\forall b \in \mathcal{B}_{oos}$; `// Zero voltage at disconnected loads`
$\mathbf{j} = \mathbf{Y}_{bus}\mathbf{v}^{(k)}$; `// Node currents at all nodes`
$\Delta\mathbf{v} = \mathbf{A}_\tau^T \mathbf{v}^{(k)}$; `// Voltage drops (equation (C.6))`
$\mathbf{i} = \mathbf{Y}_b \Delta\mathbf{v}^{(k)}$; `// Branch currents (equation (C.7))`

Algorithm 6: Y-matrix method for solving the power flow problem with only slacks and PQ nodes. In case of non-faulted nodes, $\mathcal{B}_{oos} = \emptyset$.

[88]. For simplicity, in this work it will be referred to as a Y-matrix based approach because of its explicit construction and use of the bus admittance matrix.

C.2.1 Extensions

C.2.1.1 Handling Distributed Generation

As mentioned in section C.1.1.3, small DGs can be modeled as regular PQ nodes with a known injected complex power S_b^G, so Algorithm 6 is already able to cover this case. Larger units, however, require a PV title, and the inclusion of such nodes in common methods such as regular Z-matrices [85] and Forward-Backward Sweeps [88, 102] is generally not evident because the current-injection model requires a complete characterization (magnitude and phase) of a current phasor to proceed, and only the magnitude of $\tilde{V}_b$ and the real part of S_b^G are given in this case.

This problem was dealt differently in previous works. [99] apparently neglects this case and considers only generators that can be modeled as PQ nodes, while [107] adopts a Newton-Raphson method similarly to the usual case in transmission systems, which is followed by a Backward-Forward sweep. Apparently the most popular approach is a compensation based technique, employed in many other works [96, 98, 108], which tries to estimate appropriate injected reactive power $Q_b^G = \text{Im}\{S_b^G\}$ in such nodes until the corresponding voltage magnitude $|\tilde{V}_b|$ is correct. Each work has its own different

approach. Here, we follow a simple compensation based on usual Gauss-Seidel Z-matrix methods [106].

Let $\mathbf{v}$ and $\mathbf{j}$ be the vectors of voltage and net injected currents at all nodes $\mathcal{B}$. The corresponding net injected complex power $\mathbf{s}^c$ can be computed as

$$\begin{aligned} \mathbf{s}^c &= \text{diag}(\mathbf{v})\mathbf{j}^* \\ &= \text{diag}(\mathbf{v})(\mathbf{Y}_{bus}\mathbf{v})^* \end{aligned} \tag{C.10}$$

using (C.8). For individual nodes, equation (C.10) takes the form

$$S_b^c = \tilde{V}_b \sum_{b' \in \mathcal{B}} Y_{b,b'}^* \tilde{V}_{b'}^* \tag{C.11}$$

in which $Y_{b,b'}$ is the element of $\mathbf{Y}_{bus}$ corresponding to the row of node b and column of node b'.

Let $\mathcal{B}_{PV} \subset \mathcal{B}$ be the set of PV nodes. Assume $\tilde{V}_b = |\tilde{V}_b|\angle\theta_b$, $b \in \mathcal{B}_{PV}$, is the current estimate of its voltage, with $|\tilde{V}_b|$ given as pre-determined but with a (possibly erroneous) estimate of the angle θ_b. At first, we could compute the corresponding reactive power Q_b by taking the imaginary portion of equation (C.11) for each of these nodes. However, notice that this is the *net injected power*, not necessarily the power provided by the generator. For a general node $b \in \mathcal{B}_{PV}$ modeled as in equation (C.3) and Figure C.3, after computing S_b^c, Q_b^G can be estimated by a simple manipulation of equation (C.3):

$$Q_b^G = \text{Im}\left\{ S_b^c + S_b^D(\tilde{V}_b) + \tilde{V}_b \times (\tilde{J}^D)^* - |\tilde{V}_b|^2 \times (Y_b^D)^* \right\} \tag{C.12}$$

and then, with P_b^G given for this node, we can set $S_b^G = P_b^G + \jmath Q_b^G$ and continue the process as usual. Nevertheless, notice that generators usually have limits on their output reactive power, say $Q_{b,min}^G$ and $Q_{b,max}^G$. If Q_b^G as computed by (C.12) extrapolates any of these bounds, we fix the reactive power to the appropriate limit and treat this node as a regular PQ one. Keep in mind that, in this situation, the voltage magnitude probably will not equate the pre-specified value.

The complete procedure is given in Algorithm 7, where again all parameters are assumed to be in p.u., so that the desired voltage magnitude at slacks and PV nodes is 1.0. In case of PV generators in which the reactive power is within the limits, the magnitude is corrected at each iteration in lines 16-20.

Data: Network $\mathcal{G} = (\mathcal{B}, \mathcal{L})$; Modified incidence matrix $\mathbf{A}_\tau$; Branch admittance matrix $\mathbf{Y}_b$; Sets of slacks $\mathcal{B}_{slack}$, PV $\mathcal{B}_{PV}$, out of service $\mathcal{B}_{oos}$; Real power at PV nodes P_b^G, $b \in \mathcal{B}_{PV}$; voltage tolerance ϵ_v;

Result: Node voltages $\tilde{V}_b$ and net injected currents $\tilde{J}_b$, $\forall b \in \mathcal{B}$; Branch currents $\tilde{I}_\ell$ and voltage drops $\Delta\tilde{V}_\ell$, $\forall \ell \in \mathcal{L}$;

Get in service nodes $\mathcal{B}_{ins} \longleftarrow \mathcal{B} \setminus (\mathcal{B}_{slack} \cup \mathcal{B}_{oos})$;
$\mathbf{Y}_{bus} = \mathbf{A}_\tau^* \mathbf{Y}_b \mathbf{A}_\tau^T$; // Build bus admittance matrix
$\tilde{V}_b^{(0)} \longleftarrow 1.0\angle 0°$, $\forall b \in \mathcal{B}_{ins}$; // Initial voltage profile in p.u.
$k \longleftarrow 1$;
do
  /* Estimate reactive power at PV nodes */
  for $b \in \mathcal{B}_{PV}$ **do**
    Compute net injected power S_b^c with (C.11) and using $\tilde{V}_b^{(k-1)}$;
    Compute generated reactive power Q_b^G with (C.12) and $\tilde{V}_b^{(k-1)}$;
    if $Q_b^G < Q_{b,min}^G$ *or* $Q_b^G > Q_{b,max}^G$ **then**
      Set Q_b^G to the appropriate bound;
      Flag the generator as overloaded;
    end
    $S_b^G \longleftarrow P_b^G + \jmath Q_b^G$;
  end
  /* Compute new voltage estimates */
  Compute $\tilde{J}_b^{(k)}$ for each node $b \in \mathcal{B}_{ins}$ using $\tilde{V}_b^{(k-1)}$ with equation (C.3) and Table C.1;
  Solve linear system for $\mathbf{v}^{(k)}[\mathcal{B}_{ins}]$
  $\mathbf{Y}_{bus}[\mathcal{B}_{ins}, \mathcal{B}_{ins}]\mathbf{v}^{(k)}[\mathcal{B}_{ins}] = \mathbf{j}^{(k)}[\mathcal{B}_{ins}] - \mathbf{Y}_{bus}[\mathcal{B}_{ins}, \mathcal{B}_{slack}]\mathbf{v}[\mathcal{B}_{slack}]$; // Equation (C.9)
  /* Correct voltage magnitude at PV nodes */
  for $b \in \mathcal{B}_{PV}$ **do**
    if *Generator b is not overloaded* **then**
      $\tilde{V}_b^{(k)} \longleftarrow \tilde{V}_b^{(k)} / |\tilde{V}_b^{(k)}|$;
    end
  end
  $k \longleftarrow k + 1$;
while $\max_{b \in \mathcal{B}_{ins}} \{|\tilde{V}_b^{(k)}| - |\tilde{V}_b^{(k-1)}|\} > \epsilon_v$;
$\tilde{V}_b^{(k)} \longleftarrow 0$, $\forall b \in \mathcal{B}_{oos}$; // Zero voltage at disconnected loads
$\mathbf{j} = \mathbf{Y}_{bus}\mathbf{v}^{(k)}$; // Net injected currents at all nodes
$\Delta\mathbf{v} = \mathbf{A}_\tau^T \mathbf{v}^{(k)}$; // Voltage drops (equation (C.6))
$\mathbf{i} = \mathbf{Y}_b \Delta\mathbf{v}^{(k)}$; // Branch currents (equation (C.7))

Algorithm 7: Y-matrix method for solving the power flow problem with both PQ and PV nodes, considering voltages in p.u. In case of non-faulted nodes, $\mathcal{B}_{oos} = \emptyset$. Also, notice that the PV nodes are treated as in service as well.

C.2.1.2 Alternative formulation for pure radial systems

In radial systems, the number of nodes is bigger than the edges by the number of slacks, in which case $|\mathcal{B}_{ins}| = |\mathcal{L}|$.[11] With a different combination of the Kirchhoff's Laws in equations (C.4) and (C.6) with equation (C.7), we can arrive at the following equivalent systems

[11] This can also be valid in the presence of *oos* nodes as long as branches connecting them are also discarded.

$$\mathbf{A}_\tau^*[\mathcal{B}_{ins},:]\mathbf{i} = \mathbf{j}[\mathcal{B}_{ins}] \tag{C.13}$$

$$\mathbf{Y}_b\mathbf{A}_\tau^T[:,\mathcal{B}_{ins}]\mathbf{v}[\mathcal{B}_{ins}] = \mathbf{i} - \mathbf{Y}_b\mathbf{A}_\tau^T[:,\mathcal{B}_{slack}]\mathbf{v}[\mathcal{B}_{slack}] \tag{C.14}$$

which were obtained after similar partitionings as in equation (C.9). Because $|\mathcal{B}_{ins}| = |\mathcal{L}|$, the partitioned incidence matrix is now square, and then we can solve (C.13) to obtain the branch currents and (C.14) for the node voltages.

So, what is the reward by splitting a single system into two? At first, none. However, if the network has a labeling of nodes and edges starting from the root (slack node) to the leaves – which can be obtained through a breadth first search (BFS), for instance –, then $\mathbf{A}_\tau^*[\mathcal{B}_{ins},:]$ is an upper triangular matrix (and its transpose a lower one), and thus each system can be solved by a forward/backward substitution, which may be more efficient than solving the one with the complete bus admittance matrix. This approach was proposed in [102], and Figure C.7 has coincidentally such a labeling. The reader can see how $\mathbf{A}_\tau^*$ in this example becomes upper triangular by eliminating the first column.

Keep in mind that this is not valid for weakly meshed topologies, and the BFS should be executed every time for each new network configuration, which may become more costly than considering a fixed labeling in some situations. Also, if there are mutual impedances between branches, $\mathbf{Y}_b$ is not diagonal and the product $\mathbf{Y}_b\mathbf{A}_\tau^T[:,\mathcal{B}_{ins}]$ is not necessarily lower triangular, thus hindering the efficiency. In such cases, it should be better to stick with the regular approach of equation (C.9).

C.2.2 Comparison with other methods

A first comparison can be made with Z-matrix methods [85, 106], which adopt the *bus impedance matrix* $\mathbf{Z}_{bus} = \mathbf{Y}_{bus}^{-1}$ instead of the bus admittance one. If $\mathbf{Z}_{bus}$ is available, equation (C.8) becomes $\mathbf{v} = \mathbf{Z}_{bus}\mathbf{j}$, and the iterative process of Algorithm 6 requires only matrix multiplications. Despite this advantage, there are two major shortcomings: (i) memory requirements can increase in large systems because $\mathbf{Z}_{bus}$ tends not to be sparse even if $\mathbf{Y}_{bus}$ is [102]; (ii) the construction of $\mathbf{Z}_{bus}$ is not as easy as the one of $\mathbf{Y}_{bus}$. Of course, a direct inversion is possible but usually not advisable due to numerical errors. To handle that, a more efficient building process is proposed in [109] in which the bus impedance matrix is constructed directly by means of loop and branch-path incidence matrices, and no inversions are required. These matrices involve shortest paths calculations, and thus their computation can be more costly than the usual incidence

matrix. Also, the method seems to work for radial configurations only, and not weakly meshed cases.

An approach requiring a linear order memory storage is proposed in [102], which is also valid only for radial configurations and was the one that inspired the formulation of section C.2.1.2. Instead of splitting the slack node information, however, the authors induce a self-loop in each root to make the number of edges and nodes equal, again reflecting in a square incidence matrix. The iterative process is performed by solving systems similar to equations (C.13)-(C.14), and thanks to the relabeling according to a BFS they are upper or lower triangular, and can be solved efficiently by forward and backward substitution. The drawbacks of this method are the same as discussed before: the necessity of running a BFS for each different configuration and the loss of numerical efficiency if there are mutual impedances between branches. In any case, whenever this approach is applicable, the proposed one can be reduced to it according to section C.2.1.2, but with the added contribution of handling shunts and phase-shifters.

Finally, Teng in [88] proposed a direct approach using only matrix multiplications, which can be seen as a matrix version of the Forward-Backward sweep [71]. He employs two matrices, the bus-injection to branch-current $[BIBC]$ and the branch-current to bus-voltage $[BCBV]$, which are used for the sweeps.[12] The method has the same benefits of avoiding linear systems as the Z-matrix ones, and the creation process of the structural matrices is very similar to the one previously cited in [109], but with extensions to weakly meshed cases. Apart from the same shortcomings of usual Z-matrix techniques, this method cannot handle shunts and phase-shifters. These are later treated in [103] but, as argued before in section C.1.2.3, the proposed modeling is not only more complicated but also different for each elements, which tends to make the analysis more complex.

C.3 The Power Flow in the Load Restoration Context

The previous exposition is general for any electrical network, and as such it is valid for a distribution system as well. However, some peculiarities can be explored to help "allocating" the proposed power flow algorithm into the load restoration context. Here are some guidelines that can also be used as implementation details:

- In case of networks with outages, remove all faulted nodes and their adjacent edges from it. The voltages and current in such nodes are irrelevant as they are isolated[13]

[12] As a comparison, the KCL is written as $\mathbf{i} = [BIBC]\mathbf{j}$, wherein $[BIBC]$ is equivalent to $\mathbf{A}^{-1}$. The other matrix, however, does not have an equivalent in the formulation proposed here.

[13] Or at least should be, unless the dispatch team likes to live in the edge (no pun intended)!

from the rest of the system. Alternatively, it is possible to include them in the set of *oos* nodes, but since this removal is executed only in the beginning, we at least reduce the system size by the number of these faulted buses.

- In principle, different incidence and branch admittance matrices should be constructed for each new configuration. Suppose a sequence of maneuvers M induces a configuration $\mathcal{G}_D^{CC}(M) = (\mathcal{B}, \mathcal{L}^{CC}(M))$, with $\mathcal{L}^{CC}(M) \subset \mathcal{L}$ the set of currently closed distribution lines.[14] Then, each M will associate to different $\mathbf{A}_\tau(M)$ and $\mathbf{Y}_b(M)$ [and consequently, distinct $\mathbf{Y}_{bus}(M)$] depending on $\mathcal{L}^{CC}(M)$. Instead of building these matrices every time, it should be more efficient to initially construct $\mathbf{A}_\tau$ and $\mathbf{Y}_b$ for a hypothetical configuration $\mathcal{G}_D = (\mathcal{B}, \mathcal{L})$ with all switches closed. Thus, for each new $\mathcal{G}_D^{CC}(M)$, we remove the appropriate rows and columns from these two matrices corresponding to the set of currently open $\mathcal{L}^{CO}(M) = \mathcal{L} \setminus \mathcal{L}^{CC}(M)$, and then the bus admittance matrix can be computed as usual.

- In the load restoration problem it is preferable to adopt the contracted network $\mathcal{G} = (\mathcal{N}, \mathcal{E})$, with $\mathcal{N}$ indicating sectors of buses and $\mathcal{E} = \mathcal{E}^{CC} \cup \mathcal{E}^{CO}$ the set of distribution lines containing a maneuverable switch. In that regard, if there is a mapping (or dictionary) $d(\cdot) : \mathcal{E} \mapsto \mathcal{L}$ that associates to each switch $e \in \mathcal{E}$ its respective distribution line $d(e) = \ell \in \mathcal{L}$ in the complete graph, than for a given configuration $\mathcal{G}^{CC}(M)$ we can obtain $\mathcal{G}_D^{CC}(M)$ directly and proceed with the power flow as usual.

In summary and with these considerations, given a sequence of maneuvers M and its resulting configuration $\mathcal{G}^{CC}(M)$ or $\mathcal{G}_D^{CC}(M)$, it is possible to write the following conditions to characterize a feasible configuration:

[14] Notice that lines with no associated switch are by definition always closed unless taken down by a fault (in which case they should be removed from the graph from the beginning). In that case, we extend the "currently closed" terminology to them. Similarly, healthy opened lines always have a maneuverable switch, so the "currently open" naming also holds in this case.

$$V_{b,min} \leq |\tilde{V}_b| \leq V_{b,max} \qquad \forall b \in \mathcal{B} \tag{C.15}$$

$$J_{b,min} \leq |\tilde{J}_b| \leq J_{b,max} \qquad \forall b \in \mathcal{B} \tag{C.16}$$

$$|\Delta\tilde{V}_\ell| \leq \Delta V_{\ell,max} \qquad \forall \ell \in \mathcal{L}^{CC}(M) \tag{C.17}$$

$$|\tilde{I}_\ell| \leq I_{\ell,max} \qquad \forall \ell \in \mathcal{L}^{CC}(M) \tag{C.18}$$

$$\mathbf{j}(M) = \mathbf{A}_\tau^*(M)\mathbf{i}(M) \tag{C.19}$$

$$\Delta\mathbf{v}(M) = \mathbf{A}_\tau^T(M)\mathbf{v}(M) \tag{C.20}$$

$$\mathbf{j}(M) = \mathbf{Y}_{bus}(M)\mathbf{v}(M) \tag{C.21}$$

$$\mathcal{G}^{CC}(M) \text{ is radial} \tag{C.22}$$

in which these parameters are obtained with the proposed power flow algorithm, and

- Equation (C.15) provides limits for the bus voltages $\tilde{V}_b$. Notice the upper bound takes into account possible Distributed Generations increasing too much the voltage;
- Equation (C.16) restricts the injection of currents $\tilde{J}_b$ in buses. Normally only source nodes are limited to represent feeder capacity, but this notation is more general. If a bus has no such constraint, its bounds can be set to $\pm\infty$;
- Equation (C.17) guarantees the voltage drops $\Delta\tilde{V}_\ell$ are not too large. Normally only the bus voltages are limited (in which case one can set $\Delta V_{\ell,max} \to \infty$), but this formulation is proposed to handle this case;
- Equation (C.18) confines the branch currents to the line capacity.
- Equations (C.19), (C.20) and (C.21) are the usual Kirchhoff's Laws for a given configuration obtained from a sequence of maneuvers M.
- Equation (C.22) assures the final configuration is radial, and the appropriate mathematical definition was already provided in equation (A.4).

Keep in mind that these conditions are used to characterize a feasible *configuration*, not exactly a restoration plan, modeled in this work as a sequence of maneuvers M. In its turn, a feasible sequence is one in which the configurations obtained after each operation is feasible.

Appendix D

Multi-objective optimization

D.1 Introduction to Decision Theory

Life is all about decisions. Whenever we are deciding simple matters like "which route to take to my work", "which ingredients to include in this recipe" or "should I break this egg directly into the frying pan", or more complex problems like "which career to follow", "which sequence of maneuvers to perform to restore this network" and "should we confer the title of Ph.D. to this young gentleman", we are trying to solve a *decision making problem* (or simply decision problem). In these situations, we usually have at our disposal a number of possible solutions to implement, and many of them could solve the problem. What makes it hard is that we normally look for the solution that provides us with the most satisfaction out of it, that is, we seek the *most preferred* or *optimal* solution.

In general, decision theory provides a framework for choosing between alternative courses of action when the consequences resulting from them are not perfectly known in advance [110]. In a decision making problem, we assume there is a person (or a group of people) which knows specific characteristics of the problem and is able to provide preference information about the possible solutions. This entity is called the *decision maker*[1] (DM) [112]. Therefore, the DM should be able to say "I prefer this solution over that one", or "I am indifferent regarding these two alternatives".

Apart from the DM, the elements of a decision problem can be classified as alternatives (or solutions), outcomes and states of nature [113]. The alternatives or solutions represent the possible courses of action that are open to be chosen by the decision maker (like

[1] In some contexts, the decision maker can be even a piece of software [111] with knowledge implemented into it. Then, in this chapter, I will adopt the pronoun "it" when referring to the DM in order to emphasize its impersonal character.

"Engineering", "Psychology" and "Dance" in the career example). The states of nature comprise the external factors that are not under the DM's control, such as the future availability of jobs in each of the possible careers. Lastly, the outcomes are simply the consequences of choosing a solution subjected to the corresponding state of nature.

This uncomfortable situation of not knowing exactly what we will end up with when making a decision is a great source of difficulty when solving decision problems. To overcome these scenarios, techniques involving probability theory and stochastic processes (like mentioned in [110, 114]) are often employed. Fortunately, in some cases, we can ignore the effects of these external factors and focus on a *deterministic decision problem*. The restoration problem fits in this class, so I will not treat the other cases any further in this text.

Regarding the alternatives, it is customary to divide decision problems into two categories [114]. The first includes the problems with a finite and small number of alternatives, wherein "small" means "it is feasible for the DM to evaluate each solution in order to choose its preferred". The career example can be included in this class. These problems are usually very easy to solve, unless we need to consider the states of nature, in which case we can employ multi-attribute analysis methods, like the scoring technique [114], to solve them.

Problems with infinite or finite but numerous alternatives are included in the second category. The restoration problem, in general, belongs to this class, together with others problems with continuous variables. In this category, because of the huge number of alternatives, we need to formulate a mathematical model in order to help with the solution process. This model should be able to express the outcomes in terms of performance measures, which are given by mathematical functions of the alternatives. For example, we can model the quality of a restoration plan by a combination of indices such as Time of Maneuvers, Energy not Supplied, Power not Restored etc. After this modeling, we can employ an *optimization tool* to help us selecting the final solution.

This appendix provides a description of the optimization theory required for this work, and then some approaches for adopting these tools into a decision making problem are mentioned.

A small note about notation

The main portion of this text considers a sequence of maneuvers M as a solution, while $\mathbf{x}$ was simply an abstraction representing the permutation vector. However, to maintain compatibility with most of the optimization references, in this appendix $\mathbf{x}$ will take the place of a decision variable.

D.2 Optimization Theory

Optimization is often defined as the task of computing the best (or optimal) solution of a problem. As already mentioned, two core elements of such a problem are the alternatives or solutions and the outcomes (assuming we ignore the states of nature). Let us introduce a more rigorous notation for them.

Assume that the alternatives are contained in a set $\mathbb{X}$, called *search space*. Then, we model the problem using a set of m mathematical functions $f_i(\cdot) : \mathbb{X} \mapsto \mathbb{Y}_i, i = 1, 2, \ldots, m$, with $\mathbb{Y}_i$ the *objective space* of the i-th function, which are referred to *objective functions* or *criteria* [115]. These functions associate, for each alternative $\mathbf{x} \in \mathbb{X}$, a number $f_i(\mathbf{x})$ indicating its objective value in the i-th criterion. They are commonly used to provide a quality value for a given outcome when choosing a solution $\mathbf{x}$. So, for instance, if $\mathbf{x}$ somehow represents a sequence of maneuvers of a reconfigured network, and if we model it by using $f_1(\cdot) = S_{NR}(\cdot)$ and $f_2(\cdot) = T_m(\cdot)$, then the numbers $S_{NR}(\mathbf{x})$ and $T_m(\mathbf{x})$ provide the quality of this restoration plan.

To simplify the notation, let us combine all of the m criteria into a vector function $\mathbf{f}(\cdot) = [f_1(\cdot)\ f_2(\cdot)\ \ldots\ f_m(\cdot)]^T \in \mathbb{Y}$, wherein $\mathbb{Y} = \mathbb{Y}_1 \times \mathbb{Y}_2 \times \ldots \times \mathbb{Y}_m$. Once the problem is modeled, the task of optimization can be translated into minimizing or maximizing this vector function. Since the maximum of $f_i(\cdot)$ is the minimum of $-f_i(\cdot)$, we can, without loss of generality, represent an optimization problem as[2]

$$\begin{aligned} &\text{minimize} && \mathbf{f}(\mathbf{x}) \\ &\text{subject to} && \mathbf{x} \in \mathcal{X} \subseteq \mathbb{X} \end{aligned} \tag{D.1}$$

in which $\mathcal{X}$ is called the *feasible region* representing the constraints. Its purpose is to narrow the possibilities of the solutions to the ones that make sense to the problem. For instance, $\mathcal{X}$ can represent the sequences of maneuvers such that the final network is radial and does not violate the voltage and current constraints. We call the alternatives $\mathbf{x} \in \mathcal{X}$ *feasible solutions*.

Depending on the structure of the search and objective space, problem (D.1) can be classified in different ways. First, with regards to $\mathbb{X}$, we can classify the problem into (i) continuous [116], if $\mathbb{X} = \mathbb{R}^n$, and the alternatives represent vectors with dimension n; or (ii) combinatorial [117], in which case the solutions can be (vectors of) integers,

[2] Notice the similarity of this formulation with (2.20), remembering the appropriate correspondences between $\mathbf{x}$ and M in the main text.

permutations of vectors, or even graphs, and the representation of $\mathbb{X}$ depends on the codification of the solutions (see the literature review for some examples in the restoration problem). Even though the problem of this thesis belongs to the combinatorial case, the theory explained in this chapter is valid for both types of problems.

Regarding the objective space, in this work we can assume that $\mathbb{Y} = \mathbb{R}^m$, that is, all criteria are mapped into real numbers. Also, we can classify an optimization problem into *single-objective*, if there is only one criterion $f(\cdot)$ to be minimized, and *multi-objective*, if there are two or more. The main difference in these cases is in the interpretation of equation (D.1) and in the definition of "optimal solution".

D.2.1 Single-objective optimization

If the problem can be well described by a scalar function $f(\cdot)$, equation (D.1) becomes

$$\begin{aligned} \text{minimize} \quad & f(\mathbf{x}) \\ \text{subject to} \quad & \mathbf{x} \in \mathcal{X} \subseteq \mathbb{X} \end{aligned} \tag{D.2}$$

A good example of such a case is the load restoration problem with only automated switches, when $S_{NR}(\cdot)$ alone may be enough to qualify a good plan.

Since this is a minimization, two feasible solutions $\mathbf{x}_1$ and $\mathbf{x}_2$ can be compared by saying that, if $f(\mathbf{x}_1) < f(\mathbf{x}_2)$, then the first solution is better. Therefore, the optimal solution $\mathbf{x}^*$ is the one such that there is not other feasible alternative with criterion value smaller than its own. This is what we call a global minimum, rigorously defined as follows.

Definition 1 (Global minimum). A solution $\mathbf{x}^*$ is called *global minimum* if

$$\nexists \mathbf{x} \in \mathcal{X} | f(\mathbf{x}) < f(\mathbf{x}^*)$$

Of course, *how* to actually compute $\mathbf{x}^*$ is another story. Depending on the type of the problem, there may be very efficient algorithms for solving it (like the shortest path and the minimum spanning tree in case of combinatorial optimization [117] and linear and convex problems in the continuous case [116]), while for some others we still struggle (like the traveling salesperson, general nonlinear problems and, of course, the load restoration). However, the field of single-objective optimization is very well developed, and there are many techniques that guarantee at least a good enough solution, even if not optimal.

Deterministic decision problems that use single-optimization tools can be considered "easy" in regards to choosing a final solution: just pick $\mathbf{x}^*$. With this, we manage to get the huge set $\mathbb{X}$ and reduce it into a much smaller set of alternatives, or even a singleton. This was possible because, by modeling the problem as a mathematical function, we translated the notion of preference to the concept of optimality. Thanks to this, there is very rarely the mention of a decision maker in single-objective optimization.[3] Notice, however, the importance of the modeling here: if the DM is not satisfied with the final solution, it should either lower its expectations or go back to first steps and reformulate the problem. Keep in mind: the optimization does what you tell it to do, not exactly what you want it to.

D.2.2 Multi-objective optimization

If the problem needs to be formulated using more than one criterion, we now have a vector function $\mathbf{f}(\cdot)$, and the optimization problem has the form given in equation (D.1). Its meaning, however, is less straightforward when compared to the single-objective case [118]. The reason is that, even though we may want to minimize all functions, they are usually *in conflict*, such that situations arise in which it is impossible to improve one criterion without degrading another. This is actually very evident in many situations in life: when minimizing costs and maximizing profit; when maximizing school grades, hours of sleep and social life; or when minimizing the power not restored and the time of maneuvers. In these examples we normally do not have a feasible solution that can optimize all criteria at once.

So, what is an optimal solution in a multi-objective problem? Before that, how can we compare solutions in this situation? It is easy to confront scalar numbers, but vector comparisons are, apart from a few particular cases, not so intuitive. For instance, in a two-objective example, if we have the solutions $\mathbf{f}(\mathbf{x}_1) = [2\ 4]^T$ and $\mathbf{f}(\mathbf{x}_2) = [3\ 5]^T$, since both components of the first are smaller than the ones in the second, we can readily say that $\mathbf{x}_1$ is better than $\mathbf{x}_2$. However, if we include a third alternative $\mathbf{f}(\mathbf{x}_3) = [6\ 3]^T$, we cannot say much about this new point and $\mathbf{x}_1$, since $\mathbf{x}_3$ is worse in the first criterion but excels in the second. In order to formalize these ideas, the next section presents the concept of *dominance* and *efficiency*.

[3] Actually, the only situations where it may be necessary are when there are more than one solutions with the same minimum value, or when the set $\mathbb{X}$ is not compact and, therefore, there is no minimum. In the first case, the DM can choose based on other secondary criteria or even randomly, while, in the second one, it can be satisfied with the best solution the algorithm can return. In any case, it still isn't a challenging task for the DM.

D.2.2.1 Dominance and Efficiency

Since there is no direct "less than (or equal to)" comparison in the vector case, let us start by defining *Pareto-dominance* [119].

Definition 2 (Pareto-dominance). Given two solutions $\mathbf{x}_1$ and $\mathbf{x}_2$, we say that $\mathbf{x}_1$ *Pareto-dominates*, or simply *dominates* $\mathbf{x}_2$, iff (if and only if)

- $\forall i \in \{1, 2, \ldots, m\}$, $f_i(\mathbf{x}_1) \leq f_i(\mathbf{x}_2)$, that is, $\mathbf{x}_1$ is not worse than $\mathbf{x}_2$ in all objectives; and
- $\exists i \in \{1, 2, \ldots, m\}$, $f_i(\mathbf{x}_1) < f_i(\mathbf{x}_2)$, that is, $\mathbf{x}_1$ is strictly better than $\mathbf{x}_2$ in at least one objective.

Whenever both conditions are satisfied, we write $\mathbf{x}_1 \prec \mathbf{x}_2$ or $\mathbf{f}(\mathbf{x}_1) \prec \mathbf{f}(\mathbf{x}_2)$. If $\mathbf{x}_1$ does not dominate $\mathbf{x}_2$ and neither the contrary, we say that they are *incomparable* or *non-dominated*. □

Figure D.1 shows a graphical representation of this kind of dominance for two objectives. For any point inside the region in light blue (like 2 and 3), solution 1 will possess all of its components equal with at least one better, so they will be dominated by it. Points 4 and 5 are not dominated by 1, and they also do not dominate it, being incomparable. Solution 6, however, dominates 1. Notice that, despite 1 not dominating 6, we see that $6 \prec 1$, so it is necessary to check dominance from both sides before drawing conclusions.

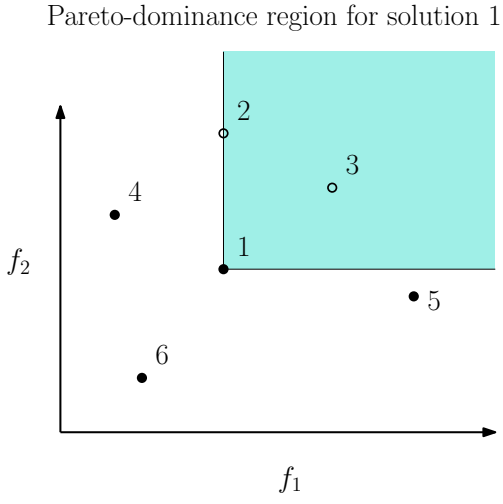


FIGURE D.1: Graphical representation of Pareto-dominance. Points 2 and 3 are inside the cone generated by 1, so they are dominated by it, while the remaining ones are not. Notice, however, that 6 dominates 1, even though its cone is not drawn.

Pareto-dominance induces a *partial ordering* in the objective space [119]. As shown in Figure D.1, this means that some solutions can be compared (like 1 and 2, 1 and 3, 1 and

6), but others cannot (like 1 and 4 and 1 and 5). This contrasts with the single-objective case, where a *total ordering* is present and, by using the "less than" operator, we can always say whether $a < b$ or $b < a$ if $a \neq b$. Because of this, in a non-empty set $\mathcal{X}$ of alternatives, there will be some solutions that are not dominated by any other. We call these alternatives *Pareto-optimal* (or simply optimal), and they are rigorously defined below.

Definition 3 (Pareto-optimality of efficiency). A feasible solution $\mathbf{x}^* \in \mathcal{X}$ is called *Pareto-optimal*, or simply *optimal*, if

$$\nexists \mathbf{x} \in \mathcal{X} | \mathbf{x} \prec \mathbf{x}^*$$

that is, if there is no other feasible solution that dominates it. □

As shown in Figure D.1, Pareto-dominance can be described by means of cones, which are called *cones of dominance*. It turns out that this is only a special case that is mostly used in applications. It is possible, in fact, to define the whole concept of optimality by means of these cones and, in these cases, the Pareto-optimal solutions are normally called *efficient* or *minimal*. This path will not be pursued here in order to simplify the discussion, but the interested and courageous reader can check references like [120] for that. Even so, the jargon "efficient" and "minimal" will occasionally be adopted as synonyms for Pareto-optimal solutions in this text.

A typical optimization problem may have many, or even infinite Pareto-optimal solutions. The set of all of these points defines the *Pareto-optimal set* and *Pareto-optimal front*, which are defined as:

Definition 4 (Pareto-optimal set and front). The set of all Pareto-optimal solutions in the search space is called *Pareto-optimal set*, or simply Pareto set, and is defined as

$$\mathcal{X}^* = \{\mathbf{x}^* \in \mathcal{X} | \mathbf{x}^* \text{ is Pareto-optimal}\}$$

In the objective space, we similarly define the *Pareto-optimal front* (or simply Pareto front) by

$$\mathcal{Y}^* = \{\mathbf{y}^* \in \mathcal{Y} | \mathbf{y}^* = \mathbf{f}(\mathbf{x}^*), \text{ and } \mathbf{x}^* \text{ is Pareto-optimal}\}$$

wherein $\mathcal{Y} \triangleq \mathbf{f}(\mathcal{X})$ is the image of the feasible region in the objective space, normally called objective feasible region.

Similarly, we can use the previous synonyms and call them efficient set/front and minimal set/front. □

Figure D.2 shows typical efficient fronts for discrete and continuous problems, showing the feasible objective region $\mathcal{Y}$ and the minimal front $\mathcal{Y}^*$.

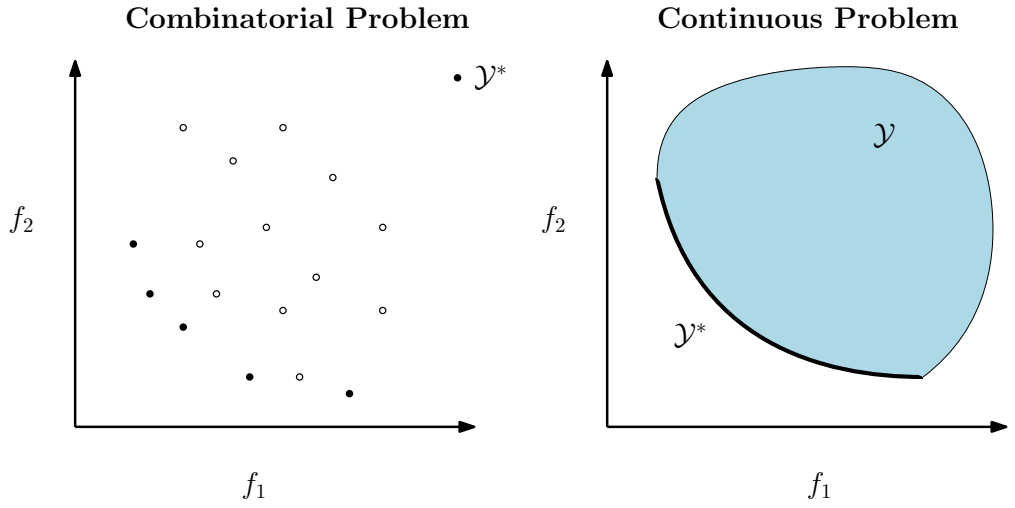


FIGURE D.2: Examples of efficient fronts for typical combinatorial (left) and continuous (right) problems. On the discrete case, all points belong to the feasible objective region, but only the black ones are efficient. In both cases, the minimal solutions are in the south-west direction.

Like previously mentioned, there are other types of efficiency that are important for a complete study of multi-objective optimization and its methods, like *weak efficiency* and *proper efficiency*. However, they are not required for the understanding of this work, and thus will not be explained. If the reader so desires, some references like [58, 119, 121] contain more details.

D.2.2.2 Boundaries of the Efficient Front and Standardization of Objectives

Pareto fronts can come in many different shapes [119], like convex, concave and disconnected, as shown in Figure D.3. This kind of information is normally relevant for some multi-objective methods in order to compute efficient solutions. Whatever the shape, there are some special points that deserve some attention in order to facilitate the decision process.

The first one is the *ideal solution*, also called *shadow minimum* in some studies (like [122]), $\mathbf{y}^\star$, and it is defined as

$$\mathbf{y}^\star \triangleq \left[\min_{\mathbf{x}\in\mathcal{X}} f_1(\mathbf{x}) \quad \min_{\mathbf{x}\in\mathcal{X}} f_2(\mathbf{x}) \quad \dots \quad \min_{\mathbf{x}\in\mathcal{X}} f_m(\mathbf{x})\right]^T \tag{D.3}$$

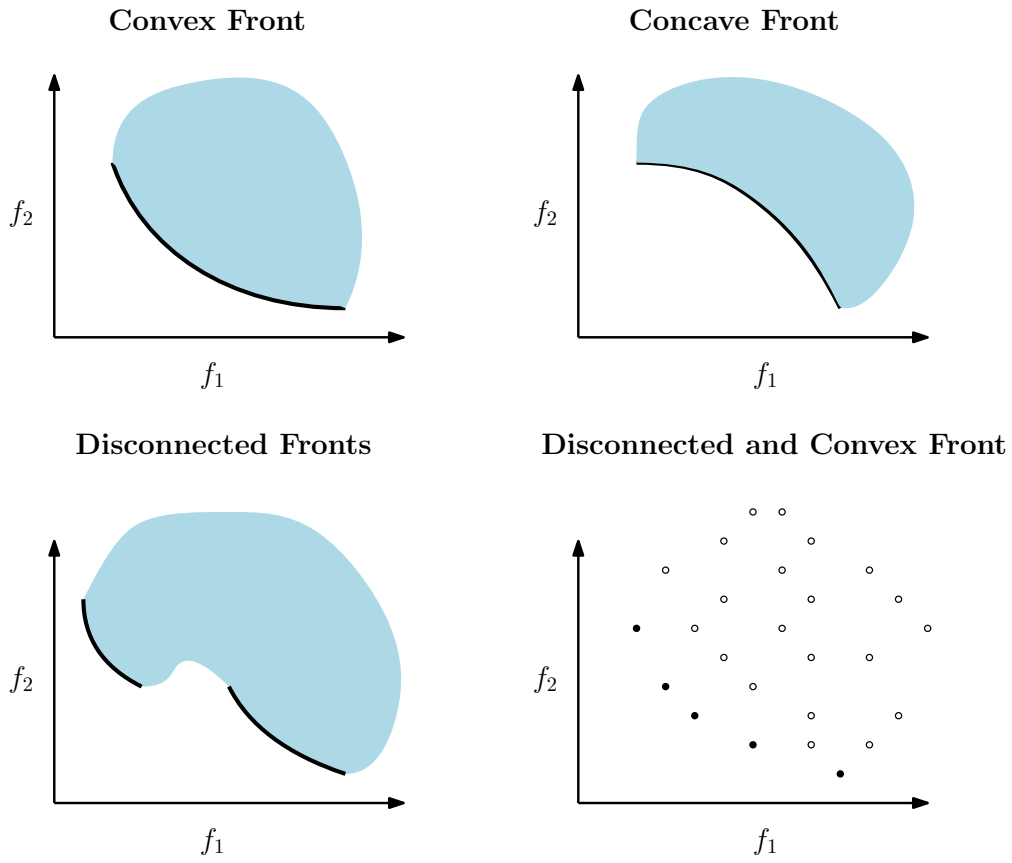


FIGURE D.3: Some shapes of efficient fronts: convex (top left), concave (top right), and disconnected (both on the bottom). Notice that combinatorial problems have disconnected fronts because the solutions are discrete, but they still can be convex or concave, for instance.

which is composed of the individual minima of each objective function, as illustrated in Figure D.4. This point is such that, if there were a feasible alternative which could provide the best out of all criteria, this would be an *ideal* situation, and it would be the chosen solution without much doubt, just like in the single-objective case. However, this is only achieved when the criteria are not in conflict (which can be an indicative of a bad modeling to begin with) or in the ideal world, so we can only use it to help with the decision process.

On the other side, there is the *maximal solution*, defined as

$$\mathbf{y}^{Max} \triangleq \left[\max_{\mathbf{x}\in\mathcal{X}} f_1(\mathbf{x}) \quad \max_{\mathbf{x}\in\mathcal{X}} f_2(\mathbf{x}) \quad \dots \quad \max_{\mathbf{x}\in\mathcal{X}} f_m(\mathbf{x})\right]^T \tag{D.4}$$

which is composed of the individual maxima of each objective (see Figure D.4). Like the ideal solution, $\mathbf{y}^{Max}$ is usually unfeasible as well, although there can be cases in which it is attainable. Together, these points provide the limits of the solutions in the objective space.

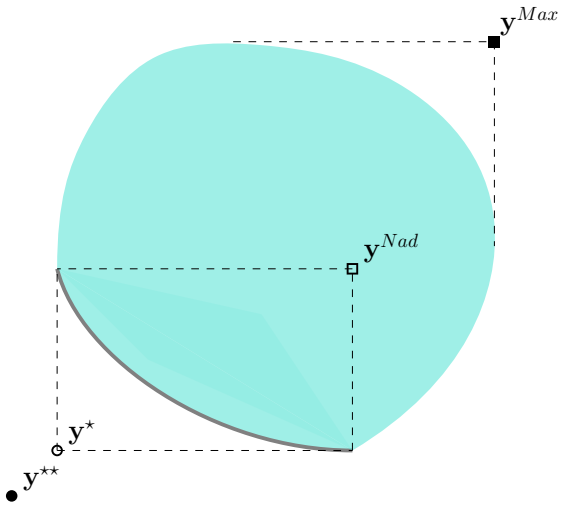


FIGURE D.4: Special points for a feasible objective region. It shows the ideal ($\mathbf{y}^{\star}$), the Nadir ($\mathbf{y}^{Nad}$), the utopian ($\mathbf{y}^{\star\star}$) and the maximal ($\mathbf{y}^{Max}$) solutions.

Unfortunately, the maximal point can be quite distant from the optimal solutions, and, in some cases, even unbounded. Since the attention is normally reserved to the efficient front, it is more useful to provide the limits for this specific region. For that, we have the *Nadir solution* (Figure D.4), defined as

$$\mathbf{y}^{Nad} \triangleq \left[\max_{\mathbf{x}\in\mathcal{X}^*} f_1(\mathbf{x}) \quad \max_{\mathbf{x}\in\mathcal{X}^*} f_2(\mathbf{x}) \quad \dots \quad \max_{\mathbf{x}\in\mathcal{X}^*} f_m(\mathbf{x})\right]^T \tag{D.5}$$

Notice that it is composed of the individual maxima of each criterion, but *restricted to the efficient set* (note the term $\mathbf{x} \in \mathcal{X}^*$). This small detail makes the computation of this point much more difficult, because we cannot simply solve m independent single-objective problems as before. For some simple cases, like with two objectives, we can use a *pay-off table* [119] to compute it exactly. But, for more objectives, this technique normally does not generate correct results. For that, we may require different techniques, as proposed in studies like [123, 124], or even some educated guesses.

The reason why it is important to "waste some time" with these points is that they provide us with the ranges of the solutions in the efficient front (in the case of the ideal and Nadir). To see why this can be relevant, try to answer the following question: suppose you have a current solution $\mathbf{x}_1$ with $\mathbf{f}(\mathbf{x}_1) = [1\ 50]^T$, and someone proposes you change to the other option $\mathbf{x}_2$ with $\mathbf{f}(\mathbf{x}_2) = [2\ 10]^T$. Would it be a nice exchange? Since these alternatives are incomparable, we may have to use some common sense here.

At first sight, we can think of this exchange as giving up one unit of the first objective to improve 40 units of the second. If we have more preference on improving the second one, then, yes, it looks a nice deal. However, what if the first criterion had a range from 0 to 10 and the second from 0 to 1000? The actual trade-off is to actually give up 1/10

$= 0.1$ on the first component and receive $40/1000 = 0.04$ of improvement on the second, which does not represent a good bargain anymore.

This kind of *trade-off* analysis is very common in some methods of solving multi-objective problems [121], and the lack of information on the boundaries of the efficient front may hinder the process. Therefore, a good practice is to work with dimensionless objectives, which is accomplished by performing the following *standardization*:

$$\tilde{f}_i(\mathbf{x}) = \frac{f_i(\mathbf{x}) - y_i^\star}{y_i^{Nad} - y_i^\star}, \quad i = 1, 2, \ldots, m \tag{D.6}$$

If the ideal or Nadir are not easily available, then good approximations of them can be used in place. "Good" here means "suitable for the problem", so they can be very different in each situation; for example, in the proposed Simulated Annealing (section 5.3.1), this scaling is performed with the limits of the current archive. Also, notice that, for some objectives, the ideal can be equal to the Nadir (what may happen in degenerate cases or when the approximations are not very "good"), so the denominator becomes zero. For that, some authors propose the use of an *utopian point* (see Figure D.4), which is just the ideal point dislocated by a very small amount, that is,

$$\mathbf{y}^{\star\star} \triangleq \mathbf{y}^\star - \boldsymbol{\epsilon} \tag{D.7}$$

wherein $\boldsymbol{\epsilon}$ is a vector of small numbers, like 10^{-3}, just to assure that the denominator of equation (D.6) is never zero.

Equation (D.6) maps all the efficient front onto the interval $[0, 1]$, so that some discussions like trade-off analysis and relative importance of criteria can become meaningful [111]. Not only that, but some methods for solving a multi-objective problem of the form (D.1) involve aggregating all functions into a single one by computing weighted sums, taking norms etc., and it does not make much sense to sum kilometers with bananas. This is why these methods also require the standardization of the objectives in order to work properly.

D.2.2.3 Computing efficient solutions

We now know what are Pareto-optimal solutions and how they can be properly compared, so we are able to move on to the process of actually computing them. Most of the methods can be classified in three types:

Scalarization or scalarizing techniques: they convert the multi-objective problem into a single-objective one, such that the optimal solution of the latter (hopefully) corresponds to an efficient candidate of the former. They comprehend techniques like the weighting method [121], ϵ-constraint technique (also called compromise programming) [119], minimization of the distance to a given reference point (e.g., weighted metrics and achievement scalarizing function approaches) [119], the Normal Boundary Intersection [122], the Pascoletti and Serafini method [125] and Adaptive Weighted Sums [126];

Deterministic Methods without Scalarization: methods that adopt deterministic algorithms and minimize the original vector function, without any aggregation. They require that the objective functions are differentiable, so they are valid only for continuous problems. These methods are able to compute provably Pareto-optimal solutions, but usually unable to control where they will lie in the efficient front. Representative methods in this class are the Steepest Descent [127] and a multi-objective version of the Newton Method [128].

Methods based on Populations: instead of evolving a single solution towards an optimal solution, population based methods employ a set of points which are iterated in parallel to approximate the optimum or the efficient front. They do not require special structures of the function (e.g., differentiability and smoothness), have less chances of getting stuck in local optima and can return good solutions in reasonable time, but this comes at the cost of no guarantee of optimality or efficiency, and the demand of a good parameter adjustment, which can be heavily problem dependent and may affect the algorithm's performance [129]. Some good examples of this class are *Evolutionary Algorithms* (EA), *Simulated Annealing* (SA), *Particle Swarm Optimization* (PSO) and *Ant Colony Optimization* (ACO) [60].

Since this is not the focus of this work, I will not go further in explaining these techniques. The reader interested in the applicability of each method, as well as in assumptions and conditions for which they return efficient solutions can check references like [58, 60, 120, 121].

D.2.2.4 Solution process in Multi-objective Optimization

Let us come back to the question of what is an optimal solution in the multi-objective case. While we usually have a single minimum in single-objective optimization, we now have a set of minimal or efficient solutions. Then, we could say that the "optimal

solution" in the vector case is composed of the Pareto-optimal points, which we learned in the previous section how to compute.

This statement can make sense if we follow the aforementioned idea of using the optimization to narrow the search space to a smaller one; in this case, the Pareto set. However, in many situations, the DM can only choose one or maybe a few solutions to actually implement [112], and the efficient set can still be too large to be of any value. Think about it: if we assume that the DM should prefer solutions that are not dominated,[4] the concept of efficiency only tells us which solutions *not to choose*. This is weaker than the optimality concept in single-objective optimization.

So, how can we adopt multi-objective tools to actually help the DM? Instead of keeping it outside of the optimization process like in the scalar case, we need to include the decision maker in the search. With this, the solution process will be composed of two steps: computation of efficient points and the inclusion of preferences. Techniques to execute the former were already mentioned, but how do the DM expresses its preferences? This is the topic of the next section.

D.3 Decision Making models

Decision theory is not a recent field. Also, it is a truly interdisciplinary one, comprising areas like engineering, biology, economics, social sciences etc. Because of that, there are numerous theories of decision making, like attribution theory, schema theory, prospect theory, expected utility, and others that can be seen in many references like [113, 131].

It is common to classify these theories into *normative* and *descriptive* [131]. In simple terms, the first type elucidates how decision makers *should* decide, while the second tries to explain how they actually take the decisions. And, apparently, both situations can be very different [132, 133]. While this simplistic definition may undermine the normative class in practice, much of the advances in the field of Economics were due to normative theories, so it is useful to discuss briefly each model.

[4] This premise is more backed up by common sense. If we modeled the problem appropriately, there is no reason to not prefer a new solution if we can improve without deterioration. The only possible exception I can think of is to consider robust solutions (the ones that have relatively small variations in objective value for small perturbations in the search space), but even so, there are some preliminary studies [130] that include the robustness in the formulation by changing the objective functions or adding penalties, so the first assumption still holds.

D.3.1 Normative theories: the utility approach

Normative theories assume we have a *rational decision maker*, which can be defined as a DM who has "a well-organized and stable system of preferences, and a skill in computation that enables it to calculate, for the alternative courses of action that are available to it, which of these will permit it to reach the highest attainable point on its preference scale" [134]. In other words, we can assume the existence of a *utility function*[5] $u(\cdot) : \mathbb{Y} \mapsto \mathbb{R}$ that maps, to each solution $\mathbf{f}(\mathbf{x})$, a real number $u(\mathbf{f}(\mathbf{x}))$ indicating how desirable it is to the DM. The goal is, then, to find the alternative that optimizes this function, which will be the final solution to the decision problem. Because of this idea, some authors call the decision makers that follow this approach *optimizers* [133].

The concept of utility function may sound abstract at first, but it is actually very natural to many of us. If you ever answered questions like "rate from 1 to 5 stars how happy you are with our services" or provided your own scores to the performance of gymnasts, then you are familiar with them. A utility function can be a table, a scale, or virtually any function that satisfies the properties that [110] $u(\mathbf{f}(\mathbf{x}_1)) > u(\mathbf{f}(\mathbf{x}_2))$ if $\mathbf{x}_1$ is preferred over $\mathbf{x}_2$ [assuming maximization of $u(\cdot)$], and $u(\mathbf{f}(\mathbf{x}_1)) = u(\mathbf{f}(\mathbf{x}_2))$ if they are equally desirable.

People tend to be very good at creating utilities, but not necessarily useful ones. Modeling a trustworthy utility requires a great deal of knowledge about all possible alternatives, and this usually demands time and effort. For instance, if a phone company asks someone to rate from 1 to 10 how good its services are, many people would not have trouble ranking that, even though there are no such numerical standards of quality. Then, if a person was a client of only one company, he could not know if there were better or worse competitors, and his utility would not be too reliable. On the other hand, if he experienced many companies' services, there would be bigger chances that the solution that optimized his utility would be actually a good one.

In the multi-objective case, we can always assume that $u(\mathbf{f}(\mathbf{x}_1)) > u(\mathbf{f}(\mathbf{x}_2))$ if $\mathbf{x}_1$ dominates $\mathbf{x}_2$, so we can concentrate again on the efficient front. Some studies like [118, 119] provide techniques for formulating utilities in very simple cases. In other situations, it may be more effective and useful to approximate a utility by some aggregation function.

[5] Actually, in our case where we ignore the states of nature, the correct term would be *value function*, while utility is reserved to the more general context. However, I will prefer this term because most of the discussion in this section is still valid for both situations.

The most popular technique is to use the weighting method, expressed as

$$\begin{aligned} \text{minimize} \quad & u(\mathbf{f}(\mathbf{x})) = \sum_{i=1}^{m} w_i f_i(\mathbf{x}) \\ \text{subject to} \quad & \sum_{i=1}^{m} w_i = 1 \\ & w_i \geq 0, \qquad i = 1, 2, \ldots, m \\ & \mathbf{x} \in \mathcal{X} \subseteq \mathbb{X} \end{aligned} \tag{D.8}$$

This method has as advantages the fact that each weight can be interpreted as an importance factor to the DM, and much of the theory of vector optimization was already described by the use of weights [135], guaranteeing that the final solution will be efficient.[6] However, there are some assumptions that need to be satisfied for that to hold. First, the functions must be in the same scale [111] [or at least standardized like shown in equation (D.6)] so the importance interpretation makes sense. Second, the Pareto front must be convex [136] (top left and bottom right of Figure D.3).

Even if these conditions do hold, the weighting method still has the drawback of not providing a good way to control the location of the final solution. Das and Dennis in [136] concluded that by showing that even equally spaced weights do not guarantee an equally spaced mapping of the front. In fact, some authors [137] even question the "user-friendliness" of specifying importance in terms of weights in some problems. For example, if you need to buy some beer and barbecue with a given amount of money, how important would one be over another? If beer is more important, can we say 80/20? Why not 90/10, or maybe 60/40? What does more or less ten percent units of importance *actually* mean? In general, these numbers are just as arbitrary as the utility scales mentioned previously (and just think how more arbitrary they can get if we add more items to the list).

There are some behavioral studies [111, 131, 132] stating that people do not actually optimize a utility function when they make decisions, that is, the concept of rational decision maker is not realistic. Additionally, some authors [133] have found that the "optimizers", i.e., the decision makers who try to optimize their utility, are more prone for regretting their final decision. This may be due to the fact that, in order to formulate a reliable utility, we need to know about the most number of alternatives as possible. However, when this becomes impractical and we have to leave some candidate solutions behind, there will be always that lingering doubt that we could do better by searching a little more. Simon in [134] provides some more critiques to these approaches for expressing the DM's preferences and, in [138], he proposes the concept of *satisficing*.

[6] Actually, properly efficient if all weights are positive, and weakly efficient if some are zero [121].

D.3.2 Descriptive theories: the satisficing approach

As already explained, descriptive theories focus on explaining how decision makers actually decide. Possibly the most famous theory in this class is the *satisficing* method [138]. It also assumes the existence of a utility function, but postulates that there is an acceptability threshold. In simple terms, the main idea of this theory is to search for the available alternatives until we find a good enough one. With this less restrictive framework, there is no need to formulate a very trustworthy utility, so the decision process becomes easier.

Although there is a good chance that better alternatives may exist, some studies [131] support the idea that people tend to satisfy rather than optimize. Also, while the optimizer may more often regret its final decision, this is less common with satisficers. In fact, if we found a good enough solution but there is still some time (or budget) available, we can keep searching and replace the solutions as we find some more attractive ones. With this regard, satisficers can go in the same direction as optimizers, but without the regrets they may have.

In the multi-objective context, the most straightforward way for expressing the DM's preferences in this case is by providing *aspiration levels* y_i^r for each objective $i = 1, 2, \ldots, m$, which collectively constitute a *reference point* $\mathbf{y}^r$. The idea is then to try to obtain a solution $\mathbf{x}^*$ such that its outcome vector $\mathbf{f}(\mathbf{x}^*)$ is as close as possible (according to some distance function) to $\mathbf{y}^r$, like shown in Figure D.5. This approach is more practical according to some authors [137]. For instance, instead of trying to say by how much is beer more important than barbecue, we can just say how many cans of beer and kilograms of meat are required, and then try to come as close as possible to this goal with our budget.

One method that follows this philosophy is the weighted metrics [121]

$$\begin{aligned} \text{minimize} \quad & \|\mathbf{f}(\mathbf{x}) - \mathbf{y}^r\|_p \\ \text{subject to} \quad & \mathbf{x} \in \mathcal{X} \subseteq \mathbb{X} \end{aligned} \tag{D.9}$$

in which the distance can be any norm, like, for instance, the Minkowski distance or L_p-norm, given by

$$L_p = \left[\sum_{i=1}^{m} (f_i(\mathbf{x}) - y_i^r)^p \right]^{(1/p)}$$

or also the goal programming approach, as in [121].

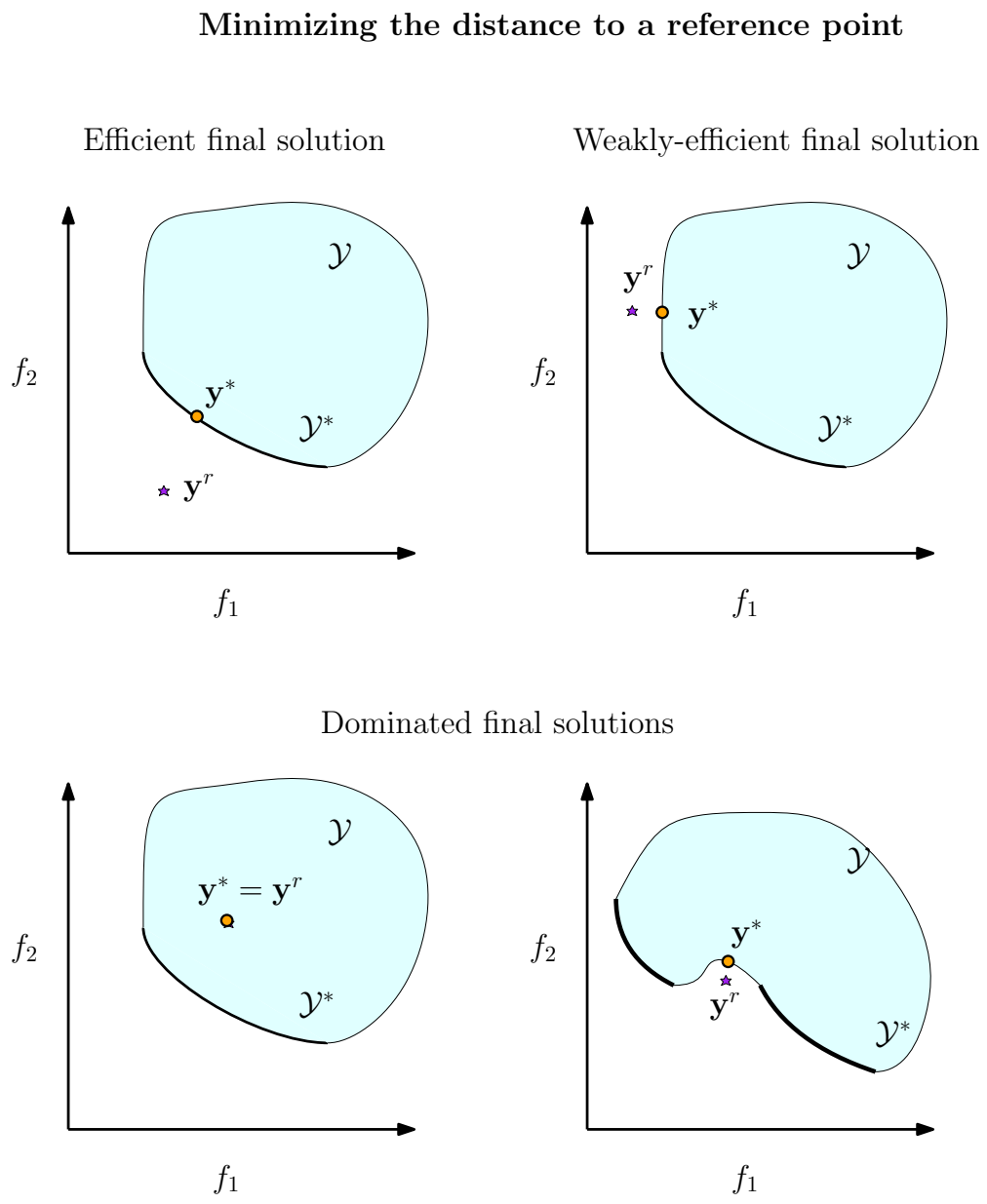


FIGURE D.5: Satisficing approach by minimizing the distance to a given reference point $\mathbf{y}^r$. If the distance is a simple metric, the final solution can be efficient (top left), sometimes weakly-efficient (top right), or even dominated (bottom panels).

The issue with these approaches is that, as also shown in Figure D.5, there is no guarantee that the final solution will be efficient. In particular, if we do not have enough knowledge about the problem and choose $\mathbf{y}^r$ as a feasible and dominated alternative, the final solution will be equal to it (bottom left panel in the figure).

The reader may wonder why would this be so troublesome with the satisficing philosophy in mind. While some DMs may actually be satisfied in this case, we may argue that there is no reason to stop if we know we can improve further. This is different from searching for a *possible* improvement: we actually *know* we can do that. Probably because of this, it was hard to compete with, for instance, the weighting method, which, even with its

drawbacks and restrictions, had the guarantee of efficiency by its side. To overcome that, Zeleny proposed in [139], with his method of *displaced ideal*, to use as reference an utopian point, that is, a point at least a little bit better than the ideal. This would prevent dominated points as solutions.

However, even with this adjustment, the technique of getting close to a reference point was not very appealing, either for the lack of efficiency guarantees, or by the impossibility of freely choosing $\mathbf{y}^r$, which limited the DM to express its preferences. But, considering that this approach is actually promising, Wierzbicki in [137] devised a survey with the entire theory of necessary and sufficient conditions of optimality and existence of minimal solutions based completely on reference points. Put simply, instead of minimizing a regular norm like in (D.9), he proposed to minimize a different, more complicated function, but capable of generating Pareto-optimal points for *any choice* of reference points. He called them *achievement scalarizing functions*, which will be described next.

D.3.2.1 Achievement scalarizing functions

Achievement scalarizing functions (ASF) can be understood as a generalization of a norm, but instead of achieving a minimum value of zero when the current solution is equal to the reference point, it allows negative values if there is room for improvement. Because of this, the DM can be free to express its preferences and, depending on the type of ASF we choose, there is always the guarantee that its minimization will result in an efficient solution. This solves both of the issues listed before.

The ASF of a point $\mathbf{x}$ is computed by a function $s(\cdot,\cdot) : \mathbb{Y} \times \mathbb{Y} \mapsto \mathbb{R}$ which returns a number $s(\mathbf{f}(\mathbf{x}), \mathbf{y}^r)$ giving its (generalized) distance to $\mathbf{y}^r$. When using them, the multi-objective problem can be converted into

$$\begin{aligned} \text{minimize} \quad & s(\mathbf{f}(\mathbf{x}), \mathbf{y}^r) \\ \text{subject to} \quad & \mathbf{x} \in \mathcal{X} \end{aligned} \tag{D.10}$$

There are some properties this function can satisfy [137].

Definition 5. For any two feasible points $\mathbf{x}_1$ and $\mathbf{x}_2$ and a given reference point $\mathbf{y}^r$, an ASF is said to be:

- **increasing**, if $\mathbf{f}(\mathbf{x}_1) \prec \mathbf{f}(\mathbf{x}_2) \implies s(\mathbf{f}(\mathbf{x}_1), \mathbf{y}^r) \leq s(\mathbf{f}(\mathbf{x}_2), \mathbf{y}^r)$;
- **strictly increasing**, if $f_i(\mathbf{x}_1) < f_i(\mathbf{x}_2),\ \forall i = 1, \ldots, m \implies s(\mathbf{f}(\mathbf{x}_1), \mathbf{y}^r) < s(\mathbf{f}(\mathbf{x}_2), \mathbf{y}^r)$;

- **strongly increasing**, if $\mathbf{f}(\mathbf{x}_1) \prec \mathbf{f}(\mathbf{x}_2) \implies s(\mathbf{f}(\mathbf{x}_1), \mathbf{y}^r) < s(\mathbf{f}(\mathbf{x}_2), \mathbf{y}^r)$;

When the ASF is strictly increasing, the solution to (D.10) is weakly optimal, or optimal if it is unique. Also, when it is strongly increasing, the solution is efficient [137]. Figure D.6 shows an example of what should we expect when solving equation (D.10) using a simple norm (Euclidean distance) and a strongly increasing ASF.

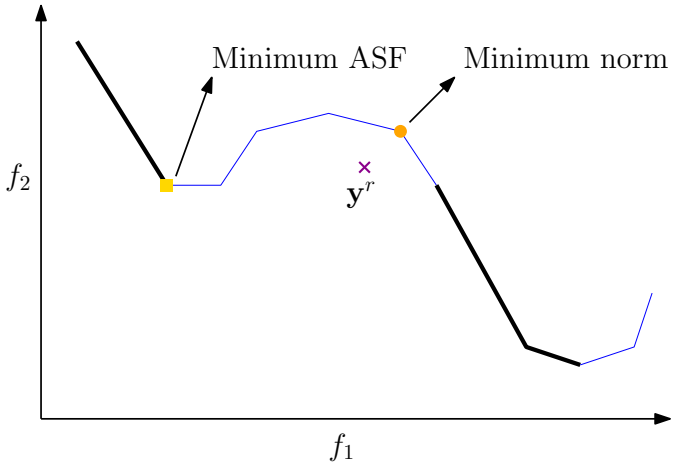


FIGURE D.6: Different results when minimizing an Euclidean distance and a strongly increasing ASF. The solution of the first case is dominated, while the second is efficient.

The ASF can assume various forms [137], but the one used in this work is the (strongly increasing) augmented Chebyshev norm:

$$s(\mathbf{f}(\mathbf{x}), \mathbf{y}^r) = \max_{i \in \{1,2,\ldots,m\}} \{f_i(\mathbf{x}) - y_i^r\} + \rho \sum_{i=1}^{m} (f_i(\mathbf{x}) - y_i^r) \tag{D.11}$$

wherein $\rho > 0$ is a small value, typically 10^{-6}.[7] Notice that this formulation assumes that the objective function values are standardized. Some authors like to include weights multiplying each $(f_i(\mathbf{x}) - y_i^r)$ in order to give different importance to the coordinates. In my opinion, this is just as user-friendly as the original weighting method, so, in order to control the final solution, it is best to change the reference point.

Just like the weighted aggregation, the ASF can also be viewed as a utility function, or, rather, an abstraction of it. We can simply let the DM express its desires in the form a reference point, and then minimize equation (D.10) to return a final solution. Despite the fact that it can be employed with this satisficing philosophy, the reference point approach was proposed to be intrinsically interactive, so that the solution that minimizes equation (D.10) do not need to be final, and the DM can use it to propose new reference points if it thinks there may be more interesting solutions. In this way,

[7] With this formulation, the solutions are, more precisely, *properly* efficient, and their trade-offs are bounded by ρ and $1/\rho$ [112].

the decision maker can *learn* about the problem and better formulate its preferences. Moreover, we have the guarantee of efficiency, that is, if these preferences are attainable with some surplus, we can still improve with this technique. Because of that, this philosophy was known as "quasi-satisficing" [111].

D.3.2.2 Interactive Reference Point method

The interactive approach proposed by Wierzbicki in [137] proceeds as follows:

1. Find the solution to (D.10) for a given reference point;
2. Create new perturbed reference points by adding, to each coordinate of $\mathbf{y}^r$, the distance value d between $\mathbf{y}^r$ and the solution obtained in step 1:

$$\mathbf{y}^{r,i}_{aux} = \mathbf{y}^r + \mathbf{e}_i d \tag{D.12}$$

   where $\mathbf{e}_i$ is the standard basis vector for the i-th coordinate, with zeros in all variables except in the i-th one, where it assumes 1;
3. Solve (D.10) for each of these perturbed reference points.

Figure D.7 illustrates this method. Notice that, when $\mathbf{y}^r$ is far from the efficient front, then the DM had too greater expectations, and the returned solutions are more distant to map other possible points. On the other hand, when the reference point is close to the efficient front, that is, the DM's preferences were more realistic, the other points are also closer, so it can "fine tune" a smaller, more preferred region. With this technique, in each iteration of the decision process, the DM is presented with the closest point to its reference, and also with the auxiliary reference points and their corresponding solutions. With that, the decision maker can opt for any of the alternatives, or maybe formulate new aspiration levels and repeat the process.

D.3.2.3 The Preference-based Adaptive Region-of-interest

The previous technique is useful specially when we are using a single-point algorithm for solving (D.10). However, in some cases, we adopt a population-based method, or even a single-point with an archive to store the best solutions so far. For these situations, I proposed, together with Prof. F. Campelo, a method in [59] very similar to Wierzbicki's, but which takes advantage of the set of points that are already stored.

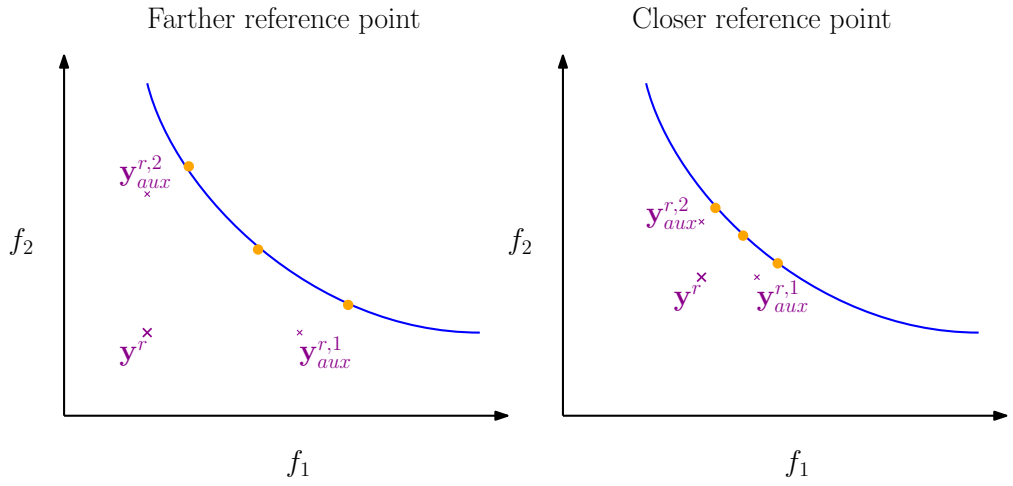


FIGURE D.7: Wierzbicki's interactive approach. Depending on the distance of the reference point to the efficient front, we map it differently.

Before presenting it, we define as a *region of interest* (ROI) the portion of the Pareto-front that possesses solutions that are more interesting to the DM. Since we usually do not have information about the efficient front from the beginning, we have to choose among the available solutions which ones seem interesting or not. There is no universal way of defining this, specially because the DM usually does not know exactly what to expect and which alternatives are optimal, but the interested reader can check some methods in [140, 141]. They tend to use external parameters to do that, which usually means more worries to the DM apart from expressing its preferences.

The method we proposed was named Preference-based Adaptive Region-of-interest (because there was no better name at the time), or PAR, and it requires no parameters apart from the reference point to define the ROI. It follows the steps:

1. Given a set of available solutions $\mathcal{P}$, compute the ASF value of each point using (D.11). Find the point with the smallest ASF, denoted by s_{min}, and let d be the (Euclidean) distance between it and $\mathbf{y}^r$;

2. Compute m auxiliary points by adding d to each coordinate of $\mathbf{y}^r$ (D.13):

$$\mathbf{y}^r_{aux,i} = \mathbf{y}^r + \mathbf{e}_i d \times \text{sign}(s_{min}) \tag{D.13}$$

   wherein sign$(\cdot)$ gives the sign of its argument (+1 if positive and -1 if negative). This is used to prevent problems in cases where the reference point is feasible and the set $\mathcal{P}$ may come to dominate it. In this situation, s_{min} can become negative, and the auxiliary points will be created in the correct direction;

3. Compute the ASF of each point in relation to each of the $\mathbf{y}^r_{aux,i}$, and find the points with the smallest value for each auxiliary point, $\mathbf{x}_{closest,i}$;

4. Determine which points belong to the ROI using the relation (D.14):

$$f_i(\mathbf{x}) \leq f_i(\mathbf{x}_{closest,i}),\ \forall i = 1, 2, \ldots, m \implies \mathbf{x} \in \text{ROI} \tag{D.14}$$

The top panels of Figure D.8 illustrates the above process for two objectives. In [59] it is suggested to include this method in an evolutionary algorithm or any other that updates an archive of points in the following way:

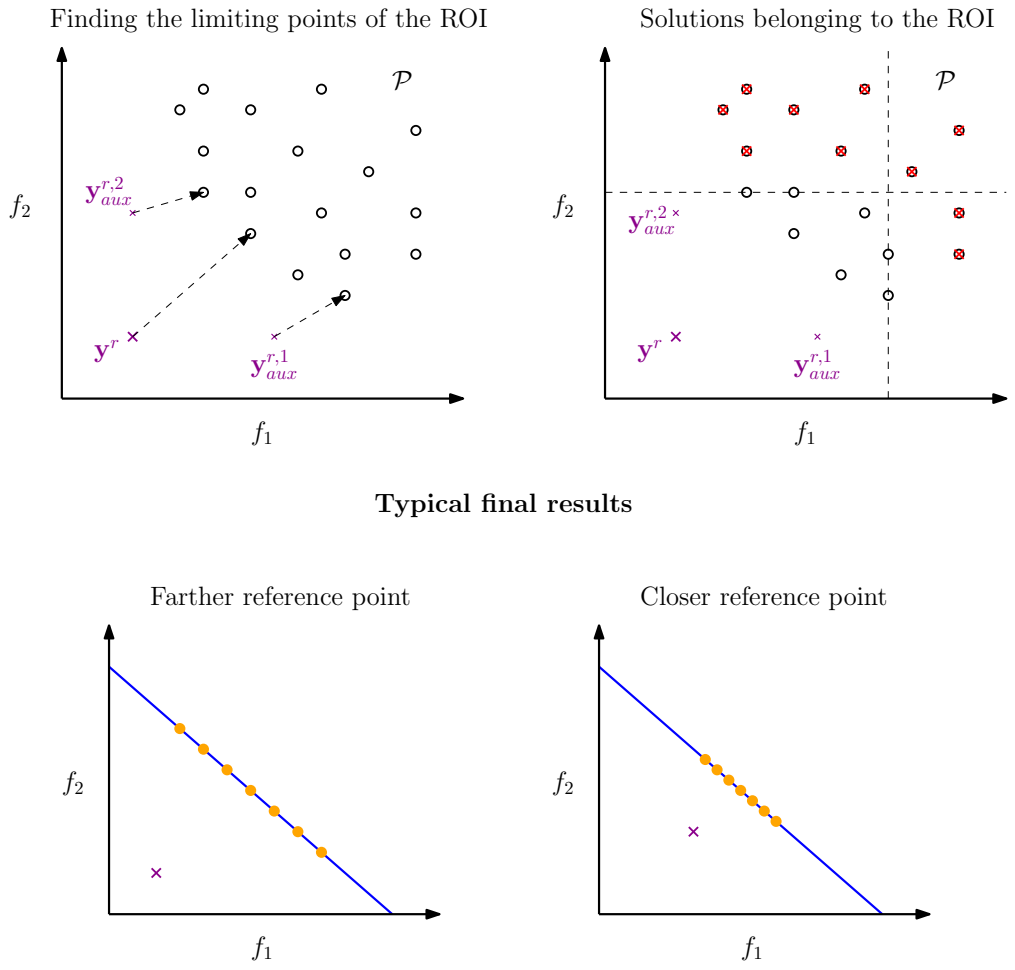


FIGURE D.8: Top panels: computing the points that belong to the region of interest among a set $\mathcal{P}$. Bottom panels: the expected final set for reference points farther (left) and closer (right) to the efficient front.

1. If $\mathcal{P}(t)$ represents the set of available points in iteration t, compute the points that fall within the region of interest according to (D.14), and call them $\mathcal{P}_{ROI}(t) \subseteq \mathcal{P}(t)$;

2. If there is a pre-specified number of solutions μ that $\mathcal{P}$ must possess, adjust this number according to:

   - If $|\mathcal{P}_{ROI}(t)| < \mu$, then include the candidates from $\mathcal{P}(t) \setminus \mathcal{P}_{ROI}(t)$ with the smallest ASF values until $|\mathcal{P}_{ROI}(t)| = \mu$;

- If $|\mathcal{P}_{ROI}(t)| > \mu$, truncate these points according to any criterion, like non-dominated sorting, indicators etc. in order to reduce its size to μ.

3. Assemble the next set $\mathcal{P}(t+1) \leftarrow \mathcal{P}_{ROI}(t)$;
4. Repeat the steps above until a stopping condition is satisfied.

In the end, we expect to have the results shown in the bottom panels of Figure D.8, which have the same interpretation of Figure D.7 when it comes to closer and farther reference points. However, instead of just showing the current best point and the limits of the region of interest, we present some alternatives in between, which may help the DM and possibly alleviate the need for devising new aspiration levels.

This method was first proposed for problems with many objectives, when the size of the Pareto-front is usually too big to be covered completely with a reasonable number of points. Also, it helps to mitigate the convergence issue that Pareto-based algorithms have in these cases. The interested reader can see [59] for more details.

D.4 Employing decision making in multi-objective optimization

As explained in section D.2.2.4, when adopting multi-objective tools to help solving a decision problem, we usually cannot isolate the steps of expressing preferences and computing minimal solutions as in single-objective problems. We now know how to execute each of them individually, so the final step is how to use them together.

The steps that compose the solution process – computation of efficient solutions and expressing preferences – do not need to be performed in any specific order. Depending on when the DM participates in the procedure, some studies [112, 121] like to classify multi-objective methods into *a priori*, *a posteriori*, interactive and "non-preference" techniques.

D.4.1 *A priori* methods

The DM expresses its preferences *before* the optimization, which normally happens by having a utility function or something that approximates it. For example, if we have a nice set of weights for each function or a good reference point, we can use the methods explained in the previous section to solve the decision problem. The advantage of the *a priori* approach is that, once the preferences are modeled, we generally need to optimize a

scalar function, and so the vast body of single-objective optimization is directly available for us.

The bad side of this was already mentioned in section D.3.1: the quality of the final solution depends on how reliable the utility function is, and this is a hard task. Even when expressing the preferences through something more intuitive like a reference point, without a great previous knowledge about the possibilities of the alternatives, the final solution - even if efficient - may be far from interesting.

Therefore, the *a priori* approach is recommended when there is a lot of information available about the problem to be solved, or when performing experiments in test problems only for comparing the performance of two or more algorithms, instead of wanting a final solution for implementation.

D.4.2 *A posteriori* methods

These methods follow a more "natural" approach: first generate efficient solutions, and afterwards select a preferred one. Because of the large number of elements that a Pareto set can contain, these techniques focus on getting a finite "good representation", which simply means that the approximating set should have solutions as close as possible to the efficient ones (in the search or objectives space), and well diversified in order to cover all of its important aspects. In other words, it should have good convergence and diversity, as shown in Figure D.9.

This technique can be performed by running any scalarizing method a number of times and changing its parameters in each execution (varying the weights, reference points etc.). This requires many single-objective optimizations, and some methods may present difficulties when trying to control the diversity of the final set. Another approach is to use population-based algorithms which, because of their ability of approximating many solutions in a single run, are vastly employed with the *a posteriori* philosophy in mind. In fact, it is not uncommon to see in the literature some authors saying that the task of these algorithms is to compute a good representation of the whole efficient front, even though this is just one way of employing them.

The strong point of this approach is that the DM does not have to worry about utilities beforehand: it can focus on the final finite set and pretend it is solving a decision problem of the first category (as explained in the introduction). However, if the efficient front has a high cardinality (which happens frequently in many-objective problems, for instance [59, 142]), the number of approximating points required for a good representation is also large, and the DM can be overwhelmed by them. In [58] there is a more thorough

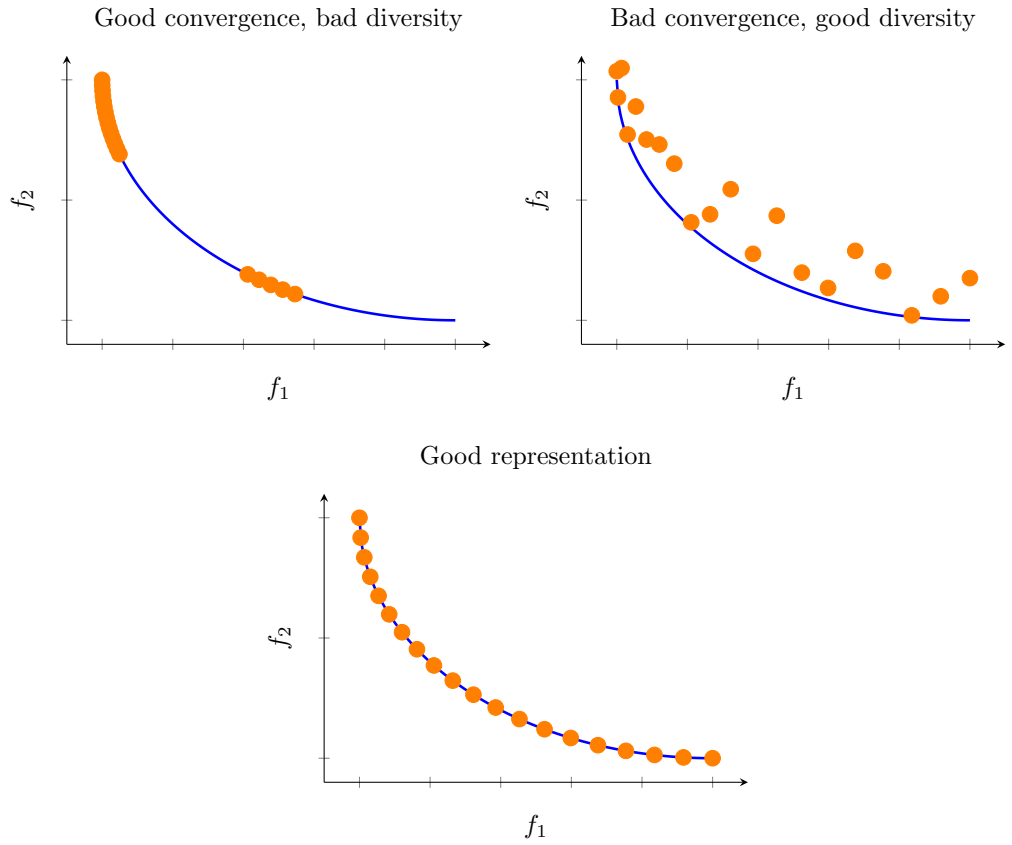


FIGURE D.9: Examples of representations in *a posteriori* methods in the objective space. On the top left, the solutions are close to optimal, but they miss many portions of the front. On the top right, the extension of the approximating set is good, but it is far from the optimal front. Finally, on the bottom we have a representation satisfying both conditions.

discussion about why the *a posteriori* approach is not interesting in many-objective optimization. Hence, it is recommended to employ these techniques in problems with typically two or three objectives.

D.4.3 Interactive methods

This is like a middle ground between the *a priori* and *a posteriori* methods, in which the expression of preferences takes turns with the computation of efficient solutions. It also assumes that the DM has a utility which needs to be optimized, but as it learns about the possibilities, it can model its utility interactively and then the chance of obtaining an interesting solution is greater than in the previous techniques.

An interactive method typically follows the structure: at each iteration, some candidate solutions are presented to the DM, which is queried with questions such as "Do you prefer this new solution over the previous one or are you indifferent with them?", "This is the trade-off between these two solutions. How interesting do you find it?", or "Are

you satisfied with these solutions or do you wish to formulate new preferences?". The success and applicability of this approach depends basically on how often the queries are performed or what type of questions are asked: too frequent questions may overload the DM and jeopardize the process, while the lack of interaction can generate non-interesting solutions.

Some examples of interactive methods are the AHP (Analytic Hierarchy Process) and ELECTRE (Elimination and Choice Translating Reality) [143], as well as the reference point interactive approach described in section D.3.2.2. Basically, any scalarizing method where we change its parameters depending on the DM's preferences can be used with this philosophy.

Interactive methods do not require complete previous knowledge about the problem, as do *a priori* methods; nor do they waste time with undesirable solutions, as is frequently the case for *a posteriori* approaches. However, they have the drawback of being very time consuming for the DM, and there has been some criticism in the literature of methods that rely too much on subjective questions [111]. Because of that, simple approaches like the reference point may be interesting for many problems.

D.4.4 Non-preference based methods

There is no DM, which resembles a situation of a manager shouting "Surprise me" and closing the door to the staff. These methods usually focus on finding a solution that provides a good balance among the objectives, which typically means solutions near the "middle" of the efficient front. These approaches are normally used to provide a starting point to some interactive methods [112].

Deterministic methods with no scalarization (mentioned in section D.2.2.3) can be used in this case, since it just looks for an efficient solution. However, if the DM requires at least a "neutral compromise" among the criteria, the weighting method with equal weights (and dimensionless objectives, of course) and the reference point fixed at the ideal solution are good options for it.

Finally, it is interesting to mention that some other classifications exist in the literature. In special, the one proposed by Rosenthal in [118] uses one class of "partial generation of the efficient front", which is exactly the idea followed by the PAR method (section D.3.2.3). In some way, it combines the first three approaches, since it requires a starting point for expressing the preferences, it returns a (smaller) set of interesting points, and it allows the DM to formulate new aspiration levels and re-execute the search. When or why to use any of these philosophies depends on the application and on the DM.

D.5 Summary

This appendix was possibly a bit overwhelming when compared to the previous ones, so here is a quick summary of the main topics presented here.

Whenever we have to choose among two or more options to solve a problem, we are solving a decision making task. Typically, if we can model it into m functions called objective functions or criteria, which are enclosed in a vector $\mathbf{f}(\cdot) : \mathbb{X} \mapsto \mathbb{Y}$ that needs to be optimized, we can use optimization theory to help finding the most preferred solution according to a decision maker (DM).

Optimization techniques replace the notion of "preference" with the concept of optimality. In single-objective problems, there is usually only one optimal solution, so these two concepts often coincide, and there is no need for the DM to be involved. In the multi-objective situation, the notion of optimality is translated to efficiency, which is weaker then in the scalar case. The consequence of this is that, instead of only one optimal solution, we have a set of *candidate* solutions, called efficient set/front (in the variable/objective spaces, respectively), which can have many, or even infinite, solutions. Because of this, the DM must now participate in a more active way.

A decision problem can be solved with multi-objective techniques using two steps:

- Computation of efficient points: there are many methods available for that. This appendix mentioned scalarization techniques, deterministic methods without scalarization, and population-based approaches.
- Inclusion of preferences: some typical methods are the use of utility functions, which can be approximated with weights; or providing aspiration levels, which collectively constitute a reference point $\mathbf{y}^r$, which is the preferred approach in this work.

These steps do not have to be executed in any specified order. Depending on when the DM expresses its preferences, optimization methods are usually classified into (i) *a priori*, (ii) *a posteriori*, (iii) interactive and (iv) non-preference based. This classification is not unique, and there are other ones in the literature, such as "partial generation of the efficient front", which describes algorithms limiting their final sets to a given region of interest, and they seem to be growing in attention. In this work, given the small number of objectives, we opted for the *a posteriori* philosophy. However, for future works, we may include the PAR approach to help with the treatment of a growing number of quality indices, if they seem required to better model the problem.

The goal of this chapter is to build enough knowledge to fill the usual gaps present in most load restoration problems (which tend to assume that as given from the reader), and also to help determining some weak points of the studies analyzed here and other ones the reader may come across.

Appendix E

Branch and Bound technique for the Load Restoration problem

The proposed algorithm works in a different space (the space of permutations of all switches in the restoration subgraph $\mathbb{X} = \mathfrak{S}_{\mathcal{E}_R}$) instead of handling directly the set of all possible sequences of maneuvers. In this section we develop an enumeration technique that works directly with them. It has the benefit of being able to approximate the true Pareto-optimal front, but it may take an impractically long time to be used in commercial applications. Nevertheless, I do not plan on selling this method, only to use it to assess the quality of the proposed method.

The Branch and Bound (BB) approach is presented in Algorithm 8, which also follows a **Case 1** version of the general restoration plan of section 4.2.2. We start with a set $\mathfrak{M}$ of empty sub-sequences (line 2), which simply means the post-fault configuration. For each *recoverable* CO switch e^{CO} (i.e., a Type 3 switch using the notation of section 4.2.1), we try to construct a stage by combining e^{CO} with all combinations of 0, 1, 2 and 3 CC switches that are downstream to it in the *oos* region (lines 11-20). If required, an isolating switch is added to prevent energizing a fault (line 14). Thus, we limit a stage to at most six maneuvers, which we believe is a large enough number to represent real world cases. If this stage generates a feasible configuration (lines 15-19), we add it to the available set of sub-sequences. We repeat this process using the configuration generated by each of the stored sub-sequences as starting point, which goes on until there is no available solution. At the end, we extract the non-dominated solutions from the archives of all available sub-sequences of maneuvers (line 23).

The tree of possibilities may grow very rapidly in large networks, so the pure enumeration version should be employed only in simple faulted scenarios. To allow its use in diverse cases of faults, we reserved in line 21 the possibility of adding any heuristic to speed up

Data: Restoration subgraph $\mathcal{G}_R$ and all available switches $\mathcal{E}_R$;
Result: Archives $\mathcal{P}^A_{\mathbf{y}}$ and $\mathcal{P}^A_M$ of Pareto-optimal maneuvers and objective values;

1 $\mathcal{P}^A_{\mathbf{y}}, \mathcal{P}^A_M \longleftarrow \emptyset, \emptyset$; `// Initialize archives of maneuvers and objective values`
2 $\mathfrak{M} \longleftarrow \emptyset$; `// Initialize set of subsequences`
3 $\mathcal{E}^{CO} \longleftarrow$ AllRecoverableCOSwitches$(\mathcal{G}_R, \mathcal{E}_R)$;
4 **do**
5 $M \longleftarrow$ FirstSubsequence$(\mathfrak{M})$;
6 $\mathfrak{M} \longleftarrow \mathfrak{M} \setminus M$;
7 RunManeuvers$(\mathcal{G}_R, M)$; `// Run maneuvers of` M `in` $\mathcal{G}_R$
8 $\mathcal{E}^{CO}_{sub} \longleftarrow$ SubsetOfRecoverableCOSwitches$(\mathcal{G}_R, \mathcal{E}^{CO})$;
9 **for** $e^{CO} \in \mathcal{E}^{CO}_{sub}$ **do**
10 $\mathcal{E}^{CC}_{down} \longleftarrow$ DownstreamCCSwitches$(\mathcal{G}_R, e^{CO})$;
11 **for** *Each combination* $\mathcal{E}^{CC}_{sub}$ *of 0, 1, 2 and 3 switches of* $\mathcal{E}^{CC}_{down}$ **do**
12 $\mathcal{S} \longleftarrow \{\mathcal{E}^{CC}_{sub}, e^{CO}\}$; `// Form a stage`
13 RunManeuvers$(\mathcal{G}_R, \mathcal{S})$;
14 $\mathcal{S} \longleftarrow$ AddIsolationSwitch$(\mathcal{G}_R, \mathcal{S})$; `// In case a fault was energized`
15 **if** $\mathcal{G}_R$ *is feasible* **then**
16 $\mathbf{y} \longleftarrow$ EvaluateSubsequence$(\mathcal{G}_R, M \cup \mathcal{S})$; `//` $\mathbf{y} = [S_{NR}, E_{NS}]$
17 $\mathcal{P}^A_{\mathbf{y}} \longleftarrow \mathcal{P}^A_{\mathbf{y}} \cup \{\mathbf{y}\}$;
18 $\mathcal{P}^A_M \longleftarrow \mathcal{P}^A_M \cup \{M \cup \mathcal{S}\}$;
19 $\mathfrak{M} \longleftarrow \mathfrak{M} \cup \{M \cup \mathcal{S}\}$
20 **end**
21 RevertManeuvers$(\mathcal{G}_R, \mathcal{S})$;
22 **end**
23 **end**
24 /* Add possible pruning heuristics here */
25 **while** $\mathfrak{M} \neq \emptyset$;
26 RemoveDominatedSolutions$(\mathcal{P}^A_{\mathbf{y}}, \mathcal{P}^A_M)$; `// Leave only Pareto-optimal solutions`

Algorithm 8: Branch and Bound procedure to compute (or approximate) the Pareto-optimal solutions of a problem.

the search. For the complex scenario considered in section 6.2.2, we suggest to replace line 21 with the following pruning heuristic

$$\text{RemoveDominatedSolutions}(\mathcal{P}^A_{\mathbf{y}}, \mathcal{P}^A_M, \mathfrak{M})$$

This heuristic, in simple terms, keeps only the non-dominated solutions after checking all feasible sequences with only one stage, then repeats it for two stages, and so on. This early pruning accelerates the search, but it may lose some solutions with bad objective values in early stages, but which may be better (or even Pareto-optimal) when considering a more complete sequence. Even so, this procedure complies better with the usual techniques of mathematical programming approaches, as explained in the literature review.

Bibliography

[1] André L Maravilha, Fillipe Goulart, Eduardo G Carrano, and Felipe Campelo. Scheduling Maneuvers for the Restoration of Electric Power Distribution Networks: Formulation and Heuristics. *Electric Power Systems Research*, 163:301–309, 2018.

[2] Fillipe Goulart, André L Maravilha, Eduardo G Carrano, and Felipe Campelo. Permutation-based optimization for the load restoration problem with improved time estimation of maneuvers. *International Journal of Electrical Power & Energy Systems*, 101:339–355, 2018.

[3] Thomas Allen Short. *Electric power distribution handbook*. CRC press, 2014.

[4] Amadeu Casal Caminha. *Introdução à proteção dos sistemas elétricos*. E. Blucher, 1st edition, 1977 (In portuguese).

[5] S.R. Bhide. *Fundamentals of Power System Protection*. Prentice-Hall Of India Pvt. Limited, 2004.

[6] Rubén Romero, John F Franco, Fábio B Leão, Marcos J Rider, and Eliane S de Souza. A new mathematical model for the restoration problem in balanced radial distribution systems. *IEEE Transactions on Power Systems*, 31(2):1259–1268, 2016.

[7] A.R. Bergen and V. Vittal. *Power Systems Analysis*. Prentice Hall, 2000.

[8] J.D. Glover, M.S. Sarma, and T. Overbye. *Power System Analysis & Design, SI Version*. Cengage Learning, 2012.

[9] J. De Kock and C. Strauss. *Practical Power Distribution for Industry*. Practical professional books from Elsevier. Elsevier Science, 2004.

[10] D. B. Campbell. Electric Power Distribution Systems Operations. Technical Report April, 1990.

[11] Angelo Baggini. *Handbook of power quality*. John Wiley & Sons, 2008.

[12] Procedimentos de Distribuição de Energia ANEEL. Elétrica no Sistema Elétrico Nacional – PRODIST: Módulo 8-Qualidade de Energia Elétrica. *Revisão*, 5:76, 2007 (In portuguese).

[13] Turan Gönen. Electrical power distribution system engineering, 1986.

[14] R. Isermann. *Fault-Diagnosis Systems: An Introduction from Fault Detection to Fault Tolerance*. Springer Berlin Heidelberg, 2005. ISBN 9783540241126. URL https://books.google.com.br/books?id=6yUfoZhGMY0C.

[15] Fahrudin Mekic, Ken Alloway, Cleber Angelo, and Robert Goodin. Fault detection isolation and restoration on the feeder (FDIR): Pick your technology. In *CI RED, 21st International Conference on Electricity Distribution, Frankfurt*, 2011.

[16] Jian Guo Liu, Xinzhou Dong, Xingying Chen, Xiangqian Tong, Xiaoqing Zhang, and Shiming Xu. *Fault Location and Service Restoration for Electrical Distribution Systems*. John Wiley & Sons, 2016.

[17] Sylvie Thiébaux and Marie-Odile Cordier. Supply restoration in power distribution systems—a benchmark for planning under uncertainty. In *Sixth European Conference on Planning*, 2014.

[18] Sylvie Thiébaux, Marie-Odile Cordier, Olivier Jehl, and Jean-Paul Krivine. Supply restoration in power distribution systems: A case study in integrating model-based diagnosis and repair planning. In *Proceedings of the Twelfth international conference on Uncertainty in artificial intelligence*, pages 525–532. Morgan Kaufmann Publishers Inc., 1996.

[19] Chia-Hung Lin, Hui-Jen Chuang, Chao-Shun Chen, Chung-Sheng Li, and Chin-Ying Ho. Fault detection, isolation and restoration using a multiagent-based Distribution Automation System. *2009 4th IEEE Conference on Industrial Electronics and Applications*, (June):2528–2533, 2009. doi: 10.1109/ICIEA.2009.5138663. URL http://ieeexplore.ieee.org/document/5138663/.

[20] Namhun Cho, Rajatha Bhat, and Jung Ho Lee. Smart fault isolation and service restoration under consideration of abnormal conditions in distribution systems. *Proceedings - 2015 International Symposium on Smart Electric Distribution Systems and Technologies, EDST 2015*, pages 362–367, 2015. doi: 10.1109/SEDST.2015.7315235.

[21] Isaiah Lankham, Bruno Nachtergaele, and Anne Schilling. *Permutations and the Determinant of a Square Matrix*, chapter Chapter 8, pages 81–94. doi: 10.1142/9789814723787_0008.

[22] Yogendra Kumar, Biswarup Das, and Jaydev Sharma. Service restoration in distribution system using non-dominated sorting genetic algorithm. *Electric Power Systems Research*, 76(9-10):768–777, 2006. ISSN 03787796. doi: 10.1016/j.epsr.2005.10.008.

[23] S. Toune, H. Fudo, T. Genji, Y. Fukuyama, and Y. Nakanishi. Comparative study of modern heuristic algorithms to service restoration in distribution systems. *IEEE Transactions on Power Delivery*, 17(1):173–181, 2002. ISSN 08858977. doi: 10.1109/61.974205.

[24] Leandro Tolomeu Marques. *Restabelecimento de energia em sistemas de distribuição considerando aspectos práticos*. PhD thesis, Universidade de São Paulo, 2018 (In portuguese).

[25] Eduardo Gontijo Carrano, Gisele P da Silva, Edgard P Cardoso, and Ricardo HC Takahashi. Subpermutation-based evolutionary multiobjective algorithm for load restoration in power distribution networks. *IEEE Transactions on evolutionary computation*, 20(4):546–562, 2016.

[26] Isamu Watanabe. An ACO algorithm for service restoration in power distribution systems. In *Proceedings of the IEEE Congress on Evolutionary Computation, CEC 2005, 2-4 September 2005, Edinburgh, UK*, pages 2864–2871, 2005. doi: 10.1109/CEC.2005.1555054. URL http://dx.doi.org/10.1109/CEC.2005.1555054.

[27] F. Mohammadi and H. Afrakhteh. Optimal load restoration in distribution network using intentional islanding. *Journal of Electrical Engineering*, 12(4):108–113, 2012. ISSN 15824594.

[28] Chen-Ching Liu, Vijay Vittal, Kevin Tomsovic, Wei Sun, Chong Wang, Raul Perez, Torrey Graf, Benjamin Wells, Hui Yuan, and Benyamin Moradzadeh. Development and Evaluation of System Restoration Strategies from a Blackout. Technical report, Power System Engin. Res. Center, 2009.

[29] Ji Zhong Zhu. Optimal reconfiguration of electrical distribution network using the refined genetic algorithm. *Electric Power Systems Research*, 62(1):37–42, 2002.

[30] T.H. Cormen, C.E. Leiserson, R.L. Rivest, and C. Stein. *Introduction To Algorithms*. MIT Press, 2001.

[31] N.D.R. Sarma, V.C. Prasad, K.S. Prakasa Rao, and V. Sankar. A New Network Reconfiguration Technique for Service Restoration in Distribution Networks. *IEEE Transactions on Power Delivery*, 9(4):1936–1942, 1994.

[32] T. Nagata, H. Sasaki, and R. Yokoyama. Power System Restoration by Joint Usage of Expert System and Mathematical Programming Approach. *IEEE Transactions on Power Systems*, 10(3):1473–1479, 1995. ISSN 15580679. doi: 10.1109/59.466501.

[33] Hassan Hijazi and Sylvie Thiébaux. Optimal distribution systems reconfiguration for radial and meshed grids. *International Journal of Electrical Power and Energy Systems*, 72:136–143, 2015. ISSN 01420615. doi: 10.1016/j.ijepes.2015.02.026.

[34] K. Aoki, K. Nara, M. Itoh, T. Satoh, and H. Kuwabara. A new algorithm for service restoration in distribution systems. *IEEE Transactions on Power Delivery*, 4(3): 1832–1839, 1989.

[35] Susheela Devi, D P Sen Gupta, and Samuel Sargunaraj. Optimal restoration of supply following a fault on large distribution systems. *International Conference on Advances in Power System Control, Operation and Management, 1991. APSCOM-91., 1991*, pages 508–513 vol.2, 1991.

[36] K. N. Miu, H. D. Chiand, B. Yuan, and G. Darling. Fast service restoration for large-scale distribution systems with priority customers and constraints. *IEEE Transactions on Power Delivery*, 13(3):789–795, 1998.

[37] Karen Nan Miu, Hsiao Dong Chiang, and Russell J. McNulty. Multi-tier service restoration through network reconfiguration and capacitor control for large-scale radial distribution networks. *IEEE Transactions on Power Systems*, 15(3):1001–1007, 2000. ISSN 08858950. doi: 10.1109/59.871725.

[38] Dariush Shirmohammadi. Service restoration in distribution networks via network reconfiguration. *IEEE Transactions on Power Delivery*, 7(2):952–958, 1992. ISSN 19374208. doi: 10.1109/61.127104.

[39] S. P. Singh, G. S. Raju, G. K. Rao, and M. Afsari. A heuristic method for feeder reconfiguration and service restoration in distribution networks. *International Journal of Electrical Power and Energy Systems*, 31(7-8):309–314, 2009. ISSN 01420615. doi: 10.1016/j.ijepes.2009.03.013. URL http://dx.doi.org/10.1016/j.ijepes.2009.03.013.

[40] T. T. Borges, S. Carneiro, P. A N Garcia, and J. L R Pereira. A new OPF based distribution system restoration method. *International Journal of Electrical Power and Energy Systems*, 80:297–305, 2016. ISSN 01420615. doi: 10.1016/j.ijepes.2016.01.024.

[41] T. D Sudhakar. Power Restoration in Distribution Network Using MST Algorithms. *New Frontiers in Graph Theory*, pages 285–306, 2012. doi: 10.5772/35897.

[42] J. Nahman and G. Strbac. A new algorithm for service restoration in large-scale urban distribution systems. *Electric Power Systems Research*, 29(3):181–192, 1994. ISSN 03787796. doi: 10.1016/0378-7796(94)90013-2.

[43] Srdjan Dimitrijevic and Nikola Rajakovic. An innovative approach for solving the restoration problem in distribution networks. *Electric Power Systems Research*, 81(10):1961–1972, 2011. ISSN 03787796. doi: 10.1016/j.epsr.2011.06.005. URL http://dx.doi.org/10.1016/j.epsr.2011.06.005.

[44] Mark. M.R. Ibrahim, H. A. Mostafa, M. M. A. Salama, R. El-Shatshat, and K. B. Shaban. A graph-theoretic service restoration algorithm for power distribution systems. *2018 International Conference on Innovative Trends in Computer Engineering (ITCE)*, pages 338–343, 2018. doi: 10.1109/ITCE.2018.8316647. URL http://ieeexplore.ieee.org/document/8316647/.

[45] Thomas Stützle. *Local search algorithms for combinatorial problems - analysis, improvements, and new applications*, volume 220 of *DISKI*. Infix, 1999.

[46] Helena R. Lourenço, Olivier C. Martin, and Thomas Stützle. Iterated Local Search: Framework and Applications Handbook of Metaheuristics. volume 146 of *International Series in Operations Research & Management Science*, chapter 12, pages 363–397. Springer US, Boston, MA, 2010. doi: 10.1007/978-1-4419-1665-5_12. URL http://dx.doi.org/10.1007/978-1-4419-1665-5_12.

[47] Eduardo M Kalinowski, Maria Teresinha, Arns Steiner, and Celso Carnieri. The Problem of Restoration of Distribution Networks: A Heuristic Method. *Journal of Computer Science*, 7(3):104–111, 2007.

[48] Vinícius Jacques Garcia and Paulo Morelato França. Multiobjective service restoration in electric distribution networks using a local search based heuristic. *European Journal of Operational Research*, 189(3):694–705, 2008. ISSN 03772217. doi: 10.1016/j.ejor.2006.07.048.

[49] Ravindra K. Ahuja, Thomas L. Magnanti, and James B. Orlin. *Network Flows: Theory, Algorithms, and Applications*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1993.

[50] Sean Luke. *Essentials of Metaheuristics*. Lulu, second edition, 2013. Available for free at http://cs.gmu.edu/∼sean/book/metaheuristics/.

[51] A Augugliaro, L Dusonchet, and E Riva Sanseverino. Multiobjective service restoration in distribution networks using an evolutionary approach and fuzzy sets. *International Journal of Electrical*

Power & Energy Systems, 22(2):103–110, 2000. ISSN 01420615. doi: 10.1016/S0142-0615(99)00040-X. URL http://www.scopus.com/inward/record.url?eid=2-s2.0-0033881744{&}partnerID=tZOtx3y1{%}5Cnhttp://linkinghub.elsevier.com/retrieve/pii/S014206159900040X.

[52] Yogendra Kumar, Biswarup Das, and Jaydev Sharma. Genetic algorithm for supply restoration in distribution system with priority customers. *2006 9th International Conference on Probabilistic Methods Applied to Power Systems, PMAPS*, 2006. doi: 10.1109/PMAPS.2006.360295.

[53] Yogendra Kumar, Biswarup Das, and Jaydev Sharma. Multiobjective, multiconstraint service restoration of electric power distribution system with priority customers. *IEEE Transactions on Power Delivery*, 23(1):261–270, 2008. ISSN 08858977. doi: 10.1109/TPWRD.2007.905412.

[54] A.C.B. Delbem, A. de Carvalho, and N.G. Bretas. Optimal energy restoration in radial distribution systems using a genetic approach and graph chain representation. *Electric Power Systems Research*, 67(3):197–205, 2003. ISSN 03787796. doi: 10.1016/S0378-7796(03)00126-3.

[55] P.M.S. Carvalho, L.A.F.M. Ferreira, and L.M.F. Barruncho. Optimization approach to dynamic restoration of distribution systems. *International Journal of Electrical Power & Energy Systems*, 29(3):222–229, 2007. ISSN 01420615. doi: 10.1016/j.ijepes.2006.07.004.

[56] Alexandre C. B. Delbem, André Carlos Ponce Leon Ferreira de Carvalho, Claudio A. Policastro, Adriano K. O. Pinto, Karen Honda, and Anderson C. Garcia. Node-depth encoding for evolutionary algorithms applied to network design. In *Genetic and Evolutionary Computation - GECCO 2004, Genetic and Evolutionary Computation Conference, Seattle, WA, USA, June 26-30, 2004, Proceedings, Part I*, pages 678–687, 2004. doi: 10.1007/978-3-540-24854-5_70. URL http://dx.doi.org/10.1007/978-3-540-24854-5_70.

[57] D.S. Sanches, J.B.A. London Jr., A.C.B. Delbem, R.S. Prado, F.G. Guimarães, O.M. Neto, and T.W. De Lima. Multiobjective evolutionary algorithm with a discrete differential mutation operator developed for service restoration in distribution systems. *International Journal of Electrical Power and Energy Systems*, 62:700–711, 2014. ISSN 01420615. doi: 10.1016/j.ijepes.2014.05.008.

[58] Fillipe Goulart. Preference-guided Evolutionary Algorithms for Optimization with Many Objectives. Master's thesis, Universidade Federal de Minas Gerais, 2014.

[59] Fillipe Goulart and Felipe Campelo. Preference-guided evolutionary algorithms for many-objective optimization. *Information Sciences*, 329:236 – 255, 2016. ISSN 0020-0255. doi: http://dx.doi.org/10.1016/j.ins.2015.09.015. Special issue on Discovery Science.

[60] AP Engelbrecht. *Computational intelligence: An Introduction*. John Wiley & Sons, 2007.

[61] Tiago Silveira Lambert-Torres, Luiz Eduardo da Silva, and Germano. Restabelecimento de sistemas elétricos de potência com a otimização por colônia de formigas. *Xlliisbpo*, pages 2123–2134, 2009 (In portuguese).

[62] W. P. Luan, Malcolm R. Irving, and Jeremy S. Daniel. Genetic algorithm for supply restoration and optimal load shedding in power system distribution networks. *IEE Proceedings-Generation, Transmission and Distribution*, 149(2):145–151, 2002.

[63] Seyhan Civanlar, JJ Grainger, Ho Yin, and SSH Lee. Distribution feeder reconfiguration for loss reduction. *IEEE Transactions on Power Delivery*, 3(3):1217–1223, 1988.

[64] Kalyanmoy Deb and Amrit Pratap. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, 2002.

[65] J. Dréo, A. Pétrowski, P. Siarry, and E. Taillard. *Metaheuristics for Hard Optimization*. Springer, 2006.

[66] Peng Tian, Jian Ma, and Dong Zhang. Application of the simulated annealing algorithm to the combinatorial optimization problem with permutation property: An investigation of generation mechanism. *Eur. J. Operational Research*, 118(1): 81–94, 1999. ISSN 03772217.

[67] P Engrand and X Mouney. Une méthode originale d'optimisation multi-objectif. *Collection de notes internes de la Direction des études et recherches. Mathématiques, informatique, télécommunications*, 1998 (In French).

[68] G.W. Oehlert. *A First Course in Design and Analysis of Experiments*. W. H. Freeman, 2000. ISBN 9780716735106.

[69] Michael Hughes. Introduction to Nonparametric Statistics Using R. Technical Report January, Miami University, 2012.

[70] Miklós Bóna. *A Walk Through Combinatorics : An Introduction to Enumeration and Graph Theory*. World Scientific Pub. cop., New Jersey, 2006.

[71] W.H. Kersting. *Distribution System Modeling and Analysis, Third Edition*. Taylor & Francis, 2012. ISBN 9781439856222.

[72] Neng Fan and Mehdi Golari. Integer Programming Formulations for Minimum Spanning Forests and Connected Components in Sparse Graphs. In *International Conference on Combinatorial Optimization and Applications*, pages 613–622. Springer, 2014.

[73] Leslie Hewitson, Mark Brown, and Ramesh Balakrishnan. *Practical power system protection*. Elsevier, 2004.

[74] Hesam Hosseinzadeh. Distribution system protection. *University of Western Ontario*, 2008.

[75] MH Hussain, SRA Rahim, and I Musirin. Optimal overcurrent relay coordination: a review. *Procedia Engineering*, 53:332–336, 2013.

[76] Radha Thangaraj, Thanga Raj Chelliah, and Millie Pant. Overcurrent relay coordination by Differential Evolution algorithm. *PEDES 2012 - IEEE International Conference on Power Electronics, Drives and Energy Systems*, (December), 2012. doi: 10.1109/PEDES.2012.6484298.

[77] Reza Mohammadi, Hossein Askarian Abyaneh, Farzad Razavi, Majid Al-Dabbagh, and Seyed H.H. Sadeghi. Optimal relays coordination efficient method in interconnected power systems. *Journal of Electrical Engineering*, 61(2):75–83, 2010. ISSN 13353632. doi: 10.2478/v10187-010-0011-x.

[78] A Akhikpemelo, MJE Evbogbai, and MS Okundamiya. Overcurrent relays coordination using matlab model. 2018.

[79] Manohar Singh, Vishnuvardhan Telukunta, and SG Srivani. Enhanced real time coordination of distance and user defined over current relays. *International Journal of Electrical Power & Energy Systems*, 98:430–441, 2018.

[80] Murali Matcha, S Kumar Papani, and Vijetha Killamsetti. Protective relaying scheme of distributed generation connected radial distribution system. *International Journal of Energy and Power Engineering*, 2(3):121–127, 2013.

[81] Ahmed Kamel, M. A. Alaam, Ahmed M. Azmy, and A. Y. Abdelaziz. Protection coordination of distribution systems equipped with distributed generations. *Electrical and Electronics Engineering: An International Journal (ELELIJ)*, 2(2): 1–13, 2013.

[82] Angel Fernández Sarabia. Impact of distributed generation on distribution system. *Aalborg University, Denmark*, 2011.

[83] Muhammad Yousaf and Tahir Mahmood. Protection coordination for a distribution system in the presence of distributed generation. *Turkish Journal of Electrical Engineering & Computer Sciences*, 25(1):408–421, 2017.

[84] Göran Andersson. Modelling and analysis of electric power systems. *ETH Zurich, september*, 2008.

[85] Lynn Powell. *Power system load flow analysis*. McGraw Hill professional, 2004.

[86] Padmaja Thatoi and Aradhana Pradhan. *Study on the performance of newton–raphson load flow in distribution systems*. PhD thesis, 2012.

[87] C Trevino. Cases of difficult convergence in load-flow problems. *IEEE Paper*, 1 (2):3, 1970.

[88] Jen-Hao Teng. A direct approach for distribution system load flow solutions. *IEEE Transactions on power delivery*, 18(3):882–887, 2003.

[89] Fawwaz T Ulaby, Umberto Ravaioli, and Eric Michielssen. *Fundamentals of applied electromagnetics*. Prentice Hall, 2014.

[90] Alcir José Monticelli. *Fluxo de carga em redes de energia elétrica*. E. Blucher, 1983 (In portuguese).

[91] Charles A Desoer. *Basic circuit theory*. Tata McGraw-Hill Education, 2009.

[92] Steven W Blume. *Electric power system basics for the nonelectrical professional*. John Wiley & Sons, 2016.

[93] Jakub Kepka. Load modelling for power system analysis. *Poland: Wroclaw University of Technology*, pages 1–4, 2014.

[94] Jianwei Liu, MMA Salama, and RR Mansour. An efficient power flow algorithm for distribution systems with polynomial load. *International Journal of Electrical Engineering Education*, 39(4):371–386, 2002.

[95] Ulas Eminoglu and M Hakan Hocaoglu. A new power flow method for radial distribution systems including voltage dependent load models. *Electric power systems research*, 76(1-3):106–114, 2005.

[96] Sarika Khushalani, Jignesh M Solanki, and Noel N Schulz. Development of three-phase unbalanced power flow using pv and pq models for distributed generation and study of the impact of dg models. *IEEE Transactions on Power Systems*, 22 (3):1019–1025, 2007.

[97] Faisal Mumtaz, MH Syed, Mohamed Al Hosani, and HH Zeineldin. A novel approach to solve power flow for islanded microgrids using modified newton raphson with droop control of dg. *IEEE Transactions on Sustainable Energy*, 7(2):493–503, 2016.

[98] SM Moghaddas-Tafreshi and Elahe Mashhour. Distributed generation modeling for power flow studies and a three-phase unbalanced power flow solution for radial distribution systems considering distributed generation. *Electric Power Systems Research*, 79(4):680–686, 2009.

[99] Ahmad Memaripour. Power flow in distribution system with consideration of distributed generation. *Int. J. Acad. Res. Appl. Sci*, 1(3):60–97, 2012.

[100] K Nagaraju, S Sivanagaraju, T Ramana, and PV Prasad. A novel load flow method for radial distribution systems for realistic loads. *Electric Power Components and Systems*, 39(2):128–141, 2011.

[101] Kultar Deep Singh and Smarajit Ghosh. A new efficient method for load-flow solution for radial distribution networks. *Electrical Review, pe. org. pl/articles/2011/12a*, pages 66–73, 2011.

[102] AC Lisboa, LSM Guedes, DAG Vieira, and RR Saldanha. A fast power flow method for radial networks with linear storage and no matrix inversions. *International Journal of Electrical Power & Energy Systems*, 63:901–907, 2014.

[103] José M Cano, Md Rejwanur R Mojumdar, Joaquín G Norniella, and Gonzalo A Orcajo. Phase shifting transformer model for direct approach power flow studies. *International Journal of Electrical Power & Energy Systems*, 91:71–79, 2017.

[104] Birron Mathew Weedy, Brian John Cory, Nick Jenkins, Janaka B Ekanayake, and Goran Strbac. *Electric power systems*. John Wiley & Sons, 2012.

[105] Ray D Zimmerman and Carlos E Murillo-Sánchez. Matpower 6.0 user's manual, 2016.

[106] J. C. Das. *Load Flow Optimization and Optimal Power Flow*. Taylor & Francis Inc, 2017.

[107] A Abur, H Singh, H Liu, and WN Klingensmith. Three phase power flow for distribution systems with dispersed generation. *14th PSCC, Sevilla*, 11(3), 2002.

[108] Hany E Farag, EF El-Saadany, Ramadan El Shatshat, and Aboelsood Zidan. A generalized power flow analysis for distribution systems with high penetration of distributed generation. *Electric Power Systems Research*, 81(7):1499–1506, 2011.

[109] Ting-Yen Hsieh, Tsai-Hsiang Chen, and Nien-Che Yang. Matrix decompositions-based approach to z-bus matrix building process for radial distribution systems. *International Journal of Electrical Power & Energy Systems*, 89:62–68, 2017.

[110] D. Warner North. A tutorial introduction to decision theory. *IEEE Transactions on Systems Science and Cybernetics*, 1968.

[111] A Wierzbicki, Marek Makowski, and J Wessels. *Model-based decision support methodology with environmental applications*, volume 2000. 2000.

[112] J Branke, K Deb, K Miettinen, and R Slowinski. *Multiobjective optimization: Interactive and evolutionary approaches*. 2008.

[113] Sven Ove Hansson. Decision Theory: A brief introduction. Technical report, 2005.

[114] Katta G. Murty. Optimization models for decision making. Available at http://www-personal.umich.edu/~murty/books/opti_model/, 2003.

[115] V. Chankong and Y.Y. Haimes. *Multiobjective Decision Making: Theory and Methodology*. Dover Books on Engineering Series. Dover Publications, 2008.

[116] D.G. Luenberger and Y. Ye. *Linear and Nonlinear Programming*. International Series in Operations Research & Management Science. Springer US, 2008.

[117] C.H. Papadimitriou and K. Steiglitz. *Combinatorial Optimization: Algorithms and Complexity*. Dover Books on Computer Science. Dover Publications, 1982.

[118] Richard E. Rosenthal. Principles of Multiobjective Optimization. *Decision Sciences*, 16(2):133–152, 1985. ISSN 1540-5915. doi: 10.1111/j.1540-5915.1985.tb01479.x.

[119] Michael Emmerich and A Deutz. Multicriteria Optimization and Decision Making. *LIACS. Leiden university, NL*, 2006.

[120] J. Jahn. *Vector Optimization: Theory, Applications, and Extensions*. Springer-Verlag Berlin Heidelberg, 2010.

[121] K. Miettinen. *Nonlinear Multiobjective Optimization*. International Series in Operations Research & Management Science. Springer US, 1999.

[122] Indraneel Das and John Dennis. Normal-Boundary Intersection: An Alternate Method for Generating Pareto Optimal Points in Multicriteria Optimization Problems. (96), 1996.

[123] Kalyanmoy Deb, Shamik Chaudhuri, and Kaisa Miettinen. Towards estimating nadir objective vector using evolutionary approaches. In *Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation*, GECCO '06, pages 643–650, New York, NY, USA, 2006. ACM. doi: 10.1145/1143997.1144113.

[124] Matthias Ehrgott and Dagmar Tenfelde-Podehl. Nadir values: Computation and use in compromise programming. 2000.

[125] A. "Pascoletti and P." Serafini. "scalarizing vector optimization problems". *"Journal of Optimization Theory and Applications"*, 42(4):"499–524", 1984. doi: 10.1007/BF00934564. URL http://dx.doi.org/10.1007/BF00934564.

[126] I. Y. Kim and O. L. de Weck. Adaptive weighted sum method for multiobjective optimization: a new method for pareto front generation. *Structural and Multidisciplinary Optimization*, 31(2):105–116, 2006. ISSN 1615-1488. doi: 10.1007/s00158-005-0557-6. URL http://dx.doi.org/10.1007/s00158-005-0557-6.

[127] J Fliege and BF Svaiter. Steepest descent methods for multicriteria optimization. *Mathematical Methods of Operations Research*, 51(3):479–494, 2000.

[128] J Fliege, LMG Drummond, and BF Svaiter. Newton's method for multiobjective optimization. *SIAM Journal on Optimization*, pages 1–24, 2009.

[129] Ruhul Sarker, Joarder Kamruzzaman, and Charles Newton. Evolutionary optimization (EvOpt): a brief review and analysis. *International Journal of Computational Intelligence and Applications*, pages 1–15, 2003.

[130] Seyedali Mirjalili and Andrew Lewis. Obstacles and difficulties for robust benchmark problems. *Inf. Sci.*, 328(C):485–509, January 2016. ISSN 0020-0255. doi: 10.1016/j.ins.2015.08.041.

[131] Arnaldo Oliveira. A Discussion of Rational and Psychological Decision-Making Theories and Models: The Search for a Cultural-Ethical Decision-Making Model. *Journal of Business Ethics and Organization Studies*, 12(2):1478–82, 2007. ISSN 00029270. doi: 10.1111/j.1572-0241.1998.00467.x.

[132] Stuart M Dillon. Descriptive Decision Making: Comparing Theory with Practice. *33rd Conference of the Operational Research Society of New Zealand*, page 10, 1998.

[133] Barry Schwartz, Andrew Ward, John Monterosso, Sonja Lyubomirsky, Katherine White, and Darrin R Lehman. Maximizing versus satisficing: happiness is a matter of choice. *Journal of personality and social psychology*, 83(5):1178–1197, 2002. ISSN 0022-3514. doi: 10.1037/0022-3514.83.5.1178.

[134] Herbert A. Simon. A Behavioral Model of Rational Choice. *The Quarterly Journal of Economics*, 69(1):99–118, February 1955. ISSN 00335533. doi: 10.2307/1884852. URL http://dx.doi.org/10.2307/1884852.

[135] M Ehrgott. *Multicriteria Optimization*, volume 39. February 2005. doi: 10.1118/1.3675601.

[136] I. Das and J. E. Dennis. A closer look at drawbacks of minimizing weighted sums of objectives for Pareto set generation in multicriteria optimization problems. *Surgical forum*, 6(1):295–6, January 1997. ISSN 0071-8041. doi: 10.1007/BF01197559.

[137] Andrzej P. Wierzbicki. A mathematical basis for satisficing decision making. *Mathematical Modelling*, 3(5):391 – 405, 1982. ISSN 0270-0255. doi: http://dx.doi.org/10.1016/0270-0255(82)90038-0.

[138] Herbert A. Simon. Rational choice and the structure of the environment. *Psychological Review*, 63(2):129–38, 1956. doi: 10.1037/h0042769. URL http://dx.doi.org.arugula.cc.columbia.edu:2048/10.1037/h0042769.

[139] Milan Zeleny. *The Theory of the Displaced Ideal*, pages 153–206. Springer Berlin Heidelberg, Berlin, Heidelberg, 1976. doi: 10.1007/978-3-642-45486-8_8. URL http://dx.doi.org/10.1007/978-3-642-45486-8_8.

[140] Kalyanmoy Deb, J. Sundar, Rao N. Udaya Bhaskara, and Shamik Chaudhuri. Reference Point Based Multi-Objective Optimization Using Evolutionary Algorithms. *International Journal of Computational Intelligence Research*, 2(3):273–286, 2006. ISSN 09741259. doi: 10.5019/j.ijcir.2006.67.

[141] Lothar Thiele, Kaisa Miettinen, PJ Korhonen, and Julian Molina. A preference-based evolutionary algorithm for multi-objective optimization. *Evolutionary Computation*, 17(3):411–436, 2009.

[142] Hisao Ishibuchi, Noritaka Tsukamoto, and Yusuke Nojima. Evolutionary many-objective optimization: A short review. *2008 IEEE Congress on Evolutionary Computation IEEE World Congress on Computational Intelligence*, (March):2419–2426, 2008. doi: 10.1109/CEC.2008.4631121.

[143] E. Triantaphyllou. *Multi-criteria Decision Making Methods: A Comparative Study*. Applied Optimization. Springer US, 2010.