UNIVERSITY OF SOUTHAMPTON

A Decentralised Graph-based Framework for Electrical Power Markets

by

Jaime Cerda Jacobo

A thesis submitted in partial fulfillment for the
degree of Doctor of Philosophy

in the
Faculty of Engineering, Science and Mathematics
School of Electronics and Computer Science

March, 2010

UNIVERSITY OF SOUTHAMPTON

ABSTRACT

FACULTY OF ENGINEERING, SCIENCE AND MATHEMATICS
SCHOOL OF ELECTRONICS AND COMPUTER SCIENCE

Doctor of Philosophy

by Jaime Cerda Jacobo

One of the main tools used to clear the electrical power market across the world is the DC optimal power flow. Nevertheless, the classical model designed for vertically integrated power systems is now under pressure as new issues such as *partial information* introduced by the deregulation process, *scalability* posed by the multiple small renewable generation units as well as microgrids, and *markets integration* have to be addressed. This dissertation presents a graph-based decentralised framework for the electricity power market based on the DC optimal power flow where Newton's method is solved using graph techniques. Based on this ground, the main principles associated to the solution of systems of linear equations using a proper graph representation are presented. Then, the burden imposed by the handling of rows and columns in its matrix representation when inequality constraints have to be enforced or not is addressed in its graph based model. To this end the model is extended introducing the notion of *conditional links*. Next, this model is enhanced to address the graph decentralisation by introducing the *weak link* concept as a mean to disregard some links in the solution process while allowing the exact gradient to be computed. Following, recognizing that the DC optimal power flow is a quadratic separable program, this model is generalised to a quadratic separable program model. Finally, an agent oriented approach is proposed in order to implement the graph decentralisation. Here the agents will clear the market interchanging some economic information as well as some non-strategic information. The main contribution presented in this document is the application of graph methods to solve quadratic separable optimisation problems using Newton's method. This approach leads to a graph model whose structure is well defined. Furthermore, when applied to the DC optimal power flow this representation leads to a graph whose solution is totally embedded within the graph as both the Hessian as well as the gradient information can be accesed directly from the graph topology. In addition, the graph can be decentralised by providing a mean to evaluate the exact gradient. As a result when applied to the DC optimal power flow, the network interconnectivity is converted into local intercommunication tasks. This leads to a decentralised solution where the intercommunication is based mainly on economic information.

Contents

List of Figures

List of Tables

Listings

List of Algorithms

Declaration of Authorship

I, Jaime Cerda Jacobo, declare that the thesis entitled **A Decentralised Graph-Based Framework for Electrical Power Markets** and the work presented in the thesis are both my own, and have been generated by me as the result of my own original research. 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 quotation, this thesis is enterely my own work;
- I have acknowledged all main sources of help;
- where the thesis is based on work done by my self jointly with others, I have made clear exactly what was done by others and what I have contributed my self;
- parts of this work have been published or presented as:
  - J. Cerda and D. De Roure . A decentralised dc optimal power flow model. Published in *Proceedings of the Third International Conference on Electric Utility Deregulation and Restructuring and Power Technologies, 2008 - DRPT08*, pages 484-490. April 6-9, 2008, Nanjing, China.
  - J. Cerda and D. De Roure. A graph-based decomposition method for quadratic separable programs. Presented at *OP08 - SIAM Conference on Optimisation*. May 10-13, 2008, Boston, USA.
  - J. Cerda, D. De Roure, and E. Gerding. An agent-based electrical power market. Published in *AAMAS08 - Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems*, pages 1655-1656. May 12-16, 2008, Estoril, Portugal.

Signed:

Date: 18 March, 2010

Acknowledgements

When I commenced my PhD research I had the idea that by immersing my self into the books and papers related to my research, I will finally end up with the idea which would shape the world. This probed to be incorrect and it just helped me to be aware about the facts already known as well as some topics already under research. I had no idea about what was ahead on my way. I discovered that a PhD thesis means spending innumerable hours working hard, implementing small changes which most of the times lead to wrong results but, fortunately, few of them proved to be correct. But I did not walk alone along this way. I walked from the beginning to the completion of my research accompanied by many people who deserve to be mention, but I will restrict this to a few of them.

I am deeply grateful to my advisor, David De Roure, who always was behind me to support my research in every aspect. It would not have been possible to complete it without his help. Extra special thanks must go to Adam Prugel Bennet and Furong Li, my internal and external examiners respectively, who went throughout this document and gave me some advices in order to improve it.

On the economic side, this research was possible thanks to the funding I received from the Programa de Mejoramiento del Profesorado (Faculty Improvement Program) and the Universidad Michoacana de San Nicolás de Hidalgo. The first for providing me financial support during the time I spent here and the second for granted me a leave to pursue my PhD. My sincere gratitude goes to them.

I would also like to thank all the members of the IAM group, a great source of new ideas, in particular the discussions held by the participants of the research seminar on agent-based computing. This changed my mind about how the problems can be addressed in a decentralised manner. I hope I will be able to apply the ideas set forth in this seminar into my area.

I am grateful to the people from the school of economics where I regularly attended lectures to understand the economics supporting the problem I was addressing. I would like to mention Professor Robin Mason and Professor Juuso Valimaki for their excellent lectures on Microeconomics and Professor Maozu Lu for providing me with the general basics of Futures and Options.

It is also a pleasure to thank the support and advice received from my friends. In particular Sebastian Stein and Ilja Ponka, who were my companions along these years. The people I met in the Mexican Society was a good way to be in a Mexican corner within the UK, my special mention to Ivo Ayala who helped me even in the very last stages of this process. On the other hand, the people in the majong and volleyball clubs provided me with a place to breath some fresh air, different of my research, they introduced me to new cultures and countries that I would not have experienced otherwise. At the other

side of the Atlantic, in Mexico, I want to thank my friend Alberto Avalos for dealing with all the paperwork I needed from there while I was here.

I have received enormous support from my family so I am very gratefull with all of them, specially my mother, María de Jesús Jacobo, who unfortunately passed away while I was here as well as my father, José Guadalupe Cerda López, who passed away one year before I started my research. I am pretty sure they would have been very happy in this moment. I have already felt your kiss and your big hug, thanks to you both, God bless you. This moment would be the happiest of my life if my brothers, Francisco and Julio Cesar, could share it with me. You will always be in my heart , big and little brothers.

My stay here was built with very happy moments brought home by my children Sandy, Jaime and Pablo, together we enjoyed countless great moments which, I am sure, they will remember forever. Thanks gang, I will never forget the moments we spent together and all the things I learnt from you. Finally, I am forever indebted to my wife Sandra for supporting me in all the activities required during my studies as well as for giving me her understanding, endless patience and encouragement when it was most required, without her, I am sure, this work could not have been brought to good end, thanks Sandra.

To Sandra, Sandy, Jaime and Pablo

for their love, support and the courage to get involved into this adventure...

Nomenclature

$\wp(S)$	The power set of set S
$\mathbb{N}$	The set of nodes
$\mathbb{T}$	The set of edges
$\mathbb{G}$	The set of generators
$\mathbb{L}$	The set of variable loads
$\mathbb{Q}$	The set of fixed loads
$\Gamma_i \in \wp(\mathbb{N})$	The set of nodes adjacent to node i
$\mathbb{G}_i \in \wp(\mathbb{G})$	The set of generators connected to node i
$\mathbb{L}_i \in \wp(\mathbb{L})$	The set of variable loads connected to node i
$\mathbb{T}_i \in \wp(\mathbb{T})$	The set of lines connected to node i
Q_i	The fixed load attached to node i
δ_i	The angle at node i
b_{ij}	The susceptance of line from i to j
$\tilde{f}$	The function associated to a given constraint
$\overline{x}$	The upper bound variable associated to an inequality constraint
$\underline{x}$	The lower bound variable associated to an inequality constraint
$\hat{x}$	The constant value of external variable x
$\lceil x \rceil$	The upper limit value of variable x
$\lfloor x \rfloor$	The lower limit value of variable x

Chapter 1

Introduction

In the last two decades several factors have been driving the electrical power market to an environment which resembles more a large scale decentralised system. First, the deregulation of the electricity industry worldwide has introduced new agents into the system. In turn this has lead to the problem of partial information. This problem has been overcome today with the creation of central authorities such as independent system operators. Second, distributed generation systems are now a reality and their integration to the power market affect its operation. On one hand, the requirements to reach the levels set by the Kyoto protocol (signed in 1997 in order to reduce the greenhouse gas emissions produced by the signing parties) have increased the support for new renewable energy sources, which in general are based on small generators distributed across large areas. On the other hand, microgrids also set a new challenge as they behave like intermittent loads in the system, switching on and off depending on the energy price. Both technologies, microgrids and renewable power-based generators also lead to a problem on energy quality as their controls are based on power electronics. Third, electrical power markets integration. Electrical power systems interconnections are a common feature nowadays. Furthermore, they have been there for a long time. However, they were just used as points where energy was exchanged. Techniques to integrate these markets have been proposed which make this challenge feasible.

In this chapter, a review of the deregulation process of the electricity industry is provided. Then, as part of the tendency toward distributed generation systems, two of them are briefly described. First, Microgrids are discussed and then, SUPERGEN, a project for sustainable power energy in the UK is described. Next, the goals of this research work are set. Finally, the structure of this document is given.

1.1 Deregulation of the Electricity Industry

From a functional point of view, an electrical power system can be decomposed into three main sectors: generation, transmission and distribution. The generation sector is responsible to produce the energy used in the industries, business and homes. The transmission sector is responsible to transmit the power produced by the generation sector to the distribution networks using the high voltage transmission system. Finally, the distribution sector is responsible to transport the energy from the high voltage transmission network to the final consumers. This decomposition is based in the electricity flow from the production center to the consumer center as shown in figure 1.1.

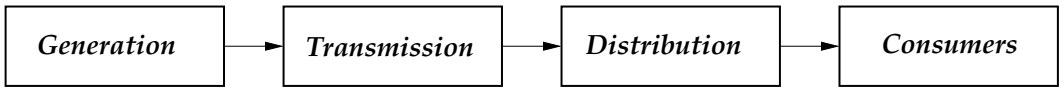


FIGURE 1.1: Electricity flow from generation to consumption.

A natural monopoly results when a single firm is able to provide one or more products at a lower cost than could be provided by more than one firm. The typical characteristics of natural monopolies are (Gilbert and Kahn, 1996):

1. Capital-intensity and minimum economic scale.
2. Non-storability with fluctuating demand.
3. Locational specificity generating location rents.
4. Necessaries, or essential for the community.
5. Involving direct connections to customers.

All these characteristics are fulfilled by the electricity industry. Furthermore, electricity utilities used to be natural regional vertically integrated monopolies. This means, they were behind the complete production chain, from production to consumption. In addition, electricity utilities were the only ones who could provide the energy in their region.

Electricity industry regulation was an inevitable demand to protect consumers from these utilities as they had all the conditions to exert market power. The main objective to introduce regulation is to hedge the consumers from those monopolies in the short-term as well as in the long term. This process is applied by periodically setting the prices the suppliers charge the consumers for the electricity as well as setting the rules which grant indiscriminate access to the electrical network.

Before the 80's, all the electrical utilities across the world were characterised as such kind of monopolies. Therefore, they had all the information about the state of the electrical

power system, as well as the ability to modify every control within the system to keep the system operating safely.

However, with the worldwide proliferation of competitive markets, the electricity industry was considered as an ideal candidate to be transformed into one with a competitive structure. In this transformation, the components in the production-consumption chain, which should be left to the free markets forces, are privatized while the others remain regulated. This process is called *deregulation*. In the extreme case this would be equivalent to privatisation when all those components are left to the free market forces.

The deregulation of electrical utilities wave which has been taking place worldwide in the last three decades was started by Chile in 1978. This was formalized in 1982 (Rudnick, 1994) aiming to implement a market-based mechanism using its marginal costs as signals to achieve economic efficiency. The United Kingdom launched its deregulation proposal in 1988 leading to the England&Wales Pool in 1992. This was a worldwide observed event which spread deregulation all over the world.

Nevertheless, deregulation has been a controversial topic over the last two decades, with people for and against its implementation. Some countries have taken this approach as a means to reach economical efficiency while others to fulfill World Bank lending requirements (Yi-chong, 2006) and eventually to achieve the goal of the first ones. Almost all the developed countries have taken this challenge (Green, 2006) and several developing countries are entering or are already on this route (Williams and Ghanadan, 2006). This process varies from country to country and even, as in the USA case, from state to state. The correct application as well as the cautionary measures are mandatory in order to achieve a successful electricity market (Haas and Auer, 2006).

This lead to the task on how these vertical structures could be broken in order to get the best market-based structure. Because electricity is a public good, there are several dimensions which have to be considered (i.e. economic, politic, geographic and so on), which are particular to each zone. However, there are two general basic stages in the deregulation process. First, each zone has to decide the unbundling degree (Joskow and Schmalensee, 1983). In this stage an assessment about which functions in the vertical structure are contestable and which ones can be regarded as natural monopolies is done. Second, the market architecture which will drive the market has to be decided. Here, the mechanics of competition will be set and will be based on the resulting electrical power market unbundled structure.

1.1.1 Main Agents in Electrical Power Markets

In 1993, William Hogan proposed several ways to unbundle the vertical natural monopoly for electricity (Hogan, 1993). The most accepted is shown in figure 1.2 which promotes a competitive wholesale market structure.

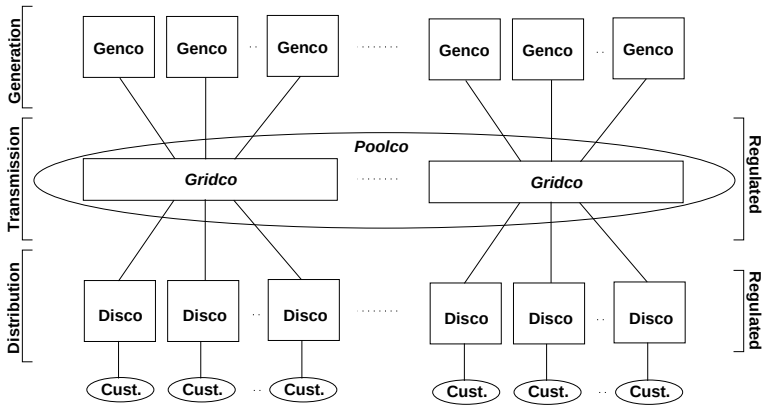


FIGURE 1.2: A Competitive Wholesale Market Structure. Source (Hogan, 1993).

Based on this structure, some countries have decided to add another agent who operates between the discos and the consumers. These agents, called retailcos, were introduced to promote competence in the retail level, and therefore lowering the prices to final consumers. In the UK, this was the decided unbundling structure which is shown in figure 1.3.

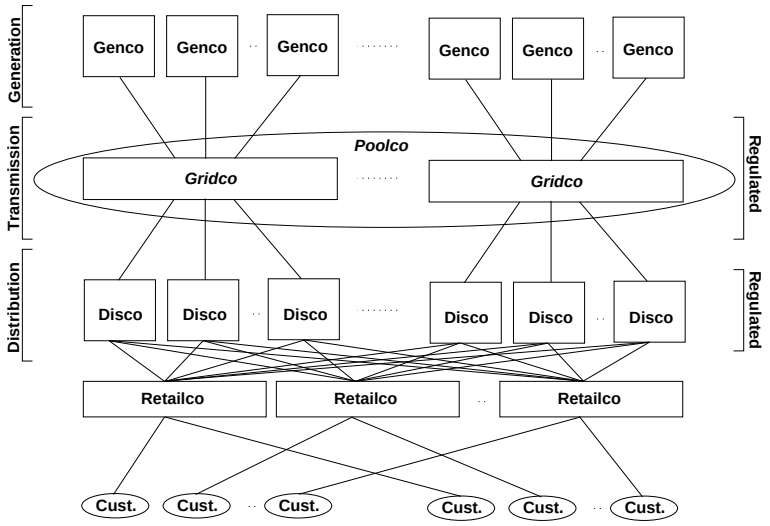


FIGURE 1.3: The unbundled structure in th UK. Based on Hogan (1993).

Taking this structure, the task to identify the agents who will participate in the market is straightforward. These agents as well as their functions are:

- **Gencos** (Generating Companies). A genco owns, operates and maintains the generating plants. They produce electricity and sell it based on the particular market rules where they are situated. PowerGen, British Energy are examples of them in the UK.

- **Gridcos** (Grid Companies). The grid company is in charge to transmit the energy from the production centers to the distribution networks using high voltages transmission lines. Their main task is to keep the system main characteristics within

its limits. This agent is represented by National Grid in England and Wales while in Scotland is represented by Scottish Power and Scottish & Southern Energy. Sometimes they are merged with the poolco as in the PJM pool in the USA.

- **Discos** (Distribution Companies). These own, operate, and maintain the distribution network. These take the energy from the High voltage network (Power grid) to the low voltage network where the final consumers are connected. However, they do not have any commercial relation with the final consumers. In the UK these are know as Distribution Network operators (DNO). Figure 1.4 shows the map for the Discos in th UK.

FIGURE 1.4: The UK distributors map (Source Energylinx - http://energylinx.co.uk).

- **Retailcos** (Retailers Companies). Also known as *suppliers*, these are the companies who bill the power each consumer takes from the distribution network (i.e. Southern Electric and British Gas, among others).

- **Custcos**. These are the agents who will consume electricity in huge quantities such as the melting industry. The retailers could be seen as a special kind of custcos. These aggregate the demand each of its clients has into a total regional demand and present it to the respective Disco who serves that region as a lumped load.

- **Poolcos** (Pool Companies). Poolcos, also known as Independent System Operators (ISO), are introduced in order to coordinate the market. These the agents are a common link among the market agents but also have to be independent from all of them in order to have credibility in its decisions. Their functions span from coordination tasks (e.g. maintenance scheduling), security tasks (e.g. congestion

management) to administrative tasks (e.g. congestion rents administration). Its existence is a consequence of the complexities posed by the electrical power markets if they were left to work in a decentralised manner.

The resulting unbundled structure (Joskow and Schmalensee, 1983) led to a consensus about what parts could be driven by the market forces. The others would need to remain either as natural monopolies owned by the state or strongly regulated. As a result of it, generation and retailing were suggested to be deregulated as their functions were contestable. On the other hand, transmission and distribution were considered as natural monopolies.

In the UK, the creation of a wholesale electricity market as well as a retail electricity market was decided but applied at different times. Competition in generation was introduced in April 1990, while retail competition was implemented nine years later, in May 1999. As for transmission, it was regarded as a monopoly owned by the state. To this end the National Grid Company was created. This, besides owning and maintaining the Transmission Network, also balances the supply and demand in real time. The last sector, distribution, was converted into private regional monopolies which are under strict surveillance by the regulator. Finally, the Office of Gas and Electricity Markets (Ofgem) was created in order to regulate this market as well as the natural gas markets. The main task of Ofgem, as for the electricity industry refers, is to provide the proper incentives in order to guarantee electrical network expansion as well as to ensure an efficient system operation.

Figure 1.5, shows the restructured electrical power market for the England and Wales pool. This scheme was active until 27 March, 2001, when it was replaced by the New Electricity Trade Agreements (NETA).

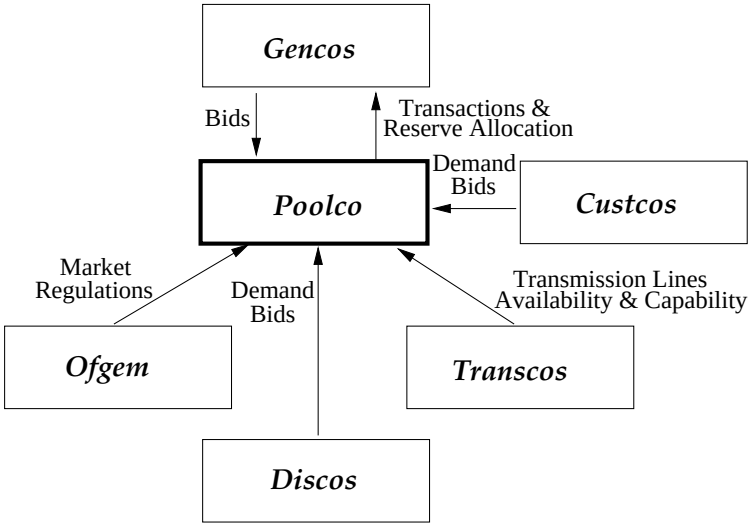


FIGURE 1.5: The restructured pool model for the electricity power market in England & Wales. Based on (Shahidehpour et al., 2002) p.316.

1.1.2 Power Markets Architectures

A relevant aspect with deregulation is the introduction of new agents within the system. Now, there are sellers wishing to sell energy, as expensive as possible, who represent the generation units. And, on the other hand, there are consumers willing to buy energy at the lowest price. Centralised mechanisms assume these agents will be willing to disclose their private data (i.e. production functions, benefit functions). Of course if this is institutional then all the agents will have the right 'incentives' to join these centralised mechanisms, otherwise they will be excluded from the game. Deregulation in this context means to unbundle this regulated vertical structure. Then, based on this unbundling it is necessary to decide which parts are convenient to regulate and which can be driven by the market forces to operate as a competitive market. Therefore, deregulation leads to a new structure in the electricity industry. Based on this new structure, a market architecture has to be defined as well as the rules which will govern its operation. The way in which the new market architectures are designed vary.

Wilson (2002), defines three main architectures in the power market, based on the agents introduced by the deregulation process. These market architectures are: Bilateral Markets, Power Exchanges, and Pools. Following, each architecture is described.

- In Bilateral Markets, trading is done among suppliers and consumers. They clear the quantity and price of each transaction through a bargaining process. However, this energy has to be sent from the supplier side to the consumer side using the transmission system. In order to accomplish this, they will use a transmission contract (Hogan, 1992), issued by some centralized entity. This entity, will take security measures in order to preserve the electrical system within its physical constraints (i.e. transmission congestion, voltage stability, etc).

- In the Power Exchange (PX), trading is done among suppliers and consumers using an auctioneer whose task is to conciliate suppliers bids with demand bids. He will take into account system loses within the market clearing procedure. This kind of market will have to coexist with another market agent called the Independent System Operator who will take the PX clearing result and decide if this is feasible (i.e. preserve the system within the physical constraints). If this is not the case it gives some feedback to the market agents in order to reach a feasible solution.

- In the Pools (Bath, 2006), as in the power exchange, trading is done among suppliers and consumers using an auctioneer. Here, besides conciliating suppliers and consumers bids, taking into account network security and losses, he will also clear the market considering the unit commitment problem, which basically means taking into account when it is worth starting a generation unit. Therefore, supply bidders have to send additional information needed by the auctioneer to clear the market.

These architectures must be enforced with the right design rules. We must remember that the agents in the system are looking for their own profit. Hence, they will try to find every loophole in the market to increment their profit. The GB's pool power model was found susceptible to manipulation (Ofgem, 1999). As a consequence, it was decided to create a new model based on a mixture of those architectures along with an energy balancing mechanism.

1.2 Distributed generation projects

Distributed generation projects pose a great challenge to the way the electrical power market is driven. These are envisioned as small generation units spread over a large area. However, its connection to the network would differ as the traditional way bulk generation units have been connected to the grid. It would be very difficult for a central authority to coordinate such a market. There are several benefits as well as costs linked to the use of distributed generation as assessed by Gumerman et al. (2003). Two main approaches are the drivers for these distributed generation systems: Microgrids and sustainable renewable power generation. These are briefly described in this section.

- Microgrids. This is a relatively new concept in power generation, where a cluster of electrical and thermal loads are served by small generation units operating as a single entity (Lasseter, 2002). These generators are located in the same place where the loads are. These microgrids could be acting as intermittent loads who would buy energy from the system if the price is low (i.e. the load is on). On the other hand, they would serve the load if the energy price is high. This would imply to serve the entire load or just part of it, by shedding the part of the load which is not essential. A great concern about how they can be integrated into the power grid is still a research area. As for this research, Microgrids could be thought as intermittent loads. This characteristic poses a great challenge to the way the Power Market is cleared. This market clearing mechanism is a centralised process. Microgrids integration to the Power Grid is a great challenge for real time operation.

- Sustainable Power Generation. In this approach, all kind of renewable energy is used to produce electricity. To this end, it gathers energy from sea, earth and sun. In the UK the project SUPERGEN, *Sustainable Power Generation and Supply*, is an EPSRC initiative aiming to help the UK meet its environmental emissions targets (SUPERGEN, EPSRC) by exploiting all kind of renewable energies. It supports projects in power generation and supply. To this end, ten different consortia have been created since November, 2003. These consortia try to find new products to generate energy which go from Biomass and Bioenergy. Also, it gives support for new projects related to alternative power generation from solar cell,

fuel cells, and so on. The final effect as for the Power Grid will be the same as the Microgrids as they will be low power generation plants distributed across the UK. This initiative envisages a future where there will be thousands of small generators supported by different technologies (i.e. photo voltaic, microgrids, wind, and hydro-systems). Therefore, the main challenge is how to incorporate these generators into the power grid and how the centralised market clearing mechanisms can be adapted for such high scale systems.

1.3 Research Aim

The electrical power market of the future has to be decentralised if the challenges above mentioned are to be addressed. Experience says centralised mechanisms are not suitable for large scale systems. A typical large scale system is the electrical power system. Centralised mechanisms lead to a bottleneck to both the clearing market mechanism and its expansion. The main aim in this research is to provide a decentralised framework to clear the Electrical Power Market. The main reasons why this market has to be decentralised, as discussed above, are: deregulation, distributed generation, and power systems markets integration. At the very heart of the electrical power market clearing procedure is the DC optimal power flow. It has been around for almost 40 years, used as a basic tool in order to clear the electrical power market. The solution to this problem is a centralised clearing mechanism. The importance of this mechanism in its distributed version is reflected in (Consentec, 2004) as an alternative to the centralised solution which would allow each country in the EU to keep their data private.

To this end, two new tools have been developed. First, a bottom-up decomposition which keeps the model as simple as the centralised model. This implies to decompose the system in its main components. Then based on their interrelations, communication strategies are derived. Second, a new decentralised graph-based algorithm is proposed. In turn, this technique is susceptible to be generalised to problems whose structure is similar to this one (i.e. quadratic separable programs).

1.4 Document Structure

This document is structured as follows:

First, the main topics which support this research are described in chapter 2. Concepts from economics such as auctions, welfare economics, and consumer behavior are provided. Then the basic components of power systems are explained from both points of view, electrical and economic. Next we move on providing the basic concepts on optimisation which are related to this research.

Chapter 3 presents a graphical representation for systems of linear equations. The solution to a subset of this kind of systems in its graphical representation is introduced. These graphs will be used and the strategies to deal with conditional evaluation as well as decentralisation aspects will be based on their topology. Based on the model developed in chapter 3 a graph-based model to solve the economical dispatch problem is proposed in chapter 4. This is done in two stages. First a relaxed version of this problem is addressed where all the bounding constraints for the variables are disregarded. This leads to a model which is able to find the solution if this is within the unconstrained space. In the second part the complete economical dispatch is solved by attaching a *conditional* or *non-conditional* label to some links which are not always active in order to deal with the solutions which are in the constrained space. Then, a decentralised model of the system is proposed in chapter 5, based on the auxiliary problem principle. This model is a decentralised mechanism to clear the electrical power market. To this end, a bottom-up decomposition approach is used. The benefits from this decomposition is twofold. First, it leads to identify the main agents in the system. Second, it uncovers the information they need to interchange in order to solve the problem in a cooperative manner.

Following, in chapter 6, a deeper analysis is done based on the model presented in chapter 5, which in turn is based on the underlying graph-based model. Based on this analysis a new characteristic for the links is proposed in order to address the decentralisation process. The attached labels are either *hard* or *soft*. Based on these values, the reduction process will be guided. This reduction process applies a graph-based formulation which leads to an efficient algorithm to solve the DC optimal power flow problem. To this end, the matrix-based formulation is converted into its corresponding graph. Algorithms to traverse this graph are then proposed in order to solve the system.

In chapter 7, based on the knowledge acquaited in chapter 6, an exploration on how to decentralise these graphs is done. Here three main kinds of decentralisation approaches are described. The first one is a completely decentralised approach where all the links are weakened leading to a gradient oriented approach. The second one is based on the notion of primal and dual variables. This decentralisation leads to a horizontal split of the graph, where all the links connecting the primal variable with the dual variables are weakened. The links which connect primal variables with their corresponding bounding dual variables are not weakened. In the third decentralisation approach, an agent oriented approach is proposed, which leads to an enhanced performance of the system to reach stable state. To this end an agent-based description is formulated based on agents which are described with a set of states and the constraints these agents are subject to.

Based on the agent-based decentralisation approach from chapter 7, in chapter 8 a methodology to generate this kind of graphs is presented. This will point out that no matrix formulation has to be done in order to derive these graphs. Finally, chapter 9 concludes this document in two stages. First, it reviews the main aspects this research

has presented. Then, it concludes this document by setting the road map with the possible research lines which can be further derived based on the work presented in this thesis.

Chapter 2

Background

Electrical power markets are the point where several streams of knowledge converge. How much we know about these streams will represent how much we understand the mechanics about how power markets work. Based on this knowledge, eventually new concepts or techniques can be proposed which can improve their operation. In the following paragraphs we delineate such knowledge streams.

- *Economics.* At the very end, markets are described as economic models. A central topic in modeling markets is auction theory. The end of an auction is to allocate commodities to those bidders who value them the most. The models used in the research until now are related to social welfare. Therefore we rely in concepts drawn from Welfare Economics in order to clear the market.

- *Electrical power systems.* These are essential in order to understand the underlying problem to be solved. The main building blocks for the the Electrical Power System need to be understood altogether with their models (i.e. nodes, lines, generators, loads, etc.). These models are the basic building blocks for the power market.

- *Optimisation.* An efficient market will have as outcome the best allocations (i.e. production levels and price). The best, depends on the context where it is used. In order to decide what "best" is, optimisation techniques have to be used to solve this problem. To this end, the mathematical concepts and tools which support these methods have to be well understood.

- *Computer Sciences.* At the very end, optimisation techniques are numerical methods. Some of the techniques we apply to these methods are drawn from computer science. In particular when we talk about decentralisation eventually we end up with its formalisation and implementation. Intercommunication processes, multi-agent systems, and so on, are some of the techniques used to that end.

In this chapter, the concepts needed from those different areas in this research are described. First, we describe auctions as the main support for competitive markets. Then, electrical power systems are discussed in two stages. First, the description of their models and then the techniques used in this area to solve them. After that we move on to set a basic background on Non-Linear Programing (NLP). Here the basic NLP problem is stated, and a description of convex programming is provided. Next, the solution to NLP problems is delineated along a discussion of the main concepts involved in this process such as Lagrange Multipliers and Karush-Khun-Tucker conditions. To finish the discussion on Non-Linear Programing, Quadratic Programing and its special case called Quadratic Separable Programing are described.

2.1 Auctions

Market-based solutions are being used in countless applications using auctions (Klemperer, 2004) as a mean to allocate scarce resources to those agents who value them the most. If this condition is met, then this auction is said to be *economic efficient*. A normal auction is formed by several agents: sellers, buyers and, if this is centralised, an auctioneer. The seller owns the good and is willing to sell it at the highest possible price. Buyers are willing to buy the good but spending as little as possible.

Three main stages can be identified in an auction:

1. *Bidding*. In this stage the participants will set their positions. Sellers will set their asking price and buyers will set their offer price. If the auction is centralised, the auctioneer will collect the buyers and sellers' offers and asks bids, respectively.
2. *Winner determination*. Here, the clearing mechanism will decide the auction winner. The commodity will be granted to him.
3. *Price settlement*. In this stage, the price for the commodity will be set and can take several forms depending on the underlying auction type.

Therefore, an auction is a price discovery mechanism which sets the price of a commodity conciliating both buyers and sellers interests and granting the commodity to those who value them the most. In some auctions, that price is known as the *Market Clearing Price*. There are many dimensions to classify auctions. Table 2.1, extracted from He et al. (2003), shows the values each auction can take along seven different dimensions.

2.1.1 The main auction types

From table 2.1, we can observe the many ways an auction can be run based on those dimensions. All of them have the same end: to discover the price of a commodity.

Dimension	Values	Description
Auction mode	O-One sided T-Two sided	Only bids or asks are permitted Both bids and asks are permitted
Time duration	S-Single-round M-Multi-round	The auction last one round The auction lasts multiple rounds
Unit of goods	O-One M-Many	Only one good is auctioned during the auction Multiple goods are auctioned
Ratio of B-S	MO-Many to one OM-One to many MM-Many to many	Multiple buyers, one seller Multiple sellers and only one buyer Multiple buyers, multiple sellers
Revealed preference	Y-Yes N-No	Intermediate revealed information exists Information about others is not available
Settlement price	F-First price S-Second price D-Different prices	Highest price among all bidders Second highest price among all bidders Trades take place any time at different prices
Closing Rules	T-Time I-Inactivity B-Budget	when time is reached when there are no more bids for a time period when a reserve price is reached

TABLE 2.1: Auction Dimensions, extracted from He et al. (2003)

However the mechanisms they use in order to reach its goal are different and in some cases the outcome is also different. There are five very well known types of auctions: English, Dutch, First Price Sealed Bid, Second Price Sealed Bid, and Continuous Double auction. In addition, in the electrical power market a Double MultiUnit Auction called *Day Ahead Auction* is extensively used in order to clear the market. Following these auctions are described and their properties for each dimension are depicted in table 2.2.

- **English Auction**. In this auction, also known as *ascending auction*, the auctioneer rises the price successively and the bidders announce if they are willing to pay this price. The bidders who are not willing to pay the price must quit the auction. This process ends when just one bidder remains willing to pay the last price announced. This last bidder will be the winner. It could also be run by the bidders announcing incremental bids themselves until no one is willing to pay more. The one who announced the highest price will be the winner.

- **Dutch Auction**. Also known as *descending auction*, works in the opposite way. The auctioneer starts the auction by announcing a very high price. Then, this price is lowered as the auction is evolving. The bidder who calls out first wins.

- **First Price Sealed Bid auction (FPSB)**. In this auction, each bidder submits his bid independently ignoring others' bids. The auctioneer will announce the auction winner who will be the one with the highest bid (i.e. government selling mineral rights), or the one with the lowest bid (i.e. government procurement). The main drawback for this auction is the winner's curse. This says the winner must have paid more than the right commodity value. Therefore, he must rethink his bidding strategy as there is a (high) possibility he is paying above the commodity value by taking into account this fact.

- **Second Price Sealed Bid Auction (SPSB)**. Also known as *Vickrey Auction*. In this auction, like in the FPSB auction, every bidder submits his bid independently from other bidders. The difference with the previous FPSB auction is that the winner is the bidder with the highest bid, but he will pay the second highest price. In this auction bidding the true value is the optimal choice.

- **Double Auction**. In this auction, the buyers submit their bids and the sellers submit their offers to an auctioneer. The auctioneer will clear the auction by constructing an agreggate demand curve with buyers' bids as well as an aggregate supply curve with sellers' offers. The market clearing price (MCP) will be that where both curves intersect each other.

- **Continuous Double Auction (CDA)** This is an iterative double auction which doesn't stop. As soon as each auction is concluded the next one is started. From here, adaptive mechanisms can be designed in order to modify the bidding and offering strategies based on the previous auctions' results.

Auction	Auction mode	Duration time	Unit goods	Ratio of B-S	Revealed information	Settlement price	Closing rules
English	O	M	O	MO-OM	Y	F	I
Dutch	O	M	O	MO-OM	Y	F	B
FPSB	O	S	O	MO-OM	N	F	T
SPSB	O	S	O	MO-OM	N	S	T
Double Auction	T	S	M	MM	N	D	B
CDA	T	M	M	MM	Y	D	I

TABLE 2.2: Comparison of different types of auctions (Based on He et al. (2003)

2.1.2 The Day Ahead Auction

In electrical power markets a periodic version of the double multiunit uniform auction called *day ahead* is used. Here, the auctioneer collects offer (request) bids, in a quantity-price pairs form, from suppliers (customers) over a given period of time, and then clears the market with a uniform price. Table 2.3, shows a typical situation, where suppliers (customers) send their offers (requests).

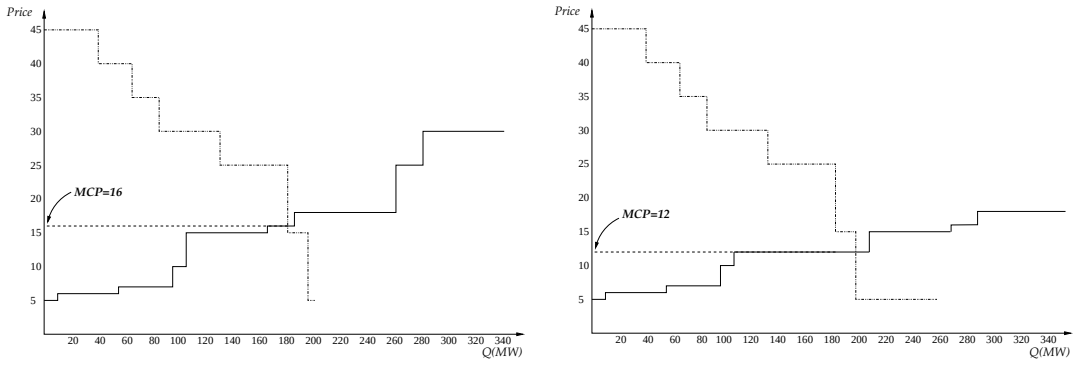
The auctioneer sort the bids in ascending (descending) order. Based on this ordering, their cumulative quantities and prices are plotted. The market clearing price (MCP) is the intersection point of those curves, as shown in figure 2.1(a), where the system would be cleared with $MCP = 16$, producing 185 MW in total. If more generation units were installed in the system, this would cause the supply curve to shift to the right. Therefore, the intersection point also will be displaced to the right. This could in turn reduce the MCP. For instance let us suppose another plant is built with a very efficient technology. This plant submit the offer bid (100 MW, 12). Figure 2.1(b), shows the effect of this offer in the MCP determination process. Here $MCP = 12$ with a total

OFFERS	
Quantity	*Price*
10	5.00
20	15.00
15	6.00
40	7.00
30	6.00
40	15.00
20	25.00
10	10.00
20	16.00
60	30.00
75	18.00

REQUESTS	
Quantity	*Price*
20	35.00
15	15.00
25	40.00
45	30.00
50	25.00
60	5.00
40	45.00

TABLE 2.3: Offer and Request bids table.

production of 190 MW. This price will be the one the gencos/custcos will receive/pay for each unit of energy. Tesfatsion and Koesrindaroto (2004) presents a framework to simulate this model.



(a) MCP for transactions submitted in table 2.3.

(b) Same bids as in (a), but the pair offer (100MW, 12) has been aggregated.

FIGURE 2.1: Market clearing price determination.

The allocation method described above would be correct in case no transaction had taken the system into a congestion state (i.e. the allocation can not be handled by the transmission system). However, if this allocation takes the electrical power system into a congestion state then the auctioneer will have to find another solution to this problem. In general, this process can be solved, in the simplest case, by using the DC Power Flow model, augmented with the transmission system constraints using optimization techniques (Rau, 2003).

This is a simplification of the kind of auction which the old England and Wales pool used to do every half hour of the next day, (i.e. 48 times/day (Bath, 2006)). This mechanism was found to be flawed, uncompetitive, and susceptible to manipulation. There, the main incumbents in the market were withholding their low-price units in order to raise the price of power (Green, 2004). As a result, this mechanism was replaced by the

New Electricity Trading Agreements (NETA) where trading is done mainly by means of bilateral contracts, one or more power exchanges and a balancing mechanism market which helps to deal with the imbalances presented in real time operation.

2.1.3 Competitive Markets

A competitive market is one where all the participants behave like price takers. This means, none of them can change the market price by themselves. As opposed to imperfect markets subject to oligopoly practices which in the extreme case would be the monopoly where the producers can set the market price or the monopsony where the consumer can set the market price.

The classical competitive market structure is the one where the producer of commodities are given direct access to the consumers of those commodities, generally under certain contracts (i.e. bilateral contracts). This situation is represented in figure Figure 2.2.

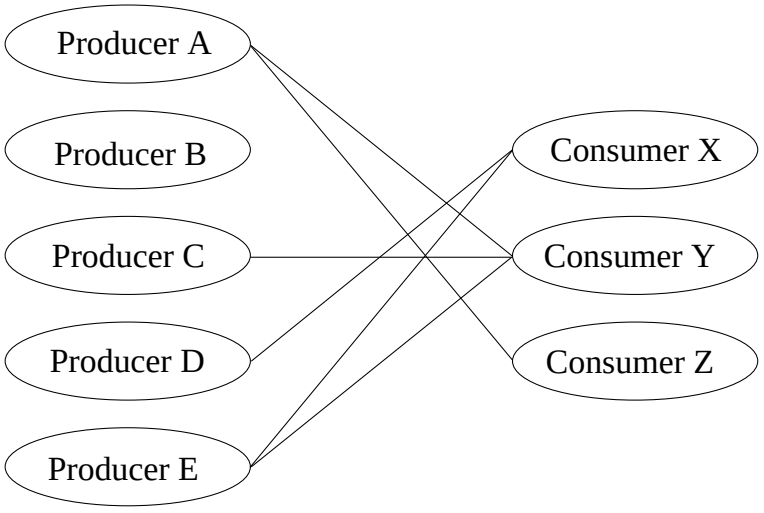


FIGURE 2.2: Classical Competitive Market Structure

Unfortunately, there are two main limitations to this model when applied to power markets. Firstly, electric energy can not be efficiently stored, therefore, once produced it has to be consumed. Perhaps this is the commodity which lasts the least time in the market. Secondly, the path it follows from the production center to the consumption center can not be controled.

Being unable to store electrical energy has a great impact on the energy price. This fact affects consumers, disabling them to store the commodity to prevent peak prices, as in any other market could have been done. As a result, the electricity price can rise very high during peak demand periods.

Without control over the path of the electrical flow, every single transaction will affect every path within the electrical network. This makes bilateral contracts very difficult

between suppliers and consumers, who will have to reinforce some security measures to avoid or compensate side effects when applied. Nevertheless, these are used extensively in the UK actual market NETA, along with a balancing mechanism.

2.2 Electrical Power Markets and their Economic Models

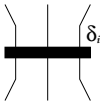
From a topological point of view, an electrical circuit can be seen as a graph formed by nodes and edges. In particular edges represent an electrical component (i.e. lines, transformers, and so on). The points where the interconnections of these components have occured are represented by nodes, which in the electrical jargon are called buses. The topological description of the circuit tells us about the circuit interconnection, which is independent of any energy source we apply to it. At this stage, only the passive properties of the circuit (i.e. resistance, admittance, etc) can be described. When an excitation (energy source) is applied to the circuit we can talk about active quantities distributed along the circuit (i.e. voltage, current, power, frequency, impedance, susceptance, etc.).

In order to perform an economical assessment to an electrical power system, we need to describe its basic components as well as their corresponding economical models. The aim of this section is to describe such components within the electrical power system from both points of view (i.e. electrical and economic).

2.2.1 Electrical Power Systems Notation

An EPS can be abstracted as a topological structure, which can be described by several sets. The set $\mathbb{N}$ represents the set of electrical nodes in the system. The set $\mathbb{E}$ represents the set of lines. These are represented as tuples (i, j), where $i, j \in \mathbb{N}$. $\mathbb{G}$ represents the set of generators which are connected across the EPS. $\mathbb{L}$ represents the set of variable loads connected across the EPS. $\mathbb{Q}$ represents the set of fixed loads connected across the EPS. Additionally, several subsets built out of the previous sets are defined. Subsets $\mathbb{G}_i$ and $\mathbb{L}_i$ represent the generators and variable loads connected to node i. Fixed loads attached to node $i, (i \in \mathbb{N})$, can be represented as a unique compounded fixed load denoted by Q_i. Finally, the set $\Gamma_i \in \wp(\mathbb{N} \setminus i)$ represents the set of nodes $\{j : (i, j) \in \mathbb{E}\}$, (i.e. the set of nodes which are neighbours of node i).

2.2.2 Nodes



The main element in an electrical power system is the node, also called bus. Nodes define the points where several elements are connected (i.e. lines, load, generators, etc.). Although a node is a "passive" component, it has several

characteristics which are the core for the analysis of electrical power networks. The generic bus model is depicted in figure 2.3.

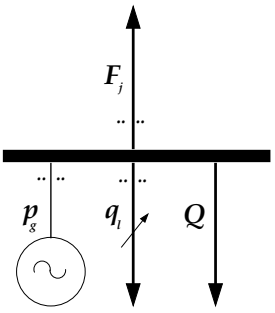
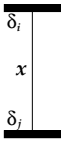


FIGURE 2.3: A generic node model.

A constraint which always has to be observed in all the nodes across the electrical power system is the power flow balance. Basically, the node can not store energy. Therefore, the energy which flows into a node must be equal to the energy which flows out of the node. The generators are injecting the power $\sum_{g\in\mathbb{G}_i} p_g$ into the node whereas the loads are extracting power $\sum_{l\in\mathbb{L}_i} q_l + Q$ out of the node. On the other hand, the flows which go through the lines connected to this node, $\sum_{j\in\Gamma_i} b_j(\delta - \delta_j)$ to other nodes do not have a determined direction. This depend on the current electrical power system state. Therefore, in this document these flows will be assumed as extracting power out of the node. A negative sign in its valuation will indicate that our assumption was wrong and the flow direction has to be reversed. The energy balance for node i, is expressed in equation 2.1 (*see* appendix A).

$$\sum_{j\in\Gamma_i} b_{ij}(\delta_i - \delta_j) - \sum_{g\in\mathbb{G}_i} p_g + \sum_{l\in\mathbb{L}_i} q_l + Q = 0 \tag{2.1}$$

2.2.3 Lines



The long line's transmission model is represented by a π-circuit model as the one shown in figure 2.4. Here, r represents the line resistance, x its reactance. The elements $(g + jb)/2$ represents the shunt admitance with respect to ground at each extreme of the line, where g is its conductance and b its susceptance. j is the imaginary operator, representing $\sqrt{-1}$. If the transmission line is short then the capacitive effect can be disregarded. Therefore, in this case, the model can be simplified by taking away both elements which are connected to ground. A further simplification can be made if this line is assumed to be lossless by disregarding its resistance r. The DC Optimal Power Flow, which is the model used in this document requires only the reactance value of the impedance.

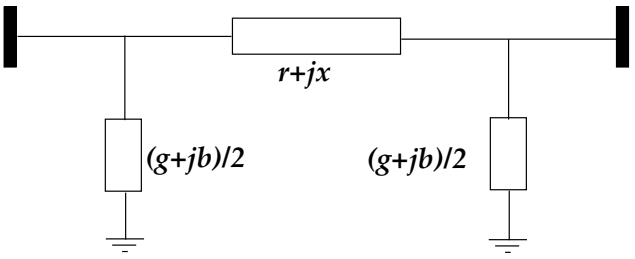


FIGURE 2.4: π-circuit model for a transmission line.

If a transmission line would transfer a power level beyond its designed capacity then this would cause a line overheat. In turn, this overheating would cause the resistance to go up and, therefore, losses would reduce the power transfer. Furthermore, this overheating would cause this line to elapse. Hence, the capacitive effect would go up as its distance to ground will be reduced. As a result from the above, the power transfer capability of the system is changing over the time affecting the characteristics of the market. Therefore, keeping the line power transfer within its limits is one of the main tasks in the everyday operation of the electrical power system. The power limit constraint, for a line connected between nodes i and j, for the DC-optimal power flow model, is defined as

$$|b(\delta_i - \delta_j)| \leq \lceil F \rceil$$

where

$$b = 1/x$$

F is the line capacity in MW

2.2.3.1 Electric and Economic Implications as a result of Congestion.

The line's power transfer constraint will limit the electric flow through that line. However, the power which has been constrained to flow through this line will be dispatched from somewhere else within the electrical power system in order to fulfill the power flow balance within the system. Therefore, from the electrical point of view, there will be just a re-allocation of resources in the system. The system will be working with a different parameter configuration but within the constraints it has to fulfill.

However, this fact will have severe consequences from the economical point of view. At the very end, power transfer imply traded energy. Congestion will limit the possible trades which could be done in an efficient market. Some of these trades are not possible as they would cause congestion. The reallocated energy will be provided by a more expensive unit located somewhere else in the electrical power system. This effect will produce different energy prices across the electrical power system; in fact, each node will have its own price. The difference in prices between two nodes gives a signal about the

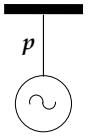
quantity of power flow which has been prevented by the flow constraint. These signal in turn can be interpreted in several ways depending on the agent type as follows

- *Gencos* : This signal will give an incentive to build new generation plants in the congested areas as they will be paid more for the energy they produce. This situation continues until the congestion constraint is cleared. Further investment has no more economical advantages for the gencos. Furthermore, if the genco does it then he would drive the energy price down as was shown in figure 2.1.2.
- *Custcos*: This is a signal which prevents the customer to augment the load in the congested zone. Furthermore, it gives an incentive to build future facilities in zones where congestion is not a problem.
- *Poolcos*: They can interpret this signal as the need to reinforce the network by building new lines. Also they can improve the network performance by using Flexible AC Transmission Systems (FACTS) (Acha et al., 2004) to control the power flow in the lines which are prone to congestion.

2.2.3.2 Locational Marginal Prices

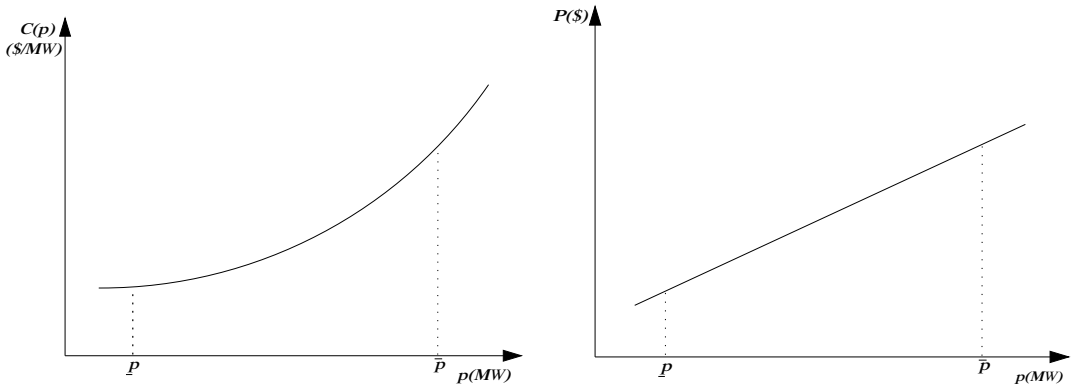
The bus energy prices cleared by the market mechanism where congestion is taken into account are called locational marginal prices (Schweppe, 1998) if losses are disregarded. If there is no congestion in the system, then the they will be equal to the market clearing price calculated by the day ahead auction. This is because none of the transmission constraints are binding, and consequently they can be disregarded. However, if congestion occurs, then each node will have its own locational marginal price. In the DC-optimal power flow, locational marginal prices are the shadow prices associated with constraint 2.1.

2.2.4 Generators



Synchronous generators are used to produce active power p as well as reactive power q. Reactive power is mainly produced in order to keep a constant voltage at its terminals. In this work we will not deal with this aspect. The power this generator can produce depends on several physical limitations relating to itself as well as the kind of technology the generator is attached to (i.e. hydro, nuclear, steam, eolic, and so on). Therefore, the units will produce power in a valid range which goes from a lower bound given by $\underline{p}$ to an upper bound defined by $\overline{p}$, as shown in figure 2.5(a). This curve which describes the production characteristic for a generator is often given as a quadratic output-input function. This function maps the production cost to generate a given power level. Nevertheless, this curve is also approximated as a piecewise curve

which is a sequence of straight line segments. In this document, the cost supply curve will be represented by a quadratic function defined as $C(p) = \alpha + \beta p + \gamma p^2, \gamma > 0$. Based on this curve, a fundamental curve is derived know as the marginal cost curve. This is shown in figure 2.5(b). This curve is built deriving the cost curve with respect to p.



(a) Cost curve (b) Marginal cost curve

FIGURE 2.5: The generator model

Generators, based on the priority to provide energy to the system in response to the demand curve, are classified as

- *Baseload*. These units will be operating all the time, except for maintenance purposes. They will supply the minimum demand, called *base load*, along the day as shown in figure 2.6. They produce electricity at almost constant rates. Usually these are the most efficient and largest units.
- *Intermediate load*. These are units which are used to supply the system to meet the intermediate energy, which is the one between the base and the peak load.
- *Peaking*. These are usually the least efficient units, and are used to meet the peak load periods. In these periods the price for the energy is high enough to recover their high cost.

The characteristics for each type is given in table 2.4. Here, *start-up costs* are the costs associated to commit the unit. *Ramp rate*, is the rate at which the generation level can be incremented or decremented over a given period of time. This rate is usually set for a period of one minute. Therefore its units are MW/min. Finally, the *heat rate* is the capacity the unit has to convert the energy contained in the fuel into electrical energy. The usual units are MBU/kWh.

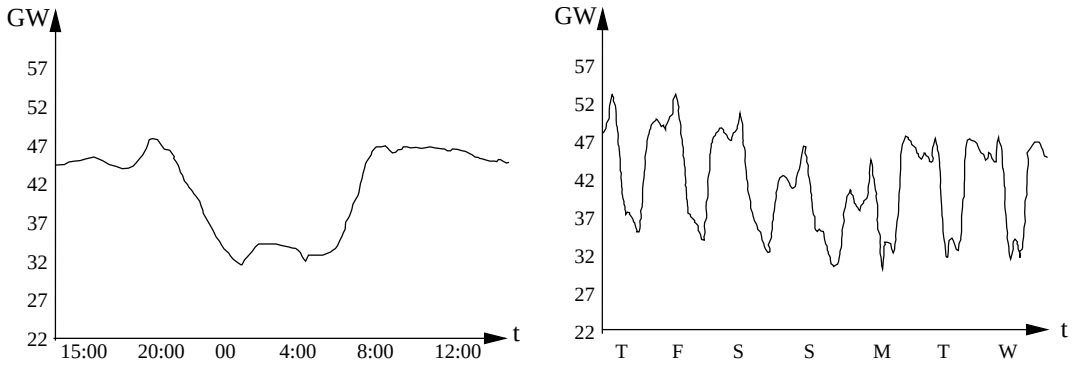
2.2.5 Loads

Loads, represent the consumers in the electrical power system and are also know as demand. The load is dispersed across all the electrical power system. Their aggregated

Characteristic\ Type	*Baseload*	*Cycling*	*Peaking*
Start-up costs	High	Moderate	Moderate
Fuels	Gas, coal, oil, nuclear	Oil, gas	Oil, gas
Ramp rates	Low	Low to moderate	High
Heat rates at Maximum Capacity	Low	Moderate	High

TABLE 2.4: Generator Operating Characteristics (Eydeland and Wolyniec (2003)).

demand is being monitored at every instant by the gridco. A curve called the *Total Demand Curve* is built from these observations . Its shape is variable along the time as shown in figure 2.6(a) which represents the demand curve from 3:00 pm, March 27, 2006 to 3:00 pm, March 28, 2006.



(a) Last 24 Hours (b) Last 8 Days

FIGURE 2.6: Electricity demand in the UK.

However, if we analyse the load curve pattern with a different time scale, as shown in 2.6(b), which represents the demand in the week from March 21 2007 to March 27 2007, a periodic pattern is observed. The total demand regular pattern is well known but the exact quantities are not. Therefore, there is a need to balance in real time generation and load. They have to be matched at every instant by the gridco.

In England and Wales, the New Electricity Trading Agreements (NETA), is based on a *Balancing Mechanism* which is used by National Grid, the Independent System Operator in England and Wales, to balance the power flows on the Grid as well as those which flow into and out of it (i.e. imports and exports).

Nevertheless, this curve also gives information about the *base load* which has to be served. This information gives the right information about the base generation level which has to be produced. The generators used for this end are known as *baseload generators* and they are operated around the clock. Together they are know as the *baseload generation capacity*.

Depending on its capacity to react to the energy price change are classified as *elastic* or *variable* loads and *perfectly inelastic* or *fixed* loads. The total demand in the system

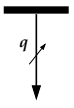
is the fixed demand plus the elastic demand. An elastic load is one which reacts to changes in prices. For example, let us assume the actual load level is q and the price is P. Now, let us suppose the price for energy changes ΔP and this change causes a change in demand of Δq. The elasticity of this load, ϵ, is given by equation 2.2.

$$\epsilon = -\frac{\%\Delta q}{\%\Delta P} = -\frac{\Delta q/q}{\Delta P/P} = -\frac{\Delta q}{\Delta P}\frac{P}{q} \tag{2.2}$$

$$\text{Elasticity can be classified as} \begin{cases} \text{perfectly elastic} & \text{if} \quad |\epsilon| \approx \infty \\ \text{elastic} & \text{if} \quad |\epsilon| > 1 \\ \text{unitary elastic} & \text{if} \quad |\epsilon| = 1 \\ \text{inelastic} & \text{if} \quad |\epsilon| < 1 \\ \text{perfectly inelastic} & \text{if} \quad |\epsilon| \approx 0 \end{cases} \tag{2.3}$$

Load elasticity plays a great role if the strategic behavior of the system is to be analysed. In the the following sections the loads and their models are described based on their ability to respond to price changes (i.e. elastic and fixed loads).

2.2.5.1 Elastic Load Model

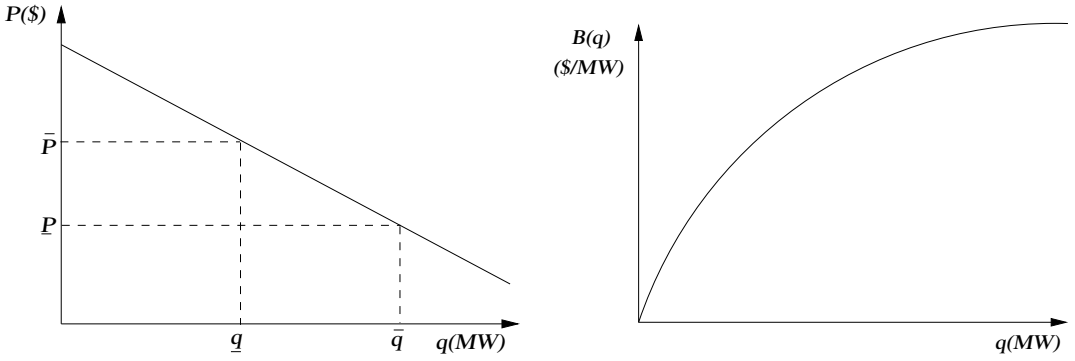


Elastic loads are those which can react to changes in energy prices by adjusting its consumption level. As an example, let us consider the melting industry. Its energy consumption level varies directly with its production. Hence, if they were able to produce their output at times where energy price is cheap, they would increment their profit. Of course, this could introduce some extra costs (i.e. more expensive labor). Nevertheless, the final decision is up to them.

In order to obtain the benefit function for this load we need to rely on a basic assumption which will ensure its concavity. In particular, a concave quadratic model will be derived. We will assume the consumers are rational agents. Therefore, the most valuable energy block for them will be the first block. This is because they will use this energy for those processes which benefit them the most. The same argument can be set for the next energy blocks (i.e. next block will be less valuable than the previous one, but more valuable than the next ones). This process is repeated until they have already fulfilled their consumption needs.

Based on the above assumption the benefit curve is built on four parameters derived from the consumer behavior. First, the minimum power consumption $\underline{q}$ he wants to consume and the price $\overline{p}$ he is willing to pay for it. Second, the maximum power consumption $\overline{q}$ he wants to consume and the price $\underline{p}$ he is willing to pay for it. This has to fulfill the constraint $\overline{p} \geq \underline{p}$ to meet the previous assumption. This curve is represented as a line

monotonically decreasing with negative slope $m = \frac{p-\overline{p}}{\underline{q}-\overline{q}}$ and intercept $\beta = \underline{p} - m\overline{q}$. This represents the marginal benefit curve for the consumer $MB(q) = \beta + mq$, as shown in figure 2.7(a).

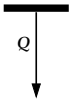


(a) Marginal benefit curve (b) Benefit curve

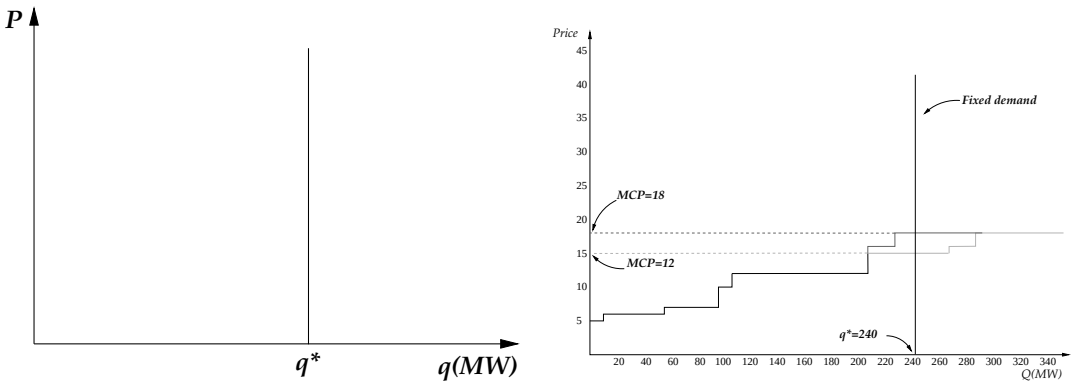
FIGURE 2.7: The elastic load model.

Based on this model, the benefit curve, $B(q)$, is straightforward. This is built integrating the marginal benefit curve $MB(q)$ over q. Therefore the benefit curve can be represented as a concave quadratic function, shown in 2.7(b). This model is described as $B(q) = \beta q + \gamma q^2$ where $\gamma = m/2$ and $m \leq 0$.

2.2.5.2 Fixed Load Model



A fixed load, whose model is shown in Figure 2.8(a), is one which does not react to changes in price, (i.e. its perfectly inelastic). This kind of load could represent the electricity suppliers. They buy energy in the wholesale market, whose price vary at every minute. This energy, is then sold in the retail market to the final consumers. Suppliers are not able to react to changes in wholesale prices. The reason is that their demand depends on the final consumer. Consumers are not subject to any restriction about how and when to use the energy. If the price the consumer faces were variable, depending on the real time wholesale market price, they would react to these changes. They would adapt their consumer behavior to the time where the energy price is cheap. Of course this would be done just if this is more suitable for them. This kind of captive demand can lead to monopolistic practices. For instance, a genco who owns several units can withhold any of them. As a result, the market clearing price will shift upward. This is because more expensive generators will provide the energy which was being produced by the cheaper generator withheld from the system. Figure 2.8(b) shows this situation where the market clearing price went up from 12 to 18.



(a) Marginal benefit price curve (b) Benefit function

FIGURE 2.8: The fixed load model.

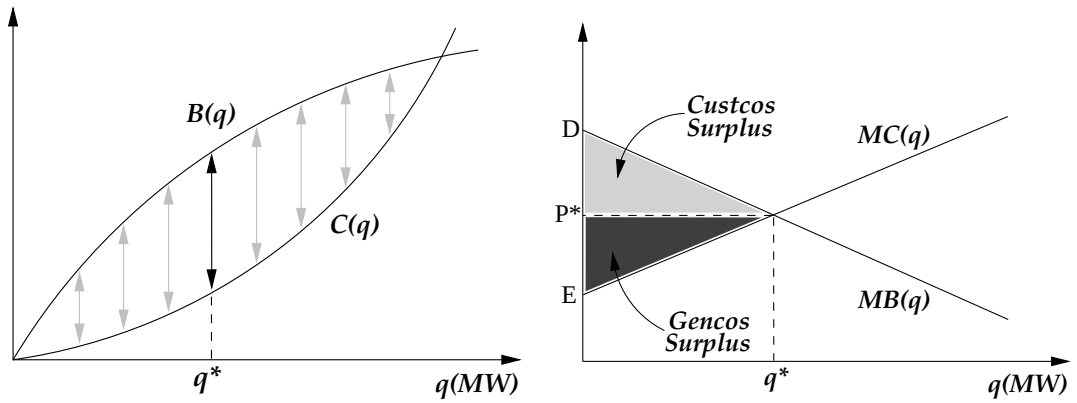
2.2.6 Market Clearing Price Revisited

The price settlement is the last stage in an auction. There are many ways to set the price for the commodity once the winner has been determined. In electrical power markets, the price settlement is a uniform price known as market clearing price. All the producers who are granted a production level will receive this price for each energy unit even if their bid was lower than that price. On the other hand, all the consumers who were granted a consumption level will pay this price for each energy unit they consume even though their request offer was greater than this price.

In this section, we will have a closer look at the market clearing price from the welfare economics point of view. Let us denote q^* as the optimal energy consumption level which has to be reached by both producers and consumers. As before, let us define $B(q)$ and $C(q)$ as the benefit function and cost function respectively. Solving this problem as the maximum social welfare, implies to maximise the total benefit gained from the consumption of q^* energy minus the cost to produce it. From figure 2.9(a), the solution to this problem is the production level q^* where the difference $B(q^*)-C(q^*)$ is the largest (i.e. we are looking for the production level q^* where the maximum separation between B(q) and C(q) takes place). On the other hand, when this problem is translated into its marginal representations for $B(q)$ and $C(q)$ (i.e. $\frac{dB(q)}{dq}$ and $\frac{dC(q)}{dq}$) the interpretation changes. Now, the production level q is that where the price p for both functions is the same (i.e. the intersection point), as shown in figure 2.9(b).

2.3 Electrical Power Systems

In general, an electrical power system, can be thought of as a huge electrical circuit where there are suppliers and consumers. Suppliers are represented by those generating companies, known as the Supply Side and consumers are represented by loads, known

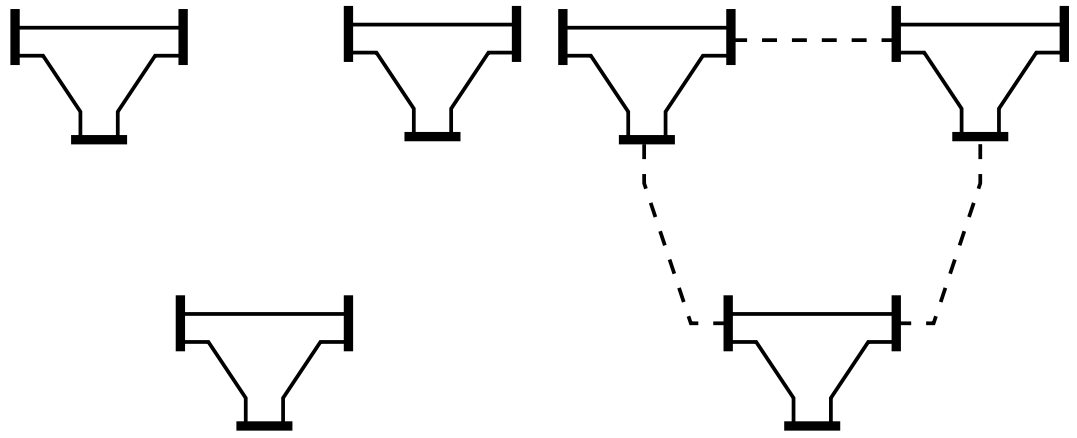


(a) $B(q)$, $C(q)$, and q^* (b) $MB(q)$, $MC(q)$, and q^*

FIGURE 2.9: The optimal production $q*$ and its relation to $B(q)$, $C(q)$, $MB(q)$, and $MC(q)$

as the Demand Side. Among these two set of players there must exist an appropriate infrastructure in order to transport the energy from the generation points to the consumer points. Essentially, this infrastructure is the transmission system itself along with its control system. The main difference between electrical power systems and normal circuits, besides the huge management of energy, is that it has to be under certain physical constraints mainly frequency, voltage, and transmission lines capacity.

This kind of electrical power systems evolved along the time in different locations worldwide as shown in figure 2.10(a). As a result of economical opportunities or political reasons, interconnections between the neighbouring areas started to appear as shown in figure 2.10(b).



(a) Isolated EPS developed worldwide (b) Then they were interconnected

FIGURE 2.10: The transition from isolated to interconnected power systems

Each country has its own electrical power system, which in general is the interconnection of several regional systems. Nevertheless, interconnections between countries are very common, as in the European Union. In this section, we will describe two main concepts

used in the analysis of these systems. First, the DC power flow method, a tool used to know the state of the system, is described. Then, the optimal power flow whose output allow us to know also the state of the system with economical efficiency. In addition, it computes the values which the controls must be set to in order to operate within the constraints of the system. Furthermore, it provides us with economical information about the price the energy must have.

2.3.1 DC Power Flow Model

An AC power flow model, is the main tool used to determine the values for the electric power system variables. It will tell us the voltage and the phase angle at each node of the system. Out of these quantities and the topological data of the system, a great variety of information can be obtained (i.e. the active power flowing in each line). This information can be used by the system operator in order to determine several aspects in the short term (i.e. line overflow), as well as in the long term (i.e. expansion planning).

A DC power flow model is a simplification of the AC power flow model (Wollenberg and Wood, 1996). In this approach, a linear model is derived which gives an easy way to calculate power flows when the power injected into the system is changed. Also by applying superposition, it is possible to know the effect of every power transaction in each line of the system. These simplifications will cost some precision. Nevertheless, as a means to explain the different situations posed by the different electrical markets decisions, is a good tool while the AC model tends to obscure such aspects. Finally, the results are fairly close with those of the AC power flow solution (Overbye et al., 2004; Purchala et al., 2005, 2006).

Let us define $\mathbb{N}$ as the set of nodes in the EPS. The DC Power Flow model is represented by equation 2.4 which has to be applied to every node $i \in \mathbb{N}$.

$$\sum_{j \in \Gamma_i} b_{ij}(\delta_i - \delta_j) - \sum_{g \in \mathbb{G}_i} p_{ig} + \sum_{l \in \mathbb{L}_i} q_{il} + Q_i = 0, \qquad \forall i \in \mathbb{N} \tag{2.4}$$

where

b_{ij}, is the susceptance of the line connecting nodes i and j.

δ_i, δ_j, are the phase angles at nodes i and j respectively,

p_{ig}, is the power produced by generation unit g connected to node i,

q_{il}, is the power consumed by the variable load l connected to node i

Q_i, is the fixed load connected to node i,

Appendix A presents a complete derivation for the DC power flow formulation.

2.3.2 The DC Optimal Power Flow

The DC power flow, solves the system from the electrical point of view. This means, it finds a solution for the system but it does not take into account the economic implications (i.e. if this solution will lead to an optimal use of the resources). The first techniques used in power markets to analyse the system taking into account economic efficiency was the economical dispatch. There the objective was to get the optimal generation costs (i.e. minimise generation costs). It did not include the transmission model. Why is the transmission model so important? When the power which flows through the line reaches its maximum power transfer limit, this line is said to be *congested*.

Therefore, in order to find a feasible solution for the electric power market, the economical dispatch was solved. Its solution was fed into the DC power flow. If the solution was feasible, then the problem had been solved. If not, another solution had to be found.

The DC optimal power flow is a quadratic separable program used to clear the power market (Rau, 2003). The optimal power flow extends the economical dispatch problem by introducing the network constraints. Congestion handling gives the difference between the economical dispatch and the optimal power flow. In systems where there is no congestion, they are equivalent. When congestion appears in the system this equivalence does not hold anymore. This model is described by equations 2.5 to 2.9.

$$\min_{p_g, q_l, \delta} \quad \sum_{g \in \mathbb{G}} C_g(p_g) - \sum_{l \in \mathbb{L}} B_l(q_l) \tag{2.5}$$

s.t.

$$\sum_{j \in \Gamma_i} b_{ij}(\delta_i - \delta_j) - \sum_{g \in \mathbb{G}_i} p_{ig} + \sum_{l \in \mathbb{L}_i} q_{il} + Q_i = 0 \tag{2.6}$$

$$\lfloor P_{ig} \rfloor \leq p_{ig} \leq \lceil P_{ig} \rceil \quad \forall g \in \mathbb{G}_i \tag{2.7}$$

$$\lfloor Q_{il} \rfloor \leq q_{il} \leq \lceil Q_{il} \rceil \quad \forall l \in \mathbb{L}_i \tag{2.8}$$

$$|b_{ij}(\delta_i - \delta_j)| \leq \lceil F_{ij} \rceil \quad \forall j \in \mathbb{T}_i \tag{2.9}$$

The objective is to maximize the social welfare which is he difference between the benefit of consuming the commodity and the cost to produce it. This is done by maximizing the difference between the total benefit the consumers obtain from the use of the energy they consume and the cost to produce it, denoted by equation 2.5. This is constrained by the generation units' physical limitations expressed by inequality 2.7 and the energy balance which has to be observed at each node and the power flow limits the lines can carry, described by equation 2.6. But it is also constrained economically by the consumer behavior. They have a minimum and maximum price they are willing to pay for the energy. This is implicit in inequality 2.8, as explained in section 2.2.5.1. Furthermore, it

is also constrained by the environment as it has to keep the power which flows through the lines within its limits, as expressed in inequality 2.9.

2.4 Optimisation

In this section a basic background on optimisation is provided as well as the mathematics which support it. First, Non Linear Programming is introduced. After this, the basics for convex programming are given. Then, Karush-Kuhn-Tucker conditions as a tool to asses optimality in convex problems are presented. Following, quadratic programming is introduced as the problem faced in this research is quadratic. Furthermore, this problem is separable. Hence, quadratic separable programming is described next. Finally, a Newton's Method revision is done.

2.4.1 Non Linear Programming

Figure 2.11 shows a non-linear function. In optimisation we are interested in the extreme points (i.e. maxima and minima). In this example those points are A, B, C, D. In general, if there exists an inflexion at point $(z, f(z))$, then this is called relative maximum or minimum. Therefore, points A, C are local maxima, while B, D are local minima. There are two special elements in this set. The global maximum is the largest element of the local maxima (i.e. point A in our example), and the global minimum is the smallest element of the local minima (i.e. point B in our example).

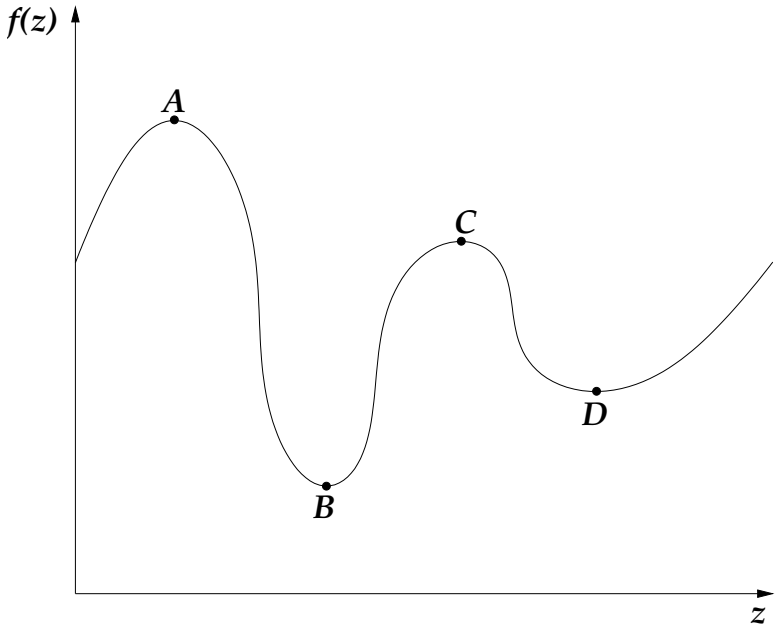


FIGURE 2.11: A nonlinear function

The technique used in order to find those points is called non linear optimisation. Nevertheless, when applied to real problems, these problems have some constraints which have to be fulfilled. Non-Linear Programming (NLP), is the general name of a set of techniques used to solve such constrained optimisation problems. If the points which are found by the unconstrained solver fulfill the constraints then, the NLP will find the same solutions. Otherwise, the NLP will find the point closest to the solution within the constraint set.

In equation 2.10, the structure of a non-linear mathematical problem is shown. Here, all functions can be non-linear.

$$\begin{aligned} &\min_{\boldsymbol{z}} f(\boldsymbol{z}) \\ \textbf{st.} \quad & g_j(\boldsymbol{z}) = 0, \qquad j = 1, 2, ..., m \end{aligned} \tag{2.10}$$

This problem alternatively can be expressed as equation 2.11

$$\begin{aligned} &\max_{\boldsymbol{z}} -f(\boldsymbol{z}) \\ \textbf{st.} \quad & g_j(\boldsymbol{z}) = 0, \qquad j = 1, 2, ..., m \end{aligned} \tag{2.11}$$

In the following sections we will assume the objective function needs to be minimised. Therefore the extreme points we are looking for are the minima.

2.4.2 Convex Programing

NLP methods where the objective function has multiple local minima are very difficult to deal with, as these must distinguish between local and global minima. Nevertheless, there are many problems which have only one minimum. Furthermore, there are functions with multiple minima which can be thought of as if it had just one extreme point. This is because in the constrained region there is just one such minimum. In order to understand the techniques used for this kind of NLP problems, the concept of convex function has to be defined.

A function $f(\boldsymbol{z})$ is called convex if for any two points $\boldsymbol{z}', \boldsymbol{z}'' \in \mathbb{R}^n$ and $0 \leq \alpha \leq 1$, constraint 2.12 holds.

$$f(\alpha \boldsymbol{z}' + (1-\alpha)\boldsymbol{z}'') \leq \alpha f(\boldsymbol{z}') + (1-\alpha) f(\boldsymbol{z}'') \tag{2.12}$$

Figure 2.12 shows a convex curve for $n = 1$

Based on model 2.10, the NLP problem is called a non-linear convex problem, if functions $f(\boldsymbol{z})$, $g_j(\boldsymbol{z})$ and $h_k(\boldsymbol{z})$ are convex. Convex programming sets the basis to solve

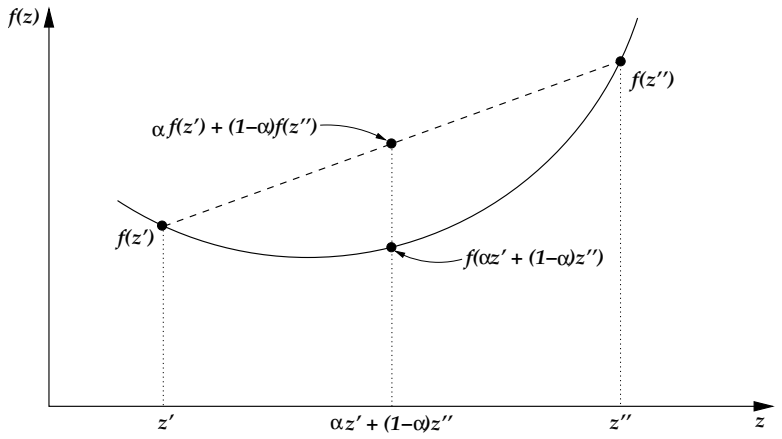


FIGURE 2.12: A convex function

optimisation problems for convex functions in $\mathbb{R}^n$. The existence of just one local minimum, and therefore a global minimum, is the main characteristic for convex functions. As a result, very efficient techniques can be applied in order to solve the system.

2.4.2.1 Lagrange Multipliers and its graphical interpretation

The solution to non-linear problems with equality constraints was developed by Joseph Louis Lagrange. A basic setting for this problem is shown in figure 2.13. Here $f(\boldsymbol{z})$ is the objective function and $g(\boldsymbol{z})$ is a linear constraint which has to be fulfilled at the solution.

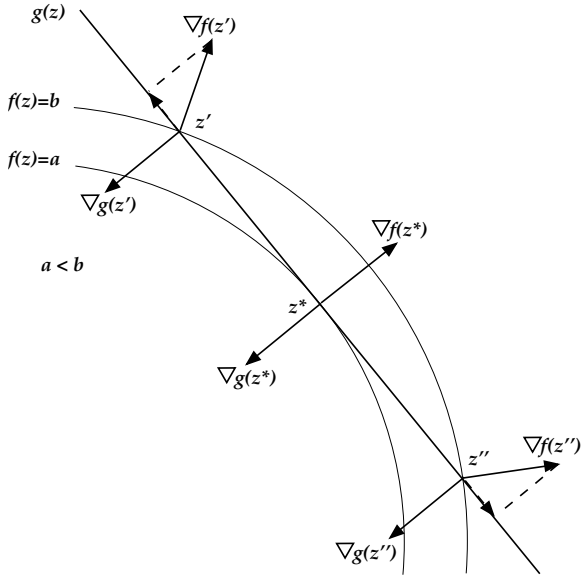


FIGURE 2.13: Graphic interpretation for Lagrange Multipliers

Let us denote the gradient for each function as $\nabla f(\boldsymbol{z})$ and $\nabla g(\boldsymbol{z})$ respectively. In this case two isocurves for $f(\boldsymbol{z})$ are presented. The first one represents the points where $f(\boldsymbol{z}) = a$, and the other one the points where $f(\boldsymbol{z}) = b$, where $a < b$. Clearly, if we

are trying to maximise $f(\boldsymbol{z})$, constrained by $g(\boldsymbol{z})$,then the solution point is where $g(\boldsymbol{z})$ touches $f(\boldsymbol{z})$ (i.e. $g(\boldsymbol{z})$ is tangent to $f(\boldsymbol{z})$). At that point both vectors $\nabla f(\boldsymbol{z})$ and $\nabla g(\boldsymbol{z})$ are parallel. Therefore the solution point has to be at point $\boldsymbol{z}^*$ where equation 2.13 holds.

$$\nabla f(\boldsymbol{z}^*) + \lambda \nabla g(\boldsymbol{z}^*) = 0 \tag{2.13}$$

In this equation, the variables represented by vector λ are know as Lagrange Multipliers or Dual Variables. These can be regarded as scaling factors for each constraint gradient vector to fulfill equation 2.13. Therefore if a NLP problem is to be solved, a mechanism is needed to build this expression. This process is achieved by defining expression 2.14 which is called the *Lagrangian*. The Lagrangian is a high dimension expression as besides solving for the original problem variables, the Lagrange Multipliers as well as the slack variables used in order to convert inequalities into equalities need to be solved.

$$\mathcal{L}(\boldsymbol{z}, \lambda) = f(\boldsymbol{z}^*) + \lambda g(\boldsymbol{z}^*) \tag{2.14}$$

Therefore in order to solve expression 2.13, the first order conditions for equation 2.14 as denoted by equation 2.15 have to be found.

$$\nabla \mathcal{L}(\boldsymbol{z}, \lambda) = \nabla f(\boldsymbol{z}^*) + \nabla \lambda g(\boldsymbol{z}^*) = 0 \tag{2.15}$$

Equation 2.15 yields to equation 2.13 whose solution leads to the optimal point for the non-linear problem. Therefore, once the non-linear program has been specified, the Lagrangian can be built and solved based on that information.

2.4.2.2 Karush-Kuhn Tucker Conditions

A discussion has been given about NLP problems with equality constraints. The Lagrange multipliers method allows us to address optimality conditions for this kind of problems. However, many problems are expressed in terms of inequality constraints as defined by 2.16.

$$\begin{aligned} & \min_{\boldsymbol{z}} f(\boldsymbol{z}) \\ \textbf{st.} \quad & g_j(\boldsymbol{z}) = 0, \qquad j = 1, 2, ..., m \\ & h_k(\boldsymbol{z}) \leq 0, \qquad k = 1, 2, ..., p \end{aligned} \tag{2.16}$$

To address this problem, the Karush-Khun-Tucker (KKT) conditions generalize the Lagrange Multipliers method by defining a minimum of conditions which, when fulfilled,

guarantee necessary conditions for the application of NLP. Furthermore, KKT conditions provide sufficient optimality conditions for convex programming problems. If we assume $\boldsymbol{z}^*$ as the optimal solution for a non linear problem with $n = |\boldsymbol{z}|$ variables, m equality constraints and p inequality constraints these conditions (Nocedal and Wright, 2006) are

$$\nabla f(\boldsymbol{z}^*) + \sum_{j=1}^{m} \lambda_j \nabla g_j(\boldsymbol{z}^*) + \sum_{k=1}^{p} \mu_k \nabla h_k(x^*) = 0 \tag{2.17}$$

$$g_j(\boldsymbol{z}^*) = 0, \qquad j = 1, 2, .., m \tag{2.18}$$

$$h_k(\boldsymbol{z}^*) \leq 0, \qquad k = 1, 2, .., p \tag{2.19}$$

$$\mu_k h_k(\boldsymbol{z}^*) = 0, \qquad k = 1, 2, .., p \tag{2.20}$$

$$\mu_k \geq 0, \qquad k = 1, 2, .., p \tag{2.21}$$

where equation 2.17 represents the equilibrium equation between the gradients of the objective and constraint functions. Equations 2.18 and 2.19 represent the feasibility of the solution at the optimal point. Equation 2.20 represents the complementarity condition (i.e. either $\mu_k = 0$ or $h_k = 0$). Finally, equation 2.21 represents the dual feasibility (i.e dual variables positiveness).

2.4.3 Quadratic Programing

Quadratic programs are non-linear programs whose objective function $f(\boldsymbol{z})$ can be expressed as a quadratic function where the constraints are linear. As before, there are n variables, m equality constraints and p inequality constraints. Let us define $\mathbf{a} \in \mathbb{R}^n$, $\mathbf{b} \in \mathbb{R}^n$, $\mathbf{C} \in \mathbb{R}^n \times \mathbb{R}^n$, $\mathbf{A} \in \mathbb{R}^m \times \mathbb{R}^n$, $\mathbf{B} \in \mathbb{R}^p \times \mathbb{R}^n$, $\mathbf{d} \in \mathbb{R}^m$ and $\mathbf{e} \in \mathbb{R}^p$. Therefore the quadratic programs can be expressed as

$$\min_{z} \mathbf{a} + \mathbf{b}'\boldsymbol{z} + \frac{1}{2}\boldsymbol{z}'\mathbf{C}\boldsymbol{z}$$

$$\begin{aligned} \mathbf{A}\boldsymbol{z} &= \mathbf{d} \\ \mathbf{B}\boldsymbol{z} &\leq \mathbf{e} \end{aligned} \tag{2.22}$$

If $\mathbf{C}$ is positive semidefinite then $f(\boldsymbol{z})$ is convex. In this case the problem is called *Convex Quadratic Programming*. To asses if $\mathbf{C}$ is positive semidefinite is a hard task. However, in our case this is guaranteed as $\mathbf{C}$ will be a diagonal matrix whose elements are all positive. In this case we are talking about Quadratic Separable Programming.

2.4.3.1 Quadratic Separable Programing

When in a quadratic program, the objective function can be separated into functions which involve just one variable, then this is called a quadratic separable program. This kind of programs are very common in real engineering systems. If the constraints are also separable, (i.e. each constraint involves just one variable), then the problem can be decomposed into subproblems, one for each variable. Of course, these are very simple systems where interactions between components do not exist. In general these constraints will involve several variables. Furthermore, these variables do not necessarily belong to the objective function. As for this research, these are the models we will be dealing with. For these problem the format is the same as for the quadratic programs. The only constraint here is that $\mathbf{C}$ is a diagonal matrix. Furthermore, this work will be dealing with diagonal positive semidefinite matrices which ensures convexity for the objective function.

2.4.4 Newton's Method

Let us consider function $f : \mathbb{R}^n \to \mathbb{R}$ and the point $\boldsymbol{z}^* \in \mathbb{R}^n$ which represents the optimal solution. Newton's method is a mathematical tool which allow us to move in $\mathbb{R}^n$ based on the information we have about point $\boldsymbol{z}_i$ from $\boldsymbol{z}_i$ to $\boldsymbol{z}_{i+1}$. It aims to reach $\boldsymbol{z}^*$ which is that one where $f(\boldsymbol{z})$ attains its extreme points (i.e. maximum or minimum). Therefore, at every step we have to end up closer to $\boldsymbol{z}^*$. Newton's Method is based on Taylor expansion of $f(\boldsymbol{z})$ at $\boldsymbol{z}_i$ as an approximation for $f(\boldsymbol{z}_{i+1})$. It is based on the values of its gradients $\nabla^{(m)} f(\boldsymbol{z}_i)$ evaluated at $\boldsymbol{z}_i$, where m stands for the gradient order. Taking this into consideration, Taylor expansion is defined as

$$f(\boldsymbol{z}_{i+1}) = \sum_{m=0}^{\infty} \frac{\nabla^{(m)} f(\boldsymbol{z}_i)}{m!} (\Delta \boldsymbol{z})^m \tag{2.23}$$

where $\Delta \boldsymbol{z} = \boldsymbol{z}_{i+1} - \boldsymbol{z}_i$ and $\nabla f(\boldsymbol{z}_i)$ is the gradient of $f(\boldsymbol{z}_i)$, evaluated at point $\boldsymbol{z}_i$.

For a grade n polynomial, a n order Taylor expansion will yield an exact result. To this end, information about its order n derivative is necessary. The functions used in this research are quadratic. Therefore, we just need a second order Taylor series polynomial to obtain the exact solution as defined in equation 2.24.

$$f(\boldsymbol{z}_{i+1}) = f(\boldsymbol{z}_i) + \nabla f(\boldsymbol{z}_i)^T \Delta \boldsymbol{z} + \frac{1}{2} \Delta \boldsymbol{z}^T H(f(\boldsymbol{z}_i)) \Delta \boldsymbol{z} \tag{2.24}$$

where $H(f(\boldsymbol{z}_i)) = \nabla^2 f(\boldsymbol{z}_i)$ known as the Hessian matrix of $f(\boldsymbol{z}_i)$. Both, ∇ and H, are evaluated at $\boldsymbol{z}_i$.

Newton's Method finds successive approximations leading to the roots of a function in $\mathbb{R}^n$. Applied to the Tailor expansion, we can see it is a function of $\Delta \boldsymbol{z}$. At the optimal point $\Delta \boldsymbol{z}$ must be zero. Therefore, applying the first order conditions to equation 2.24 and solving for $\Delta \boldsymbol{z}$ leads to equation 2.25.

$$\nabla f(\boldsymbol{z}_i) + H(f(\boldsymbol{z}_i))\Delta \boldsymbol{z} = 0 \tag{2.25}$$

this in turn yields to equation 2.26

$$H(f(\boldsymbol{z}_i))\Delta \boldsymbol{z} = -\nabla f(\boldsymbol{z}_i) \tag{2.26}$$

which in turn can be solved by using matrix computations as shown in equation 2.27

$$\Delta \boldsymbol{z} = -H(f(\boldsymbol{z}_i))^{-1}\nabla f(\boldsymbol{z}_i) \tag{2.27}$$

The approach taken to solve this system using equation 2.26 or equation 2.27 will have a great impact on the efficient solution of the system.

2.5 Concluding remarks

In this chapter the general mathematical tools which will be used in this document have been described. Firstly, a general review of the concepts underlying auctions has been presented, where the main type of auction analysed was the day ahead auction which is a periodic double multiunit auction. Then the different concepts underlying the electrical power markets and their economic models were presented. There a description for the different agents involved in such a market was given as well as the economical implications for each one of them. In the next part, the concepts needed from electrical power systems have been given along with a description of the different elements which they are built from. The principal constraints which the electrical power system has to fulfill have also been described. Finally the main topics on optimisation needed for this document have been depicted. In the next chapter a discussion will be given on how to address these optimisation problems by using graph-based methods.

Chapter 3

Systems of Linear Equations and their Graphical Solution

The solution to a large kind of problems can be addressed by solving a system of linear equations (SLE). In particular, the main tool used to solve non-linear optmisation problems are based on Newton's method. This method solves iteratively an SLE until convergence is reached (if the solution exists). Therefore if the goal is to derive efficient algorithms to solve non-linear optimisation problems then a deeper insight into how an SLE is solved has to be done. In this chapter the graphical solution to an SLE is described and various special cases are analysed. Section 3.1 presents a graphical representation for an SLE. Then, based on the observation that Newton's method is based on the Hessian matrix $H(\mathcal{L}(\boldsymbol{z}))$ (*see* section 2.4.4) whose structure is symmetrical, section 3.2 moves on the focus to symmetrical SLEs (SSLE) and their graphical representation. Next section 3.3 presents a closer look to special SSLEs whose structure are represented by a tree (TSSSLE), including the algebraic solution for such a system. After becoming familiar with the different classes of SLE and their graph representation, the solution to those graphs is addressed. Section 3.4 presents the graphical interpretation for the Gaussian elimination. Next, section 3.5 compares the different strategies to solve the TSSSLE graph using the transformations introduced in section 3.4. Following, a graph-based solution for tree structured symmetric systems of linear equations is given in section 3.6. Then section 3.7 presents a graph-based representation for the Newton's method. Finally section 3.8 provides some concluding remarks.

3.1 Graphical Representation for Systems of Linear Equations

Systems of linear equations represent systems of the form of equation 3.1.

$$\begin{array}{ccccc} a_{11}x_1 & \dots & a_{1j}x_i & \dots & a_{1n}x_n = b_1 \\ \vdots & & \vdots & & \vdots \\ a_{i1}x_1 & \dots & a_{ij}x_i & \dots & a_{in}x_n = b_i \\ & & \vdots & & \\ a_{in}x_1 & \dots & a_{nj}x_i & \dots & a_{nn}x_n = b_n \end{array} \tag{3.1}$$

Defining the following variables

$$\boldsymbol{A} = \begin{pmatrix} a_{11} & \dots & a_{1j} & \dots & a_{1n} \\ \vdots & & \vdots & & \vdots \\ a_{i1} & \dots & a_{ij} & \dots & a_{in} \\ \vdots & & \vdots & & \vdots \\ a_{in} & \dots & a_{nj} & \dots & a_{nn} \end{pmatrix} \qquad \boldsymbol{x} = \begin{pmatrix} x_1 \\ \vdots \\ x_i \\ \vdots \\ x_n \end{pmatrix} \qquad \boldsymbol{b} = \begin{pmatrix} b_1 \\ \vdots \\ b_i \\ \vdots \\ b_n \end{pmatrix}$$

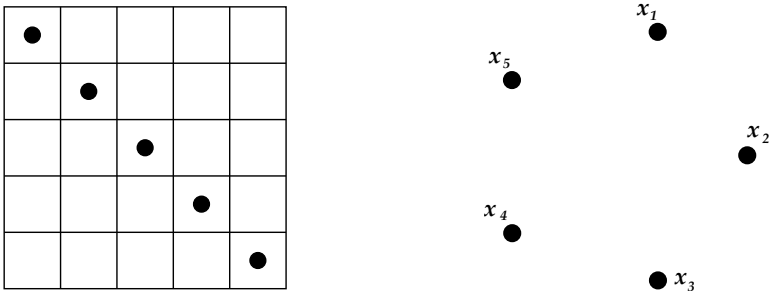
where $\boldsymbol{A} \in \mathbb{R}^n \times \mathbb{R}^n$, $\boldsymbol{x} \in \mathbb{R}^n$ and $\boldsymbol{b} \in \mathbb{R}^n$. Therefore, expression 3.1 can be described in matrix notation by equation 3.2.

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \tag{3.2}$$

The non-zero elements in $\boldsymbol{A}$, excluding those in the diagonal, define the topology of the graph which represents the SLE. If $\boldsymbol{A}$ is very sparse then the graph will have very few interconnections. Sparse systems can be solved using special algorithms which may need special data structures. Therefore a measure of sparsity is needed in order to evaluate how sparse a matrix is. Let us define n_z as the number of zeroed off-diagonal elements in $\boldsymbol{A}$, where $0 \leq n_z \leq n(n-1)$. Equation 3.3 defines the sparsity degree for matrix $\boldsymbol{A}$.

$$s = \frac{n_z}{n(n-1)} \tag{3.3}$$

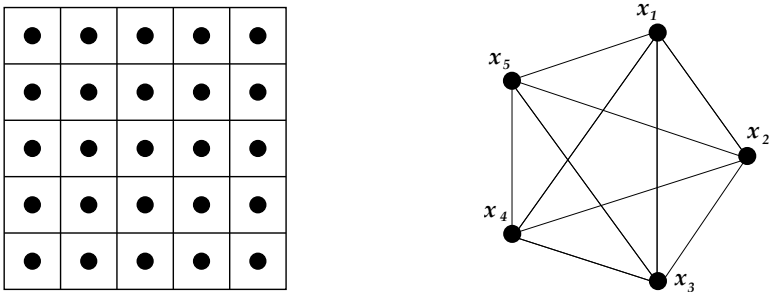
The possible values for s lie in the interval [0,1]. As the ratio between the number of zeroed elements and the number of elements if it were full grows, the sparsity degree will grow. In particular, when $s = 1$, its corresponding graph would represent a totally decoupled system. Figure 3.1 shows a decoupled system with five nodes representing a SLE with five equations and five unknowns.



(a) Decoupled matrix (b) Decoupled graph

FIGURE 3.1: A totally decoupled SLE.

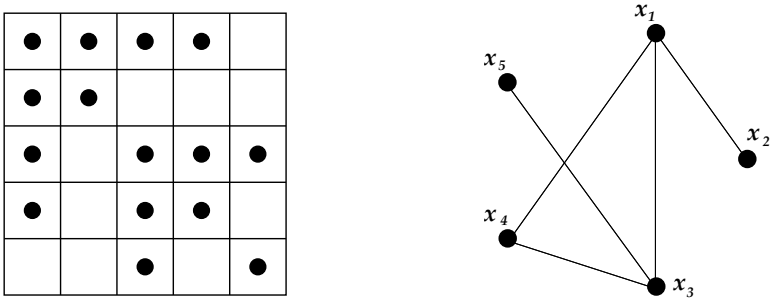
On the other hand, $s = 0$ represents a fully connected system (i.e. every node is connected to every other node), as shown in figure 3.2 representing a fully coupled SLE with five equations and five unknowns.



(a) Fully coupled matrix (b) Fully coupled graph

FIGURE 3.2: A fully coupled SLE.

Between these two extremes (i.e. $0 < s < 1$), $\boldsymbol{A}$ is neither full nor decoupled and its corresponding graph will not be fully connected (i.e. not all the nodes are connected among them). Figure 3.3 representing a SLE with five equations and five unknowns.



(a) Its matrix (b) Its graph representation

FIGURE 3.3: A sparse connected system.

It turns out that many physical systems solved using linear systems are very sparse i.e $s \to 1$. In particular, in electrical power systems, the matrices which represent transmission networks are very sparse. This is the main reason why sparsity techniques have been

improved by research whose goal is to solve efficiently the actual state of the power system; such as the best elimination ordering (Markowitz, 1957; Tinney and Walker, 1967). Sparsity techniques have been around since at least 1970 using a technique known as bifactorization (Zollenkopf, 1970). The main principles used in bifactorization are strongly directed toward exploiting the underlying matrix graph.

Therefore in order to approach the solution using its graph representation, first an appropriate model has to be derived. This model has to be able to represent the complete SLE elements (i.e. $\boldsymbol{A}, \boldsymbol{x}$, and $\boldsymbol{b}$). The model proposed in this document is based on a per equation basis. This representation is given in figure 3.4.

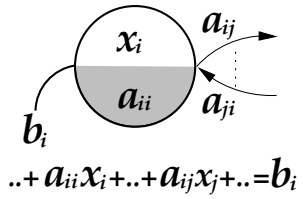


FIGURE 3.4: Conversion from a linear system of equations to its graph model

In this model an equation is represented by two components: a node and a set of links. The node is a well defined component which consists of two subcomponents: a circle consisting of two half parts and an arc. The upper part of the circle represents the variable related to this equation which has to be solved by the system (i.e. x_i) and the lower part represents the coefficient related to this variable in equation i (i.e. a_{ii}). The arc represents the $i-th$ component in $\boldsymbol{b}$ (i.e. b_i). The second part depends on the SLE topology and is represented by links which connect the nodes. These links will be denoted as (i, j) where i and j represent the row and the column number respectively. There can be zero or more links which connect the node with some other nodes in the graph. Each link has an associated value for the coefficient located in the row i column j (i.e. a_{ij}). Perhaps it is a little absurd to consider the case where there are no external links. However, as will be shown later, this is the basic configuration which will always be pursued in order to solve the SLE.

In order to illustrate these concepts let us define an example of a SLE to illustrate this process. Consider the SLE represented by equations 3.4 to 3.7

$$0.5x_1 - x_2 - x_3 = 0 \tag{3.4}$$

$$-x_1 + 2x_2 - x_3 = 0 \tag{3.5}$$

$$-x_1 - x_2 + x_3 - x_4 = -5 \tag{3.6}$$

$$-2x_3 + 4x_4 = 0 \tag{3.7}$$

Instantiating equation 3.1 with equations from from 3.4 to 3.7 leads to equation 3.8

$$\begin{pmatrix} 0.5 & -1 & -1 & 0 \\ -1 & 2 & -1 & 0 \\ -1 & -1 & 1 & -1 \\ 0 & 0 & -2 & 4 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ -5 \\ 0 \end{pmatrix} \tag{3.8}$$

Applying the model defined in figure 3.4 to each equation, the graph shown in figure 3.5 is obtained.

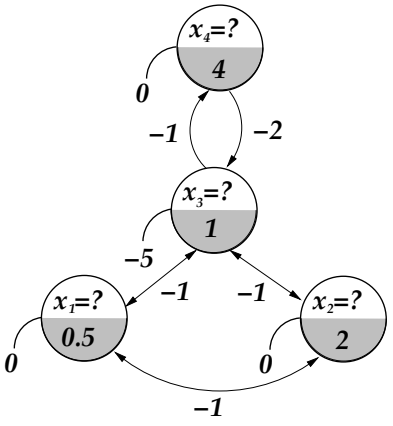


FIGURE 3.5: Graph corresponding to the system defined by equation 3.8

Unidirectional links ($\rightarrow$ and $\leftarrow$) have to be used as in general $a_{ij} \neq a_{ji}$, as shown by links $(3,4)$ and $(4,3)$, representing elements a_{34} and a_{43}. If $a_{ij} = a_{ji}$ then these elements are represented with a bidirectional link ($\leftrightarrow$) as shown by the link $(1,3)$, representing elements $a_{1,3}$ and $a_{3,1}$. This graph represents an asymmetric SLE (ASLE). An ASLE is a SLE where there exists at least one pair of links $(i,j),(j,i)$ where $i \neq j$, such that $a_{ij} \neq a_{ji}$ holds.

In this document the main aim is to express Newton's method to solve non-linear optimisation problems using a graph approach. The kind of matrices involved in such problems are symmetric, therefore the main focus will be on this subset of SLE.

3.2 Symmetric Systems of Linear Equations and Its Graphical Representation.

Symmetric systems of linear equations (SSLE) are SLEs where $a_{ij} = a_{ji}$ holds for all i, j. This document will not deal with the analysis of the properties these systems hold. The main interest here is how to represent such systems using graphs and how to exploit them in order to solve the SLE. SSLEs are very common in physical systems, in particular, a great range of problems in electrical power systems can be addressed with SSLEs. To

represent SSLEs into its graphical form, the graph representation proposed in figure 3.4 is modified as shown in figure 3.6.

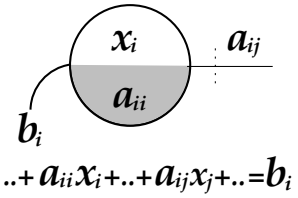


FIGURE 3.6: Conversion from a symmetric linear system of equations to its graph model

The only modification in this variant is with regard to the unidirectional links. The graphs representing SSLEs must contain only bidirectional links. The link representation has been modified and the arrows are no longer used as they do not give any extra information. In order to illustrate these concepts let us instantiate equation 3.1 with equation 3.9 which is basically equation 3.8 where the element a_{43} has been set to -1 in order to be equal with element a_{34}.

$$\begin{pmatrix} 0.5 & -1 & -1 & 0 \\ -1 & 2 & -1 & 0 \\ -1 & -1 & 1 & -1 \\ 0 & 0 & -1 & 4 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ -5 \\ 0 \end{pmatrix} \tag{3.9}$$

Applying the model defined in figure 3.6, yields the graph shown in figure 3.7

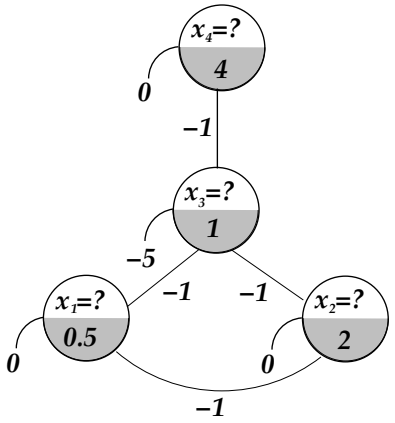


FIGURE 3.7: Graph corresponding to the system defined by equaion 3.9

SSLEs can be described as perfect SLEs; they are well behaved and their properties have been known for a long time. However, there exists a subset of SSLE which besides all those properties possessed by them, have another property. They can be represented with a graph known as a tree and as a consequence all the well known algorithms regarding trees can be applied to them. For reasons which will be explained in the

following chapters these will be the SLEs this document will be dealing with. Therefore, the attention will be focused on this kind of systems.

3.3 Tree Structured Symmetric Systems of Linear Equations and Its Graphical Representation.

Tree structured symmetrical systems of linear equations (TSSSLE) are SSLE where the graph representing the SSLE is a tree. Based on this structure, efficient algorithms can be derived in order to solve this kind of systems. These algorithms emerge naturally, just by exploiting the properties of trees. This kind of graphs have been applied to solve electrical distribution networks whose main characteristic is its radial shape (i.e. no loops exists in the network). Therefore a tree structure can be derived for the SLE representing these systems. Algorithms to solve different problems with different degree of complexity have been proposed for distribution networks based on this structure in Goswami and Basu (1991); Das et al. (1994); Chen et al. (2000); Mekhamer et al. (2002).

A tree-shaped graph has to be free of loops. This work does not deal with how to identify and remove loops from graphs. Decentralisation techniques proposed in later chapters will remove the possible loops resulting from the transmission network. Therefore, graph 3.7 will be converted into a graph representing a TSSLE by removing the link $(1, 2)$. This implies removing elements a_{12} and a_{21} from matrix $\boldsymbol{A}$. Let us instantiate equation 3.1 with equation 3.10 which is basically equation 3.9 where elements a_{12} and a_{21} have been set to 0

$$\begin{pmatrix} 0.5 & 0 & -1 & 0 \\ 0 & 2 & -1 & 0 \\ -1 & -1 & 1 & -1 \\ 0 & 0 & -1 & 4 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ -5 \\ 0 \end{pmatrix} \tag{3.10}$$

Applying the model defined in figure 3.6, leads to the graph shown in figure 3.8 which is graph 3.7 where link $(1, 2)$ has been removed.

The TSSSLE graph model will be used along this document to address different problems which must be solved in the electricity power market. This model will be enhanced in the following chapters to address different aspects which emerge during non-linear optimisation problem solving tasks, such as conditional evaluation and graph decentralisation. Before that, in the following sections different solution strategies are described for such systems.

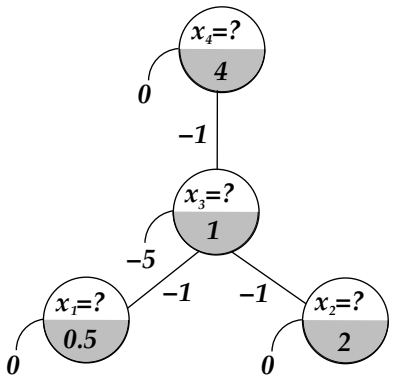


FIGURE 3.8: Graph corresponding to the system defined by equation 3.10

3.3.1 Algebraic Solution

In this section the system shown in equation 3.10 will be solved algebraically, adding rows, subtracting rows and substituting variables. To this end let us expand this system as the one given from equation 3.11 to equation 3.14

$$.5x_1 - x_3 = 0 \tag{3.11}$$

$$2x_2 - x_3 = 0 \tag{3.12}$$

$$-x_1 - x_2 + x_3 - x_4 = -5 \tag{3.13}$$

$$-x_3 + 4x_4 = 0 \tag{3.14}$$

Subtracting equation 3.12 from equation 3.11 yields equation 3.15

$$.5x_1 - 2x_2 = 0 \tag{3.15}$$

Then adding equations 3.13 and 3.14 leads to equation 3.16

$$-x_1 - x_2 + 3x_4 = -5 \tag{3.16}$$

Now adding equations 3.12 and 3.13 equation 3.17 is obtained

$$-x_1 + x_2 - x_4 = -5 \tag{3.17}$$

Equation 3.15 yields to equation 3.18

$$x_1 = 4x_2 \tag{3.18}$$

Substituting 3.18 in 3.16 and 3.17, equations 3.19 and 3.20 are derived, respectively.

$$-5x_2 + 3x_4 = -5 \tag{3.19}$$

$$-3x_2 - x_4 = -5 \tag{3.20}$$

From equation 3.20, equation 3.21 is obtained

$$x_4 = 5 - 3x2 \tag{3.21}$$

Substituting 3.21 in 3.19 x_2 is calculated

$$-5x_2 + 3(5 - 3x_2) = -5 \Longrightarrow x_2 = \frac{10}{7}$$

Using $x_2 = 10/7$ in 3.20, x_4 is solved

$$x_4 = 5 - 3(\frac{10}{7}) = \frac{5}{7}$$

Now x_1 is computed using x_2 in equation 3.18

$$x_1 = 4(\frac{10}{7}) = \frac{40}{7}$$

Finally using $x_1 = 40/7$ in equation 3.11, x_3 is solved.

$$x_3 = .5(\frac{40}{7}) = \frac{20}{7}$$

Therefore, the solution for the TSSSLE denoted by equation 3.10 is the vector 3.22

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} 40/7 \\ 10/7 \\ 20/7 \\ 5/7 \end{pmatrix} \tag{3.22}$$

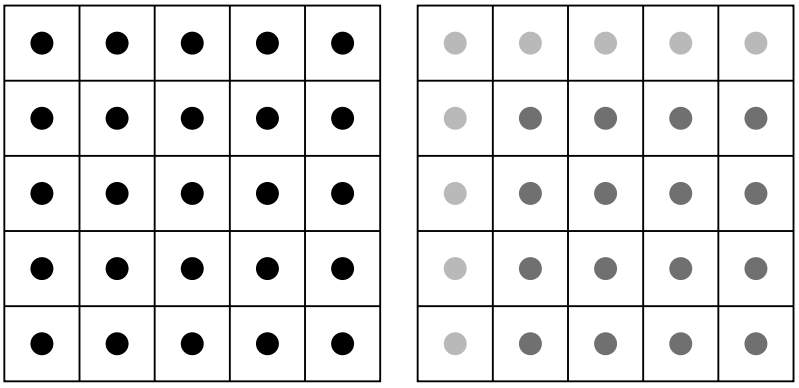
This example will be used throughout this chapter. The system will be solved using different strategies which have to lead to the same solution. To this end Gaussian elimination will be used as the main tool to solve the system.

3.4 Gaussian Elimination and Its graphical Interpretation

Gaussian elimination is a general method to solve a SLE. It consists of the iterative application of elementary row operations which lead the system to an echelon form. This is achieved by modifying each of the elements which do not belong to the column and row to the equation under reduction. These elements are modified using expression 3.23.

$$a_{ij}^{'} = a_{ij} - \frac{a_{ik}a_{kj}}{a_{kk}} \tag{3.23}$$

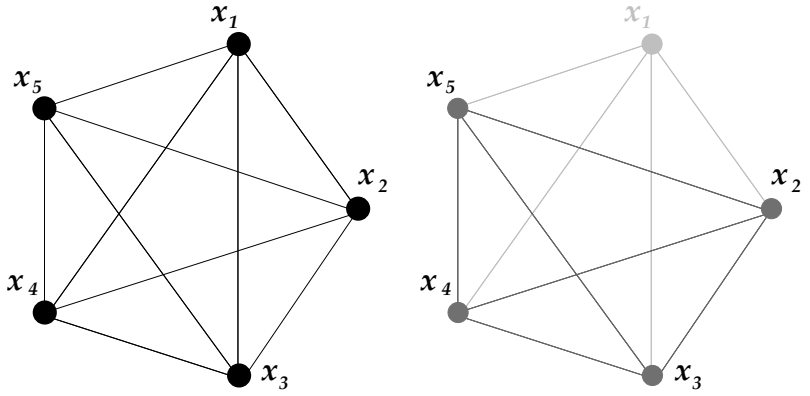
Gaussian elimination can be regarded as a matrix transformation from $\mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathbb{R}^{n-1} \times \mathbb{R}^{n-1}$. The resulting system has all the information needed to solve the subsystem resulting from the transformation. Let us assume $\boldsymbol{A}$ is a full matrix. Therefore Gaussian elimination, as shown in figure 3.9, can be thought as a transformation where the row and column (shown in light gray) where it was applied have been deactivated and the resulting SLE consists only of the remaining rows and columns which have already been modified by equation 3.23 (shown in dark gray).



(a) Before the elimination (b) After the elimination

FIGURE 3.9: Gaussian elimination and its matrix interpretation

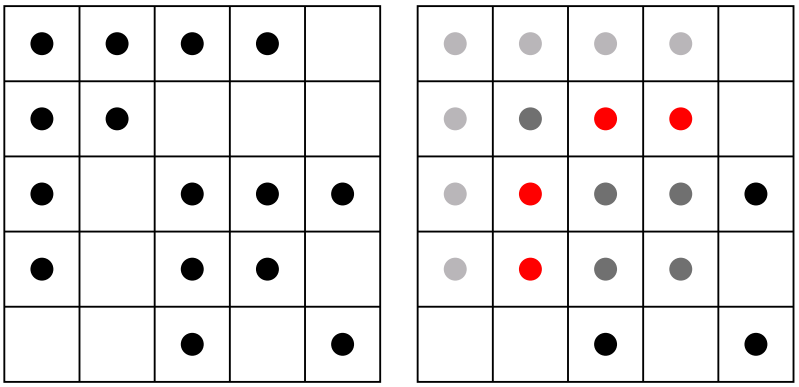
A graph interpretation for this transformation when applied to node k is shown in figure 3.10 where it can be seen all the links and nodes of the graph are modified in order to reflect such an equivalence.



(a) Before the elimination (b) After the elimination

FIGURE 3.10: Gaussian elimination and its graph interpretation.

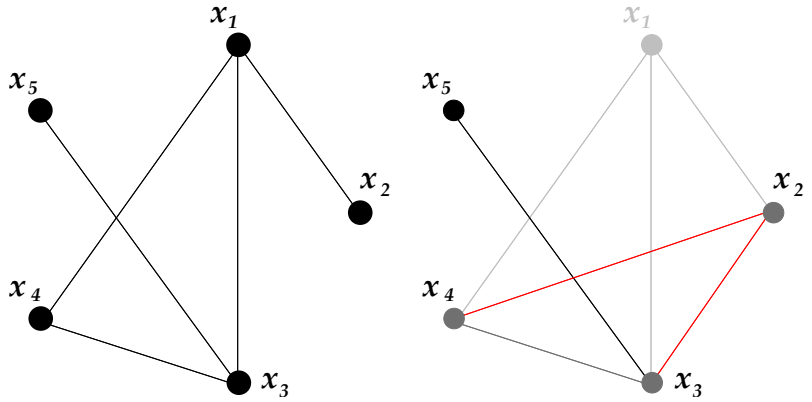
When dealing with sparse systems several observations have to be done. Figure 3.11(a) reproduces the sparse matrix defined in figure 3.3(a) and figure 3.11(b) shows the transformation it undertakes when Gaussian elimination is applied to x_1. The entries in red denote elements whose values were zero before the transformation. Nevertheless, it just denotes a change in value.



(a) Before the elimination (b) After the elimination

FIGURE 3.11: Gaussian elimination and its matrix interpretation for a sparse matrix

Now, let us analyse the transformation in its graphical representation. To this end, let us define Γ_k as the set of nodes connected to node k. In this case let us instantiate $k = 1$ as the node to be eliminated; consequently, $\Gamma_1 = \{2, 3, 4\}$. A graph interpretation for this transformation is shown in figure 3.12. Here figure 3.12(a) represents the state of the graph before the transformation is applied and figure 3.12(b) represents the state of the graph after the transformation has been applied.



(a) Before the elimination (b) After the elimination

FIGURE 3.12: Gaussian elimination and its graph interpretation for a sparse matrix

This shows that when node k is eliminated then the nodes which are connected to it, Γ_k, will form a complete graph among them as a result of this transformation. This is reflected by equation 3.24

$$\Gamma'_j \leftarrow (\Gamma_j \cup \Gamma_k) \setminus \{j, k\} \qquad \forall j \in \Gamma_k \tag{3.24}$$

Where Γ_j and Γ'_j denote the neighbour nodes of node j before and after the transformation, respectively. If nodes i and j were connected before the transformation then the value for the link (i, j) which was connecting them (shown in dark gray) will be updated

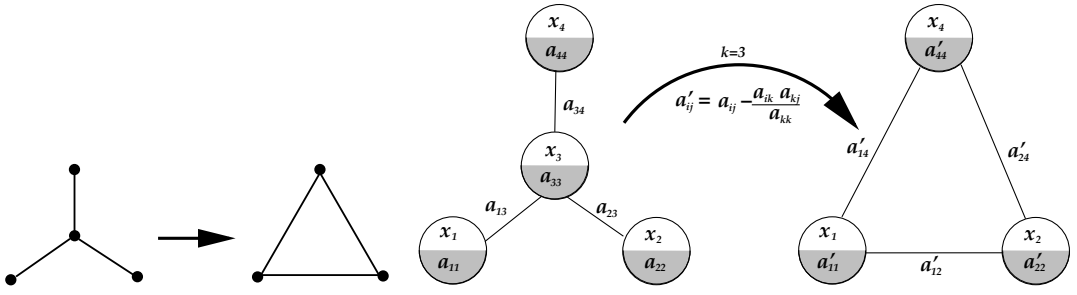
by equation 3.23. On the other hand, if they were not connected (i.e. $a_{ij} = 0$) then these interconnections would have to be created (shown in red). If no pair of nodes i, j, where $i, j \in \Gamma_k$, were connected before the transformation then a complete subgraph would be created among them. The number of links needed to build this subgraph, N_k, is given by equation 3.25.

$$N_k = \frac{|\Gamma_k| \left(|\Gamma_k| - 1\right)}{2} \tag{3.25}$$

Therefore, Gaussian elimination has two costs: one which has to be applied every time is updating, and the second one is the creation of new links, known in the literature as *fill-ins*. Furthermore, from figure 3.12(b) link $(3, 5)$ and node 5 were not used at all in the transformation. Here is where the power of sparse methods appears in systems whose components are lossely coupled as they do not deal with elements not involved in the transformation. Obviously, the burden set by sparse methods have to be avoided if the systems under study are known to be very full matrices which derive almost complete graphs tranformations. Berry and Heggernes (2003) gives a deeper description about the modifications of the graph as the reduction process is applied.

3.4.1 Special Cases for Gaussian Elimination

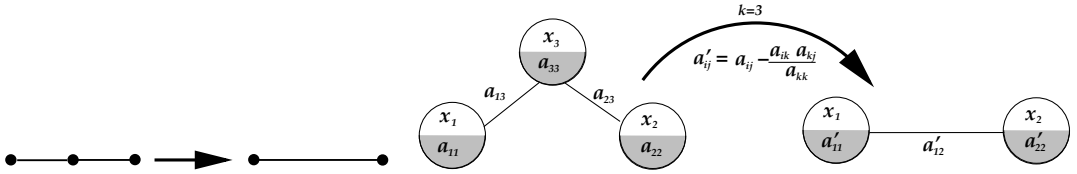
As mentioned previously, Gaussian elimination has two main costs: updating and link creation. The first one is unavoidable but the second one can be optimised if an elimination order is achieved such that the number of links created are minimal, or in the ideal case no links are created. There are three special cases which deserve special attention. The first one is shown in figure 3.13. This is the well known as star-delta transformation in electrical engineering. Here $|\Gamma_k| = 3$ and by equation 3.25 at most three new links will be generated.



(a) Schematic representation (b) Complete transformation

FIGURE 3.13: Gaussian elimination for $|\Gamma_k| = 3$.

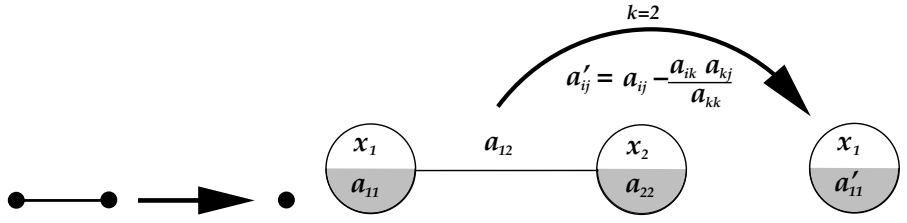
The second one is shown in figure 3.14. This is the well known series reduction in electrical engineering. Here $|\Gamma_k| = 2$ and by equation 3.25 at most one new link will be generated.



(a) Schematic representation (b) Complete transformation

FIGURE 3.14: Gaussian elimination for $|\Gamma_k| = 2$.

The third and most important case regarding this document , as it will be shown along this chapter, is shown in figure 3.15. This configuration can be regarded as a dangling node which can be reduced into the node where it is dangling from. Here $|\Gamma_k| = 1$ and by equation 3.25 there will be no new links generated. This basic fact will be used in order to reduce the TSSSLE as the leafs can be regarded as dangling nodes. Once they are reduced, then the nodes where they were dangling from can be reduced, and so on.



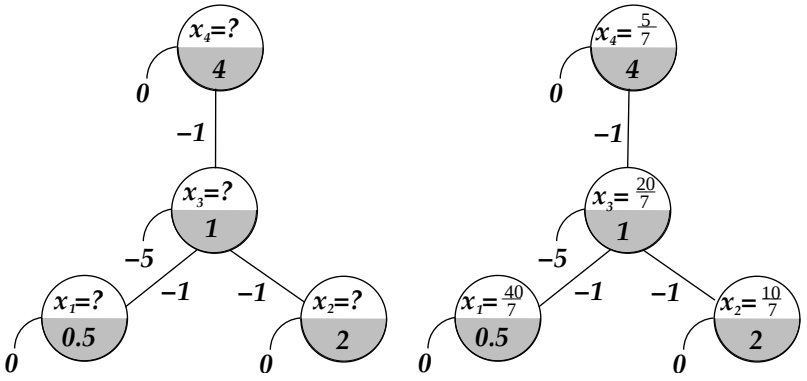
(a) Schematic representation (b) Complete transformation

FIGURE 3.15: Gaussian elimination for $|\Gamma_k| = 1$.

3.5 Graph-based Solution for Symmetric Systems of Linear Equations

Solving a SLE, when translated to its graph counterpart, means to go from the configuration shown in figure 3.16(a) to the configuration show in figure 3.16(b), where all the variables have been solved as it was done in detail in section 3.3.1 using an algebraic approach. It is desirable to end up with the same values as the initial configuration but as it will be seen this is not possible as the successive application of the Gaussian elimination will modify these values. Furthermore, the final configuration will depend on the order in which the Gaussian elimination was applied.

How were these values obtained? There are several methods to solve SLE which are based on Gaussian Elimination. Here the graph is reduced by applying Gaussian elimination using the reduction shown in figure 3.15(b), one node at a time iteratively, until the graph is reduced to just one node. This method is known as forward elimination. At this point the system can be solved as its configuration is



(a) Its original configuration (b) Its final configuration

FIGURE 3.16: A graph system.

$$a'_{ii}x_i = b'_i$$

from this x_i is solved with a value of

$$x_i = \frac{b'_i}{a'_{ii}}$$

Then a process called backward substitution can be applied by solving the previous node and so on. It is important to keep the tree structure in the elimination process as it will perform the fastest and cheapest solution for the SLE. Let us apply the Gaussian elimination to the example graph which in figure 3.16 is reproduced.

The first question is which node has to be applied the elimination on? A more advanced question is which elimination order has to be applied?. This is an open question and has been addressed in different scenarios. There are several elimination orders which can be taken in order to go from configuration 3.16(a) to configuration 3.16(b). In fact, the total number of elimination orders, N_e, for a system with n variables and n equations is n^n. However, some of them can not be applied as they would lead to an inconsistent system. An inconsistent configuration appears when the node where Gaussian elimination is to be applied is zero. This would lead equation 3.23 to an undefined value and would stop the reduction process. If we derive a configuration where all the nodes are zero then the system is said to be singular. Therefore, in its matrix version, this situation is avoided by interchanging (or renumbering) rows and columns, provided the system is not singular. In the graph representation just an inspection have to be done at the actual node where the Gaussian elimination is to be applied. If its value is zero then its reduction is delayed to a later moment.

In this section, three different orders will be applied to the graph system shown in figure 3.16(a) in order to obtain some insight about the Gaussian elimination process

(applying all the possible orders implies $4! = 24$ elimination orders). The first elimination order shown in 3.17 is $1 \rightarrow 2 \rightarrow 4 \rightarrow 3$. First, node one, using the reduction shown in figure 3.15(b), is reduced into node 3 as shown in figure 3.17(a). In the same way, node 2 is reduced into node 3 as shown in figure 3.17(b). Finally, as for the reduction process, node 4 is reduced into node 3 as shown in the upper part of figure 3.17(c). Now x_3 can be solved, as shown in bottom part of figure 3.17(c). Once x_3 is solved, the substitution process can be applied as node $1, 2$, and 4 were connected to node 3 only. This process leads to the configuration shown in figure 3.17(d). As it can be appreciated, no new links are created. Furthermore, node 3 is the only node whose initial configuration is modified.

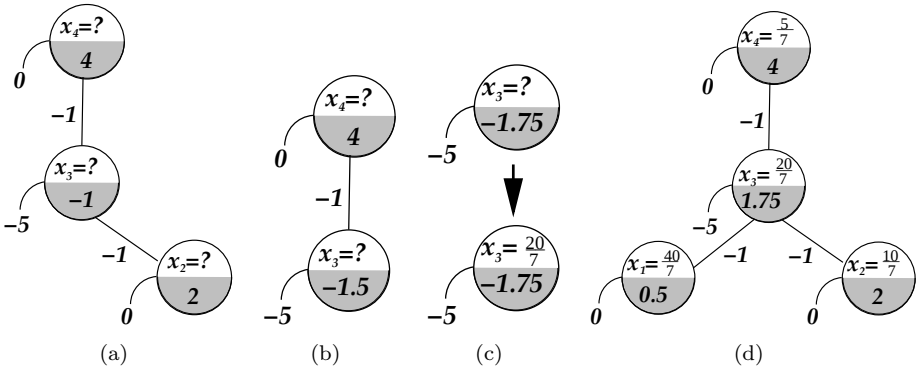


FIGURE 3.17: The graph solution with elimination order $1 \rightarrow 2 \rightarrow 4 \rightarrow 3$

The second elimination order shown in 3.18 is $1 \rightarrow 2 \rightarrow 3 \rightarrow 4$. Here, no new links are created. However, the initial configuration for nodes 3 and 4 has been modified. Furthermore, once x_4 is solved just x_3 can be solved and the rest of the variables after this using backward substitution.

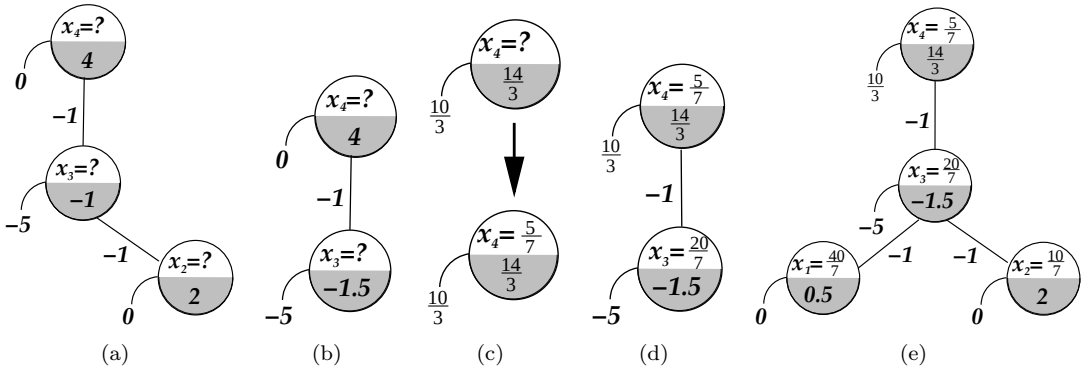


FIGURE 3.18: The graph solution with elimination order $1 \rightarrow 2 \rightarrow 3 \rightarrow 4$

The third elimination order shown in 3.19 is $3 \rightarrow 4 \rightarrow 2 \rightarrow 1$. Here, three new links are created when node 3 is eliminated as $|\Gamma_3| = 3$. The initial configuration for all nodes has been modified. Furthermore, in order to solve x_3 the rest of the variables have to be solved. To solve x_4, first x_1 and x_2 have to be solved. Finally, to solve x_2, first x_1 must be solved.

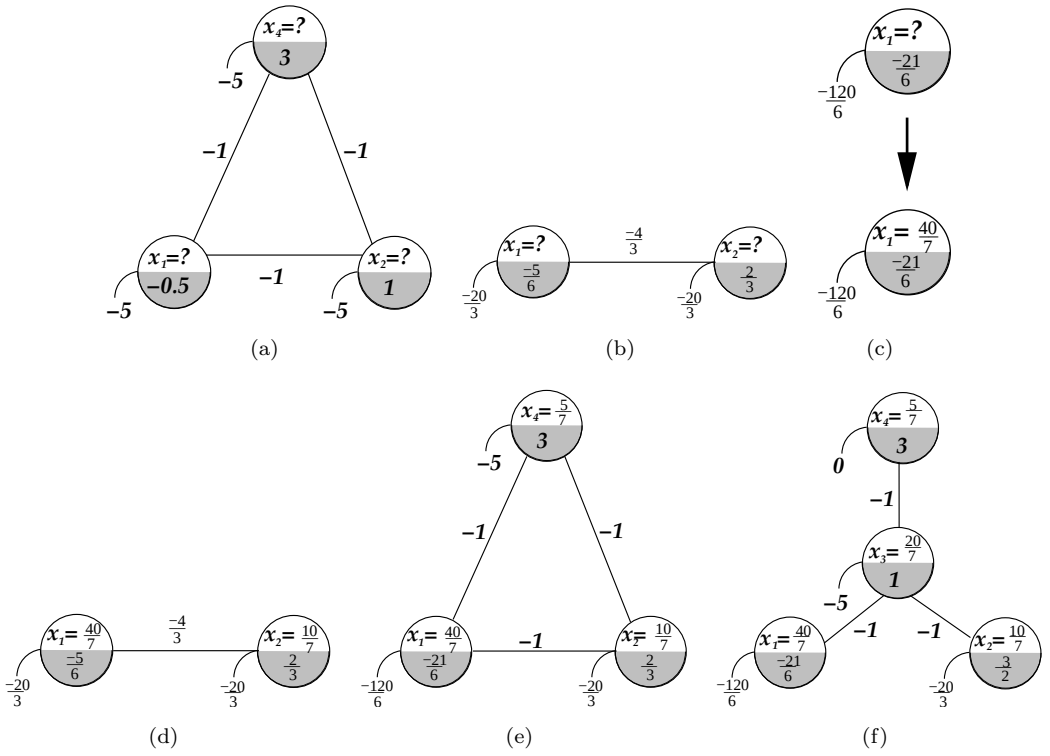


FIGURE 3.19: The graph solution with elimination order $3 \rightarrow 4 \rightarrow 2 \rightarrow 1$

3.5.1 About the Importance of the Initial Configuration

In the previous examples, the modification to the initial configuration was mentioned. This is important to preserve or at least try to preserve it as much as possible as there are iterative algorithms which will be using this configuration in order to reach the solution. If the algorithm which solves the graph modifies this configuration at every iteration, then this will have to be reinstantiated in each of them.

3.6 Graph-based Solution for Tree Structured Symmetric Systems of Linear Equations

A tree is a special kind of graph whose main characteristic is the absence of loops. A standard representation for a tree is presented in figure 3.20. A tree has n layers, $n \geq 1$, numbered from 0 to $n-1$. The number of layers represents its depth. A function $layer(x)$ can be defined to express the layer where node x is located, for instance $layer(E) = 2$. Another useful function is $parent(x)$ which returns the node where node x is hanging from or nil if it has no parent, for instance $parent(E) = B$ and $parent(A) = nil$.

Let us denote Υ_i as the set of indices j which corresponds to variables x_j which are hanging from x_i as denoted by equation 3.26 (i.e. the *children* of x_i).

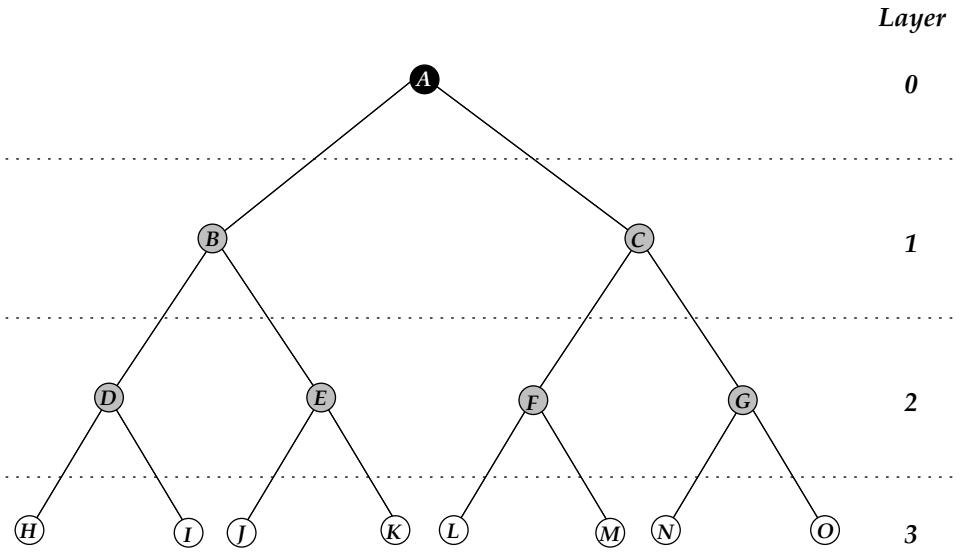


FIGURE 3.20: An example of a tree.

$$\Upsilon_i = \{j : \exists(i,j), parent(i) \neq j\} \tag{3.26}$$

For instance $\Upsilon_A = \{B, C\}$, and $\Upsilon_G = \{N, O\}$ in figure 3.20. Based on these definitions, the nodes in a tree can be classified in three types

- *root* node: There is just one of them in the graph such that $layer(root) = 0$ and $parent(root) = nil$. For instance the root node in figure 3.20 is A (in black),
- *internal* node: are those nodes x such that $\Upsilon_x \neq \emptyset$ and $x \neq root$. The set of internal nodes, I, in figure 3.20 is $\{B, C, D, E, F, G\}$ (in gray),
- *terminal* node: also called *leaf* node, are those nodes x where $\Upsilon_x = \emptyset$. Therefore the set L representing the leaf nodes in figure 3.20 is $\{H, I, J, K, L, M, N, N, O\}$ (in white).

This kind of graphs are one of the most useful structures in computer science. Their application spans from context-free grammar analysers in compiler theory, XML representation in internet technologies, and so on. A common characteristic about those applications is that only the leaf nodes have real information. However, when applied to TSSSLEs, the internal nodes will have some information which need to be processed in order to solve the SLE.

It is very important to preserve the tree structure of the system in the reduction process. The best way to reduce these graphs is by finding an order which does not generate new elements at all. From figures 3.13, 3.14 and 3.15 a basic fact can be asserted. The only reductions which do not generate new elements are those which are applied to nodes

k where $|\Gamma_k| = 1$. Therefore those are the ideal candidates to apply the elimination process.

Gaussian elimination when applied to all nodes j where $j \in \Upsilon_i$, is expressed in equations 3.27 and 3.28 for the coefficient corresponding to variable x_i and the independent term respectively.

$$a'_{ii} = a_{ii} - \sum_{j \in \Upsilon_i} \frac{a_{ij}^2}{a_{jj}} \tag{3.27}$$

$$b'_i = b_i - \sum_{j \in \Upsilon_i} \frac{b_j a_{ij}}{a_{jj}} \tag{3.28}$$

The solution for the SLE once the reduction process has been applied is straightforward. To this end, advantage is taken from the modifications the reduction process in its way up has made to the tree. It has updated the value of b_i and a_{ii} when their dependencies with the lower layer were eliminated. Let us suppose $parent(i) = k$. Then, the equation to be solved for x_i is

$$a_{ii}x_i + a_{ik}x_k = b_i \tag{3.29}$$

which is solved as

$$x_i = \frac{b_i - a_{ik}x_k}{a_{ii}} \tag{3.30}$$

clearly for this equation if node i is the tree root, then there will be no more layers up so this equation becomes

$$x_i = \frac{b_i}{a_{ii}} \tag{3.31}$$

Therefore, a top down propagation approach can be applied in order to solve the nodes in the lower layers. This can be based on a Breadth-first (BF) or a Depth-first (DF) algorithm (Cormen et al., 2001). Nevertheless, if the shape of the graph is known apriori, then more efficient algorithms can be proposed in order to speed up the computations avoiding recursivity or the handling of other auxiliary data structures.

3.7 Converting the Newton's Method into a Graph-based System

As denoted in section 2.4.4, Newton's method is a linearization for $\mathcal{L}(\boldsymbol{z})$ defined by the SSLE denoted in 3.32 when applied to the Lagrangian (i.e. $f(\boldsymbol{z}) \equiv \mathcal{L}(\boldsymbol{z})$).

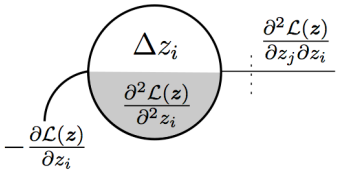
$$H(\mathcal{L}(\boldsymbol{z}))\Delta\boldsymbol{z} = -\nabla\mathcal{L}(\boldsymbol{z}) \tag{3.32}$$

Expanding this expression leads to

$$\begin{pmatrix} \frac{\partial^2 \mathcal{L}(\boldsymbol{z})}{\partial z_1 \partial z_1} & \cdots & \frac{\partial^2 \mathcal{L}(\boldsymbol{z})}{\partial z_1 \partial z_j} & \cdots & \frac{\partial^2 \mathcal{L}(\boldsymbol{z})}{\partial z_1 \partial z_n} \\ \vdots & & \vdots & & \vdots \\ \frac{\partial^2 \mathcal{L}(\boldsymbol{z})}{\partial z_i \partial z_1} & \cdots & \frac{\partial^2 \mathcal{L}(\boldsymbol{z})}{\partial z_i \partial z_j} & \cdots & \frac{\partial^2 \mathcal{L}(\boldsymbol{z})}{\partial z_i \partial z_n} \\ \vdots & & \vdots & & \vdots \\ \frac{\partial^2 \mathcal{L}(\boldsymbol{z})}{\partial z_n \partial z_1} & \cdots & \frac{\partial^2 \mathcal{L}(\boldsymbol{z})}{\partial z_n \partial z_j} & \cdots & \frac{\partial^2 \mathcal{L}(\boldsymbol{z})}{\partial z_n \partial z_n} \end{pmatrix} \begin{pmatrix} \Delta z_1 \\ \vdots \\ \Delta z_i \\ \vdots \\ \Delta z_n \end{pmatrix} = - \begin{pmatrix} \frac{\partial \mathcal{L}(\boldsymbol{z})}{\partial z_1} \\ \vdots \\ \frac{\partial \mathcal{L}(\boldsymbol{z})}{\partial z_i} \\ \vdots \\ \frac{\partial \mathcal{L}(\boldsymbol{z})}{\partial z_n} \end{pmatrix} \tag{3.33}$$

A fundamental fact here is that $H(\mathcal{L}(\boldsymbol{z}))$ is symmetric as $\frac{\partial^2 \mathcal{L}(\boldsymbol{z})}{\partial z_i \partial z_j} = \frac{\partial^2 \mathcal{L}(\boldsymbol{z})}{\partial z_j \partial z_i}$.

The graph-based model for a general equation in the SLE described by equation 3.33 is shown in figure 3.21. To convert this matrix system into its corresponding graph model, the following steps are required. A node is defined for each variable in $\Delta \boldsymbol{z}$. In that node the upper half circle refers to the variable whose value has to be solved, in this case Δz_i. The lower half circle refers to the coefficient for variable Δz_i, actually $\frac{\partial^2 \mathcal{L}(\boldsymbol{z})}{\partial^2 z_i}$, which in this document will be denoted as a_{z_i}. Then the independent term, represented as the small arc incident to the node, initialized with $-\nabla_{z_i} \mathcal{L}(\boldsymbol{z})$, which will be denoted as b_{z_i}. Out of this node there will possibly be interconnections with other variables represented by edges. The expression attached to these edges are the values for $H(\mathcal{L}(\boldsymbol{z}))$ in the positions relating the variables connected at each end (i.e. $\frac{\partial^2 \mathcal{L}(\boldsymbol{z})}{\partial z_i \partial z_j}$). Finally, the actual value for each variable, $\boldsymbol{z}_i$, will also be stored in the node (not shown).



$$.. + \frac{\partial^2 \mathcal{L}(\boldsymbol{z})}{\partial^2 z_i} \Delta z_i + .. + \frac{\partial^2 \mathcal{L}(\boldsymbol{z})}{\partial z_j \partial z_i} \Delta z_j + .. = -\frac{\partial \mathcal{L}(\boldsymbol{z})}{\partial z_i}$$

FIGURE 3.21: Graph model for the Newton's method

3.8 Concluding Remarks

In this chapter the graph representations for SLEs, SSLE and TSSSLE have been presented. Here we have learnt the differences among them and the fact that $TSSSLE \subset SSLE \subset SLE$. Then Gaussian elimination and its graphical interpretation has been presented. Even that different elimination orders have the same solution when Gaussian elimination is applied, some will require less operations to reach the solution. Also depending on this elimination order a variable number of new links will be created or not. Furthermore, some elimination orderings are not allowed as they will derive graphs whose pivot where gaussian elimination is to be applied is zero.

When solving a SLE, the objective is to find the values for variables represented by vector $\boldsymbol{x}$. The basic tool to find those values will be based on the Gaussian elimination. The processing task for this graph will address the previous features so no links are created at all. A TSSSLE is a SLE which can be represented with a tree. The solution to TSSSLE does not require the creation of links, therefore is a very efficient structure which this work will try to derive. Finally, a graph-based representation has been derived for the Newton's method which will be the basis for the models to be developed in the next chapters.

Chapter 4

The Economical Dispatch and its graph-based solution

The economical dispatch is a convex quadratic separable problem (QSP). The solution to a convex QSP with $\boldsymbol{z}$ representing all the variables used in the solution process (i.e. problem variables, slack variables and dual variables), is based on the Lagrangian $\mathcal{L}(\boldsymbol{z})$, its gradient ($\nabla(\mathcal{L}(\boldsymbol{z}))$) and its Hessian Matrix $H(\mathcal{L}(\boldsymbol{z}))$. The solution to the system is given as an iterative process where the initial assumption z_o is improved by solving a Newton step (i.e. $\Delta\boldsymbol{z} = -H(\mathcal{L}(\boldsymbol{z}))^{-1}\nabla(\mathcal{L}(\boldsymbol{z}))$). In the case of QSP, and if no constraint is violated, then just one step is needed in order to reach the optimal solution. If some constraints are violated, then these are activated and the process is applied again until all constraints are fulfilled. In this case the solution will be the closest valid point to the optimal solution, as some of the constraints will be binding. For large scale systems this approach is inconvenient as the size of $H(\mathcal{L}(\boldsymbol{z}))$ will grow accordingly with the number of variables in the system, $|\boldsymbol{z}|$. To complicate this procedure, the active constraint set is not static along the solution process. $H(\mathcal{L}(\boldsymbol{z}))$ has to be modified in every solution step in two ways. First, the new constraints which became active in the last step have to be added. Second, the constraints which became inactive as a result of the last step application have to be removed. From this, it can be concluded that the size and structure of $H(\mathcal{L}(\boldsymbol{z}))$ will be varying along the process.

In this chapter the Hessian Matrix structure is exploited. First a relaxed version for the economical dispatch problem is proposed where the physical limits of the generators are relaxed. The matrix formulation is transformed into a graph system as delineated in section 3.7. The topology of the resulting graph is a tree. This model is solved with the model developed in section 3.6. Then noting the results this structure provides for the problem under certain conditions are infeasible this model is enhanced. This is done to deal with the original economical dispatch problem where the limits on the generators are included. The former matrix formulation is extended to include such constraints

and its graph system is obtained according to section 3.7. The complex procedure of removing and adding constraints into $H(\mathcal{L}(\boldsymbol{z}))$ is translated into decisions whether to visit part of the graph or not based on Karush-Kuhn-Tucker (KKT) conditions.

4.1 The Economical Dispatch Problem

The economical dispatch is a problem where a set of producers are connected to a single node trying to met demand Q which is connected also to that node. Its goal is to find the production levels for each generator (i.e. p_g where $g \in \mathbb{G}$), as well as the price for the energy λ with economic efficiency. The production level for each generation unit has lower and upper limits $\lfloor p_g \rfloor$ and $\lceil p_g \rceil$ respectively. The production function is assumed to be quadratic (i.e. $C_g(p_g) = \alpha_g + \beta_g p_g + \gamma_g p_g^2$). In the same way, the benefit function is assumed to be quadratic concave (i.e. $B_l(q_l) = \beta_l q_l - \gamma_l q_l^2$) as obtained in section 2.2.5.1. Figure 4.1 shows the graphical representation for this problem.

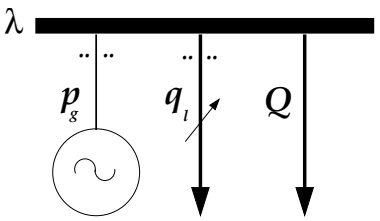


FIGURE 4.1: A general representation for the economical dispatch problem

An instance of this problem to test the graph model which will be used along this work is shown in figure 4.2 (extracted from Wollenberg and Wood (1996)).

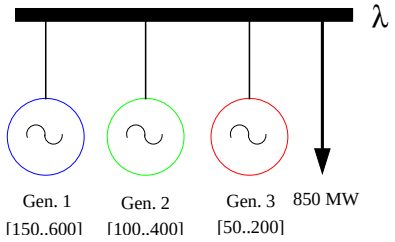


FIGURE 4.2: A system with one node and three generators

The data for this system is given in table 4.1. Two cases are analysed. The second case ($1a$) is the same as case one, the only difference being the coal price which is cheaper than in case 1 represented by smaller coefficients for generator 1 whose generation is coal-based. These values are in the second row as compared with those in the first row. The production functions with their associated limits for each generator in table 4.1 are shown in figure 4.3

gen	α	β	γ	$\lfloor p \rfloor$	$\lceil p \rceil$
1	510	7.92	0.001562	150	600
1(a)	459	6.48	0.00128	150	600
2	310	7.85	0.00194	100	400
3	78	7.97	0.00482	50	250

TABLE 4.1: Data for the system components

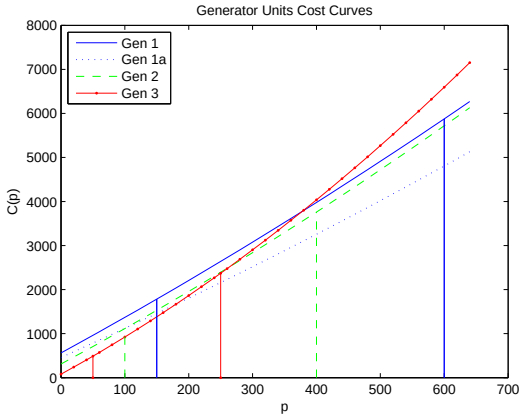


FIGURE 4.3: Production functions for the generators.

4.2 The Economical Dispatch with Ideal Generators and Its Graph-based Solution

The economical dispatch with ideal generators problem is defined within this work as the relaxed version of the previous problem where the capacity constraints associated to the active elements (i.e. generators and elastic loads) are dismissed as shown in the problem described by equations 4.1 and 4.2. Therefore the limits imposed in the generation level for the generators shown in the last two columns of table 4.1 will be disregarded in the solution process.

$$\min_{p_g, q_l} \quad \sum_{g \in \mathbb{G}} C_g(p_g) - \sum_{l \in \mathbb{L}} B_l(q_l) \tag{4.1}$$

s.t.

$$\sum_{g \in \mathbb{G}_i} p_g - \sum_{l \in \mathbb{L}_i} q_l = Q \tag{4.2}$$

Solving this NLP implies building the Hessian and the applying the Newton method to solve the system. In order to build $H(\mathcal{L}(\boldsymbol{z}))$, the Lagrangian $\mathcal{L}(\boldsymbol{z})$ has to be built. The expression for the Lagrangian for the relaxed economical dispatch problem is given by equation 4.3

$$\mathcal{L}(\boldsymbol{z}) = \sum_{g\in\mathbb{G}} C_g(p_g) - \sum_{l\in\mathbb{L}} B_l(q_l) + \lambda \left(\sum_{g\in\mathbb{G}_i} p_g - \sum_{l\in\mathbb{L}_i} q_l - Q \right) \tag{4.3}$$

where $\boldsymbol{z} = [p_g, q_l, \lambda]^T$

The equation can be solved by using a Newton approach. To this end the system represented by equation 4.4 has to be solved.

$$H(\mathcal{L}(\boldsymbol{z}))\Delta\boldsymbol{z} = -\nabla(\mathcal{L}(\boldsymbol{z})) \tag{4.4}$$

Therefore, in order to solve $\Delta\boldsymbol{z}$ first $\nabla(\mathcal{L}(\boldsymbol{z}))$ and $H(\mathcal{L}(\boldsymbol{z}))$ have to be known. Notice the solution has not been expressed as a function of $H(\mathcal{L}(\boldsymbol{z}))^{-1}$. Table 4.2, describes the relation among $\boldsymbol{z}$, $\mathcal{L}(\boldsymbol{z})$, $\nabla(\mathcal{L}(\boldsymbol{z}))$ and $H(\mathcal{L}(\boldsymbol{z}))$.

$\mathcal{L}(\boldsymbol{z}) = \sum_{g\in\mathbb{G}} C_g(p_g) - \sum_{l\in\mathbb{L}} B_l(q_l) + \lambda \left(\sum_{g\in\mathbb{G}_i} p_g - \sum_{l\in\mathbb{L}_i} q_l - Q \right)$				
	$\nabla(\mathcal{L}(\boldsymbol{z}))$	$H(\mathcal{L}(\boldsymbol{z}))$		
$\boldsymbol{z}$		p_g	q_l	λ
p_g	$\beta_g + 2\gamma_g p_g - \lambda$	$2\gamma_g$		-1
q_l	$-\beta_l + 2\gamma_l q_l + \lambda$		$2\gamma_l$	1
λ	$Q + \sum_{l\in\mathbb{L}} q_l - \sum_{g\in\mathbb{G}} p_g$	-1	1	

TABLE 4.2: Relation among $\boldsymbol{z}$, $\mathcal{L}(\boldsymbol{z})$, $\nabla(\mathcal{L}(\boldsymbol{z}))$ and $H(\mathcal{L}(\boldsymbol{z}))$.

The structure for this graph, shown in figure 4.4, complies with the TSSSLE described in section 3.3. Therefore the solution to this system can be approached using the methodology described in section 3.6. In this case the tree has two layers. Another characteristic is that the coefficient related to the dual variable λ is zero. In general this will be the case for every dual variable as there is not second order information about the dual variables.

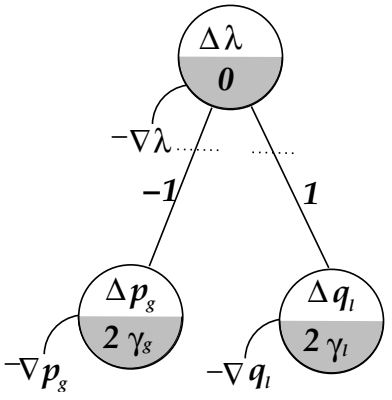


FIGURE 4.4: Graph corresponding to the LSE described by table 4.2

4.2.1 Algorithms

The graph structure described by figure 4.4 represents a *tree*. A system of linear equations whose graph representation is a tree can be solved straightforward; the only requirement is to define an algorithm which exploits its particular shape and content in an ordered way. Let us denote layer 0 as that where the root node is. Therefore, the depth of this graph is 2, and the breadth will be in function of the number of elements connected to this node (i.e. generators and loads).

The next two sections will propose algorithms to exploit this graph. These will only perform the operations needed in the evaluation of the graph (i.e. exploiting completely the sparsity of $H(\mathcal{L}(\boldsymbol{z}))$). The first section proposes an algorithm to travel up the graph applying a sparse forward elimination. The second section proposes an algorithm to solve the system based on the propagation the first algorithm performed. This is accomplished by traveling down the graph.

4.2.1.1 The Graph Forward EDIG Algorithm (gFEDIG)

Let us denote Γ_i as the set of nodes j such that there exists a link between i and j. Now, let us assume nodes i and j are connected by a_{ij}. Also, let us assume i is in layer n and j is in layer $n+1$ and $|\Gamma_j| = 1$. Hence, if Gaussian elimination is applied to node j then the only node which has to be affected by this process is node i which is one layer above node j. The graph transformation process when Gaussian elimination is applied to node j is outlined in figure 4.5.

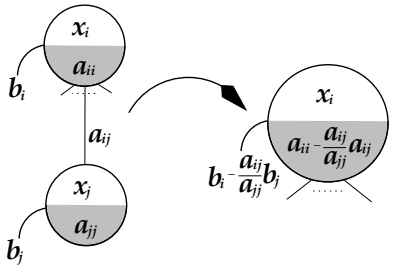


FIGURE 4.5: Graph reduction when applying Gaussian elimination

The gFED algorithm will apply this basic process to all the nodes in the tree in a bottom-up way until no more reductions can be done. This bottom-up approach has to be guided by the graph structure, and can be done in a modular way depending only on the constraint type. Two types of components can be distinguished: Generators and variable loads. As can be seen from the graph, Δ_λ depends on the characteristics of the other two components. Therefore, in order to reduce the λ-related values, it is needed to deduce the other object-related values (i.e. generators and variable loads before). Algorithm 1 will apply the principles above mentioned by processing first the generators related values and then the variable loads related values.

Algorithm 1 gFEDIG($\mathbb{G}, \mathbb{L}$)

$\nabla_\lambda \leftarrow Q$
for all $g \in \mathbb{G}_i$ **do**
  $a_{p_g} \leftarrow 2\gamma_g$
  $b_{p_g} \leftarrow \beta_g + 2\gamma_g p_g - \lambda$
  $\nabla_\lambda \leftarrow b_\lambda - p_g$
end for
for all $l \in \mathbb{L}_i$ **do**
  $a_{q_l} \leftarrow 2\gamma_l$
  $b_{p_g} \leftarrow -\beta_l + 2\gamma_l q_l + \lambda$
  $\nabla_\lambda \leftarrow b_\lambda + q_l$
end for
$a_\lambda \leftarrow -\sum_{g\in\mathbb{G}} (1/a_{p_g}) - \sum_{l\in\mathbb{L}} (1/a_{q_l})$
$b_\lambda \leftarrow -\nabla_\lambda - \sum_{g\in\mathbb{G}} (b_{p_g}/a_{p_g}) + \sum_{l\in\mathbb{L}} (b_{q_l}/a_{q_l})$

4.2.1.2 The Graph Backwards EDIG Algorithm (gBEDIG)

The $gBEDIG$ algorithm shown in listing 2 will perform the backward substitution. This will be guided by the graphic structure in a top-down approach. Basically it takes advantage from the propagation that in its way up the tree the $gFEDIG$ algorithm performed. It has updated the value of b_i and a_{ii} when their dependencies with the lower layer were eliminated. Let us suppose node i is in layer n and node k is in layer $n+1$. Then, the next equation has to be solved for x_i,

$$a_{ii}x_i + a_{ik}x_k = b_i \tag{4.5}$$

which is solved as

$$x_i = \frac{b_i - a_{ik}x_k}{a_{ii}} \tag{4.6}$$

clearly for this equation if node i is the tree root, then there will be no more layers up so this equation becomes

$$x_i = \frac{b_i}{a_{ii}} \tag{4.7}$$

Therefore this algorithm will apply this simple principle following the inverse process $gFEDIG$ took when it was applied.

Algorithm 2 gBEDIG($\mathbb{G}, \mathbb{L}$)

$\Delta_\lambda \leftarrow b_\lambda / a_\lambda$
for all $g \in \mathbb{G}$ **do**
  $\Delta_{p_g} \leftarrow (b_{p_g} - \Delta_\lambda)/a_{p_g}$
end for
for all $l \in \mathbb{L}$ **do**
  $\Delta_{q_l} \leftarrow (b_{q_l} - \Delta_\lambda)/a_{q_l}$
end for

4.2.1.3 The Graph EDIG Algorithm (gEDIG)

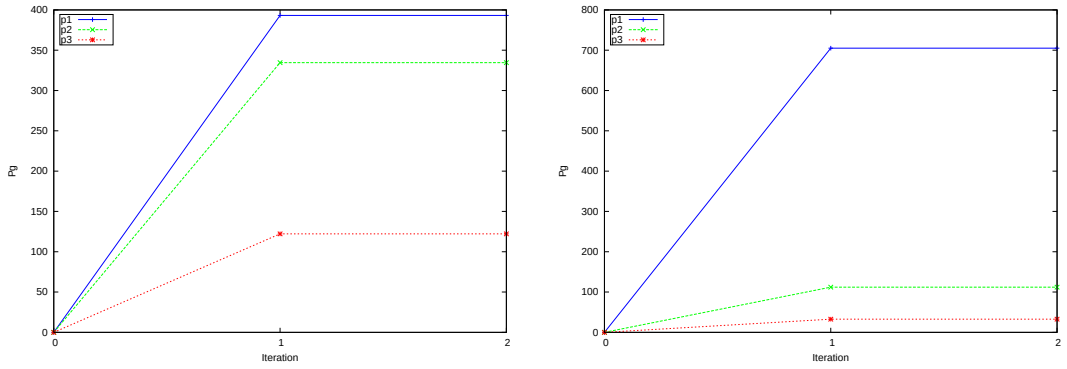
This section presents a simple iterative algorithm which will apply the algorithms *gFEDIG* and *gBEDIG* until the convergence criterion is reached.

Algorithm 3 gEDIG

1: Initialise $\boldsymbol{z}$
2: **repeat**
3: Evaluate $\nabla(\mathcal{L}(\boldsymbol{z}))_i$
4: Apply gFEDIG$(\mathbb{G}, \mathbb{L})$
5: Apply gBEDIG$(\mathbb{G}, \mathbb{L})$
6: $\boldsymbol{z} \leftarrow \boldsymbol{z} + \Delta\boldsymbol{z}$
7: **until** Convergence reached

4.2.2 Study Cases

Figure 4.2.2 shows the results for the system described in figure 4.2. As can be noticed, the tree has to be tranversed up and down once in order to reach the solution. No more iterations are required as Newton's method is a second order approximation and the EDIG model used is quadratic.



(a) Results for case 1 (Coal price=1)

(b) Results for case 1a (Coal price=0.9)

FIGURE 4.6: Results for the relaxed ED study under different coal prices.

Table 4.3 shows the results. Both results are correct [1] as they fullfill the equilibrium constraint. Nevertheless the results shown in case 1(a) for generator 1 and unit 3 (in red) are not within the limits where these units are supposed to work in. Furthermore, there could possibly be some allocations which would allow negative values. This imply to motorise those generation units, not a very desirable outcome. Therefore the mechanism used to solve this graph has to be modified in order to cope with these situations. This aspect will be addressed in the next section.

[1] Actually the original values there for case 1(a) are slightly different from these, as $p_1 + p_2 + p_3 = 849MW$ as compared with $p_1 + p_2 + p_3 = 850MW$ here as it should be.

g	p_g 1 λ=9.148	p_g 1(a) λ=8.285
1	393.170	705.147
2	334.604	112.159
3	122.226	32.695

TABLE 4.3: Results for system shown in figure 4.2

4.3 The Economical Dispatch Problem and its solution

In this section the economical dispatch mathematical model is developed in order to obtain a graph-based model. The development will be based on the model presented in figure 4.1. To this end, the previous model is augmented with the active elements capacity constraints. Equations from 4.8 to 4.11 describe the model for figure 4.1.

$$\max_{p_g, q_l} \quad \sum_{l \in \mathbb{L}} B_l(q_l) - \sum_{g \in \mathbb{G}} C_g(p_g) \tag{4.8}$$

s.t.

$$\sum_{g \in \mathbb{G}_i} p_g - \sum_{l \in \mathbb{L}_i} q_l - Q = 0 \tag{4.9}$$

$$\lfloor P_g \rfloor \leq p_g \leq \lceil P_g \rceil \quad \forall g \in \mathbb{G} \tag{4.10}$$

$$\lfloor Q_l \rfloor \leq q_l \leq \lceil Q_l \rceil \quad \forall l \in \mathbb{L} \tag{4.11}$$

$$\tag{4.12}$$

4.3.1 Normalizing the Mathematical ED Model

The method proposed needs to "dissect" the way the previous model is solved. To this end, the problem above has to be normalised as there are some inequality constraints. First the problem is converted into a minimisation problem by multiplying by -1 the objective function. Then, the inequality constraints are converted into equality constraints by introducing slack convex quadratic terms into the inequality constraints. The above QSP is transformed into the normalised QSP described by the model from 4.13 to 4.18.

$$\min_{p_g, q_l} \quad \sum_{g \in \mathbb{G}} C_g(p_g) - \sum_{l \in \mathbb{L}} B_l(q_l) \tag{4.13}$$

$$\textbf{s.}t. \quad \sum_{g \in \mathbb{G}} p_g - \sum_{l \in \mathbb{L}} q_l - Q = 0 \tag{4.14}$$

$$\lfloor p_g \rfloor - p_g + \underline{p_g}^2/2 = 0, \quad \forall g \in \mathbb{G} \tag{4.15}$$

$$p_g - \lceil p_g \rceil + \overline{p_g}^2/2 = 0, \quad \forall g \in \mathbb{G} \tag{4.16}$$

$$\lfloor q_l \rfloor - q_l + \underline{q_l}^2/2 = 0, \quad \forall l \in \mathbb{L} \tag{4.17}$$

$$q_l - \lceil q_l \rceil + \overline{q_l}^2/2 = 0, \quad \forall l \in \mathbb{L} \tag{4.18}$$

Let us denote constraint 4.14 as $\tilde{g}(p_g, q_l)$, 4.15 as $\tilde{h}_{\underline{p_g}}(p_g, \underline{p_g})$, 4.16 as $\tilde{h}_{\overline{p_g}}(p_g, \overline{p_g})$, 4.17 as $\tilde{h}_{\underline{q_l}}(q_l, \underline{q_l})$, and 4.18 as $\tilde{h}_{\overline{q_l}}(q_l, \overline{q_l})$. To obtain the Lagrangian $\mathcal{L}(\boldsymbol{z})$ based on this model a Lagrange Multiplier needs to be introduced for each constraint. There is just one equality constraint, $\tilde{g}(p_g, q_l)$ which has to be enforced at every time. The rest of them are inequality constraints which have to be taken into account only if they are binding. Let us allocate the following Lagrange Multipliers: λ to $\tilde{g}(p_g, q_l)$, $\underline{\rho}_g$ to $\tilde{h}_{\underline{p_g}}(p_g, \underline{p_g})$, $\overline{\rho}_g$ to $\tilde{h}_{\overline{p_g}}(p_g, \overline{p_g})$, $\underline{\mu}_l$ to $\tilde{h}_{\underline{q_l}}(q_l, \underline{q_l})$, and $\overline{\mu}_l$ to $\tilde{h}_{\overline{q_l}}(q_l, \overline{q_l})$. Therefore $\mathcal{L}(\boldsymbol{z})$ can be expressed in expression 4.19.

$$\begin{aligned} \mathcal{L}(\boldsymbol{z}) \quad = \quad & \sum_{g \in \mathbb{G}} \left[C_g(p_g) + \underline{\rho}_g \tilde{h}_{\underline{p_g}}(p_g, \underline{p_g}) + \overline{\rho}_g \tilde{h}_{\overline{p_g}}(p_g, \overline{p_g}) \right] \\ & - \sum_{l \in \mathbb{L}} \left[B_l(q_l) - \underline{\mu}_l \tilde{h}_{\underline{q_l}}(q_l, \underline{q_l}) - \overline{\mu}_l \tilde{h}_{\overline{q_l}}(q_l, \overline{q_l})) \right] \\ & + \lambda \tilde{g}(p_g, q_l) \end{aligned} \tag{4.19}$$

where $\boldsymbol{z} = [p_g, q_l, \underline{p_g}, \overline{p_g}, \underline{q_l}, \overline{q_l}, \lambda, \underline{\rho}_g, \overline{\rho}_g, \underline{\mu}_l, \overline{\mu}_l]^T$

4.3.2 Solution of the Mathematical ED Model

Expression 4.19 represents a high dimensional function whose optimal point can be found by using the Newton approach, as mentioned in section 2.4.4. To this end, the system represented by equation 4.20 has to be solved

$$H(\mathcal{L}(\boldsymbol{z}))\Delta\boldsymbol{z} = -\nabla(\mathcal{L}(\boldsymbol{z})) \tag{4.20}$$

Therefore, in order to solve $\Delta\boldsymbol{z}$, the values for $\nabla(\mathcal{L}(\boldsymbol{z}))$ and $H(\mathcal{L}(\boldsymbol{z}))$ are needed. Notice the solution has not been expressed as a function of $H(\mathcal{L}(\boldsymbol{z}))^{-1}$. Table 4.4 describes the relation among $\boldsymbol{z}$, $\mathcal{L}(\boldsymbol{z})$, $\nabla(\mathcal{L}(\boldsymbol{z}))$ and $H(\mathcal{L}(\boldsymbol{z}))$. Compared to table 4.2, this table has augmented the previous set of variables. Especifically, there are four more variables for each generator or variable load, which will be used to keep each generator within the limit constraints. This system of linear equations has a graph representation. In turn, this graph can be used to solve the system. Hence, some knowledge about how to transform this system into its graphical representation is needed, as well as how to exploit it.

$$\mathcal{L}(\boldsymbol{z})=\lambda\tilde{g}(p_g,q_l)+\sum_{g\in\mathbb{G}}(C_g(p_g)+\underline{\rho}_g\tilde{h}_{\underline{p_g}}(p_g,\underline{p_g})+\overline{\rho}_g\tilde{h}_{\overline{p_g}}(p_g,\overline{p_g}))$$
$$-\sum_{l\in\mathbb{L}}(B_l(q_l)-\underline{\mu}_l\tilde{h}_{\underline{q_l}}(q_l,\underline{q_l})-\overline{\mu}_l\tilde{h}_{\overline{q_l}}(q_l,\overline{q_l}))$$

	$\nabla(\mathcal{L}(\boldsymbol{z}))$	$H(\mathcal{L}(\boldsymbol{z}))$									
$\boldsymbol{z}$		p_g	q_l	$\underline{p_g}$	$\overline{p_g}$	$\underline{q_l}$	$\overline{q_l}$	λ	$\underline{\rho}_g$	$\overline{\rho}_g$	$\underline{\mu}_l$	$\overline{\mu}_l$
p_g	$\beta_g+2\gamma_g p_g-\lambda-\underline{\rho}_g+\overline{\rho}_g$	$2\gamma_g$						-1	-1	1		
q_l	$-\beta_l+2\gamma_l q_l+\lambda-\underline{\mu}_l+\overline{\mu}_l$		$2\gamma_l$					1			-1	1
$\underline{p_g}$	$\underline{\rho}_g\underline{p_g}$			$\underline{\rho}_g$					$\underline{p_g}$			
$\overline{p_g}$	$\overline{\rho}_g\overline{p_g}$				$\overline{\rho}_g$					$\overline{p_g}$		
$\underline{q_l}$	$\underline{\mu}_l\underline{q_l}$					$\underline{\mu}_l$					$\underline{q_l}$	
$\overline{q_l}$	$\overline{\mu}_l\overline{q_l}$						$\overline{\mu}_l$					$\overline{q_l}$
λ	$\sum_{g\in\mathbb{G}}p_g-\sum_{l\in\mathbb{L}}q_l-Q$	-1	1									
$\underline{\rho}_g$	$\lfloor p_g\rfloor-p_g+\underline{p_g}^2/2$	-1		$\underline{p_g}$								
$\overline{\rho}_g$	$p_g-\lceil p_g\rceil+\overline{p_g}^2/2$	1			$\overline{p_g}$							
$\underline{\mu}_l$	$\lfloor q_l\rfloor-q_l+\underline{q_l}^2/2$		-1			$\underline{q_l}$						
$\overline{\mu}_l$	$q_l-\lceil q_l\rceil+\overline{q_l}^2/2$		1				$\overline{q_l}$					

TABLE 4.4: Relation among $\boldsymbol{z}$, $\mathcal{L}(\boldsymbol{z})$, $\nabla(\mathcal{L}(\boldsymbol{z}))$ and $H(\mathcal{L}(\boldsymbol{z}))$.

4.3.3 Equivalent Graph Model

Table 4.4 shows the sparse structure for $H(\mathcal{L}(\boldsymbol{z}))$. This sparsity can be exploited in order to apply the minimal number of operations in the solution process. Several approaches have been proposed in the literature (Tinney et al. (1985); Gilbert et al. (1992)). However, they are very general and even though they reduce the number of operations this is not optimal as they do not take advantage of the explicit structure $H(\mathcal{L}(\boldsymbol{z}))$ has for this problem.

In this chapter, a graph-based approach is applied to the solution of equation 4.20. It is important to notice that equation 4.20 was not written as a function of $H(\mathcal{L}(\boldsymbol{z}))^{-1}$. The solution process will be based on the equivalent graph when the system represented in table 4.4 is translated to its graph representation applying the procedure given in section 3.7. Figure 4.7 is the graph representation obtained when this procedure is applied. Its structure fulfills the one defined for the TSSSLE described in section 3.3.

4.4 Inequality Constraints and Its Handling

From the problem description there is a basic fact which has to be taken into account. Not all the constraints need to be enforced along the solution process. Furthermore, there are mutually excluding constraints (i.e. at most one of them can be enforced). The standard mathematical tools used to assess this is by using KKT complementarity conditions. These will indicate if it is needed to enforce the constraints which are violated in the solution process. Additionally, they will signal which constraints have to be released along the solution process. Hence, they will define the graph traversal order.

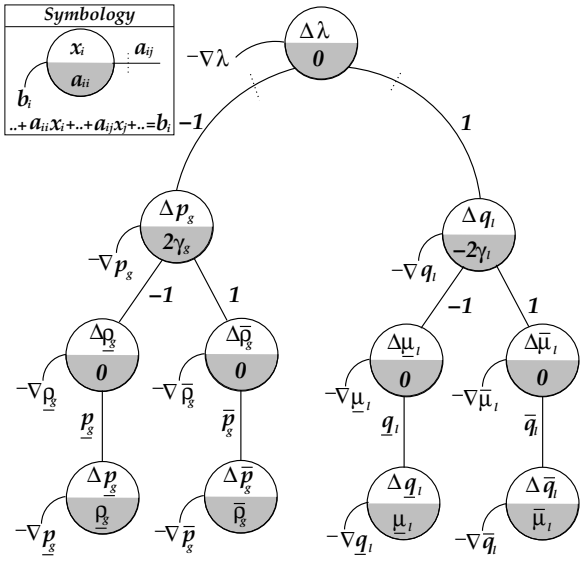


FIGURE 4.7: Graph corresponding to the economic dispatch model

Notice the difference between this approach and that from the matrix solution approach. There, KKT complementarity conditions are used in order to modify $H(\mathcal{L}(\boldsymbol{z}))$, by adding or eliminating the rows and columns corresponding to the Lagrange multiplier and slack variable related to the constraint being tested.

KKT conditions set the minimum requirements, which, when fulfilled, guarantee optimality conditions for convex non-linear programming problems. If $\boldsymbol{z}^*$ is assumed to be the optimal solution for a problem with $n = |\boldsymbol{z}|$ variables, m equality constraints and p inequality constraints, then KKT conditions are

$$\nabla f(\boldsymbol{z}^*) + \sum_{j=1}^{m} \lambda_j \nabla g_j(\boldsymbol{z}^*) + \sum_{k=1}^{p} \mu_k \nabla h_k(z^*) = 0 \tag{4.21}$$

$$g_j(\boldsymbol{z}^*) = 0, \qquad j = 1, 2, .., m \tag{4.22}$$

$$h_k(\boldsymbol{z}^*) \leq 0, \qquad k = 1, 2, .., p \tag{4.23}$$

$$\mu_k h_k(\boldsymbol{z}^*) = 0, \qquad k = 1, 2, .., p \tag{4.24}$$

$$\mu_k \geq 0, \qquad k = 1, 2, .., p \tag{4.25}$$

where equation 4.21 represents the equilibrium equation between the objective and the gradient of the constraint. Equations 4.22 and 4.23 represent the feasibility of the solution at the optimal point. Equation 4.24 represents the complementarity condition (i.e. either $\mu_k = 0$ or $h_k = 0$). Finally, equation 4.25 enforces the inequalities to act in just one direction.

KKT complementarity condition $\mu_k h_k(\boldsymbol{z}^*) = 0$ in equation 4.24 becomes $\mu_k \overline{x}_k = 0$ as the inequality has been converted into an equality introducing the slack term $\overline{x}^2/2$ to the constraint. KKT positiveness condition in equation 4.25 allows to assert that if $\mu > 0$

then the constraint is tight or binding since $\overline{x}$ must be zero. In the matrix approach this criterion is used to decide if the row and columnn belonging to the dual variable as well as the row and column related to the slack variable attached to such constraint have to be included in the computation (shown in figure 4.8 with the grey rows and columns). Obviously this implies modifying the matrix at every iteration by removing or adding rows and columns until convergence is reached. This is a complex task which impacts directly both the implementation and the computation time.

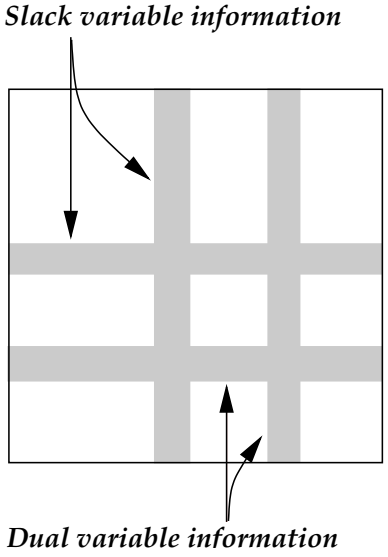


FIGURE 4.8: Conditioned row and columns in the matrix approach.

In comparison, in the graph approach, this criterion is used to decide if the part of the graph belonging to such constraint has to be evaluated. Therefore, based on this principle, the graph-based representation for the system described in table 4.4 is shown in figure 4.9. In this figure, the grey lines denote conditional evaluation which will be performed whenever its associated constraint is active.

4.4.1 Algorithms

The graph structure described by figure 4.9 represents a *tree*. As described in section 3.3, a system of linear equations whose graph representation is a tree can be solved straightforwardly by exploiting its particular shape and content in an ordered way. Let us denote layer zero as that where the root node is. Therefore, the depth of this graph is three, and the breadth will be a function of the number of elements connected to this node (i.e. generators and loads). KKT complementarity conditions, will guide the graph traversal.

The next two sections propose algorithms to exploit this graph. These algorithms will only perform the operations needed in the evaluation of the graph (i.e. exploiting completely the sparsity of $H(\mathcal{L}(\boldsymbol{z}))$). Section 4.4.1.1, proposes an algorithm to travel up the

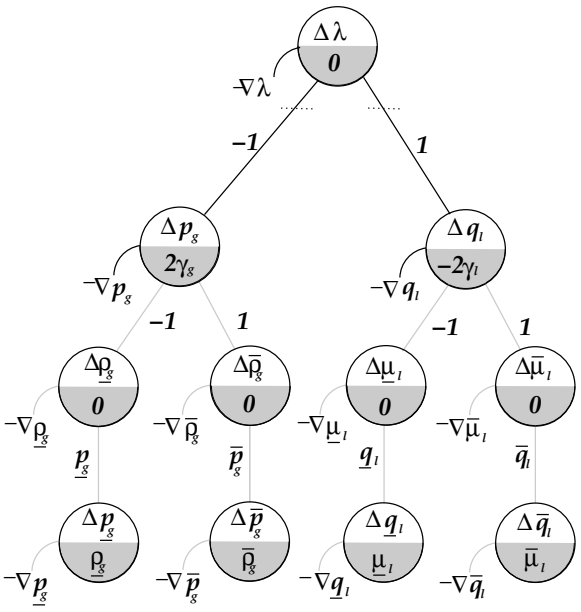


FIGURE 4.9: Conditional sub-graphs in the graph approach.

graph applying a sparse forward elimination. Then, section 4.4.1.2 proposes an algorithm to solve the system based on the propagation the previous algorithm performed. This is accomplished by traveling down the graph.

4.4.1.1 The Graph Forward ED Algorithm (gFED)

Let us denote Γ_i as the set of nodes j such that there exists a link between nodes i and j. Now, let us assume nodes i and j are connected by a_{ij}. Also, let us assume i is in layer n and j is in layer $n+1$, and $|\Gamma_j| = 1$. Hence, if Gaussian elimination is applied to node j then the only node which has to be affected by this process is node i which is one layer above node j. The graph transformation process when Gaussian elimination is applied to node j is outlined in figure 4.10.

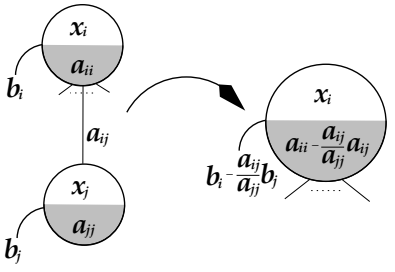


FIGURE 4.10: Graph reduction when applying Gaussian elimination

The gFED algorithm will apply this reduction to all the nodes in the tree in a bottom-up way until no more reductions can be done. This bottom-up approach has to be guided by the graph structure, and can be done in a modular way depending only on the constraint type. Two types of components can be distinguished: Generators and

variable loads. As can be seen from the graph, Δ_λ depends on the other two components' characteristics. Therefore, in order to reduce the λ-related values, first the other object-related values (i.e. generators and variable loads) have to be deduced. Algorithm 4 applies the principles above mentioned by processing first the generators related values and then the variable loads related values. First, lines from 1 to 15 apply the reduction process for all the variables used in the model of each generator. Next the same process is applied to all the loads in lines from 16 to 30. Finally, lines 31 and 32 do the reduction process related to the energy price given by λ.

Algorithm 4 gFED($\mathbb{G}, \mathbb{L}$)

1: **for all** $g \in \mathbb{G}$ **do**
2: $a_{p_g} \leftarrow 2\gamma_g$
3: $b_{p_g} \leftarrow -\nabla_{p_g}$
4: **if** p_g is lower binding **then**
5: $a_{\underline{\rho}_g} \leftarrow -\underline{p_g}^2/\underline{\rho}_g$
6: $b_{\underline{\rho}_g} \leftarrow -\nabla_{\underline{\rho}_g} + \nabla_{\underline{p_g}\,\underline{p_g}}/\underline{\rho}_g$
7: $a_{p_g} \leftarrow a_{p_g} - 1/a_{\underline{\rho}_g}$
8: $b_{p_g} \leftarrow b_{p_g} + b_{\underline{\rho}_g}/a_{\underline{\rho}_g}$
9: **else if** p_g is upper binding **then**
10: $a_{\overline{\rho}_g} \leftarrow -\overline{p_g}^2/\overline{\rho}_g$
11: $b_{\overline{\rho}_g} \leftarrow -\nabla_{\overline{\rho}_g} + \nabla_{\overline{p_g}\,\overline{p_g}}/\overline{\rho}_g$
12: $a_{p_g} \leftarrow a_{p_g} - 1/a_{\overline{\rho}_g}$
13: $b_{p_g} \leftarrow b_{p_g} + b_{\overline{\rho}_g}/a_{\overline{\rho}_g}$
14: **end if**
15: **end for**
16: **for all** $l \in \mathbb{L}$ **do**
17: $a_{q_l} \leftarrow 2\gamma_l$
18: $b_{q_l} \leftarrow -\nabla_{q_l}$
19: **if** q_l is lower binding **then**
20: $a_{\underline{\mu}_l} \leftarrow -\underline{q_l}^2/\underline{\mu}_l$
21: $b_{\underline{\mu}_l} \leftarrow -\nabla_{\underline{\mu}_l} + \nabla_{\underline{q_l}\,\underline{q_l}}/\underline{\mu}_l$
22: $a_{q_l} \leftarrow a_{q_l} - 1/a_{\underline{\mu}_l}$
23: $b_{q_l} \leftarrow b_{q_l} + b_{\underline{\mu}_l}/a_{\underline{\mu}_l}$
24: **else if** q_l is upper binding **then**
25: $a_{\overline{\mu}_l} \leftarrow -\overline{q_l}^2/\overline{\mu}_l$
26: $b_{\overline{\mu}_l} \leftarrow -\nabla_{\overline{\mu}_l} + \nabla_{\overline{q_l}\,\overline{q_l}}/\overline{\mu}_l$
27: $a_{q_l} \leftarrow a_{q_l} - 1/a_{\overline{\mu}_l}$
28: $b_{q_l} \leftarrow b_{q_l} + b_{\overline{\mu}_l}/a_{\overline{\mu}_l}$
29: **end if**
30: **end for**
31: $a_\lambda \leftarrow -\sum_{g\in\mathbb{G}}(1/a_{p_g}) - \sum_{l\in\mathbb{L}}(1/a_{q_l})$
32: $b_\lambda \leftarrow -\nabla_\lambda - \sum_{g\in\mathbb{G}}(b_{p_g}/a_{p_g}) + \sum_{l\in\mathbb{L}}(b_{q_l}/a_{q_l})$

4.4.1.2 The Graph Backwards ED Algorithm (gBED)

The gBED algorithm shown in listing 5 performs the backward substitution. This algorithm is guided by the graphic structure in a top-down approach. Basically, it takes advantage from the propagation that in its way up the tree, the gFED algorithm performed. It has updated the value of b_i and a_{ii} when their dependencies with the lower layer were eliminated. Let us suppose node i is in layer n and node k is in layer $n+1$. Then, the next equation has to be solved for x_i,

$$a_{ii}x_i + a_{ik}x_k = b_i \tag{4.26}$$

which is solved as

$$x_i = \frac{b_i - a_{ik}x_k}{a_{ii}} \tag{4.27}$$

clearly, for this equation, if node i is the tree root, then there will be no more layers up so this equation becomes

$$x_i = \frac{b_i}{a_{ii}} \tag{4.28}$$

Therefore this algorithm applies this simple principle following the inverse process algorithm $gFED$ took when it was applied. First, line 1 applies the solution process related to the energy price given by λ using equation 4.28. Then lines from 2 to 11 apply the solution process for all the variables used in the model of each generator, respecting the position they have within the tree using equation 4.27. Finally the same process is applied to all the loads in lines from 12 to 21.

4.4.1.3 The Graph ED Algorithm (gED)

This section presents an iterative algorithm which applies algorithms $gFED$ and $gBED$ until a convergence criterion is reached. First in line 1 vector $\boldsymbol{z}$ is initialized, then in line 3 the gradient for $\mathcal{L}(\boldsymbol{z})$ with respect to each variable is evaluated. After this in line 4 algorithm $gFED$ is invoked. This will traverse the tree up. Then the inverse process is applied in line 5 with algorithm $gBED$ as this traverses down the tree. In line 6 $\boldsymbol{z}$ is updated. Finally, in line 7 the convergence will be tested. If this is not reached, then the sequence before described will be repeated until the convergence criterion is fulfilled.

4.4.2 Study Cases

In this section two study cases are analysed. The first one is a three generator case based on data extracted from Wollenberg and Wood (1996). The second case is a ten generators case based on data extracted from Han et al. (2001). Both cases are analysed under different conditions.

Algorithm 5 gBED($\mathbb{G}, \mathbb{L}$)

$\Delta_\lambda \leftarrow b_\lambda / a_\lambda$
for all $g \in \mathbb{G}$ **do**
  $\Delta_{p_g} \leftarrow (b_{p_g} - \Delta_\lambda)/a_{p_g}$
  if p_g is lower binding **then**
    $\Delta_{\underline{\rho}_g} \leftarrow (b_{\underline{\rho}_g} - \Delta_{p_g})/a_{\underline{\rho}_g}$
    $\Delta_{\underline{p_g}} \leftarrow (\nabla_{\underline{p_g}} + \underline{p_g}\Delta_{\underline{\rho}_g})/\underline{\rho}_g$
  else if p_g is upper binding **then**
    $\Delta_{\overline{\rho}_g} \leftarrow (b_{\overline{\rho}_g} + \Delta_{p_g})/a_{\overline{\rho}_g}$
    $\Delta_{\overline{p_g}} \leftarrow (\nabla_{\overline{p_g}} + \overline{p_g}\Delta_{\overline{\rho}_g})/\overline{\rho}_g$
  end if
end for
for all $l \in \mathbb{L}$ **do**
  $\Delta_{q_l} \leftarrow (b_{q_l} - \Delta_\lambda)/a_{q_l}$
  if q_l is lower binding **then**
    $\Delta_{\underline{\mu}_l} \leftarrow (b_{\underline{\mu}_l} - \Delta_{q_l})/a_{\underline{\mu}_l}$
    $\Delta_{\underline{q_l}} \leftarrow (\nabla_{\underline{q_l}} + \underline{q_l}\Delta_{\underline{\mu}_l})/\underline{\mu}_l$
  else if q_l is upper binding **then**
    $\Delta_{\overline{\mu}_l} \leftarrow (b_{\overline{\mu}_l} + \Delta_{q_l})/a_{\overline{\mu}_l}$
    $\Delta_{\overline{q_l}} \leftarrow (\nabla_{\overline{q_l}} + \overline{q_l}\Delta_{\overline{\mu}_l})/\overline{\mu}_l$
  end if
end for

Algorithm 6 gED($\mathbb{G}, \mathbb{L}$)

Initialise $\boldsymbol{z}$
repeat
    Evaluate $\nabla(\mathcal{L}(\boldsymbol{z}))_i$
    Apply gFED($\mathbb{G}, \mathbb{L}$)
    Apply gBED($\mathbb{G}, \mathbb{L}$)
    $\boldsymbol{z} \leftarrow \boldsymbol{z} + \Delta\boldsymbol{z}$
until Convergence reached

4.4.2.1 Three-Generator Case

The system shown in figure 4.11 shows the structure for this system.

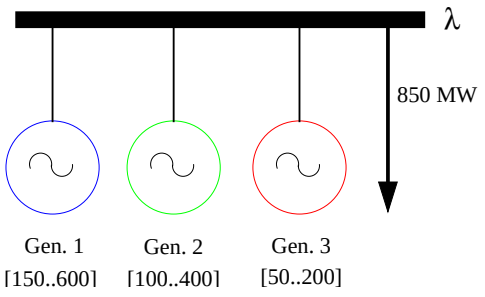


FIGURE 4.11: Three generators system

These cases are built based on the same model with some slight changes on their characteristics. In the first case, three generators whose production functions are shown in figure 4.12 are considered.

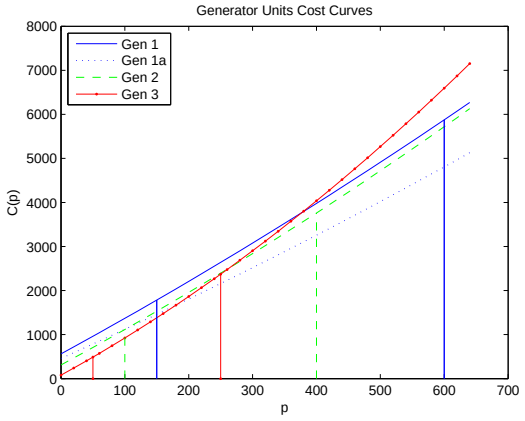
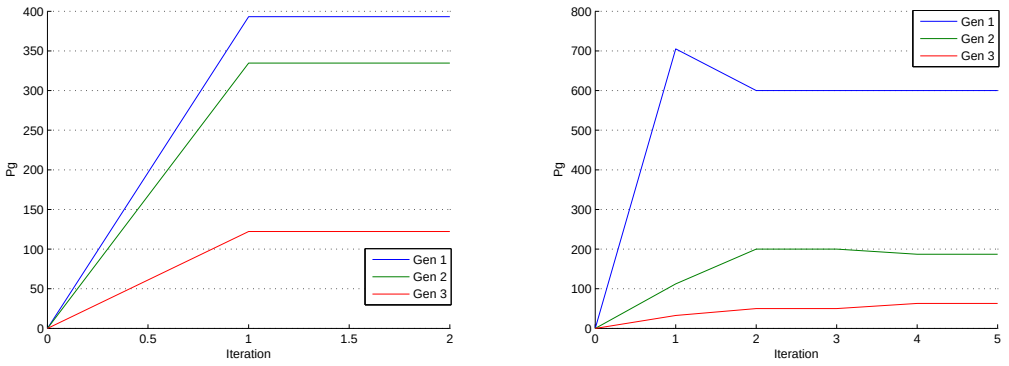


FIGURE 4.12: Production functions for the generators.

Table 4.5, shows the data used for the different cases. Gen 1a data are the data where the coal is more expensive than the case in generator 1b.

generator	α	β	γ	$\lfloor p \rfloor$	$\lceil p \rceil$
1	510	7.92	0.001562	150	600
1(a)	459	6.48	0.00128	150	600
2	310	7.85	0.00194	100	400
3	78	7.97	0.00482	50	250

TABLE 4.5: System components' data



(a) Case a

(b) Case b

FIGURE 4.13: Three generators case results

Figure 4.13 shows the convergence behavior for λ. In the first case as shown by figure 4.13(a) it reaches its value in just one iteration, as expected from the second order approximation. Also a reason behind this, is that the solution is within the constrained solution space, as none of the generators are overloaded. As a consequence, this is the

optimal operation point. Furthermore, the marginal cost for each generator will be the same. Therefore, $\frac{\partial C_1(393.17)}{\partial p_1} = \frac{\partial C_2(334.6)}{\partial p_2} = \frac{\partial C_3(122.23)}{\partial p_3} = \lambda$ must hold.

However in the second case things change as shown in figure 4.13(b) where the convergence curve changes its shape and the algorithm takes more iterations to solve the system. Table 4.6 shows the energy allocation each generator will be producing in each configuration. Notice the value for λ has gone down compared to case 1.

generator	p	
	(a)	(b)
	$\lambda = 9.1482$	$\lambda = 8.576$
1	393.17	600
2	334.6	187.13
3	122.23	62.87

TABLE 4.6: Results for three generators system case, gen 1(a) non binding $\lambda = 9.1482$ (b) binding $\lambda = 8.576$

The reason behind this changes is that the energy produced by generator 1 now is cheaper. Therefore, it will be able to sell all the energy it can produce up to its upper limit. The method will be aware of this fact and will introduce the upper binding constraint into the system. This is done by setting the power produced by gen 1 to its maximum (i.e. 600 MW). The power produced by generator 3 is below its limits in the first iteration (not shown). As a result, the binding constraint for the lower limit for generator 3 is introduced. KKT conditions will be deciding when the constraint have to be enforced. Another iteration is needed, but with new equations added to the system in order to meet the constraints. Given that gen 1 is binding the maximum price he would ask for at its binding generation level is $\lambda_1 = \frac{\partial C_1(600)}{\partial p_1} = 8.016$. λ will be set by gen 2 and 3, and has to be greater or equal than the one for generator 1 when is binding. Therefore, the relation $\lambda_2 = \lambda_3 \geq \lambda_1$ must hold. From table 4.1, $\frac{\partial C_2(187.13)}{\partial p_2} = \frac{\partial C_3(62.87)}{\partial p_3} = 8.576$. This result fulfills the previous requirement. Figure 4.14 shows the convergence behaviour for λ in both cases.

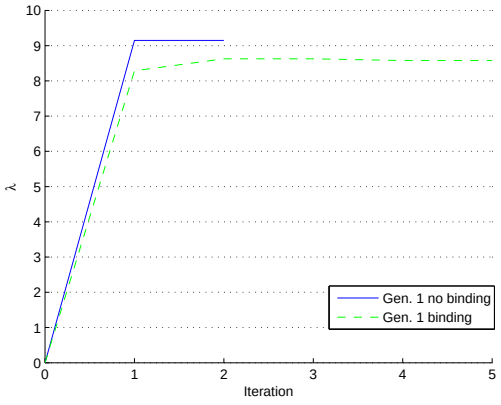


FIGURE 4.14: Three generators system's λ convergence

4.4.2.2 Ten-Generator Case

Here a system involving ten generators is analysed. The data corresponding to this case is shown in table 4.7. This analysis is done in order to test the model under different load conditions. First the model is tested with a 1223MW load. Then it is tested with a 478MW load in order to test the convergence to the lower limit in all the generators. Finally, a 1563 MW load is defined so the upper limits constraints in all the generators are binding.

gen	α	β	γ	$\lfloor p \rfloor$	$\lceil p \rceil$
1	15	2.2034	0.0051	12	73
2	25	1.9101	0.0040	26	93
3	40	1.8518	0.0039	42	143
4	32	1.6966	0.0038	18	70
5	29	1.8015	0.0021	30	93
6	72	1.5334	0.0026	100	350
7	49	1.2643	0.0029	100	248
8	82	1.2130	0.0015	40	190
9	105	1.1954	0.0013	70	190
10	100	1.1285	0.0014	40	113

TABLE 4.7: Ten generators case data

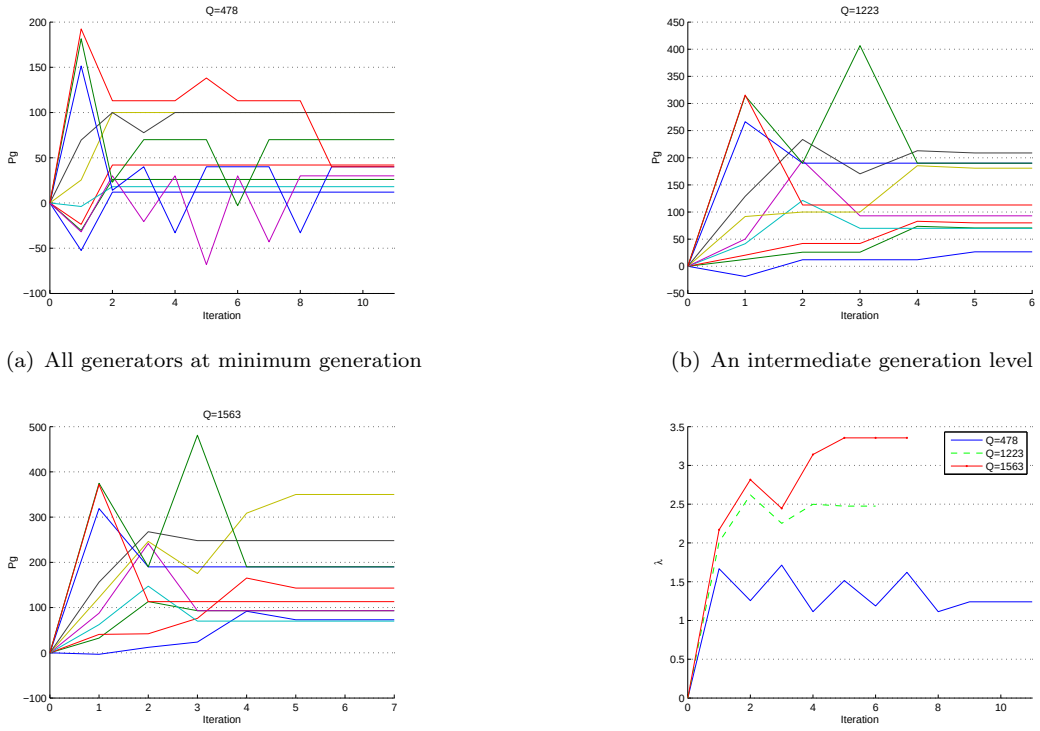
The results for this case are shown in table 4.8. The second column shows all the units are lower binding and the fourth column all of them are upper binding. This means, the solution to this problem is located in the lower and upper hypercorner of the polytope represented by the set of limit constraints.

Case	Q=478MW $\lambda = 1.2405$	Q=1223MW $\lambda = 2.4756$	Q=1563MW $\lambda = 3.3554$
p_1	12	26.6861	73
p_2	26	70.6874	93
p_3	42	79.9742	143
p_4	18	70.0000	70
p_5	30	93.0000	93
p_6	100	180.8075	350
p_7	100	208.8447	248
p_8	40	190.0000	190
p_9	70	190.0000	190
p_{10}	40	113.0000	113

TABLE 4.8: Ten generators case results

Figure 4.15 shows the convergence behaviour for the different cases defined by their load profile. Several remarks can be made from it. Firstly, it would be expected the behaviour in the system minimal load and system maximal load would lead to a similar convergence pattern. This is not the case as shown by figure 4.15(a) and 4.15(c). It took ten iterations to reach convergence for the minimal generation boundary which;

compared to six iterations it took to reach convergence for the maximal generation limit; is significant. On the other hand the results obtained for the case where not all the generators are binding, shown in figure 4.15(b), converge in just one more iteration compared to the three generators case. Finally, figure 4.15(d) shows the convergence behavior for λ, which in general follow the same behavior as the primal variables in each case.



(a) All generators at minimum generation

(b) An intermediate generation level

(c) All generators at maximum generation

(d) λ price convergence

FIGURE 4.15: Ten generation system results under different load conditions

4.5 Concluding Remarks

By taking a graph-based approach, it has been possible to propose a new set of algorithms for the economical dispatch based on the basic graph model. Based on table 4.4, a new graph-based solution has been proposed. This approach allows us to scale up the system in a very efficient way (i.e. linear in the number of generators and loads attached to the electrical power system). To this end; an economical dispatch algorithm based on graphs has been presented which uses two basic algorithms the Forward ED ($gFED$) and the Back Forward ED ($gBED$) . These algorithms perform the minimal number of operations as a result of the topological structure of the model. Comparing this approach with the $H(\mathcal{L}(\boldsymbol{z}))^{-1}$ based solution, where it would be necessary to invert $H(\mathcal{L}(\boldsymbol{z}))$ and then multiply it by $\nabla(\mathcal{L}(\boldsymbol{z}))$, is much more efficient. Basically, this algorithm performs the Gaussian elimination in the $gFED$ and $gBED$ implements the backward

substitution. This chapter presents two main contributions. First, the explicit graph model derived from the mathematical model allows us to think about the model in a modular fashion. This in turn prevents us from using general sparse matrix methods such as those in Tinney et al. (1985). Second, the order in which this reduction has to be accomplished. This order is a consequence of the matrix graph analysis and is inspired in its topological structure.

Chapter 5

A Bottom Up Distributed Optimal Power Flow Model

5.1 Introduction

Figure 5.1 represents the structure of a typical electrical power system. Their main components are nodes representing electrical buses and lines representing transmission lines. The solid lines represent internal transmission lines. The dotted lines represent interconnections between different electrical power systems belonging to different countries or different zones within a country for power exchange purposes among them. These are called *interconnectors*. These are modeled as generators or loads depending if they are importing or exporting power respectively. Attached to each node there can be a number of generators willing to sell energy as well as a number of consumers willing to buy that energy as described in section 2.2.5.

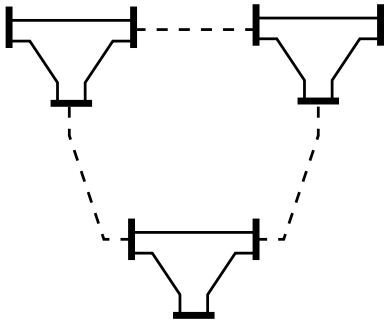


FIGURE 5.1: Generic electrical power market representation

The Optimal Power Flow (OPF) is a centralised market clearing mechanism to conciliate both suppliers, trying to sell energy as expensive as possible, and buyers, trying to buy energy as cheap as possible. The clearing prices must give some signals to the market participants in order to keep the market efficient. The OPF considers limits on the

devices attached to the electrical power system (i.e. upper and lower capacity of the generation units) as well as constraints posed by the environment such as transmission constraints. If there are no transmission constraints, then the OPF will result in the MCP, where accumulated market demand and supply curves intersect. This price will be the marginal cost of the most expensive unit committed. This price will be uniform for all the nodes in the system.

On the other hand, if transmission constraints appear then the clearing mechanism sets a different price for each node in the electrical power system. The energy price will be higher in those places where congestion exists. This is a signal which can be interpreted as an incentive for the suppliers to build more plants in those areas and also gives a signal for the gridco to build more lines in order to alleviate congestion in the congested zones.

The centralised OPF is a Non Linear Mathematical Program whose generalised description for $\boldsymbol{z} \in \Re^n$, is given by equations 5.1 to 5.3.

$$\max_{\boldsymbol{z}} \; f(\boldsymbol{z}) \tag{5.1}$$

$$\textbf{st.} \quad g_j(\boldsymbol{z}) = 0, \qquad j = 1, 2, ..., m \tag{5.2}$$

$$h_k(\boldsymbol{z}) \leq 0, \qquad k = 1, 2, ..., p \tag{5.3}$$

The task is to maximize $f(\boldsymbol{z})$, representing the social welfare defined by the total benefit for the consumers minus the total cost to produce the energy they consume. Alternatively, this problem can also be formulated as the minimisation of the total cost to produce the energy minus the total benefit for the consumers. The equality constraints, $g_j(\boldsymbol{z}) = 0$, represent the energy equilibrium constraints at each node. This is done by incorporating the DC Power Flow into this model. The inequality constraints, $h_k(\boldsymbol{z}) \leq 0$, are those related to the maximum and minimum generation limits for each generation unit and the flow limit each line is subject to. This model will be described in detail in section 5.3, . The objective function is convex as this is formed with the cost functions and the benefit functions which are quadratic. The constraints are linear. Therefore this guarantees the uniqueness of the solution, if it exists.

In the last decade there have been several proposals to decentralise this problem. The motivations behind all these proposals are:

- *Deregulation*. This process, introduces new agents into the market. The information about the system is dispersed across the market and therefore nobody has the whole picture. Furthermore, this information now is an strategic element for every agent. It is not likely they will be willing to disclose it. Therefore, nobody

has a complete view of the system. Methods to solve this problem with partial information are needed.

- *Market complexity*. Here, as the market grows, the resources to solve the problem increases in a nonlinear manner. The market complexity is not due to the complexity of its components; it is complex because of the interconnections of those components. Large scale approaches have to be taken in order to derive simpler models interacting in order to solve the otherwise complex problem.
- *Market integration*. As the time evolves, more and more markets are merging. Tools are needed to face the integration problem so that the final market can be solved as a communication problem, without having to rely on an centralised authority.

Perhaps the biggest challenge in a decentralised power market is how all the agents involved in it will reach agreements sharing just the minimal information to reach the equilibrium. Even more, the first question would be what is understood by *minimal information* and its *nature*? Is this information of electrical nature? If so, how then will they be aware of the global economical behavior? Is this information of economical nature? If so, how will they be sure the system is working within its physical limits? Perhaps the information shared among the agents should be of both types electrical and economical.

Until now this problem has been overcome by taking a centralised approach. An Independent System Operator (ISO) is introduced in order to solve the global problem preserving the system within its physical constraints. This global problem is formulated by the ISO who will gather all the information to clear the market. It assumes all the agents in the system will be willing to share their internal information with him. Given that this is an institutional agent, there is no doubt all the agents will be "willing" to share it. If they do not provide it, then they would be left out of the game.

By taking such an assumption, this centralised model is limiting the main deregulation goal: to transform the electricity industry into a competitive market. There, all the agents (i.e. Gencos and Custcos), should be able to reach agreements based on their bids. However, this is not the case as they are constrained to just a static set of coefficients representing both the quadratic cost from the suppliers and benefit function from the customers. These coefficients have to be submitted to the ISO in order to clear the market. Hence they are revealing the information which in a competitive scenario could be used to reach more profitable trades.

In a competitive market, all the agents will try to reach the equilibrium by varying those coefficients, until no more gain can be obtained without disadvantaging the other agents (i.e. Pareto efficient). Therefore, an ideal competitive deregulated power market would be one where the interaction among the agents (i.e. Custcos and GenCos), would

determine the Electricity price, without the need for the ISO. But this is not free. As the ISO has been removed from the market, then each agent would have to take into account the system security. Their local behavior towards security must guarantee that the overall global physically constrained system behavior will be within its limits.

Several distributed algorithms have been proposed to overcome this bottleneck. These approaches decompose the electrical power system into subareas which are linked by lines called *interconnectors*. Each subarea solves a NLP which needs some external information in order to fulfill its task. Figure 5.2 illustrates this approach. The circles represent the NLP domains which have to be solved. Therefore, the variables inside these circles are local variables and the variables outside this circle are external variables. If a variable is external, then it has to be treated as a constant by the local NLP. The dotted links represent interconnections between the different areas. From a functional point of view they represent coordination and intercommunication tasks. The timing, kind, and amount of information varies depending on the approach used to solve the decentralised system.

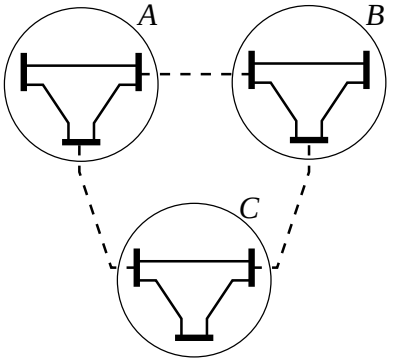


FIGURE 5.2: Multiarea based system approach

Those methodologies share a common characteristic: based on the centralized model, they split the model focusing in the sub area concept. Then, they try to find how these subarea problems can be defined so that their local behaviors and the intercommunication with their neighbouring areas guarantee the global behavior. Nevertheless, the private information problem remains unsolved.

A different approach has to be taken in order to solve this problem. After all, the global power system behavior is the result of the behavior of its individual components and the environment where they are situated. In this case, the generation units and loads are the individual components. This environment would be represented by the node models as well as the interconnections among them.

In general, a large scale system is complex not because of the number of basic components, which in general have simple models. They are complex because of the interconnection or dependencies among these basic components. A basic consideration when designing solutions in decentralised systems would be to replace these dependencies by

intercommunication and coordination strategies. However, the intercommunication task as a whole has to be moderate so it does not turn into an intercommunication bottleneck.

The methodology proposed in this work is based on a bottom-up approach. A basic decentralised model will be derived using decomposition techniques. Figure 5.3 illustrates this approach. The circles represent the problem to solve for each node. The lines represent information interchange between individual nodes. Here every node represents a subproblem and there is some intercommunication with its neighbouring nodes. Each local problem is solved taking into account the information gathered from the other nodes. Each node also provides its own information to each of their neighbours.

A central point here is that the information they interchange must not reveal any strategic information. In particular, they interchange market information (i.e. prices), as well as information describing temporary states (i.e. voltage angles). The main endeavor is to derive a model based on local information as well as market intercommunication which leads to a market equilibrium. By equilibrium, we mean to reach a configuration where nobody can get better off without making someone else worse off (i.e. Pareto Efficient).

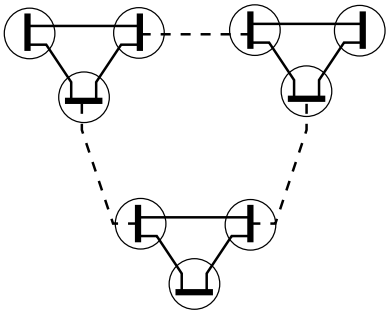


FIGURE 5.3: Multinode based system approach

This chapter is organised as follows. In section 5.2, a review about the work on decentralisation applied to the optimal power flow is provided. In section 5.3 the centralised model is defined based on a basic node model. Then, in section 5.4 a distributed model is derived. Having done that, section 5.5 proposes two algorithms to exploit the model. Section 5.6 presents several study cases. Finally, section 5.7 concludes highlighting the main contributions of this chapter.

5.2 Related Work

The Optimal Power Flow is at the heart of every power market clearing tool. There is a vast literature which goes from the early 1960s (Huneault and Galiana, 1991). The main aim of this research is to provide a decentralised framework based on simple models and their interaction. This section presents a literature review of decentralisation applied

to the optimal power flow. As a result from this review, a general knowledge of the methodologies applied in this area has been learnt.

Kim and Baldwick (1997), present a coarse-grained distributed methodology to solve the optimal power flow. In order to apply this methodology, they need to create dummy nodes so that coordination can be achieved. The solution is based on the Auxiliary Problem Principle (Cohen, 1978). In order to achieve this, they add some constraints to the model based on those dummy nodes introduced artificially. This model is implemented on the GAMS environment (Rosenthal, 2007).

Conejo and Aguado (1998), propose a multiarea DC optimal power flow. This model is solved using the decomposition and coordination principle using a Lagrangian relaxation procedure. They also introduce the losses effect within the mathematical model. They achieve the coordination phase by introducing one or two fictitious buses. This aspect increments the system size with its consequences as constraints related to these nodes are introduced. The solution is accomplished using a dual programming approach . This involves two steps. First, it maximizes the dual variables by fixing the original problem variables. Second, it minimizes the primal problem taking the dual variables as given (i.e. constants).

Aguado, Quintana, and Conejo (1999), extend the previous work to deal with reactive power as well as using a cutting-plane solution approach. Here the interconnecting lines called tie lines, are modeled as generators or loads depending on whether they are importing or exporting energy. The Lagrange multipliers associated with the constraints at the fictitious buses are the spot prices for power exchange.

The work presented by Aguado and Quintana (2001), presents a market-oriented approach to coordinate inter-utilities power exchange. In their formulation, they introduce some constraints in the border nodes to force the variables involved in it to be the same. The dual variables they calculate are used as the import or export spot prices.

Bakirtzis and Biskas (2003), implement the DC Optimal Power Flow solution by decomposing the entire system into subareas. The solution process is accomplished using a coordination mechanism based on the Lagrange multipliers which can be interpreted as the spot prices for the imported/exported energy between the zones this line is connecting. In order to achieve this, they introduce two more constraints into each area. This work differs from the other as it does not introduce any virtual node into the model.

Finally, based on the previous work, Bakirtzis, Biskas, Macheras, and Pasialis (2000) implement the previous system on a network of computers and also extends it by eliminating the need of a reference node. In order to achieve this, a new set of constraints are introduced into the model. They also rely on the concept of ϵ-effective branch power flow constraints to reduce the optimisation search space. Finally Bakirtzis, Biskas, Macheras, and Pasialis (2000) extend this work to solve the AC-OPF.

5.3 Centralised Optimal Power Flow Model

The basic node model shown in figure 5.4 can be modeled by the centralised NLP given by equations from 5.4 to 5.8.

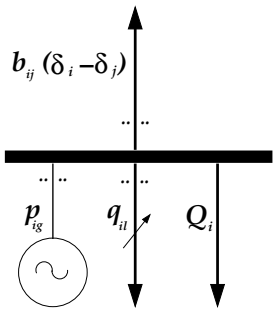


FIGURE 5.4: Basic node system

The mathematical program objective, as described by equation 5.4 is to maximise the social welfare in the system. This is given by the difference between the Social Benefit $B(q)$ and the cost to produce the energy $C(p)$ in the entire electrical power system.

$$\min_{p_{ig},q_{il},\delta_i} \quad \sum_{g\in\mathbb{G}} C_g(p_g) - \sum_{l\in\mathbb{L}} B_l(q_l) \tag{5.4}$$

s.t.

$$\sum_{j\in\Gamma_i} b_{ij}(\delta_i - \delta_j) - \sum_{g\in\mathbb{G}_i} p_{ig} + \sum_{l\in\mathbb{L}_i} q_{il} + Q_i = 0 \tag{5.5}$$

$$\lfloor P_{ig} \rfloor \leq p_{ig} \leq \lceil P_{ig} \rceil \quad \forall g \in \mathbb{G}_i \tag{5.6}$$

$$\lfloor Q_{il} \rfloor \leq q_{il} \leq \lceil Q_{il} \rceil \quad \forall l \in \mathbb{L}_i \tag{5.7}$$

$$|b_{ij}(\delta_i - \delta_j)| \leq \lceil F_{ij} \rceil \quad \forall j \in \mathbb{T}_i \tag{5.8}$$

There are some constraints since some solutions to this problem are infeasible due to physical considerations. Constraint 5.5 is used to ensure energy balance at every node. In practice all the physical devices have lower and upper limits which have to be satisfied. In order to reinforce those limits, constraint 5.6 is defined to bound the solution to a zone where the generator operation is safe, the same criterion is defined for the loads in constraint 5.7. Finally, constraint 5.8 is used to ensure the power flowing along the lines will not exceed their capacity. This last constraint is referred to the absolute value, as there is no certainty about the flow direction.

5.3.1 Full Centralised Model for the 2-node system

In this section a basic two node model is presented. Then the centralised mathematical model for this system is analysed. Finally, by analyzing this basic system a "hidden cooperative behavior" is discovered which will allow us to reduce the set of constraints.

5.3.1.1 The Two-node OPF Formulation

In this section a two-node complete system model is explicitly analysed based on the system shown in figure 5.5.

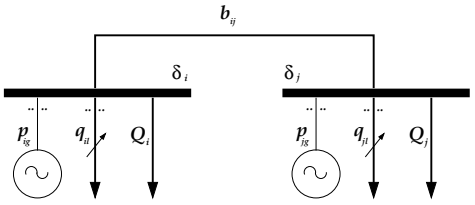


FIGURE 5.5: Two node system

Equations from 5.9 to 5.15 represent the OPF problem for this system.

$$\min_{p_{ig},q_{il},\delta_i,\delta_j} \sum_{g\in\mathbb{G}_i\cup\mathbb{G}_j} C_g(p_g) - \sum_{l\in\mathbb{L}_i\cup\mathbb{L}_j} B_l(q_l) \tag{5.9}$$

s.t.

$$b_{ij}(\delta_i - \delta_j) - \sum_{g\in\mathbb{G}_i} p_g + \sum_{l\in\mathbb{L}_i} q_l + Q_i = 0 \tag{5.10}$$

$$b_{ji}(\delta_j - \delta_i) - \sum_{g\in\mathbb{G}_j} p_g + \sum_{l\in\mathbb{L}_j} q_l + Q_j = 0 \tag{5.11}$$

$$\lfloor P_g \rfloor \leq p_g \leq \lceil P_g \rceil \quad \forall g \in \mathbb{G}_i \cup \mathbb{G}_j \tag{5.12}$$

$$\lfloor Q_l \rfloor \leq q_l \leq \lceil Q_l \rceil \quad \forall l \in \mathbb{L}_i \cup \mathbb{L}_j \tag{5.13}$$

$$|b_{ij}(\delta_i - \delta_j)| \leq \lceil F_{ij} \rceil \tag{5.14}$$

$$|b_{ji}(\delta_j - \delta_i)| \leq \lceil F_{ji} \rceil \tag{5.15}$$

5.3.1.2 Hidden Cooperation

The cooperative behavior to keep the transmission system within its limits is explored in this section. This behavior is concerned with constraints 5.14 and 5.15. These constraints basically are provided to keep the line power flow within the line capability. Because the sign of the difference between $(\delta_i - \delta_j)$ is not known beforehand then the

absolute value of this difference is checked at each node, in this case just i and j. When the complete constrained system is expanded and analysed, a cooperative behavior emerges which allows to simplify this model. Therefore, by a further analysis of inequalities 5.14 and 5.15, the following proposition can be established.

Proposition 1. The constraint set

$$|b_{ij}(\delta_i - \delta_j)| \leq \lceil F_{ij} \rceil$$
$$|b_{ji}(\delta_j - \delta_i)| \leq \lceil F_{ji} \rceil$$

is equivalent to the following constraints set

$$b_{ij}(\delta_i - \delta_j) \leq \lceil F_{ij} \rceil$$
$$b_{ji}(\delta_j - \delta_i) \leq \lceil F_{ji} \rceil$$

Proof. constraints 5.14 and 5.15 can be rewritten as:

$$-\lceil F_{ij} \rceil \leq b_{ij}(\delta_i - \delta_j) \leq \lceil F_{ij} \rceil \tag{5.16}$$
$$-\lceil F_{ji} \rceil \leq b_{ji}(\delta_j - \delta_i) \leq \lceil F_{ji} \rceil \tag{5.17}$$

on the other hand, it is known that $b_{ij} = b_{ji}$ and $\lceil F_{ij} \rceil = \lceil F_{ji} \rceil$. Let $\Phi = \lceil F_{ij} \rceil / b_{ij} = \lceil F_{ji} \rceil / b_{ji}$ then the above system can be rewritten as:

$$-\Phi \leq \delta_i - \delta_j \leq \Phi \tag{5.18}$$
$$-\Phi \leq \delta_j - \delta_i \leq \Phi \tag{5.19}$$

multiplying constraint 5.19 by -1, these two constraint systems become

$$-\Phi \leq \delta_i - \delta_j \leq \Phi \tag{5.20}$$
$$\Phi \geq \delta_i - \delta_j \geq -\Phi \tag{5.21}$$

inequalities 5.20 and 5.21 essentially tell there is a replicated constraint. By taking just one of them, this can be split into two parts. Let us take equation 5.20, which can be rewritten as

$$\delta_i - \delta_j \leq \Phi \tag{5.22}$$
$$-\Phi \leq \delta_i - \delta_j \tag{5.23}$$

multiplying equation 5.23 by -1, these two inequalities system thus become

$$\delta_i - \delta_j \leq \Phi \tag{5.24}$$

$$\delta_j - \delta_i \leq \Phi \tag{5.25}$$

finally, substituting Φ in inequalities 5.24 and 5.25

$$b_{ij}(\delta_i - \delta_j) \leq \lceil F_{ij} \rceil \tag{5.26}$$

$$b_{ji}(\delta_j - \delta_i) \leq \lceil F_{ji} \rceil \tag{5.27}$$

□

This proposition shows formally the equivalence between inequalities 5.26 and 5.27 with respect to inequalities 5.14 and 5.15.

In the original approach there are two ways to deal with this by the optimisation routine. One is to square each side of the inequality which in turn leads to fill some places in the Hessian matrix. The second way is to explicitly split the inequality as a two-inequality system as given by equation 5.16 which in turn leads to the introduction of another slack variable and its associated Lagrange multiplier impacting the Hessian matrix size and therefore the optimisation routine performance.

The central difference is with respect to the behavior. Here, the subsystems would be cooperating to deal with the transmission constraints as show in figure 5.3.1.2. Node i will take care of the transmission limit from i to j, as show in figure 5.6(a), and node j will overlook the one from j to i, as show in figure 5.6(b).

(a) Constraint $b_{ij}(\delta_i - \delta_j) \leq \lceil F_{ij} \rceil$ (b) Constraint $b_{ji}(\delta_j - \delta_i) \leq \lceil F_{ji} \rceil$

FIGURE 5.6: The constraint enforcing policy is shared between nodes i and j.

As for the performance, if we analyze the European Interconnected System presented by Bialek and Zhou (2005), where the system contained 1254 nodes, 378 generators and 1944 lines, an important improvement is highlighted when applied to the node model. By applying the second approach these 1944 lines would generate 4 constraints each. Each constraint will generate 2 equations in the non-linear optimisation solver when using the Lagrange Multipliers method. These sum up to 15552 equations. With the cooperative model this number goes down to 2 constraints per line, summing up to 7776

equations. This is a really great improvement which surely would be reflected in the optimisation routine.

5.4 The Decentralised Model

The decentralised mathematical model will be derived based on the Auxiliary Principle Problem with explicit coupling constraints (Cohen, 1978). These are constraints which involve variables from both subproblems. Let us denote $\boldsymbol{z}_i$ and $\boldsymbol{z}_j$, as the related local variables of nodes i and j respectively. In order to derive the decomposed model, three steps are required.

1. *Problem split.* The system has to be split into two or more subproblems. Each one of them must preserve the structure of the global problem.
2. *Coupling constraint identification.* The most suitable coupling constraint, a constraint which involves variables from both subproblems $CC(\boldsymbol{z}_i, \boldsymbol{z}_j)$, has to be identified in the system once it has been split.
3. *Objective function modification.* The constraint multiplied by its associated Lagrange multiplier must be added to the objective function corresponding to each subproblem.

5.4.1 Splitting the Centralized Model

To this end, the global objective function will be split into two parts. Each of them will deal with the local variables at each node. The set of constraints will be split also using the same criterion.

The optimisation subproblem for node i is given by the model described from 5.28 to 5.32:

$$\min_{p_g, q_l, \delta_i} \sum_{g \in \mathbb{G}_i} C_g(p_g) - \sum_{l \in \mathbb{L}_i} B_l(q_l) + \mu_j CC(\boldsymbol{z}_i, \boldsymbol{z}_j) \tag{5.28}$$

s.t.

$$b_{ij}(\delta_i - \delta_j) - \sum_{g \in \mathbb{G}_i} p_g + \sum_{l \in \mathbb{L}_i} q_l + Q_i \;=\; 0 \tag{5.29}$$

$$\lfloor P_g \rfloor \leq p_g \leq \lceil P_g \rceil \quad \forall g \in \mathbb{G}_i \tag{5.30}$$

$$\lfloor Q_l \rfloor \leq q_l \leq \lceil Q_l \rceil \quad \forall l \in \mathbb{L}_i \tag{5.31}$$

$$b_{ij}(\delta_i - \delta_j) \leq \lceil F_{ij} \rceil \tag{5.32}$$

and the model represented from 5.33 to 5.37 describe the optimisation subproblem for node j. This model essentially mirrors the problem for problem i, by interchanging subscript i by j and viceversa.

$$\min_{p_{jg},q_{jl},\delta_j} \sum_{g\in\mathbb{G}_j} C_g(p_g) - \sum_{l\in\mathbb{L}_j} B_l(q_l) + \mu_i CC(\boldsymbol{z}_j, \boldsymbol{z}_i) \tag{5.33}$$

s.t.

$$b_{ji}(\delta_j - \delta_i) - \sum_{g\in\mathbb{G}_j} p_g + \sum_{l\in\mathbb{L}_j} q_l + Q_j = 0 \tag{5.34}$$

$$\lfloor P_g \rfloor \leq p_g \leq \lceil P_g \rceil \quad \forall g \in \mathbb{G}_j \tag{5.35}$$

$$\lfloor Q_l \rfloor \leq q_l \leq \lceil Q_l \rceil \quad \forall l \in \mathbb{L}_j \tag{5.36}$$

$$b_{ji}(\delta_j - \delta_i) \leq \lceil F_{ji} \rceil \tag{5.37}$$

The models derived above show the same structure as the global problem. Notice the inclusion of the term $\mu_i CC(\boldsymbol{z}_i, \boldsymbol{z}_j)$. This term represent the coupling constraint and its associated Lagrange Multiplier each subproblem has to add in its objective function.

5.4.2 The Coupling Constraint

A coupling constraint is a constraint which is defined in terms of local and external variables with respect to the subproblem, (i.e. they couple these subproblems). Taking any of the two subproblems, we can identify two possible coupling constraints. One would be the power flow equation (eq. 5.29). The other one would be the constraint related to the line capability (eq. 5.32). Given that in this scenario we want to know what the economical behavior of the system is, it makes sense to chose that constraint which provides us with that information.

The Lagrange Multiplier associated with the power flow equation λ, is also the nodal spot price (Schweppe, 1998) when no losses are considered. λ represents how much it costs to produce the next unit at each node. On the other hand, we are also interested in knowing about the electrical behavior of the system. This information is also provided by this equation as it will give us the angle δ at that node. Therefore, the power flow equation is chosen as the coupling constraint. At first glance it seems the objective function is not anymore the original one. However, $CC(\boldsymbol{z}_i, \boldsymbol{z}_j)$ will be 0; if not it would mean the power flow balance at that node would be violated. Hence, taking constraint 5.37 and its associated Lagrange multiplier from the model for node j to the model for node i, the objective function for node i can be expressed with equation 5.38

$$\mu_j CC(\boldsymbol{z}_j, \boldsymbol{z}_i) = \hat{\lambda}_j [b_{ji}(\hat{\delta}_j - \delta_i) - \sum_{g \in \mathbb{G}_j} \hat{p_g} + \sum_{l \in \mathbb{L}_j} \hat{q_l} + \hat{Q_j}] \tag{5.38}$$

where the hatted symbol is used to denote an external variable. These variables must be treated as constants as the local problem can not modify its value. This value will be changed by the second problem and communicated to the first one. These will be interchanged among the subproblems in the solution process. Furthermore, as for the optimisation process, just the variable part of this expression with respect to the actual node is needed. Therefore, the constant part of this expression can be taken off. This leads to equation 5.39 as

$$- \hat{\lambda}_j b_{ji} \delta_i \tag{5.39}$$

finally, as for the node i, the coupling constraint from node j has to be added to its objective function. Equation 5.40 is the original objective function with the coupling expression added.

$$\sum_{g \in \mathbb{G}_i} C_g(p_g) - \sum_{l \in \mathbb{L}_i} B_l(q_l) - \delta_i \hat{\lambda}_j b_{ij} \tag{5.40}$$

5.4.3 The Final Distributed Model

Finally, to extend this to the general case where each node is connected to one or more nodes is a straightforward procedure applying the same principle. The distributed optimisation problem is defined by the optimisation problem represented by equations 5.41 to 5.45.

$$\min_{p_{ig}, q_{il}, \delta_i} \quad \sum_{g \in \mathbb{G}_i} C_g(p_g) - \sum_{l \in \mathbb{L}_i} B_l(q_l) - \delta_i \sum_{j \in \Gamma_i} b_{ij} \hat{\lambda}_j \tag{5.41}$$

s.*t*.

$$\sum_{j \in \Gamma_i} b_{ij}(\delta_i - \hat{\delta}_j) - \sum_{g \in \mathbb{G}_i} p_g + \sum_{l \in \mathbb{L}_i} q_l + Q_i = 0 \tag{5.42}$$

$$\lfloor P_{ig} \rfloor \leq p_g \leq \lceil P_{ig} \rceil \quad \forall g \in \mathbb{G}_i \tag{5.43}$$

$$\lfloor Q_{il} \rfloor \leq q_l \leq \lceil Q_{il} \rceil \quad \forall l \in \mathbb{L}_i \tag{5.44}$$

$$b_{ij}(\delta_i - \hat{\delta}_j) \leq \lceil F_{ij} \rceil \quad \forall j \in \mathbb{T}_i \tag{5.45}$$

Based on this model, a graph model approach will be derived in chapter 6.

5.5 Algorithms

In this section a sequential algorithm to solve each subproblem at every iteration is proposed. On each iteration, each subproblem gets the external information needed to perform its task. An "intercommunication" task will be performed in order to obtain the information each individual subproblem needs in the iterative solution process.

Algorithm 7 will be solving the model for each node. In line 1, it assigns the set $NodeS$ converted into a Double Linked List by function $DoubleLinkedList(\mathbb{N})$ to $dList$. Then, solves the problem for each node in a back and forth manner. From the first node in the list to the last node, represented by lines 6 to 9. After this, from the last node to the first node, represented by lines 10 to 13. This process will continue until convergence is reached, as indicated by line 14.

Algorithm 7 dOPF

```
1:  dList ← DoubleLinkedList(ℕ)
2:  nodeDoubleLinkedList   i ← dList.First()
3:
4:  repeat
5:     Intercommunicate values
6:     while (i.next() ≠ nil) do
7:        Compute Δz for i.node
8:        i ← i.next()
9:     end while
10:    while (i.previous() ≠ nil) do
11:       Compute Δz for i.node
12:       i ← i.previous()
13:    end while
14: until converge(Δz)
```

5.6 Study Cases

In this section a number of cases are analysed based on data extracted from Wollenberg and Wood (1996). These cases are built based on the same model with some slight changes in their characteristics. In the first case, we have three generators whose production cost functions are shown in figure 5.7.

The second case, depicted in figure 5.8, the same three generators are analysed but in this case the coal price has been lowered hence modifying the production function curve for generator 1.

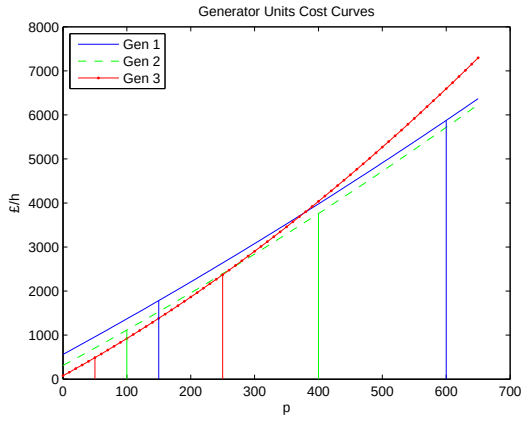


FIGURE 5.7: Production functions for the generators (Coal cost=1.1 £/MBtu)

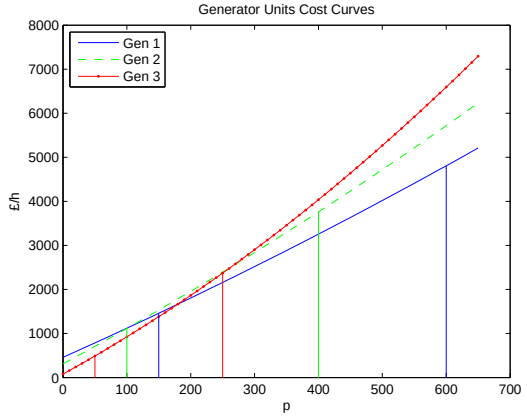


FIGURE 5.8: Production functions for the generators (Coal cost =0.9 £/MBtu)

Table 5.1, shows the data used for the different cases. Gen 1a data are the data where the coal is more expensive than the case in generator 1b.

gen	α	β	γ	$\lfloor p \rfloor$	$\lceil p \rceil$
1(a)	510	7.92	0.001562	150	600
1(b)	459	6.48	0.00128	150	600
2	310	7.85	0.00194	100	400
3	78	7.97	0.00482	50	250

TABLE 5.1: Data for the system components

The analysis will be done firstly assuming there exists just one node in section 5.6.1 and then in section 5.6.2 the analysis is extended introducing another node and the resources are split among both nodes.

5.6.1 Isolated Bus with Three generators and a Fixed Load

This is the simplest model, and in fact it can be shown that this model represents the classical *economical dispatch problem*, where all the transmission system constraints are

ignored. Figure 5.9 shows the structure for this system.

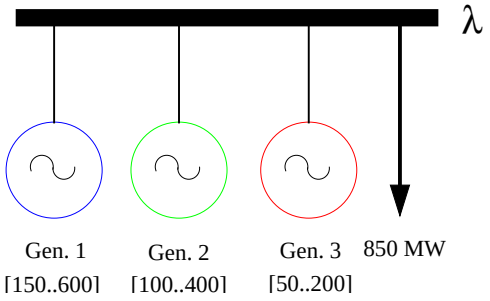
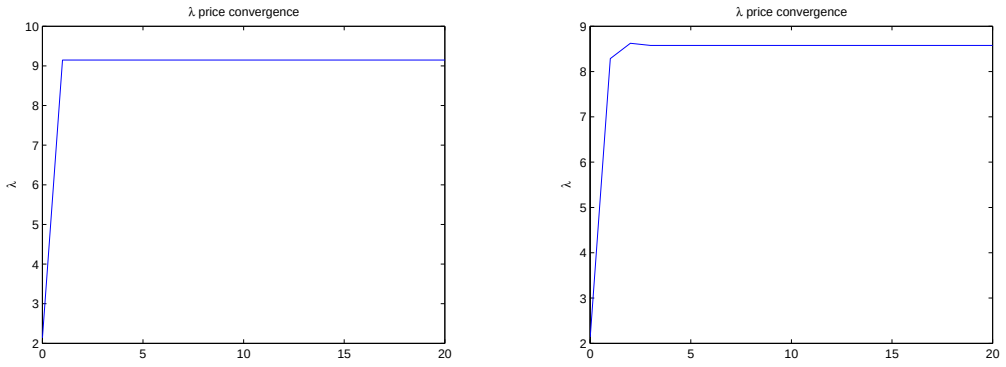


FIGURE 5.9: Isolated bus case

Figure 5.10a shows the convergence behavior for λ. As can be seen it reaches its value in just one iteration. As expected from the second order approximation. Also a reason behind this is because the solution is within the constrained solution space, as none of the generators are overloaded. As a consequence this is the optimal operation point. Furthermore, the marginal cost for each node will be the same. Therefore, $\frac{\partial C_1(393.17)}{\partial p_1} = \frac{\partial C_2(334.6)}{\partial p_2} = \frac{\partial C_3(122.23)}{\partial p_3} = \lambda$ must hold.

However, in figure 5.10b, convergence changes and it takes more iterations to solve the system. The reason behind this fact is that the energy produced in generator 1 now is cheaper. Therefore, it will be able to sell all the energy it can produce up to its upper limit. The method recognizes of this fact and introduces the upper binding constraint into the system. This is done by setting the power produced by gen 1 to its maximum (i.e. 600 MW). The power produced by generator 3 is below its limits in the first iteration (not shown). As a result, the binding constraint for the lower limit for generator 3 is introduced. KKT conditions will be deciding when the constraint has to be enforced. Another iteration is needed, but with new equations added to the system in order to meet the constraints. Given that gen 1 is binding, then λ will be set by gen 2 and 3. The relation $\frac{\partial C_2(187.13)}{\partial p_2} = \frac{\partial C_3(62.87)}{\partial p_3} = \lambda$ must hold.



(a) Gen 1 not binding

(b) Gen 1 binding

FIGURE 5.10: Isolated bus case results

Table 5.2 shows the energy allocation each generator will be producing in each configuration. Also by taking a look into λ, we notice its value is lower. It means the energy price went down as price in the coal for generator 1 is cheaper.

gen	p	
	(a)	(b)
	$\lambda = 9.1482$	$\lambda = 8.576$
1	393.17	600
2	334.6	187.13
3	122.23	62.87

TABLE 5.2: Results for one node system case, gen 1 (a) non binding $\lambda = 9.1482$ (b) binding $\lambda = 8.576$

5.6.2 Two-node System with Three Generators and Two Fixed Loads

This system will allow us to test the algorithm with the most basic transmission system, two nodes connected by one line. Figure 5.11 shows the structure for this system. Furthermore, it allow us to test the model capabilities with a decentralised case.

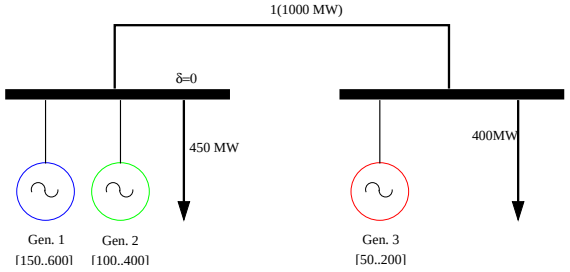
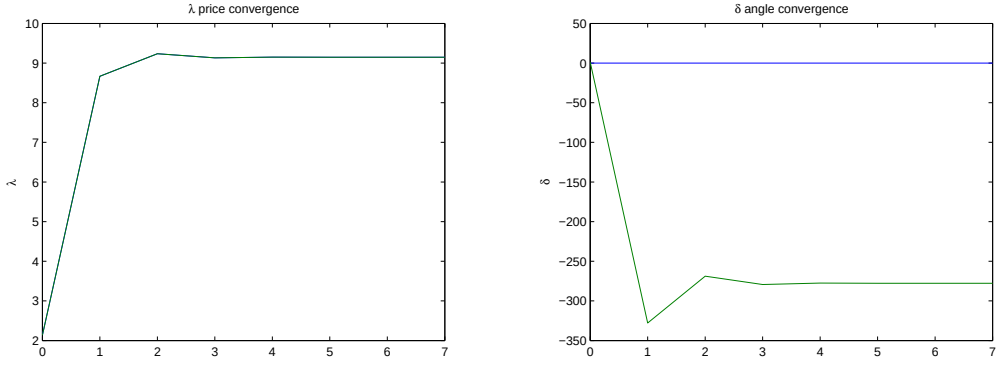


FIGURE 5.11: Two node transmission system

Figure 5.12 shows how the system price converges almost in three iterations as now it has to adjust its results based on the information provided by the other bus.



(a) λ Convergence

(b) δ Convergence

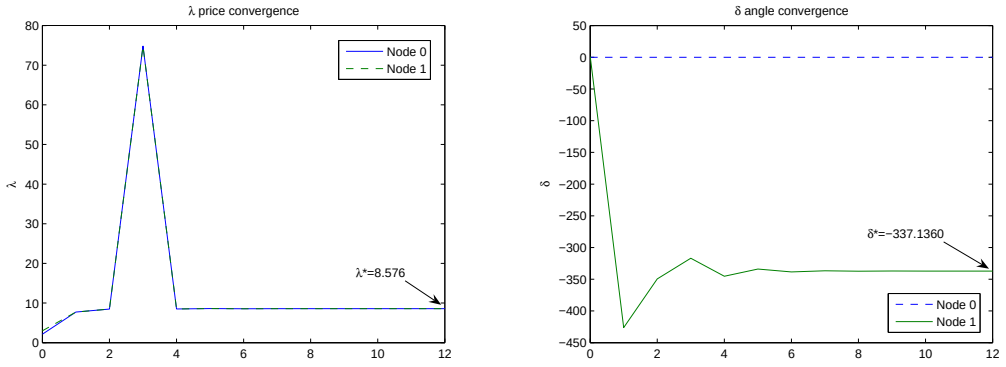
FIGURE 5.12: Two-node system, gen 1 non-binding

gen	p	node	λ	δ	line	Flow
1(a)	393.17	0	9.1482	0	0-1	277.77
2	334.6	1	9.1482	-277.77	-	-
3	122.23	-	-	-	-	-

TABLE 5.3: Results for the two-node system case, gen 1 non binding

In this case the sum of the power produced by $gen1$ and $gen2$ goes beyond the needed to feed the load attacehed to node 1. The exceding energy is exported to node 2. This is $393.17 + 334.6 - 277.77 = 450$. On the other hand the power produced in node 2 is that of $gen3$, but is not enough to feed the load attached to node 2, so the missing power is imported from node 1. This is $122.23 + 277.77 = 400$.

Now, figure 5.13 shows the results when gen 1 is binding. In this case λ converges in five iterations. The reasons behind this are the same as above, but also have to be added the constraints which have to be observed in this case.



(a) λ Convergence

(b) δ Convergence

FIGURE 5.13: Two-node system, gen 1 binding

gen	p	node	λ	δ	line	Flow
1(b)	600.0	0	8.576	0	0-1	337.14
2	187.11	1	8.576	-337.14	-	-
3	62.87	-	-	-	-	-

TABLE 5.4: Results for the two-node system case, gen 1 binding

In this case the sum of the power produced by $gen1$ and $gen2$ goes again beyond the needed to feed the load attacehed to node 1. The exceding energy is exported to node 2. This is $600 + 187.11 - 337.14 = 449.97$. On the other hand the power produced in node 2 is that of $gen3$, but is not enough to feed the load attached to node 2, so the missing power is imported from node 1. This is $62.87 + 337.14 = 400.01$.

Finally, comparing table 5.2 column (a) with table 5.3 and table 5.2 column (b) with table 5.4, we conclude the results are essentially the same. This has to be remarked given that each node is an independent entity from each other, whom by interchanging some

minimal information have been able to reach agreements which allocate the resources at optimality. Also, we can see the convergence behavior in the two-node system was almost identical, with different values but the constrained case was as fast as the unconstrained case.

5.7 Concluding Remarks

By taking a decentralised approach a simpler model for the Optimal Power Flow problem has been proposed. Based on a basic centralised model, the analysis lead us to discover certain cooperative behavior which when exploited simplifies the constraint set. This simplification reduce the constraint set described by the transmission system and therefore really improves the algorithm performance when applied to large scale systems.

Next, by using the auxiliary problem principle a distributed solution to the OPF has been proposed. As a positive side effect of the application of this decomposition, the complexity of the system, as a result of the network interconnectivity, has been overridden and converted into an intercommunication task. In turn, as shown in chapter 6, this will allow us to propose algorithms which are extremely efficient to solve the OPF problem. The model flexibility must be emphasized, as a parallel implementation is straightforward. Even more if advantage is taken of both algorithms by applying the sequential solution in parallel the interarea problem can be solved with no major modifications to the model.

This is an alternative approach between bilateral trades pricing and centralised pricing (Stoft, 2002). Both of them have bottlenecks. On the centralised price approach as the market grows up the centralised model grows also up in a non-linear way. As for the bilateral trades pricing approach, the possible set of bilateral trades among all the generators grows up in a non linear manner also. In this approach, the decentralisation is achieved by interchanging δ and λ between each node and its neighbours. This means the size of the problem will grow up linearly with the number of nodes. Just as the right set of bilateral trades, this decentralised approach will yield to the same outcome as the centralised market clearing mechanism.

Finally, the main contribution in this chapter is the decentralisation which allows the different parties to keep their data private and just sharing the information resulting from this data (i.e. δ and λ). This suggests that a centralised market can be decomposed into submarkets which in turn can be cleared based on local information and economic communication among the submarkets. In the next chapter this model will be formulated with the graph approach presented in the previous chapters. In order to cope with the decentralisation, some new features will be incorporated to the model in order to achieve it.

Chapter 6

A Decentralised Graph-based Algorithm to Solve the DC-OPF Problem

6.1 Introduction

The model presented in chapter 5 is based on a bottom-up approach model to solve the Optimal Power Flow problem. In a system with many agents interacting among them, scalability is the most appealing characteristic. In that model the solution to the decentralised model was achieved by solving each subproblem model on its own. There was some intercommunication between the subproblems in order to reach the solution. As previously stated, the solution to a convex NLP is usually implemented using a Newton-based approach (*see* section 2.4.4) which heavily relies on the Hessian Matrix $H(\mathcal{L}(\boldsymbol{z}))$. The size of $H(\mathcal{L}(\boldsymbol{z}))$ varies with the number of variables in the system as well as with the number of constraints which have to be considered. Let us assume n is the size of $H(\mathcal{L}(\boldsymbol{z}))$. Therefore the solution to the system is at least as complex as to invert this matrix (i.e. $O(n^3)$), following by a multiplication by the gradient vector (i.e. $O(n^2)$). Hence, we are dealing with a large scale problem whose complexity grows in a non linear way when the size of the system grows. Inversion becomes impractical very fast. In practice, other approaches are used which rule out the inverse; the Cholesky factorisation. However, to complicate this procedure, the constraint set is not static along the solution process as shown in chapter 4. $H(\mathcal{L}(\boldsymbol{z}))$ has to be modified in every solution step in two ways. First, the new constraints which became active in the last step have to be added. Second, the constraints which became inactive from the last step have to be removed. This leads to the conclussion that the size and structure of $H(\mathcal{L}(\boldsymbol{z}))$ will be varying along the process.

In this chapter we will exploit the Hessian Matrix structure derived from the Lagrangian, just as it was done in chapter 4, but now there are several constraints which have to be observed at the solution point. This matrix system is transformed into a graph system using simple rules. The topology of the resulting graph is very close to the network topology under study. However in this case we can derive a layered topology where the dual variables will be apart from the primal variables. Based on this graph, the decentralisation of the graph is achieved by declaring the variables which are not under the control of the subproblem as external variables (i.e. they are regarded as constants). By inspecting the Hessian for each subproblem, the terms related to such variables vanish and then, for this problem, every subproblem can be represented by a tree. Based on this tree, algorithms to reduce it are proposed. After the reduction process, an algorithm to propagate the results is proposed. However, intercommunication has to be allowed in order to interchange the values of the variables involved in the complicating constraint as well as the values for the dual variables (i.e. Lagrange multipliers).

Again, the complex procedure of removing and adding constraints into $H(\mathcal{L}(\boldsymbol{z}))$ is translated into decisions whether to visit part of the graph or not.

6.2 From the Mathematical Model to the Graph-based Model

In this section the single-node mathematical model is developed in order to obtain the graph based model. This model is flexible enough to allow the inclusion of all the market agents (i.e. Gencos and Custcos). Equations from 6.1 to 6.5 describe the model for node $i \in \mathbb{N}$.

$$\min_{p_{ig},q_{il},\delta_i} \quad \sum_{g\in\mathbb{G}_i} C_{ig}(p_{ig}) - \sum_{l\in\mathbb{L}_i} B_{il}(q_{il}) - \delta_i \sum_{j\in\Gamma_i} \hat{\lambda}_j b_{ij} \tag{6.1}$$

s.t.

$$\sum_{j\in\Gamma_i} b_{ij}(\delta_i - \hat{\delta}_j) - \sum_{g\in\mathbb{G}_i} p_{ig} + \sum_{l\in\mathbb{L}_i} q_{il} + Q_i = 0 \tag{6.2}$$

$$\lfloor P_{ig} \rfloor \leq p_{ig} \leq \lceil P_{ig} \rceil \quad \forall g \in \mathbb{G}_i \tag{6.3}$$

$$\lfloor Q_{il} \rfloor \leq q_{il} \leq \lceil Q_{il} \rceil \quad \forall l \in \mathbb{L}_i \tag{6.4}$$

$$b_{ij}(\delta_i - \hat{\delta}_j) \leq \lceil F_{ij} \rceil \quad \forall j \in \Gamma_i \tag{6.5}$$

The economical cost model for each generator is given as a quadratic function, generally $C_g(p_g) = \alpha_g + \beta_g p_g + \gamma_g p_g^2$, as described in section 2.2.4. In the same way the economical

benefit model for each variable load is represented by the quadratic function $B_l(q_l) = \beta_l q_l - \gamma_l q_l^2$ as described in section 2.2.5.1. Note the model just checks for the power flow from i to j. This model differs from other models where the formulation takes care of the flow limit in both ways. The basic reason for this is that we assume every agent in the system formulates this in the same way. Hence, agent i will take care of flow from i to j and agent j will be aware of the flow from j to i. Therefore, they will complement each other and their cooperation will preserve the condition that the flow in line ij must be in the interval $[-F_{ij} \dots F_{ij}]$. A formal demonstration of this fact was given in section 5.3.1.2.

6.2.1 The Mathematical Node Model

The node model proposed in section 5.4.3, shown again in figure 6.1, will be used as the starting point for the analysis. In this model, a variable number of elements can be attached to it (e.g. generators, variable loads and lines as well as a constant load). In case there are more than one constant load attached to the node; these can be summed up and treated as a compound fixed load.

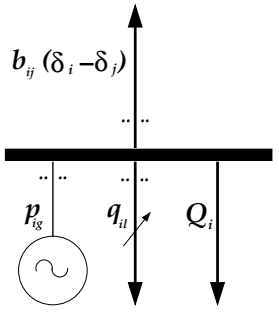


FIGURE 6.1: The Basic Node Model

In the rest of this document node i will be always considered. Taking into account this fact, index i will be stripped off in all the equations. Therefore, the next sets are equivalents $\mathbb{N} \equiv \mathbb{N}_i, \mathbb{G} \equiv \mathbb{G}_i, \mathbb{L} \equiv \mathbb{L}_i$ and $\mathbb{T} \equiv \mathbb{T}_i$. Now, introducing slack variables into the inequality constraints, the above NLP is transformed into the NLP described by the model from 6.6 to 6.12.

$$\min_{p_g, q_l, \delta} \quad \sum_{g \in \mathbb{G}} C_g(p_g) - \sum_{l \in \mathbb{L}} B_l(q_l) - \delta \sum_{j \in \Gamma} \hat{\lambda}_j b_j \tag{6.6}$$

$$\text{s.t.} \quad \sum_{j \in \Gamma} b_j(\delta - \hat{\delta}_j) - \sum_{g \in \mathbb{G}} p_g + \sum_{l \in \mathbb{L}} q_l + Q = 0 \tag{6.7}$$

$$\lfloor p_g \rfloor - p_g + \underline{p_g}^2/2 = 0, \quad \forall g \in \mathbb{G} \tag{6.8}$$

$$p_g - \lceil p_g \rceil + \overline{p_g}^2/2 = 0, \quad \forall g \in \mathbb{G} \tag{6.9}$$

$$\lfloor q_l \rfloor - q_l + \underline{q_l}^2/2 = 0, \quad \forall l \in \mathbb{L} \tag{6.10}$$

$$q_l - \lceil q_l \rceil + \overline{q_l}^2/2 = 0, \quad \forall l \in \mathbb{L} \tag{6.11}$$

$$\delta - \hat{\delta}_j - \lceil F_j \rceil / b_j + \overline{f_j}^2/2 = 0, \quad \forall j \in \Gamma \tag{6.12}$$

Let us denote each constraint as

$$\begin{aligned}
\tilde{g}(p_g, q_l, \delta) &\equiv \sum_{j\in\Gamma} b_j(\delta - \hat{\delta}_j) - \sum_{g\in\mathbb{G}} p_g + \sum_{l\in\mathbb{L}} q_l + Q \\
\tilde{h}_{\underline{p_g}}(p_g, \underline{p_g}) &\equiv \lfloor p_g \rfloor - p_g + \underline{p_g}^2/2, \quad \forall g \in \mathbb{G} \\
\tilde{h}_{\overline{p_g}}(p_g, \overline{p_g}) &\equiv p_g - \lceil p_g \rceil + \overline{p_g}^2/2, \quad \forall g \in \mathbb{G} \\
\tilde{h}_{\underline{q_l}}(q_l, \underline{q_l}) &\equiv \lfloor q_l \rfloor - q_l + \underline{q_l}^2/2, \quad \forall l \in \mathbb{L} \\
\tilde{h}_{\overline{q_l}}(q_l, \overline{q_l}) &\equiv q_l - \lceil q_l \rceil + \overline{q_l}^2/2, \quad \forall l \in \mathbb{L} \\
\tilde{h}_{\overline{f_j}}(\overline{f_j}, \delta) &\equiv \delta - \hat{\delta}_j - \lceil F_j \rceil / b_j + \overline{f_j}^2/2, \quad \forall j \in \Gamma
\end{aligned}$$

From this system we obtain its Lagrangian, $\mathcal{L}(\boldsymbol{z})$. To obtain the Lagrangian $\mathcal{L}(\boldsymbol{z})$ out of this model we need to introduce a Lagrange Multiplier for each constraint. We just had one equality constraint, $\tilde{g}(p_g, q_l, \delta)$, and all the others are inequality constraints. Let us allocate the following Lagrange Multipliers λ to $\tilde{g}(p_g, q_l, \delta)$, $\underline{\rho}_g$ to $\tilde{h}_{\underline{p_g}}(p_g, \underline{p_g})$, $\overline{\rho}_g$ to $\tilde{h}_{\overline{p_g}}(p_g, \overline{p_g})$, $\underline{\mu}_l$ to $\tilde{h}_{\underline{q_l}}(q_l, \underline{q_l})$, $\overline{\mu}_l$ to $\tilde{h}_{\overline{q_l}}(q_l, \overline{q_l})$ and $\overline{\eta}_j$ to $\tilde{h}_{\overline{f_j}}(\overline{f_j}, \delta)$. Therefore $\mathcal{L}(\boldsymbol{z})$ can be expressed as:

$$\begin{aligned}
\mathcal{L}(\boldsymbol{z}) \quad = \quad & \sum_{g\in\mathbb{G}} \left[C_g(p_g) + \underline{\rho}_g \tilde{h}_{\underline{p_g}}(p_g, \underline{p_g}) + \overline{\rho}_g \tilde{h}_{\overline{p_g}}(p_g, \overline{p_g}) \right] \\
& - \sum_{l\in\mathbb{L}} \left[B_l(q_l) - \underline{\mu}_l \tilde{h}_{\underline{q_l}}(q_l, \underline{q_l}) - \overline{\mu}_l \tilde{h}_{\overline{q_l}}(q_l, \overline{q_l})) \right] \\
& + \sum_{j\in\Gamma_i} \left[\overline{\eta}_j \tilde{h}_{\overline{f_j}}(\overline{f_j}, \delta) - \delta \hat{\lambda}_j b_j \right] + \lambda \tilde{g}(p_g, q_l, \delta)
\end{aligned}$$

where $\boldsymbol{z} = [p_g, q_l, \underline{p_g}, \overline{p_g}, \underline{q_l}, \overline{q_l}, \overline{f_j}, \delta, \lambda, \underline{\rho}_g, \overline{\rho}_g, \underline{\mu}_l, \overline{\mu}_l, \overline{\eta}_j]^T$

This equation can be solved by using the Newton approach, as defined in section 2.4.4. To this end, the system represented by equation 6.13 has to be solved.

$$H(\mathcal{L}(\boldsymbol{z}))\Delta\boldsymbol{z} = -\nabla(\mathcal{L}(\boldsymbol{z})) \tag{6.13}$$

Therefore, in order to solve $\Delta\boldsymbol{z}$, first we need to obtain $\nabla(\mathcal{L}(\boldsymbol{z}))$ and $H(\mathcal{L}(\boldsymbol{z}))$. Table 6.1, describes the relation among $\boldsymbol{z}$, $\mathcal{L}(\boldsymbol{z})$, $\nabla(\mathcal{L}(\boldsymbol{z}))$ and $H(\mathcal{L}(\boldsymbol{z}))$, where $\Psi = \sum_{j\in\Gamma} b_j$.

$$\mathcal{L}(\boldsymbol{z}) = \sum_{g\in\mathbb{G}} (C_g(p_g) + \underline{\rho}_g \tilde{h}_{\underline{p_g}}(p_g, \underline{p_g}) + \overline{\rho}_g \tilde{h}_{\overline{p_g}}(p_g, \overline{p_g})) - \sum_{l\in\mathbb{L}} (B_l(q_l) - \underline{\mu}_l \tilde{h}_{\underline{q_l}}(q_l, \underline{q_l}) - \overline{\mu}_l \tilde{h}_{\overline{q_l}}(q_l, \overline{q_l})) + \sum_{j\in\Gamma_i} (\overline{\eta}_j \tilde{h}_{\overline{f_j}}(\overline{f_j}, \delta) - \delta\hat{\lambda}_j b_j) + \lambda\tilde{g}(p_g, q_l, \delta)$$

$\boldsymbol{z}$	$\nabla(\mathcal{L}(\boldsymbol{z}))$	$H(\mathcal{L}(\boldsymbol{z}))$												
		p_g	q_l	$\underline{p_g}$	$\overline{p_g}$	$\underline{q_l}$	$\overline{q_l}$	$\overline{f_j}$	δ	λ	$\underline{\rho}_g$	$\overline{\rho}_g$	$\underline{\mu}_l$	$\overline{\mu}_l$	$\overline{\eta}_j$
p_g	$\beta_g + 2\gamma_g p_g - \lambda - \underline{\rho}_g + \overline{\rho}_g$	$2\gamma_g$								-1	-1	1			
q_l	$-\beta_l - 2\gamma_l q_l + \lambda - \underline{\mu}_l + \overline{\mu}_l$		$-2\gamma_l$							1			-1	1	
$\underline{p_g}$	$\underline{\rho}_g \underline{p_g}$			$\underline{\rho}_g$							$\underline{p_g}$				
$\overline{p_g}$	$\overline{\rho}_g \overline{p_g}$				$\overline{\rho}_g$							$\overline{p_g}$			
$\underline{q_l}$	$\underline{\mu}_l \underline{q_l}$					$\underline{\mu}_l$							$\underline{q_l}$		
$\overline{q_l}$	$\overline{\mu}_l \overline{q_l}$						$\overline{\mu}_l$							$\overline{q_l}$	
$\overline{f_j}$	$\overline{\eta}_j \overline{f_j}$							$\overline{\eta}_j$							$\overline{f_j}$
δ	$\sum_{j\in\Gamma_i} [(\lambda - \hat{\lambda}_j) b_j + \overline{\eta}_j]$									Ψ					1
λ	$\sum_{j\in\Gamma_i} b_j(\delta - \hat{\delta}_j) - \sum_{g\in\mathbb{G}} p_g + \sum_{l\in\mathbb{L}} q_l + Q$	-1	1						Ψ						
$\underline{\rho}_g$	$\lfloor p_g \rfloor - p_g + \underline{p_g}^2/2$	-1		$\underline{p_g}$											
$\overline{\rho}_g$	$p_g - \lceil p_g \rceil + \overline{p_g}^2/2$	1			$\overline{p_g}$										
$\underline{\mu}_l$	$\lfloor q_l \rfloor - q_l + \underline{q_l}^2/2$		-1			$\underline{q_l}$									
$\overline{\mu}_l$	$q_l - \lceil q_l \rceil + \overline{q_l}^2/2$		1				$\overline{q_l}$								
$\overline{\eta}_j$	$(\delta - \hat{\delta}_j) - \lceil F_j \rceil / b_j + \overline{f_j}^2/2$							$\overline{f_j}$	1						

TABLE 6.1: Relation among $\boldsymbol{z}$, $\mathcal{L}(\boldsymbol{z})$, $\nabla(\mathcal{L}(\boldsymbol{z}))$ and $H(\mathcal{L}(\boldsymbol{z}))$ ($\Psi = \sum_{j\in\Gamma} b_j$).

If this system in table 6.1 is expanded, the following linear equations system is obtained:

$$2\gamma_g \Delta_{p_g} - \Delta_\lambda - \Delta_{\underline{\rho}_g} + \Delta_{\overline{\rho}_g} = -\nabla_{p_g}, \quad \forall g \in \mathbb{G} \tag{6.14}$$

$$-2\gamma_l \Delta_{q_l} + \Delta_\lambda - \Delta_{\underline{\mu}_l} + \Delta_{\overline{\mu}_l} = -\nabla_{q_l}, \quad \forall l \in \mathbb{L} \tag{6.15}$$

$$\underline{\rho}_g \Delta_{\underline{p_g}} + \underline{p_g} \Delta_{\underline{\rho}_g} = -\nabla_{\underline{p_g}}, \quad \forall g \in \mathbb{G} \tag{6.16}$$

$$\overline{\rho}_g \Delta_{\overline{p_g}} + \overline{p_g} \Delta_{\overline{\rho}_g} = -\nabla_{\overline{p_g}}, \quad \forall g \in \mathbb{G} \tag{6.17}$$

$$\underline{\mu}_l \Delta_{\underline{q_l}} + \underline{q_l} \Delta_{\underline{\mu}_l} = -\nabla_{\underline{q_l}}, \quad \forall l \in \mathbb{L} \tag{6.18}$$

$$\overline{\mu}_l \Delta_{\overline{q_l}} + \overline{q_l} \Delta_{\overline{\mu}_l} = -\nabla_{\overline{q_l}}, \quad \forall l \in \mathbb{L} \tag{6.19}$$

$$\overline{\eta}_j \Delta_{\overline{f_j}} + \overline{f_j} \Delta_{\overline{\eta}_j} = -\nabla_{\overline{f_j}}, \quad \forall j \in \Gamma_i \tag{6.20}$$

$$\Psi \Delta_\lambda + \sum_{j\in\Gamma_i} \Delta_{\overline{\eta}_j} = -\nabla_{\delta_j} \tag{6.21}$$

$$-\sum_{g\in\mathbb{L}} \Delta_{p_g} + \sum_{l\in\mathbb{G}} \Delta_{q_l} + \Psi \Delta_{\delta_j} = -\nabla_\lambda \tag{6.22}$$

$$-\Delta_{p_g} + \underline{p_g} \Delta_{\underline{p_g}} = -\nabla_{\underline{\rho}_g}, \quad \forall g \in \mathbb{G} \tag{6.23}$$

$$\Delta_{p_g} + \overline{p_g} \Delta_{\overline{p_g}} = -\nabla_{\overline{\rho}_g}, \quad \forall g \in \mathbb{G} \tag{6.24}$$

$$-\Delta_{q_l} + \underline{q_l} \Delta_{\underline{q_l}} = -\nabla_{\underline{\mu}_l}, \quad \forall l \in \mathbb{L} \tag{6.25}$$

$$\Delta_{q_l} + \overline{q_l} \Delta_{\overline{q_l}} = -\nabla_{\overline{\mu}_l}, \quad \forall l \in \mathbb{L} \tag{6.26}$$

$$\overline{f_j} \Delta_{\overline{f_j}} + \Delta_{\delta_j} = -\nabla_{\overline{\eta}_j}, \quad \forall j \in \Gamma_i \tag{6.27}$$

6.3 Equivalent Graph Model

From 6.1, we notice how disperse the structure of $H(\mathcal{L}(\boldsymbol{z}))$ is. Several approaches have been proposed (Tinney et al., 1985; Gilbert et al., 1992) in order to exploit this structure. However, they are very general and even though they reduce the number of operations this is not optimal as they do not take advantage of the explicit structure $H(\mathcal{L}(\boldsymbol{z}))$ has for this problem. Furthermore, the decomposition approach proposed here leads to a natural multiagent scenario. Here, the interconnections which can be transformed into intercommunications task are directly identifiable.

In this section, a graph-based approach is applied to the solution of equation 6.13. It is important to notice equation 6.13 was not written as a function of $H(\mathcal{L}(\boldsymbol{z}))^{-1}$. The solution process will be based on the equivalent graph when the system represented in table 6.1 is translated to its graph representation.

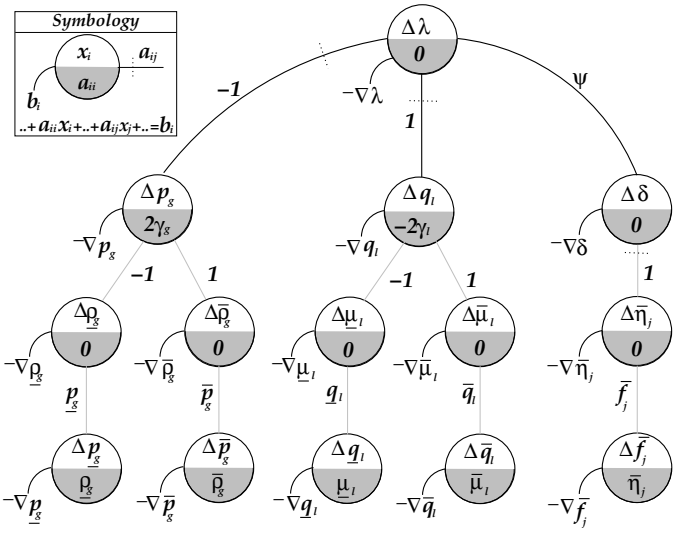


FIGURE 6.2: Graph corresponding to the basic node model

Figure 6.2, based on table 6.1, represents this system in a graphical way. This representation was introduced in section 3.3 for a general system of linear equations. Let us describe again the process to convert this matrix into its corresponding graph model. The graph-based model for a general equation is shown in figure 6.3. A node is defined for each variable in $\Delta\boldsymbol{z}$. In that node the upper half circle refers to the variable whose value has to be solved, in this case Δz_i. The lower half circle refers to the coefficient for variable Δz_i, actually $\frac{\partial^2\mathcal{L}(\boldsymbol{z})}{\partial^2 z_i}$, which in this document will be denoted as a_{z_i}. Then the independent term, represented as the small arc incident to the node, initialized with $-\nabla_{z_i}\mathcal{L}(\boldsymbol{z})$, which will be denoted as b_{z_i}. Out of this node there will possibly be interconnections with other variables represented by edges. The expression attached to these edges are the values for $H(\mathcal{L}(\boldsymbol{z}))$ in the positions relating the variables connected at each end (i.e. $\frac{\partial^2\mathcal{L}(\boldsymbol{z})}{\partial z_i \partial z_j}$). Finally, the actual value for each variable, $\boldsymbol{z}_i$, will also be stored in the node (not shown). The gray lines denote conditional evaluation which will be

performed whenever its corresponding constraint is active. This is achieved by looking at KKT conditions.

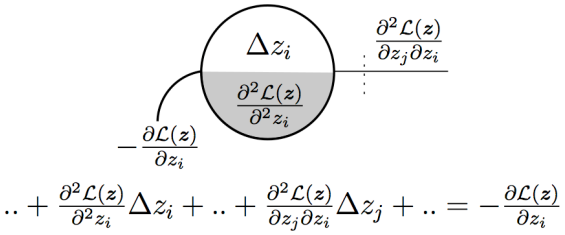


FIGURE 6.3: Graph model for the Newton's method

6.4 Algorithms

6.4.1 Fundaments

The graph structure described by figure 6.2 is amazingly regular. When the nodes are viewed in a bottom-up fashion, an evident characteristic is shared by all these nodes. If there is an upper layer, then each node of this layer is connected to that upper layer with just one node. Of course this is a well known characteristic of a special kind of graphs called *trees*, but what is special about this one? Basically, when a tree represents a system of linear equations, the solution to this system is straightforward. An algorithm has to be defined which exploits its particular shape in an ordered way. Let us denote layer 0 as that where the root node is. Therefore, the depth of this graph is 3, and the breadth will be a function of the number of elements connected to this node (i.e. generators, loads, and lines).

KKT conditions, described in section 2.4.2.2, will guide the travel into the graph. They indicates if is needed to enforce the constraints in the solution process. They also signal which constraints have to be released along the solution process. Hence, they define the way the graph is visited. Notice the difference between this approach and that from the matrix solution approach. There, KKT conditions are used to modify $H(\mathcal{L}(\boldsymbol{z}))$; this is done by adding or eliminating the rows and columns corresponding to the Lagrange Multiplier and slack variable related to the constraint being tested.

The next two sections propose algorithms to exploit this graph. These only perform the operations needed in the evaluation of the graph (i.e. exploiting completely $H(\mathcal{L}(\boldsymbol{z}))$ sparsity). The first section proposes an algorithm to travel up the graph applying a sparse forward elimination. The second section proposes an algorithm to solve the system based on the propagation the first algorithm performed. This is accomplished by traveling down the graph.

6.4.2 Notation

$\mathbb{N}, \mathbb{G}, \mathbb{L}$, and $\mathbb{T}$ represent the set of nodes, generators, variable loads, and lines in an electrical power system, respectively. Γ_i is the set of neighbouring nodes of node i (i.e. the nodes $j \in \mathbb{N} \setminus i$ such that there exists a line between nodes i and j). $\mathbb{G}_i \in \wp(\mathbb{G}), \mathbb{L}_i \in \wp(\mathbb{L}), \mathbb{T}_i \in \wp(\mathbb{T})$ are the generators, variable loads, and lines connected to node i, respectively.

6.4.3 The Graph Forward OPF Algorithm ($gFOPF$)

Let us assume nodes i and j are connected by a_{ij}. Also, let us assume i is in layer n and j is in layer $n+1$ and $|\Gamma_j| = 1$. Hence, if Gaussian elimination is applied to node j then the only node which has to be affected by this process is node i which is one layer above node j. The graph transformation process when Gaussian elimination is applied to node j is outlined in figure 6.4.

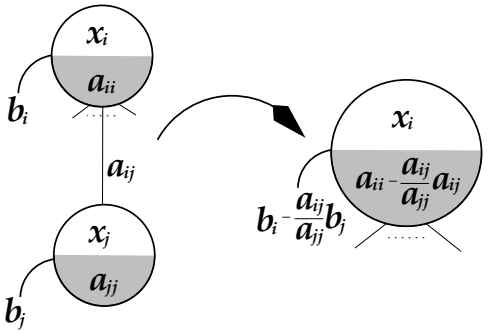


FIGURE 6.4: Graph reduction when applying Gaussian elimination

The $gFOPF$ algorithm applies this basic process to all the nodes in the tree in a bottom-up way until no more reductions can be done. This bottom-up approach has to be guided by the graph structure, and can be done in a modular way depending only on the constraint type. Having said that, we can distinguish four types of components: generators, variable loads, lines, and nodes. As can be seen from the graph, the node characteristics (Δ_λ and Δ_{δ_j}) are based on the other three characteristics of the components. Therefore, in order to reduce the node-related values first, we have to deduce the other object-related values (i.e. generators, variable loads, and lines). Algorithm 8 applies the principles mentioned above by processing first the generators related values, then the variable loads related values, following the lines related values, and finally the bus related values.

6.4.4 The Graph Backward OPF Algorithm ($gBOPF$)

The FBOPF algorithm shown in listing 9 performs the backward substitution, which is guided by the graph structure in a top-down approach. Basically, it takes advantage

Algorithm 8 gFOPF($i \in \mathbb{N}$)

for all $g \in \mathbb{G}_i$ **do**
  $a_{\underline{\rho}_g} \leftarrow -\underline{p_g}^2/\underline{\rho}_g$
  $b_{\underline{\rho}_g} \leftarrow -\nabla_{\underline{\rho}_g} + \nabla_{\underline{p_g}\underline{p_g}}/\underline{\rho}_g$
  $a_{\overline{\rho}_g} \leftarrow -\overline{p_g}^2/\overline{\rho}_g$
  $b_{\overline{\rho}_g} \leftarrow -\nabla_{\overline{\rho}_g} + \nabla_{\overline{p_g}\overline{p_g}}/\overline{\rho}_g$
  $a_{p_g} \leftarrow 2\gamma_g - 1/a_{\underline{\rho}_g} - 1/a_{\overline{\rho}_g}$
  $b_{p_g} \leftarrow -\nabla_{p_g} + b_{\underline{\rho}_g}/a_{\underline{\rho}_g} - b_{\overline{\rho}_g}/a_{\overline{\rho}_g}$
end for
for all $l \in \mathbb{L}_i$ **do**
  $a_{\underline{\mu}_l} \leftarrow -\underline{q_l}^2/\underline{\mu}_l$
  $b_{\underline{\mu}_l} \leftarrow -\nabla_{\underline{\mu}_l} + \nabla_{\underline{q_l}\underline{q_l}}/\underline{\mu}_l$
  $a_{\overline{\mu}_l} \leftarrow -\overline{q_l}^2/\overline{\mu}_l$
  $b_{\overline{\mu}_l} \leftarrow -\nabla_{\overline{\mu}_l} + \nabla_{\overline{q_l}\overline{q_l}}/\overline{\mu}_l$
  $a_{q_l} \leftarrow -2\gamma_l - 1/a_{\underline{\mu}_l} - 1/a_{\overline{\mu}_l}$
  $b_{q_l} \leftarrow -\nabla_{q_l} - b_{\underline{\mu}_l}/a_{\underline{\mu}_l} + b_{\overline{\mu}_l}/a_{\overline{\mu}_l}$
end for
for all $j \in \mathbb{T}_i$ **do**
  $a_{\overline{\eta}_j} \leftarrow -\overline{f_j}^2/\overline{\eta}_j$
  $b_{\overline{\eta}_j} \leftarrow -\nabla_{\overline{\eta}_j} + \nabla_{\overline{f_j}\overline{f_j}}/\overline{\eta}_j$
end for
$a_\delta \leftarrow -\sum_{j\in\Gamma}(1/a_{\overline{\eta}_j})$
$b_{\delta_j} \leftarrow -\nabla_{\delta_j} - \sum_{j\in\Gamma}(b_{\overline{\eta}_j}/a_{\overline{\eta}_j})$
$a_\lambda \leftarrow -\sum_{g\in\mathbb{G}_i}(1/a_{p_g}) - \sum_{l\in\mathbb{L}_i}(1/a_{q_l}) - \Psi^2/a_\delta$
$b_\lambda \leftarrow -\nabla_\lambda + \sum_{g\in\mathbb{G}_i}(b_{p_g}/a_{p_g}) - \sum_{l\in\mathbb{L}_i}(b_{q_l}/a_{q_l}) - \Psi b_\delta/a_\delta$

from the propagation that in its way up the tree the *gFOPF* algorithm performed. FFOPF has updated the value of b_i and a_{ii} when their dependencies with the lower layer were eliminated. Let us suppose node i is in layer n and node k is in layer $n+1$. Equation 6.28 has to be solved for x_i,

$$a_{ii}x_i + a_{ik}x_k = b_i \tag{6.28}$$

solving for x_i

$$x_i = \frac{b_i - a_{ik}x_k}{a_{ii}} \tag{6.29}$$

clearly from this equation, if node i is the tree root, then there are no more layers up so equation 6.29 becomes

$$x_i = \frac{b_i}{a_{ii}} \tag{6.30}$$

In summary, this algorithm applies this simple principle following the inverse process *gFOPF* took when was applied.

Algorithm 9 gBOPF($i \in \mathbb{N}$)

$\Delta_\lambda \leftarrow b_\lambda / a_\lambda$
for all $g \in \mathbb{G}_i$ **do**
  $\Delta_{p_g} \leftarrow (b_{p_g} - \Delta_\lambda)/a_{p_g}$
  $\Delta_{\underline{\rho}_g} \leftarrow (b_{\underline{\rho}_g} - \Delta_{p_g})/a_{\underline{\rho}_g}$
  $\Delta_{\underline{p_g}} \leftarrow (\nabla_{\underline{p_g}} + \underline{p_g}\Delta_{\underline{\rho}_g})/\underline{\rho}_g$
  $\Delta_{\overline{\rho}_g} \leftarrow (b_{\overline{\rho}_g} + \Delta_{p_g})/a_{\overline{\rho}_g}$
  $\Delta_{\overline{p_g}} \leftarrow (\nabla_{\overline{p_g}} + \overline{p_g}\Delta_{\overline{\rho}_g})/\overline{\rho}_g$
end for
for all $l \in \mathbb{L}_i$ **do**
  $\Delta_{q_l} \leftarrow (b_{q_l} - \Delta_\lambda)/a_{q_l}$
  $\Delta_{\underline{\mu}_l} \leftarrow (b_{\underline{\mu}_l} - \Delta_{q_l})/a_{\underline{\mu}_l}$
  $\Delta_{\underline{q_l}} \leftarrow (\nabla_{\underline{q_l}} + \underline{q_l}\Delta_{\underline{\mu}_l})/\underline{\mu}_l$
  $\Delta_{\overline{\mu}_l} \leftarrow (b_{\overline{\mu}_l} + \Delta_{q_l})/a_{\overline{\mu}_l}$
  $\Delta_{\overline{q_l}} \leftarrow (\nabla_{\overline{q_l}} + \overline{q_l}\Delta_{\overline{\mu}_l})/\overline{\mu}_l$
end for
$\Delta_\delta \leftarrow (b_\delta - \Psi\Delta_\lambda)/a_\delta$
for all $j \in \mathbb{T}_i$ **do**
  $\Delta_{\overline{\eta}_j} \leftarrow (b_{\overline{\eta}_j} - a_\delta)/a_{\overline{\eta}_j}$
  $\Delta_{\overline{f_j}} \leftarrow (b_{\overline{f_j}} - \overline{f_j}\Delta_{\overline{\eta}_j})/\overline{\eta}_j$
end for

6.4.5 The Graph OPF Algorithm ($gOPF$)

This section presents a simple iterative algorithm, called $gOPF$, which applies the algorithms $gFOPF$ and $gBOPF$ until some convergence criterion is reached. A more elaborated algorithm to propagate the most truthful values across the graph is proposed in section 5.5.

Algorithm 10 gOPF

Initialize $\boldsymbol{z}$
repeat
  Intercommunicate values
  for all $i \in \mathbb{N}$ **do**
    Evaluate $\nabla(\mathcal{L}(\boldsymbol{z}))_i$
    Apply gFOPF(i)
    Apply gBOPF(i)
    $\boldsymbol{z} \leftarrow \boldsymbol{z} + \Delta\boldsymbol{z}$
  end for
until Convergence reached

6.5 Concluding Remarks

By taking a graph-based approach, it has been possible to propose a new set of algorithms based on the basic node model. Based on table 6.1, a new graph-based solution has been

proposed. This approach allow us to scale up the system in a very efficient way (i.e. linear in the number of generators, loads and lines attached to the electrical power system). To this end, an OPF algorithm has been presented which is based on two basic algorithms. the graph forward OPF ($gFOPF$), and the graph backward OPF ($gBOPF$). These algorithms perform the minimal number of operations as a result of the topological structure of the model. Comparing this approach with the $H(\mathcal{L}(\boldsymbol{z}))^{-1}$ based solution, where it would be necessary to invert $H(\mathcal{L}(\boldsymbol{z}))$ and then multiply it by $\nabla(\mathcal{L}(\boldsymbol{z}))$, this method is much more efficient. Basically, this algorithm performs the Gaussian elimination in its $gFOPF$ phase and $gBOPF$ implements the backward substitution. This chapter has two main contributions. First, the explicit graph model derived from the mathematical model allows us to think about the model in a modular fashion. This in turn prevents us from using sparse matrix methods such as those in Tinney et al. (1985). Second, the order in which this reduction has to be realized is a consequence of a matrix graph analysis and is inspired in its topological structure.

Chapter 7

Graph Decentralisation

Chapter 6 presented a decentralised approach which relies on the auxiliary principle problem. The choice of weakening links was guided using such approach. That implied to weak the links which were coupling the problem in order to achieve such decentralisation. In this chapter a deeper analysis of these links is done which leads to its complete understanding. It will be seen that the main effect of the link weakening operation is to allow the computation of the exact gradient. However, the solution will be reinforced by taking into account second order information provided by the linking structure. This chapter is structured as follows. First, the self contained characteristic of the graph is presented. Then a QSP topological model is presented, which gives a standard graph-based representation for QSP. Also, a simpler representation is derived. Following, a graph analysis is undertaken based on the linear equation each node and its links represents. Finally, different decentralisation schemes are presented based on the previous analysis.

7.1 Self Contained Graphs

An interesting fact when this kind of graphs is used is that the information to rebuild the gradient is contained in the graph topology. First, the case where the gradient for a primal variable, p_g, is analysed. The discussion will be focused on the subgraph delineated in figure 7.1. From table 6.1, it is known that $\nabla p_g = \beta_g + 2\gamma_g p_g - \lambda - \underline{\rho}_g + \overline{\rho}_g$.

Figure 7.2 shows the gradient evaluation process for a primal variable. From section 3.7, it is known that every node has the value related to the variable itself represents. Therefore, the node gradient evaluaton starts by taking into account the gradient information within the node which in this case would be $\beta_g + 2\gamma_g p_g$, as shown in figure 7.2(a).

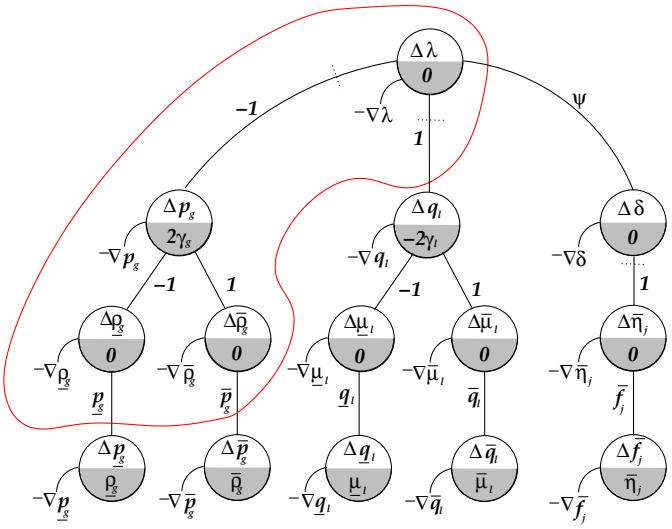
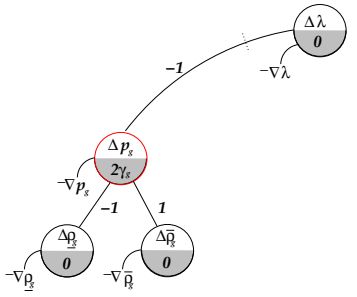
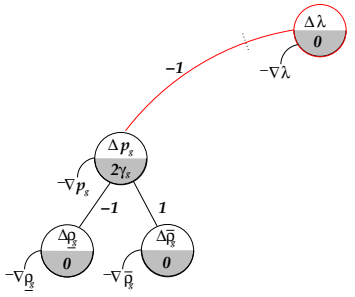


FIGURE 7.1: Subgraph for a primal variable (i.e. p_g).

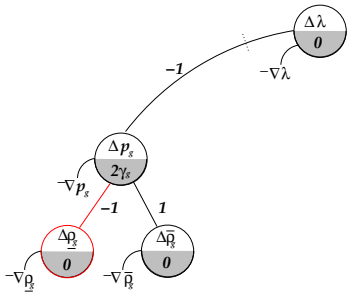
Then it starts to evaluate the portion of the gradient which is a function of the variables contained by the neighbours of p_g, as shown in figures 7.2(b), 7.2(c) and 7.2(d).



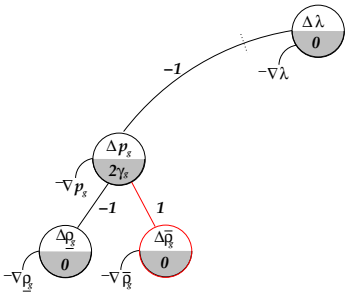
(a) $\nabla_{p_g} = \beta_g + 2\gamma_g p_g$



(b) $\nabla_{p_g} = \beta_g + 2\gamma_g p_g - \lambda$



(c) $\nabla_{p_g} = \beta_g + 2\gamma_g p_g - \lambda - \underline{\rho}_g$



(d) $\nabla_{p_g} = \beta_g + 2\gamma_g p_g - \lambda - \underline{\rho}_g + \overline{\rho}_g$

FIGURE 7.2: Graph-based gradient evaluation for a primal variable (i.e. p_g).

Now, let us turn the attention to the case where the gradient for a dual variable is to be found, λ in this case. The discussion will be focused on the subgraph delineated in figure 7.3.

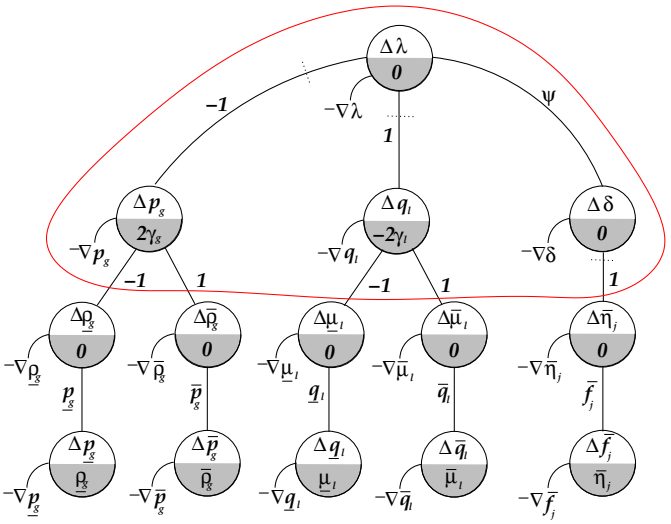
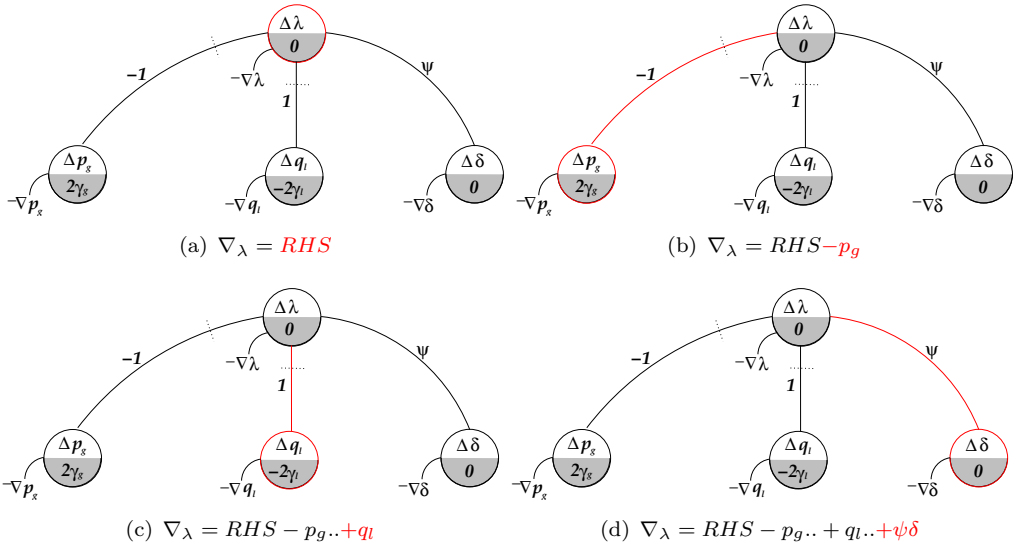


FIGURE 7.3: Subgraph for a dual variable (i.e. λ).

Figure 7.2 shows the gradient evaluation process for a dual variable. From table 6.1 it is known that $\nabla_\lambda = RHS - p_g + q_l + \psi\delta$. As before, the gradient evaluation starts by taking into account the information contained within the node itself, in this case RHS, as shown in figure 7.4(a). Then the evaluation of the links attached to this node and the variables at the other extreme of the link is performed as shown in figures 7.4(b), 7.4(c) and 7.4(d)



(a) $\nabla_\lambda = RHS$ (b) $\nabla_\lambda = RHS - p_g$

(c) $\nabla_\lambda = RHS - p_g .. + q_l$ (d) $\nabla_\lambda = RHS - p_g .. + q_l .. + \psi\delta$

FIGURE 7.4: Graph-based gradient evaluation for a dual variable (i.e. λ).

From the previous discussion, as there exist only dual variables and primal variables, the gradient for every variable can be derived straightforwardly from the graph topology. Therefore, the graph can be said to be self-contained as no external information is needed.

7.2 A QSP Topological Model Proposal

The graph-based model in chapter 4 was derived to solve the economical dispatch problem. This problem involves just one constraint and the generators bounds. However, this model can be generalised to represent a more general system. In this section a graph topology for the Newton step method is proposed. For this purpose, let us base the discussion with the QSP described by the objective function given by expression 7.1 and the constraints described by expressions from 7.2 to 7.4. This QSP consists of N variables, L equality constraints, and M inequality constraints.

$$\min_{\boldsymbol{z}_i} \quad \sum_{i=1}^{N} C_i(\boldsymbol{z}_i) \tag{7.1}$$

s.t.

$$\sum_{i=1}^{N} a_{li}\boldsymbol{z}_i = \quad b_l \qquad 1 \leq l \leq L \tag{7.2}$$

$$\sum_{i=1}^{N} c_{mi}\boldsymbol{z}_i \leq \quad d_m \qquad 1 \leq m \leq M \tag{7.3}$$

$$\lfloor \boldsymbol{z}_1 \rfloor \leq \boldsymbol{z}_n \leq \lceil \boldsymbol{z}_N \rceil \qquad 1 \leq n \leq N \tag{7.4}$$

The graph represented in this QSP is presented in figure 7.5. Each constraint is represented by a dual variable and a set of links which represent the linear terms within the constraint. The terms in the constraints are represented by links which join the primal variables with the dual variables. The only difference between equality contraints and inequality constraints is the kind of links used to build the linking structure. In the case of equality constraints, the linking structure will be active along the whole solution process. On the other hand, the linking structure for the inequality contraints will be active only when the constraint is binding.

7.3 An Equivalent Graph Representation

In this section, once the characteristics of this graph have been analysed, a simpler graph model representation is derived in order to make its handling easier. This simplification is based on two main observations: first, the bounding structures are fixed, and second the different kind of variables can be represented in such a way that the content of that node will be inferred by its representation.

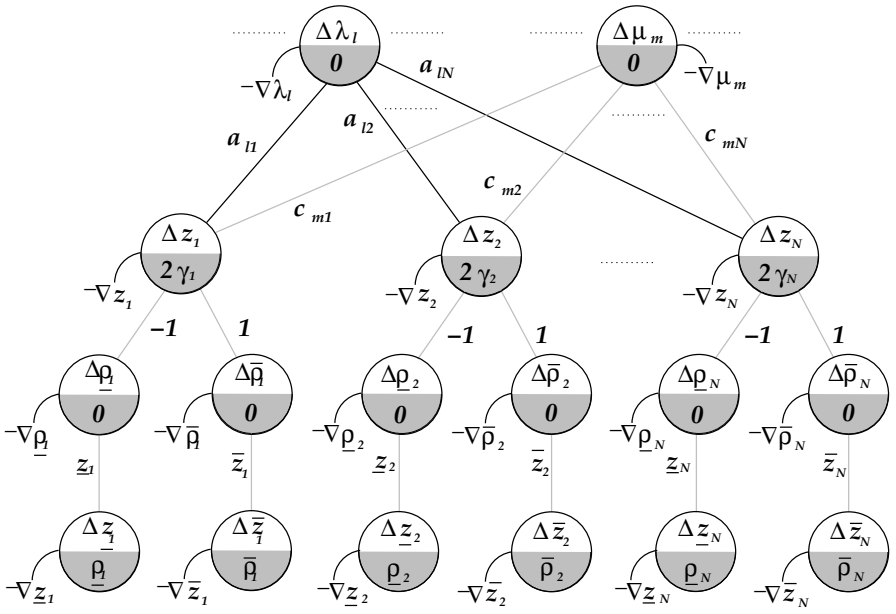


FIGURE 7.5: Proposed topology for the Newton step method

7.3.1 An Equivalent Graph Bounding Structure Representation

The subgraphs which represent the bound on the variables are well defined. Therefore a special graph notation will be derived in order to handle them, as shown in figure 7.6. Here all the links and nodes contained in such subgraph are embedded within the triangle. The link value will can be as follows:

$$link.value = \begin{cases} 1 & \text{if the constraint is lower binding,} \\ -1 & \text{if the constraint is upper binding,} \\ -- & \text{if is not binding.} \end{cases}$$

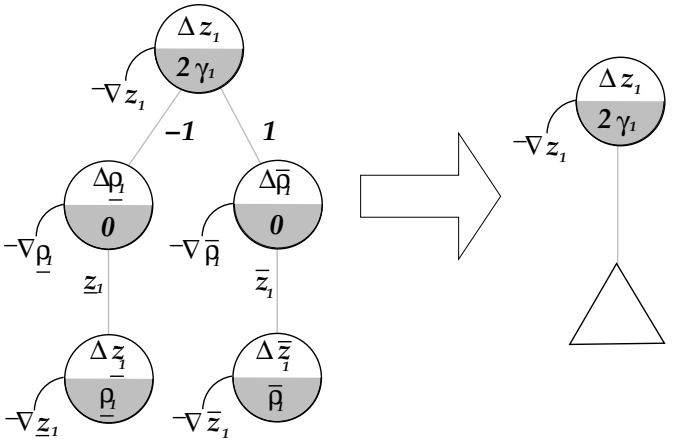


FIGURE 7.6: Bound subgraph representation

This leads to the representation shown in figure 7.7.

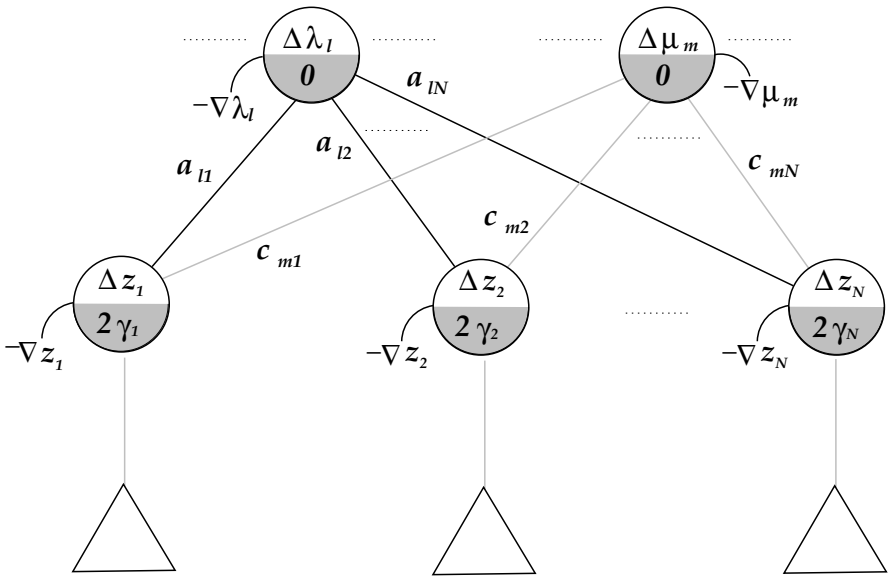


FIGURE 7.7: Hessian topology - modified representation

7.3.2 An Equivalent Node Type Representation

From the previous section, it has been learnt that the gradient is embedded within the graph topology and the information attached to each node. Therefore, a simplification for the graph representation will be derived. The last section has presented a representation for the bounding structures which control the limits on the primal variables. Therefore, now the only nodes in the remaining graph are those representing the primal variables and the dual variables. As mentioned above, the primal variables set can be further divided into two sets. The first one represents those variables which are part of the objective function. The second one contains those primal variables which appear only within the constraints. An instance of these would be the variable representing the electrical angle in the electric power market example (i.e. δ). These two subsets will be called objective and non-objective variables respectively. Therefore, there are three kinds of variables which have to be represented within the graph. These representations are shown in figure 7.8. Based on the type of variable this node is representing, the information attached to it will be known. This information is as follows

- Objective variables: Attached to this node will be the linear coefficient as well as the quadratic coefficient in order to be able to compute its gradient,
- Non objective variables: To this node there will be no additional information since its coefficients in the constraints are given by the values of the links which are attached to it,
- Dual variables: The information attached to this kind of node will be the right hand side of the constraints which will allow its gradient computation.

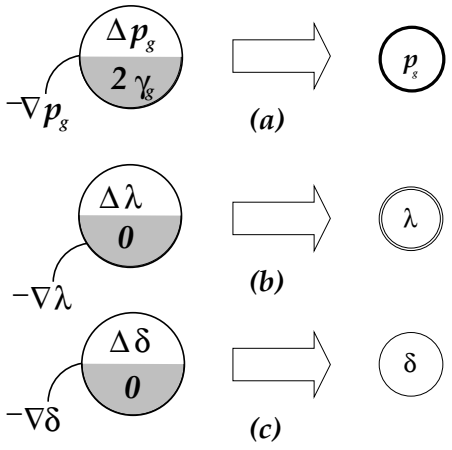


FIGURE 7.8: Variables representation. (a) Primal variable, (b) Dual Variable, (c) Non Objective Variable

This leads to the representation shown in figure 7.9, where z_2 is assumed as a nonobjective variable. Based on this graph the appropriate classes for each type of variable can be defined. Once these definitions have been implemented, the operations to solve the graph can be implemented straightforward.

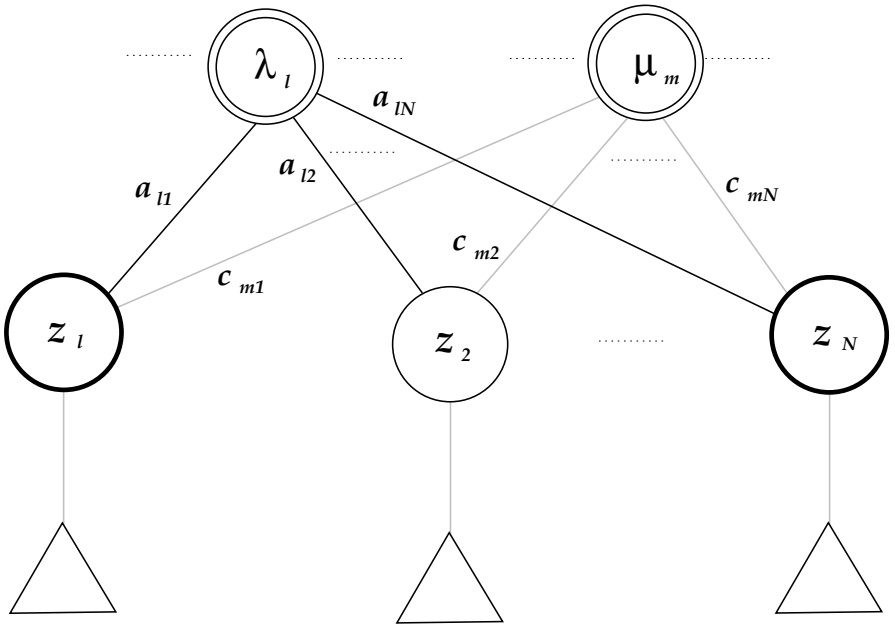


FIGURE 7.9: Hessian topology - final equivalent representation

7.4 Graph Analysis

A node and the links which are attached to it represent an equation as previously described in section 3.7. In this section the analysis for a node and the equation it represents is done. To this end let us extract the equation corresponding to z_i from the system

of linear equations which describes the Newton step. This is given by equation 7.5.

$$\frac{\partial^2 \mathcal{L}(\boldsymbol{z})}{\partial^2 \boldsymbol{z}_i} \Delta \boldsymbol{z}_i + \sum_{\forall j \in \Gamma_i} \frac{\partial^2 \mathcal{L}(\boldsymbol{z})}{\partial \boldsymbol{z}_i \partial \boldsymbol{z}_j} \Delta \boldsymbol{z}_j = -\nabla_{\boldsymbol{z}_i} \mathcal{L}(\boldsymbol{z}) \tag{7.5}$$

solving for $\Delta \boldsymbol{z}_i$ leads to

$$\Delta \boldsymbol{z}_i = \frac{-\nabla_{\boldsymbol{z}_i} \mathcal{L}(\boldsymbol{z}) - \sum\limits_{\forall j \in \Gamma_i} \frac{\partial^2 \mathcal{L}(\boldsymbol{z})}{\partial \boldsymbol{z}_i \partial \boldsymbol{z}_j} \Delta \boldsymbol{z}_j}{\frac{\partial^2 \mathcal{L}(\boldsymbol{z})}{\partial^2 \boldsymbol{z}_i}} \tag{7.6}$$

This can be rewritten as

$$\Delta \boldsymbol{z}_i = \frac{-\nabla_{\boldsymbol{z}_i} \mathcal{L}(\boldsymbol{z})}{\frac{\partial^2 \mathcal{L}(\boldsymbol{z})}{\partial^2 \boldsymbol{z}_i}} - \sum_{\forall j \in \Gamma_i} \frac{\frac{\partial^2 \mathcal{L}(\boldsymbol{z})}{\partial \boldsymbol{z}_i \partial \boldsymbol{z}_j}}{\frac{\partial^2 \mathcal{L}(\boldsymbol{z})}{\partial^2 \boldsymbol{z}_i}} \Delta \boldsymbol{z}_j \tag{7.7}$$

This expression can be thought as the improvement in the solution for the component in the orthogonal axis $\boldsymbol{z}_i$. It can be split into two parts. The first part, described by expression 7.8, is a component which always appear in any decentralisation approach for the graph.

$$\frac{-\nabla_{\boldsymbol{z}_i} \mathcal{L}(\boldsymbol{z})}{\frac{\partial^2 \mathcal{L}(\boldsymbol{z})}{\partial^2 \boldsymbol{z}_i}} \tag{7.8}$$

This is the contribution based on $-\nabla_{\boldsymbol{z}_i} \mathcal{L}(\boldsymbol{z})$ just like in the steepest descent methods. However the length of the step will be reinforced with the second order information provided by $1/\frac{\partial^2 \mathcal{L}(\boldsymbol{z})}{\partial^2 \boldsymbol{z}_i}$. This will be the case for the primal variables, however for the dual variables there will not be second order information, and therefore the gradient step size will have to be controlled by other means.

The second part, described by expression 7.9, is composed by all the second order contributions which will be collected by $\boldsymbol{z}_i$ from its neighbours (i.e. Γ_i).

$$-\sum_{\forall j \in \Gamma_i} \frac{\frac{\partial^2 \mathcal{L}(\boldsymbol{z})}{\partial \boldsymbol{z}_i \partial \boldsymbol{z}_j}}{\frac{\partial^2 \mathcal{L}(\boldsymbol{z})}{\partial^2 \boldsymbol{z}_i}} \Delta \boldsymbol{z}_j \tag{7.9}$$

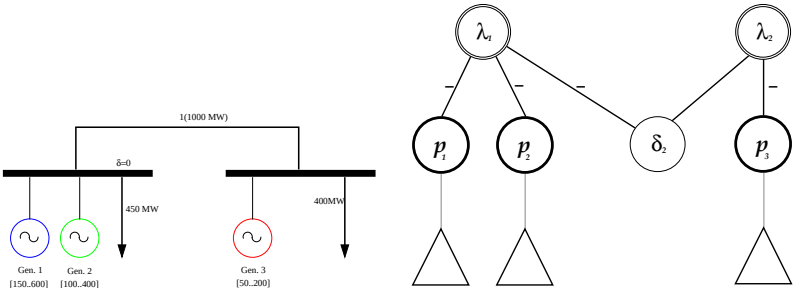
This part will have a variable number of components which will depend on the applied decentralisation approach. These can go from taking into account the second order information from all the neighbours to the other extreme where no second order information from them will be collected at all. The first approach would be the full centralised Newton step and the second one would result in the steepest descent reinforced with

the second order information for the same orthogonal axis. Nevertheless, between these two approaches there is a plethora of options about which of the components of second order information can be taken into account. In fact there are $|\wp(\Gamma_i)|$ choices and the choice at any point will be based on the particular decentralisation approach. Let us define $\mathcal{L}_h$ as the set of links which are taken into account for this process. With this in mind expression 7.9 can be rewritten as expression 7.10

$$- \sum_{\substack{\forall j \in \Gamma_i \\ (i,j) \in \mathcal{L}_h}} \frac{\frac{\partial^2 \mathcal{L}(z)}{\partial z_i \partial z_j}}{\frac{\partial^2 \mathcal{L}(z)}{\partial^2 z_i}} \Delta z_j \tag{7.10}$$

7.5 The Graph and Its Decentralisation

In this section the graph and its decentralisation is addressed. To this end an operation over the links of the graph, called *link weakenning*, is defined. Then, three different approaches to decentralise the graph are proposed. The first one is a complete decentralised, the second approach is based on the notion of the kind of variables within the graph (i.e. primal and dual variables); and the third approach will be based on agency definitions. The last approach will be described in the next chapter. To this end let us take the system which has been used along the document as shown in figure7.10(a). The graph representation corresponding to this example is shown in figure 7.10(b), where the minus sign represents a -1 value for the link.



(a) Two nodes system. (b) Its graph representation.

FIGURE 7.10: Two nodes system and its graph representation

7.5.1 Link Weakening

Before going into the decentralisation approaches, let us define the link weakening operation which allows the decentralisaton process. This operation labels the links with one of the following two labels.

- HARD - This labeling will be granted to those links which are not part of the decentralisation process. The graph reduction process will take into account these links,
- SOFT - These links provide the means to decentralise the graph. If the link posseses this property, then the reduction process will not pass through them. Nevertheless, by using its connectivity, they will provide a means to retrieve the actual value of the variable at the other end of the link which will allow the gradient to be computed, as described in section 7.1.

7.5.2 A Gradient-oriented Approach

The first approach is to decentralise the graph in an extreme way by weakening all the links as shown in figure 7.11. This method leads to a model where the gradient method has to be applied at each node in the graph.

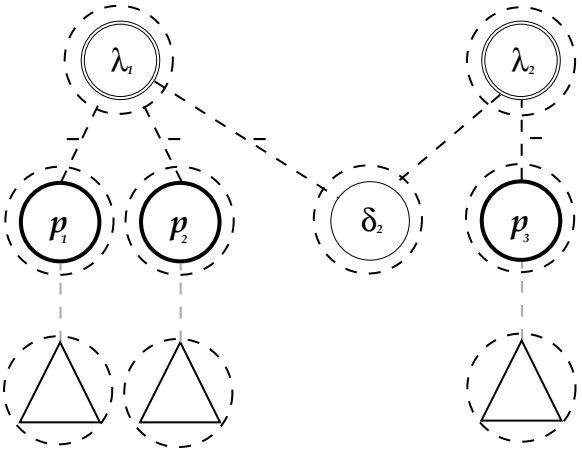


FIGURE 7.11: An gradient-based decentralisation approach

From this figure and based on equation 7.7 we can assert it will become equation 7.11, where the second order information of all of its neighbours is disregarded. In this case $\mathcal{L}_h = \emptyset$. Therefore equation 7.7 becomes equation 7.11.

$$\Delta \boldsymbol{z}_i = \frac{-\nabla_{\boldsymbol{z}_i} \mathcal{L}(\boldsymbol{z})}{\frac{\partial^2 \mathcal{L}(\boldsymbol{z})}{\partial^2 \boldsymbol{z}_i}} \tag{7.11}$$

Nevertheless, those nodes which have proper second order information will be able to use it in order to speed up the convergence process. In particular, all the nodes related to primal variables have this information. Dual variables do not have second order information at all and therefore they will have to use equation 7.12

$$\Delta \boldsymbol{z}_i = -\kappa \nabla_{\boldsymbol{z}_i} \mathcal{L}(\boldsymbol{z}) \tag{7.12}$$

The main drawback of gradient methods known also as steepest descent methods is the hardness to estimate κ. Therefore in the primal nodes equation 7.13 holds.

$$\kappa = \frac{1}{\frac{\partial^2 \mathcal{L}(z)}{\partial^2 z_i}} \tag{7.13}$$

7.5.3 A Dual-oriented Approach

The second natural approach to decentralise the graph is the dual-oriented approach. From the model proposed in section 7.2 it is known the dual variables are in only one layer so if a line across both layers is drawn dissecting the graph, the links which connected the dual variables with the primal variables will be weakened as shown in figure 7.12.

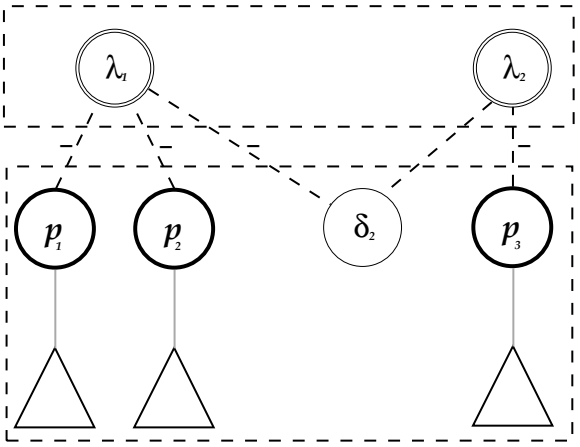


FIGURE 7.12: An dual-oriented decentralisation approach

The only dual variables considered in this case are those related with constraints involving two or more primal variables (i.e. bound dual variables are not split from the primal variables set). Let us denote $\mathcal{D}$ as the set of those dual variables. Therefore equation 7.7 becomes equation 7.14, where all the second order information about the dual variables are disregarded by the primal variables. On the other hand, as the dual variables only have links with primal variables, they are now isolated just as in the gradient approach.

$$\Delta z_i = \frac{-\nabla_{z_i} \mathcal{L}(z)}{\frac{\partial^2 \mathcal{L}(z)}{\partial^2 z_i}} - \sum_{\substack{\forall j \in \Gamma_i \\ z_j \notin \mathcal{D}}} \frac{\frac{\partial^2 \mathcal{L}(z)}{\partial z_i \partial z_j}}{\frac{\partial^2 \mathcal{L}(z)}{\partial^2 z_i}} \Delta z_j \tag{7.14}$$

7.5.4 An Agent-oriented Approach

In this approach the decentralisation process is made based on concepts drawn from the multiagent community. A classical definition for an agent is

"An agent is a computer system *situated* in an *environment*, and capable of *flexible autonomous action* in this *environment* in order to meet its *design objectives*" (adapted from Jennings and Wooldridge (1995)).

The interpretation in this work for an agent is an entity which posseses some states or variables and presents an independent and proactive behaviour, represented by their objective function. Furthermore, it is situated in an environment which he can sense and act accordingly. However, he is also constrained by its own limitations as well as the constrains presented by the environment. Going back to the problem we are addressing in this thesis, the DC OPF. There the agents would be acting on behalf of each node. Therefore, their own limitations would be the generation levels for the generators and the power balance at each node. On the other hand the constraints represented by the environment would be represented by the transmission constraints. To cope with this paradigm, the graph is split into subsets of primal variables, links, and dual variables. Based on the membership of these components, the graph is decentralised as shown in figure 7.13. This decentralisation approach is described in chapter 8.

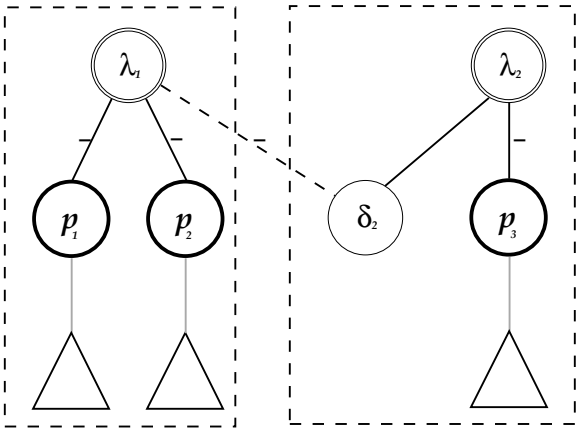


FIGURE 7.13: An agend-based decentralisation approach

7.6 Concluding Remarks

In this section the main concepts on how to decentralise the graph have been presented. First the self-containing characteristic has been uncovered. Then a standard topological model proposal for the Newton step, when applied to QSP, has been presented. This topological model lends itself to a simpler representation which allows its direct implementation. Then the underlying decentralisation principles have been presented based on the analysis of the equation represented by the node and its links. This allow us to compute the gradient directly from the graph, provided the correct information is attached to each node. Finally, three decentralisation approaches have been described. The first one is a totally decentralised approach which will lead us to a gradient oriented model reinforced with its proper second order information. The second one is based on

the type of variables (i.e. primal or dual), and leads us to a horizontal graph split. The last approach is based on the agency concepts and will be further described in chapter 8.

Chapter 8

An Agent-based Decentralisation Approach for QSP

In the previous chapter several approaches to decentralise the graph have been described. The last one addressed the decentralisation task based concepts drawn from the multi-agents community. A classical definition for an agent is

"An agent is a computer system *situated* in an *environment*, and capable of *flexible autonomous action* in this *environment* in order to meet its *design objectives*" (adapted from Jennings and Wooldridge (1995)).

Jennings and Wooldridge follow by asserting a weak notion of agency as a computer system which posesses the following properties:

- *autonomy*- agents operate without the direct intervention of humans or others, and have some kind of control over their actions and internal state;
- *social ability*- agents interact with other agents (and possibly humans) via some kind of agent-communication language;
- *reactivity*- agents perceive their environment, (which may be the physical world, a user via a graphical user interface, a collection of other agents, the INTERNET, or perhaps all of these combined), and respond in a timely fashion to changes that occur in it;
- *pro-activeness*- agents do not simply act in response to their environment, they are able to exhibit goal-directed behaviour by taking the initiative.

These properties can be asserted in the decentralisation approach as the decentralised components

- Are *autonomous*, as their decisions are taken on their own. These decisions are based on their own information as well as the information they gather from their local environment,
- Have *social ability*, as they intercommunicate their corresponding states with their neighbours in order to exchange their actual state so everybody can take better decisions about their future states,
- Are reactive, as they perceive their environment represented by a set of constraints and respond in a timely fashion to changes that could activate such constraints; and
- *Act Proactively*, given that they are adapting their state accordingly with the changes in the environment, always acting as selfish agents trying to minimize their own costs and by doing so a global optimum allocation of resources is reached.

Therefore, the agents are solving their own problem and interchanging some information which is required from the other agents and, by doing so, the global problem is solved. To this end, in this chapter a representation for the agents is presented based on these characteristics. Before that, a literature review about multiagent concepts applied to electrical power markets is presented in section 8.1. Then in section 8.2 the agent-based decentralisation approach is developed in order to obtain the agent-based decentralised graph. Finally, section 8.4 presents some concluding remarks.

8.1 Previous Work

In the last fifteen years research on Multi-Agent Systems (MAS) applied to electrical power markets has consistently been published. A review was done with the main goal to identify what kinds of agents are being under research as well as the way they are used in order to support the electrical power market. This review is presented in a chronological order. Talukdar and Ramesh (1994) present a MAS that solves a contingency constrained optimal power flow by decomposing the OPF in smaller problems, which in turn can be solved in parallel by the agents using asynchronous communication. It focuses on continuous control actions to correct the effects of non-planned changes on the system. A MAS solution is proposed consisting of the following agents *Data Importers, Probes, Voyagers, Inhibitors* and *Destroyers*. Ygge and Akkermans (1996) performed a study for power load management as a computational market based on load agents called *homebots*. They assume concave preferences for the customers and their task is to find the consumption level each customer has. The system is solved using Newton's method and as a result they report superlinear rate of convergence, a natural characteristic of the method. Newton's method is a centralised approach to solve non linear optimisation problems. This problem is avoided by interchanging first and second

order information among the agents which leads to an intercommunication bottleneck as the system grows up. As a result, the desirable characteristic of autonomy is lost as they have to share their private data in order to solve the problem collectively. Furthermore, there is a centralised aspect as they rely on a centralised algorithm to cope with the bounds which constrain the loads. Besides this, as a demand-side management system, it does not take into account the generation side leading to a non realistic problem solution. A MAS framework is presented by Lam and Wu (1999), as a mean to simulate electricity markets in order to discover possible flaws in the general market models when applied to a particular context. This simulator is built on top of JATLite (Java Agent Template Lite), using mSQL accessed with JDBC. The systems participant agents are Independent System Operators, Power Exchanges, Schedule Coordinators, Utility Distribution Companies, Generators, Customers, Retailers, Brokers and Aggregators. This work delineates the composition of the electrical power market as well as the interaction among them. It also points out the technologies available at that time in order to implement the system. Krish and Ramesh (1998a) present a negotiation protocol based on cooperative game theory. Here, no knowledge about the strategies of the other players is assumed. Coalitions are permitted as long as they do not become an oligopoly and the grand coalition is not allowed. The negotiation model is based on two actions: To prioritise possible coalition partners and bargain with the selected partners, if coalition is appropriate. After that, the solution is fed into the market game, whose output is used as a feedback for future transactions. This algorithm is applied in Krish and Ramesh (1998b), to the Power Markets area. The production cost curve, which in general is represented by a quadratic curve, is discretized to two values, *low-cost* and *high cost*, and a third value is deduced as the average of these two. A decentralised coalition scheme is applied by Contreras and Wu (1999) to the transmission expansion planning problem. Here, the goal is to determinate the optimal number, as well as location, of lines to an existing transmission system. This is done in order to meet the forecasted load in a given year as economically as possible. It addresses three issues, determining how coalitions are formed, the implementation of a bargaining algorithm and the distribution of the costs for every single agent. A DC optimal power flow is used whose model has two main constraints: the first one preserves the power balance at each bus, the second one keeps the lines below their load limits. In this game, the goal is to allocate the total cost among the agents in a fair way. The minimal coalition of agents is constituted by one generator, one load, and one transmission line. A set of axioms is enumerated which in turn are used to generate all the possible coalitions for the particular example. The Bilateral Shapley Value (BSV) is used to create a fair distribution of resources among two agents only. BSV is a method to calculate the share an agent has to contribute to all the possible coalitions where it can be enrolled. They follow Klush and Shehory's coalition formation method (Klush and Shehory, 1996). This method consists essentially of four phases: Self Calculation, Communication, BSV Calculation, and Bilateral Negotiation. Finally a backward induction method is developed to allocate

the costs. Yeung, Poon, and Wu (1999), proposed a MAS approach in order to solve the coalition formation for multilateral trades. As in (Contreras and Wu, 1999), Klush and Shehory's coalition formation method is used. Each agent represents all the functions provided in each bus (generator, consumer). Two agent models are presented, the first one without network transmission costs. The second one introduces linear transmission costs into the first model. In this model, transmission costs vary proportionally with the line load. All the trade is done using a bilateral trade strategy. This implies a communication bottleneck as the system grows. Furthermore, for large scale systems, coalition formation algorithms still are hard even though there have recently been advances in this area (Rahwan and Jennings, 2007). Harp, Bruce, Wollemberg, and Samad (2000), present a simulator for the electric power industry. A decomposition of the system is undertaken based on a general MAS framework. They propose some design guidelines for the system as well as the intercommunication mechanism. However, they rely in a centralised mechanism to clear the market. This fact sets a bottleneck if this system is to be scaled. The use of a MAS-based system is presented by Wei, Yang, Yen, and Wu (2001) to perform a decentralised approach for wholesale cross-border trade planning. First, it presents the centralised solution to the problem. Then, it introduces the decentralised approach based on a first come, first served. A lemma showing the equivalence between the decentralised and the centralised approach is given. A centralised security agent is provided to approve the transaction among the agents. This proposal is based on a shortest path search algorithm. Zhou, Tu, Talukdar, and Marshall (2004) present a Genetic Algorithm-based MAS to perform the simulation of the wholesale electricity market. This leads to a new market design, contrasting it with the year 2000 California failure which became a crisis. The model implemented is the Poolco used in California, using the Last Accepted Offer (LAO) pricing rule. In this model, the auctioneer imposes a price cap. Each agent is based on a GA, which has 20 genes each one of them with 20 boolean values, representing their bids in each stage. An infinitely repeated multi-unit auctions is proposed as a learning method. Their results report the GA is run for 100 times with 10,000 periods each. It reaches equilibrium whose results suggest to offer the cheapest generation units, withhold the upper ones but offer the one on the top. It is claimed these are the results which best fit the California crisis situation. Then in Tolbert, Qi, and Pendg (2001) a scalable multi-agent paradigm is presented. It is a proposal which brings scalability as a desired characteristic of a MAS if this is to be used in the Electrical Power Area. Also, it proposes a dynamic hybrid MAS to fulfill this requirement. They also consider ancillary agents (FACTS, transformers, and so on) besides the traditional agents (Generators, loads, transmission lines, etc.).

As it can be seen, the kind of interpretations as for the agency concepts vary widely. Furthermore, some of them rely on centralised mechanisms in order to solve the system and some others reach equilibrium in a decentralised way. As previously stated a multi-agent system is one composed by agents who act independently, are able to act on the environment, and can cooperate. All this in a proactive manner. However they are

constrained by their own capabilities and also by the environment. In the next section a methodology is proposed in order to derive the decentralised graph based on the notions of states and constraints.

8.2 An Agent-based Decentralisation Approach

The agent paradigm defines the multiagent system as a set of agents who are acting independently, constrained by the environment, and always proactively. To fulfill these concepts, the system is represented by three sets. The set of agents $\mathcal{A}$, the set of variables $\mathcal{V}$, and the set of constraints $\mathcal{C}$. The last two sets are split among the agents. Therefore every agent a_i is characterised by two sets:

- The set of states or variables, $\mathcal{V}_{a_i} \subseteq \mathcal{V}$,
- The set of constraints the agent is subject to, $\mathcal{C}_{a_i} \subseteq \mathcal{C}$,

These sets have the following characteristics

- The variable subsets are disjoint (i.e. $\forall a_i, a_j \in \mathcal{A}, i \neq j, \mathcal{V}_{a_j} \cap \mathcal{V}_{a_i} = \emptyset$),
- The constraint subsets are disjoint (i.e. $\forall a_i, a_j \in \mathcal{A}, i \neq j, \mathcal{C}_{a_j} \cap \mathcal{C}_{a_i} = \emptyset$),
- All the variables in the system belong to some agent (i.e. $\mathcal{V} = \cup_{a_i \in \mathcal{A}} \mathcal{V}_{a_i}$),
- All the constraints in the system belong to some agent (i.e. $\mathcal{C} = \cup_{a_i \in \mathcal{A}} \mathcal{C}_{a_i}$).

8.2.1 Algorithm

Algorithm 11 takes as input the sets $\mathcal{A}, \mathcal{V}$, and $\mathcal{C}$ representing the set of agents, the set of program variables, and the set of constraints, respectively. It derives the agent-based graph model. The algorithm consists of five stages

- Primal variables creation and bounding,
- Dual variables creation,
- Linking structure creation,
- Agents creation, and
- Non proper links weakening.

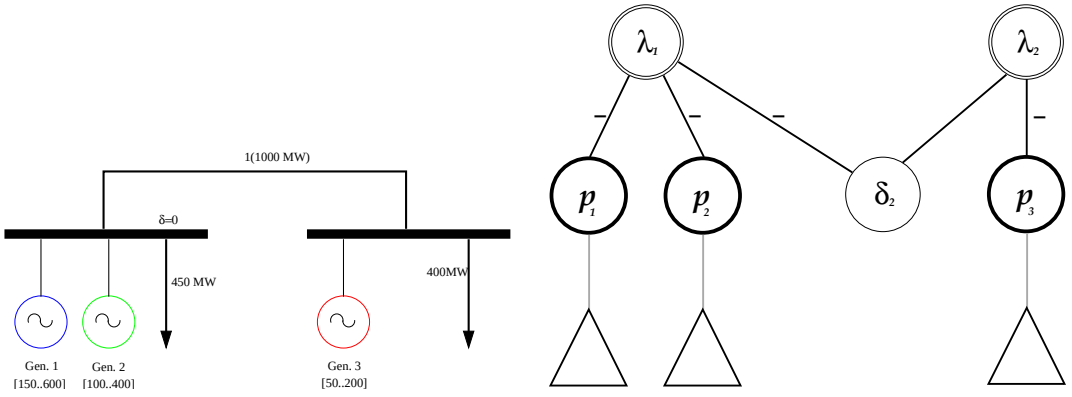
All these stages are explained with an example in section 8.2.2.

Algorithm 11 SQPAgentification ($\mathcal{V}$: The set of program variables, $\mathcal{C}$: The set of constraints, $\mathcal{A}$: The set of agents)

for all $v \in \mathcal{V}$ **do**
  if $v.type$ ==OBJECTIVEVARIABLE **then**
    Create objective variable node for variable v
  else
    Create nonObjective variable node for variable v
  end if
  if v.bounded==**TRUE then**
    Create the node bounding structure for variable v
  end if
end for
for all $c \in \mathcal{C}$ **do**
  if c.type==EQUALITY **then**
    Create equality dual variable node λ_c for constraint c
  else
    Create inequality dual variable node λ_c for constraint c
  end if
  $\mathcal{L} \leftarrow \emptyset$
  for all $term \in c$ **do**
    {Create link l connecting the dual variable}
    { from λ_c to $term.ProgramVariable$}
    $\mathcal{L} \leftarrow \mathcal{L} \cup l$
    l.value $\leftarrow term.Coefficient$
    l.dualVariable $\leftarrow \lambda_c$
    l.programVariable $\leftarrow term.ProgramVariable$
    if c.type==INEQUALITY **then**
      l.conditionalType $\leftarrow$ CONDITIONAL
    else
      l.conditionalType $\leftarrow$ PERMANENT
    end if
  end for
end for
for all $a \in \mathcal{A}$ **do**
  for all $v \in \mathcal{V}_a$ **do**
    $v.owner \leftarrow a$
  end for
  for all $c \in \mathcal{C}_a$ **do**
    $c.owner \leftarrow a$
  end for
end for
for all $l \in \mathcal{L}$ **do**
  if l.programVariable.owner $\neq$ l.dualVariable.owner **then**
    l.connectivityType $\leftarrow$ SOFT
  else
    l.connectivityType $\leftarrow$ HARD
  end if
end for

8.2.2 Building the Agent-based Model: An Example

In this section algorithm 11 is applied to the example which has been used along this document. Therefore, our aim is to transform the EPM model described in figure 8.1(a) to its graph-based representation shown in figure 8.1(b). To this end, an ontology has been defined in order to describe the electrical power market (*see* appendix B), as well as an ontology to describe separable quadratic programs (*see* appendix E). A translator between the EPM and QSP ontologies is used. In this case the listing provided in appendix D is the description for the system in figure 8.1(a) and the listing in appendix G is its corresponding QSP description. This is the one used by algorithm 11.



(a) EPM model (b) Graph based model

FIGURE 8.1: An Electrical Power Market ant its corresponding graph model

In the EPM there are two participating agents, represented by the set $\mathcal{A} = \{a_1, a_2\}$. The variables which represent the state of the system as well as the controls are represented by the set of variables $\mathcal{V} = \{p_1, p_2, p_3, \delta_2\}$. On the other hand this system is subject to a set of constraints imposed by the own agents limitations as well as the limitations imposed by the environment where the agents are working in. Specifically, the set of constraints is $\mathcal{C} = \{(450 - p_1 - p_2 - \delta_2 = 0), (400 - p_3 + \delta_2 = 0)\}$. The subsets of variables each agent has to deal with are $\mathcal{V}_{a_1} = \{p_1, p_2\}$, $\mathcal{V}_{a_2} = \{p_3, \delta_2\}$. Finally, the set of constraints each agent is subject to are $C_{a_1} = \{(450 - p_1 - p_2 - \delta_2 = 0)\}$ and $C_{a_2} = \{(400 - p_3 + \delta_2 = 0)\}$

8.2.2.1 Program Variables Creation

The code between line 2 and 6 refers to the variable creation process. In this stage all the variables or states $\mathcal{V}$ are created. The elements in this set can be of two kinds:

- Objective variables. These are variables which appear in the objective function. They must have a quadratic coefficient in one of the terms where they appear. Attached to this variable the values for the constant, linear and quadratic coefficients (i.e. α, β, γ), are stored in order to evaluate the gradient completely.
- Non-objective variables. These are variables which appear within the constraints set only. No special information is stored as the information attached to the links contains the actual coefficients for these variables.

Therefore, in this case p_1, p_2, and p_3 are objective variables whereas δ_2 is a non-objective variable. Notice the difference between the graph representation for each node. A thick circle represents an objective variable and the non-objective variable is represented by a normal thin circle. This process is depicted in figure 8.2, and its final result is shown in figure 8.3 where the variable δ_2 is a non-objective variable.

FIGURE 8.2: Program variables creation.

8.2.2.2 Program Variables Bounding

The code between line 7 and 9 refers to the variable creation process. In this step, the slack variables and dual variables needed to handle the bounds for the program variables are created and their interconnections are built. This process is described in figure 8.3.

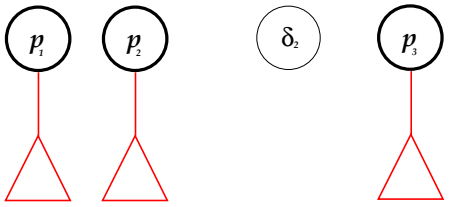


FIGURE 8.3: Program variables bounding.

8.2.2.3 Dual Variables Creation

The code between line 11 and 31 refers to the dual variables creation process. Now, based on the systems declared constraints, one dual variable is created for each constraint. The right hand side (RHS) constant of the constraint is attached to the information within the node representing this variable. This is needed in order to evaluate the gradient of the dual variable. Figure 8.4 illustrates this step.

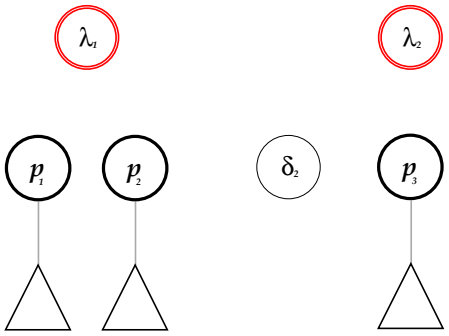


FIGURE 8.4: Dual variables creation.

8.2.2.4 Links Creation

The code between line 17 and 30 refers to the link creation process, based on the information contained within the constraints, a set of links are created. Specifically a link is created for every term in the constraint. This link connects the dual variable related to the constraint under process and the program variable within the term. The link is assigned as value the coefficient of such term. Also the link has information about which variables are connected to each of its ends. Furthermore, for the case of QSP, we know the links can only connect dual variables with program variables. The link creation process is represented by figure 8.5.

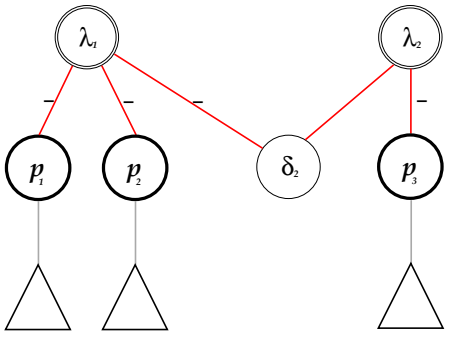


FIGURE 8.5: Links creation.

8.2.2.5 Agents Creation

The code between line 32 and 39 refers to the agent creation process. Now the agents are created. Based on the information they possess a partition of the variable space is done. Each agent has ownership of a subset of program variables and a subset of dual variables which are related to the constraints they are subject to. This information is stored within the node representing each variable, (i.e. who this variable belongs to). This process is depicted in figure 8.6.

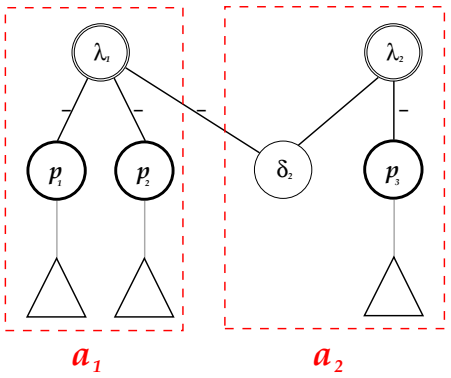


FIGURE 8.6: Agents creation.

8.2.2.6 Non-proper Links Identification

The code between line 40 and 46 refers to the non proper links identification process. Once the agents have been defined, the next step is to identify the links which do not belong to just one agent, these are called *non-proper links*. This is done by looking at the variables at each end of the link. If they belong to different agents then they are of such a type. In this case the link between variables λ_1 and δ_2 is the only one which is shared between different agents as shown in figure 8.7.

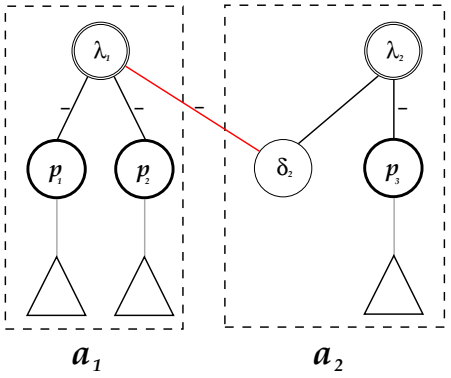


FIGURE 8.7: Non proper links identitification.

8.2.2.7 Final Agentified Graph System by Non-Proper Links Weakening

In the final stage the non-proper links are labelled as *soft* links. This is done in line 42. Therefore the graph shown in figure 8.8 is the final model which is taken as input by the QSP solver. The labeling of these links has the effect, when running the solver, to disregard them when the tree is being traversed up and down. Nevertheless, it serves as a mean to get the value related to the variable at the other end of the link. This is used to compute the gradient of the variable at this end. This process is illustrated in figure 8.8.

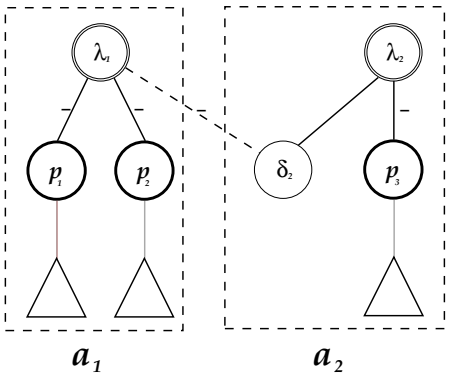


FIGURE 8.8: Final agentified graph system by non proper links weakening.

8.2.3 Building a larger Agent-based Model: A three node example

In this section algorithm 11 is applied to the three node example shown in figure 8.9. This will give us an insight about how an EPM with arbitrary topology is converted into a set of trees. In this system, there are three agents $\mathcal{A} = \{a_1, a_2, a_3\}$, the set of variables is $\mathcal{V} = \{p_1, p_2, p_3, \delta_2, \delta_3\}$, and the set of constraints is $\mathcal{C} = \{(250 - p_1 - \delta_2 - \delta_3 = 0), (200 - p_2 + 2\delta_2 - \delta_3 = 0), (400 - p_3 + 2\delta_3 - \delta_2 = 0), (150 \leq p_1 \leq 600)), (100 \leq p_2 \leq 400), (50 \leq p_3 \leq 200)\}$. Therefore, the subsets of variables each agent has to deal with are $\mathcal{V}_{a_1} = \{p_1\}$, $\mathcal{V}_{a_2} = \{p_2, \delta_2\}$, y $\mathcal{V}_{a_3} = \{p_3, \delta_3\}$. Finally, the set of constraints each agent is subject to are $C_{a_1} = \{(250 - p_1 - \delta_2 - \delta_3 = 0), (150 \leq p_1 \leq 600)\}$, $C_{a_2} = \{(200 - p_2 + 2\delta_2 - \delta_3 = 0), (100 \leq p_2 \leq 400)\}$, $C_{a_3} = \{(400 - p_3 + 2\delta_3 - \delta_2 = 0), (50 \leq p_3 \leq 200)\}$.

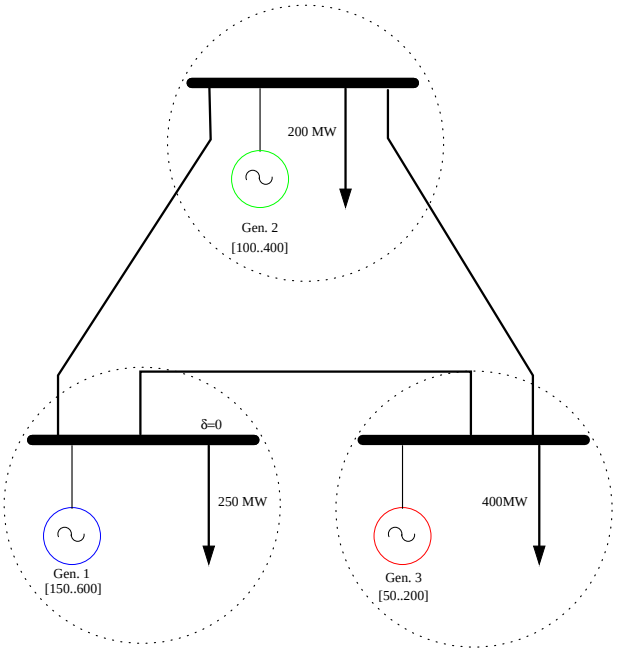


FIGURE 8.9: Three node EPM diagram

8.2.3.1 Program Variables Creation

In this case p_1, p_2, and p_3 are objective variables whereas δ_2 and δ_3 are non-objective variable. This process is depicted in figure 8.10.

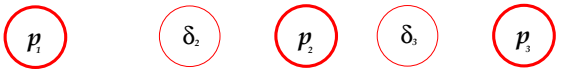


FIGURE 8.10: Program variables creation.

8.2.3.2 Program Variables Bounding

In this step, the slack variables and dual variables needed to handle the bounds for the program variables are created and their interconnections are built. In this case there are three program variables p_1, p_2, and p_3 so a bounding structure has to be built for each of them. This process is described in figure 8.11.

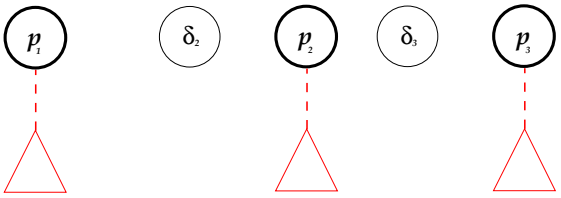


FIGURE 8.11: Program variables bounding.

8.2.3.3 Dual Variables Creation

Now, based on the systems declared constraints, one dual variable is created for each of the three constraints. Figure 8.12 illustrates this step.

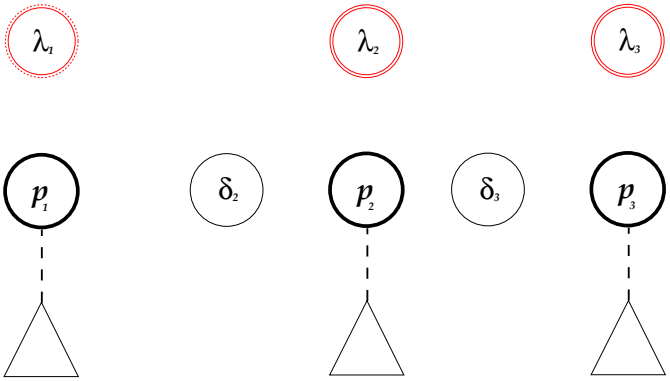


FIGURE 8.12: Dual variables creation.

8.2.3.4 Links Creation

In this case there will be nine links connecting dual variable to program variables. These links are given by the terms the constraints posses. Specifically a link is created for every term in each constraint. The link creation process is represented by figure 8.13.

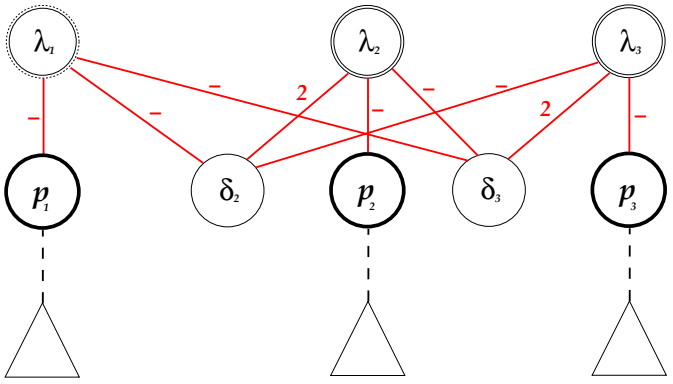


FIGURE 8.13: Links creation.

8.2.3.5 Agents Creation

Now the agents are created. Based on the information they possess a partition of the variable space is done. Each agent has ownership of a subset of program variables and a subset of dual variables which are related to the constraints they are subject to. This information is stored within the node representing each variable, (i.e. who this variable belongs to). This process is depicted in figure 8.14.

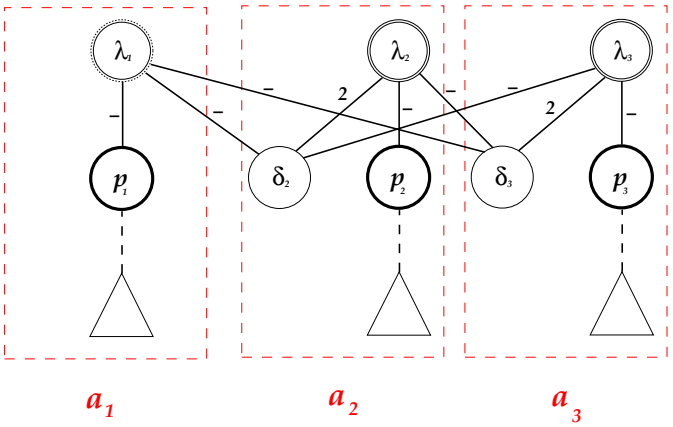


FIGURE 8.14: Agents creation.

8.2.3.6 Non-proper Links Identification

Once the agents have been defined, the next step is to identify the links which do not belong to just one agent, these are called *non-proper links*. This is done by looking at

the variables at each end of the link. If they belong to different agents then they are of such a type. In this case, as opposed to the two-nodes example, there are four links: $\lambda_1 \rightarrow \delta_2$, $\lambda_1 \rightarrow \delta_3$, $\lambda_2 \rightarrow \delta_3$, and $\lambda_3 \rightarrow \delta_2$. These links are shared between different agents, as shown in figure 8.15.

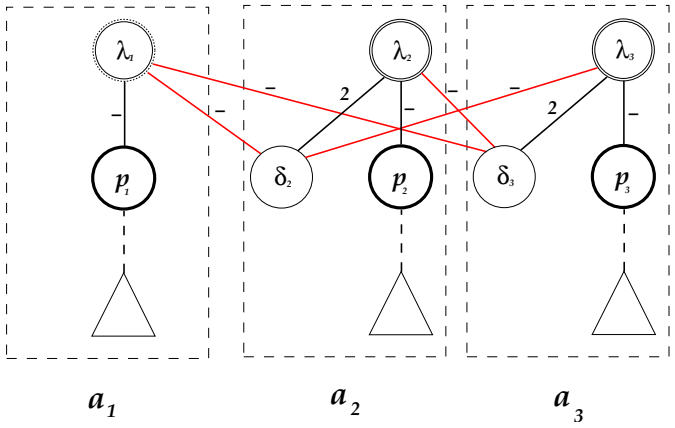


FIGURE 8.15: Non proper links identitification.

8.2.3.7 Final Agentified Graph System by Non-Proper Links Weakening

In the final stage the non-proper links are labelled as *soft* links. Do notice the graph for every agent is a tree, i.e. the graph denoted with solid lines. Therefore the graph shown in figure 8.16 is the final model which is taken as input by the QSP solver. As in the two-nodes example, the labeling of these links has the effect, when running the solver, to disregard them when each tree is being traversed up and down. Nevertheless, it serves as a mean to get the value related to the variable at the other end of the link. This is used to compute the gradient of the variable at this end. This process is illustrated in figure 8.16.

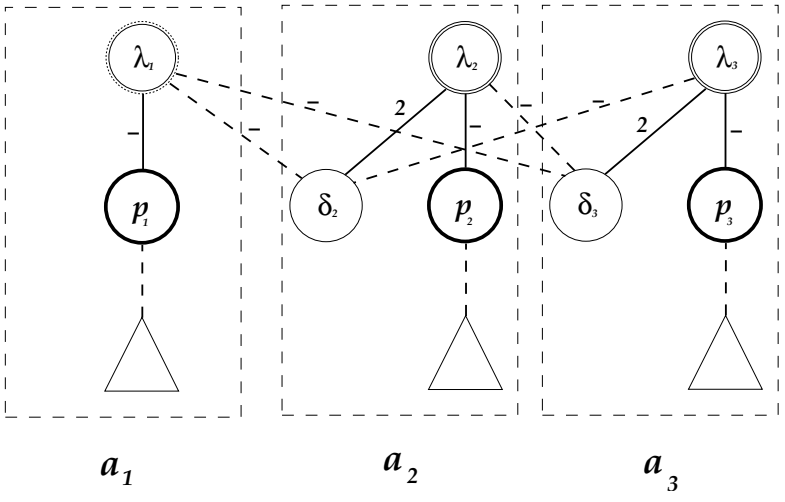


FIGURE 8.16: Final agentified graph system by non proper links weakening.

8.3 Cooperative Intercommunication

This section analyses the intercommunication process is analysed for the case when the decentralised model is implemented using a decentralised approach where agents are represented by different processes in order to solve the model. This analysis is based on the properties of the soft links. The links have a twofold objective. First they allow a decentralised gradient computation. Second they provide the paths to perform the graph reduction. When a link is declared soft, then the second objective is cancelled but the first one remains as was shown in chapter 7.

It is important to underline that there are only links between primal variables and dual variables. This would not be the case if the constraints were not linear or the objective function was not separable. The intercommunication task is done by using the information provided by the soft links. Therefore, this gives us some insight as how it has to be enforced.

Let us suppose we have the soft link l (i.e. $l.connectivityType = SOFT$), whose other characteristics are $l.dualVariable = d$, $l.primalVariable = p$, and $l.value = a$, where d is the dual variable related to constraint c. From algorithm 11, line 37, we derive that d is related to the agent who actually has to fulfill constraint c (i.e. that is the effect of the constraint ownership). Let us refer to this agent as a_i. Therefore, this agent knows he has to fulfill constraint c. However, in order to meet this requirement, it knows it needs the information about the primal variables which are contained within the terms of the constraint. Because l is a soft link, the primal variable p related to this link belongs to some other agent. Let us call this agent a_j. Therefore, at the other side of the link resides agent a_j who owns p.

This agent has a very different perspective about the world. Specifically, it is not aware of constraint c, which is only known by agent a_i, even more it may not be aware agent a_i exists. Because of this it is not able to initiate the intercommunication process with agent a_i. As a result, agent a_i is the only one who can initiate the intercommunication process with agent a_j. However, how can it guarantee agent a_j is willing to share its information (i.e. p)?. The only way this can be done is by setting up a mutual agreement to interchange this information. The only reason why this kind of behavior is preserved is because every agent has a set of constraints to be fulfilled and a set of primal variables. Therefore, it is in their best interest to share their information. Specifically, if agent a_j rejects to share the information with agent a_i then agent a_i would deny its information to agent a_j. As a result they would not be able to compute their gradients and therefore the direction which they take would not be toward the optimum. This would lead to a non global optimal solution. Therefore, we can think of the soft links as intercommunication channels of agreements between the parties which are connected by the soft links which have to be honored.

8.4 Concluding Remarks

In this chapter a methodology to decentralise the graph based on the core agent concepts has been presented. This takes the quadratic separable program specification and creates the graph on which the solution process is applied. This process involves the creation of primal variables and its appropriate bounding subgraph when the variable has limits on its value. Then, the creation of dual variables which are attached to each constraint and, depending on its type, they are labelled as INEQUALITY or EQUALITY. Once the primal and dual variables have been created, the link creation process follows. To this end, a link is created for each term in each constraint. This link connects the dual variable attached to the constraint to the primal variable which is contained within the term. The value for the link is the coefficient which was multiplying the primal variable. At the same time, if the constraint which is actually being processed is an inequality then the link is labelled as CONDITIONAL and PERMANENT otherwise. Then, an ownership property is attached to each one of the variables (i.e. primal and dual), where the owner belongs to the set of agents participating in the system. Finally, the labeling of the links is undertaken based on the ownership property, that the dual and the primal variables attached to this link have. If they belong to the same agent then the link is labeled as HARD, otherwise it is labeled SOFT.

Chapter 9

Conclusions and Future Work

This final chapter concludes the thesis by reviewing its contributions to the field of power markets and by identifying some opportunities for future work. To this end, in Section 9.1, we provide a high-level overview of the techniques we have proposed to address it. Then, in Section 9.2, we discuss in more detail our research contributions. Finally, in Section 9.3, we propose several ways in which this work can be extended in the future.

9.1 Research Summary

This document has introduced the main characteristics for electrical power markets in chapter 2. It has been argued that the electrical power markets must be operated in a decentralised manner. The are two main reasons for this decentralisation: first, the partial information problem derived from the deregulation, as well as the integration of electrical power markets; second, the tendency to highly distributed power generation systems represented by technologies such as microgrids and renewable power generation plants. This effect is also reinforced by the integration of electrical power systems.

To address such a challenge, this thesis has developed a methodology based on a decentralised graph-based solution for systems of linear equations representing the Newton method. The main contribution derived from this thesis is to address QSP optimisation problems with graph techniques. It has been argued that no fill-ins are created in the solution process. This assertion is justified by using just tree-like graphs which have this particular property as shown in section 3.3. Whenever the graph cannot be represented by a tree, a decentralisation process is applied which derives such structures. This is not for free as, by applying the decentralisation process, the system will take more time to find the stationary point. Nevertheless, the need for a central agent in order to clear the market has been eliminated. With this, the value of the strategic information grows

as no agent will want to disclose it to the other agents. This was not the case in the centralised version where everybody has to report this data to the central agent in order to solve the system.

It is important to highlight that this methodology works with QSP problems whose underlying topology have the same characteristics as the one which resulted for the DC-OPF. In this case there was just one equality constraint for each sub-system and therefore the dual variable associated to such constraint could be set as the root of the tree corresponding to that subgraph. As for the inequality constraints corresponding to the transmission system and the limits on the generation units, they can be treated as bounding constraints.

In the following section, we provide a more detailed summary of the approach presented in this document, highlighting the novel contributions we have made to the state of the art.

9.2 Research Contributions

In this thesis, we set out to design a graph-based decentralised framework for electrical power markets. This framework is developed in a gradual and consistent manner. We depart from a basic graph-based representation of a system of linear equations and its solution. Then we, gradually, attach several properties to the links of the graph until its complete decentralisation is achieved. This process is described in the following sections, where we outline the main contributions of this thesis (Sections 9.2.1-9.2.5).

9.2.1 A Graph-based Approach Framework for the Solution of Systems of Linear Equations

In chapter 3 a graph-based model for systems of linear equations has been proposed. Based on these graphs representing a system of linear equations our attention has turned to a subset of systems of linear equations, the symmetric systems of linear equations. The graphs representing the Hessian matrix belong to this class of systems of linear equations. In addition, the kind of graphs this document is dealing with are those that can be represented with a tree. Therefore, our attention was then switched to the tree structured symmetric systems of linear equations. These are very important as they provide us with a means to reduce graphs with no new elements generated at all. As a result the sparsity of the system is completely preserved. Finally a graph-based representation for the Newton step has been proposed as the basic graph model we will use for the power market. This in turn is further analised in chapter 7.

9.2.2 A Graph-based Economical Dispatch Solution

In chapter 4, a solution based on the graph model developed in chapter 3 was presented for the economical dispatch problem with the limits on the generators relaxed. This kind of graphs were not able to deal with the inequality constraints that are needed in order to keep the generators within their bounds. Therefore, that graph-based model had to be extended in order to deal with the actual active constraint set at each iteration. This aspect is addressed by attaching a conditional property to the links of the graph. These conditional links are active only when the right conditions are fulfilled. Specifically, in our approach we only need to decide if parts of the graphs need to be visited or not. This is done by using Karush-Khun-Tucker conditions. This leads to a very efficient solution approach compared with the matrix-based approach, where handling rows and columns is the burden. There, in order to avoid that, normally barriers methods are used which penalize the objective function whenever the solution point tries to go out of the barriers.

9.2.3 A Distributed Model for the DC-OPF

In chapter 5, a distributed algorithm for the DC optimal power flow has been presented. This model was implemented using the auxiliary problem principle. It does not add any overhead to the centralised model, which is replicated by each node. Furthermore, it simplifies the centralised model by letting the different parties cooperate by sharing the constraints of the transmission system.

Out of the analysis done in the previous decentralised model, a graph-based decentralisation approach is proposed in chapter 6. Based on the matrix contained within table 6.1, a new graph-based algorithm has been presented. This algorithm takes advantage of the graph structure to solve the problem. This approach allow us to scale up the system in a very efficient way (i.e. linear in the number of generators, loads, and lines attached to the electrical power system). To this end a graph-based algorithm to solve the OPF, called $gOPF$, has been developed. gOPF is based in two basic algorithms; the $gFOPF$ algorithm and the $gBOPF$ algorithm. $gFOPF$ performs the Gaussian elimination and $gBOPF$ implements the backward substitution on the proposed graph-based representation of the system. These algorithms perform the minimum number of operations as a result of the topological structure of the model. Also, as the sub-graphs derived for each node can be represented using a tree structure, no fill-ins are created.

Summing up, in this part the main contributions are:

The explicit *graph model* derived from the mathematical model allows us to think about the model in a modular fashion. This in turn will prevent us from using general sparse matrix methods such as those in Tinney et al. (1985).

The *order* in which this reduction has to be realized is a consequence of matrix graph analysis, and is inspired in its topological structure.

In order to deal with the decentralisation, the concept of *soft link* and *hard link* have been introduced. The hard link is the link which is regularly used in order to reduce the graphs. On the other hand, the soft link is used as a means to compute the gradient but no reduction process is performed with it.

9.2.4 A Graph-based Decentralised Model

Chapter 7 has presented a novel graph-based representation for the Newton method. Based on this representation different decentralisation approaches have been proposed. The main contribution in this chapter, besides these decentralisation approaches, is the notion of decentralisation degree which this graph provides. This degree goes from a completely decentralised approach where all the links are weaken (i.e. the gradient method), to a full centralised approach where all the links are hard (i.e. the Newton method). Within these two approaches there is a pletora of decentralisation possibilities (i.e. the set of links that will be weakened). However, if the decentralisation process has to be implemented in a systematic way, then some criteria has to be provided in order to achieve it. Two such criteria have been proposed

- *Dual-oriented approach*: This is based on the type of variable and derives a decentralisation scheme which dissects the graph into two layers: one for the primal variables and the other corresponding to the dual variables.
- *Agent-oriented approach*: This scheme is based on the multi-agent paradigm where they are sharing some constraints. This approach derives graphs where the variables and links are decomposed based on an agent membership criterion.

Additionally, we have shown that these graphs provide a direct means to compute the gradient. Therefore, we can claim that these graphs are self contained as no more additional structures are needed in order to solve them.

9.2.5 An Agent-based Model

Finally, chapter 8 has presented an algorithm based on the agency concepts. This algorithm derives the graphs presented along the previous chapters. To this end, the needed ontologies to describe electrical power markets as well as separable quadratic programs were defined. Therefore, the matrix formulation model is not needed in order to obtain these graphs. This was the procedure used in chapters 4 and 6 where the matrix formulation was needed in order to derive the graphs. These graphs can be reformulated

using concepts drawn from the multi-agent systems Wooldridge (2002). They have a goal in mind which is to optimise their operation with just the local information. But they are constrained by several physical constraints which must be preserved at every time. The components derived from the decomposition process can be though as an agent system as they

- are *autonomous*, their decisions are based on their own information as well as the information they gather from their local environment,
- have *social ability*, they intercommunicate their corresponding states (e.g. δ, λ) with their neighbours in order to exchange their actual state so everybody can take better decisions about their future states.
- *act Proactively*, they will be adapting their state accordingly with the changes in the environment always acting as selfish agents, but as the economic theory predicts the "invisible hand" will lead the market to an optimum allocation of resources.
- *cooperate* in order to preserve the transmission constraints below their limits.

9.3 Future work

The work presented in this thesis can be extended to further research lines. These are described in the next paragraphs.

9.3.1 Addressing the AC-OPF

As described in appendix A, the DC-OPF uses a power flow model which is a simplification of the AC-OPF. Therefore, all the second derivatives would not necessarily be linear. To address this aspect the links functionality must be enhanced so they can deal with expressions instead of scalars. To this end, an expression tree would have to be attached to each link which would be evaluated whenever is needed. In fact, in the actual work we use an expression when the bounding constraints are evaluated, but this expression is well known. This evaluation is done in the matrix approach before attempting to solve the Newton step. The difference here is that all this evaluation would be executed in line as the graph is traversed.

9.3.2 Profit Maximisation

The problem addressed until now in this work, assumes a cooperative behaviour as all the agents are cooperating in order to solve the system. Nevertheless, in a competitive

market scenario, all the agents will be trying to maximize their own profit. The constraint space would be the same but the objective function would change. The objective function each generator g would be searching for is given by equation 9.1

$$\max_{p_g,\lambda}(\lambda p - C_g(p_g)) \tag{9.1}$$

Here the objective function itself has a dual variable within its formulation. Furthermore, the objective function is not convex anymore. Therefore Newton's method will lead the search toward a local maximum. The research challenge in this setting is how the graph representation has to be modified in order to solve such formulation.

9.3.3 Unit Commitment Problem

The unit commitment problem determines which generators have to be committed at each stage along some predetermined time horizon. This would involve the solution of the DCOPF at each stage. However, there are some intertemporal constraints which couple the variables from one point in time to the next. The optimisation problem in this case is given by the social welfare maximisation along the period of time considered. Two of the questions where research has to be done are: What is the graph topology for this problem? and How the decentralisation of the graph is to be addressed?

9.3.4 Addressing Non-separable Convex Quadratic Problems

With the methodology developed in this work, links were only created between primal variables and dual variables. This kind of topoloogy has its roots at the separability of the objective function. These functions as well as the constraints were convex. Nevertheless, there are other quadratic problems which are non separable but convex. Therefore, this methodology could be applied to this kind of systems with some modifications to the way the problems are defined as well as processed.

Appendices

Appendix A

DC Power Flow Derivation

Let us consider the node i, $i \in \mathbb{N}$, displayed in figure A.1 where the symbol j represent the imaginary operator. In this model $|\mathbb{G}_i|$ generators producing $p_{ig} + jq_{ig}$ MV, $|\mathbb{L}_i|$ variable loads consuming $p_{il} + jq_{il}$ MV and a fixed load consuming $P_i + jQ_i$ MV are attached to node i where $g \in \mathbb{G}_i$ and $l \in \mathbb{L}_i$. Out of this node there are $|\Gamma_i|$ interconnections to some other nodes using lines whose admittance is $g_{ik} + jb_{ik}$ where $k \in \Gamma_i$.

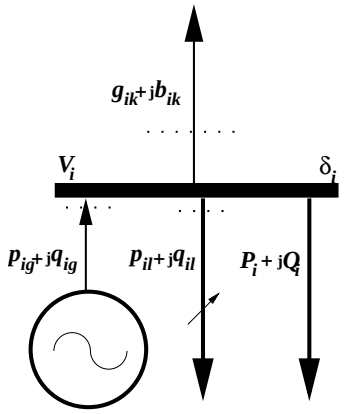


FIGURE A.1: Node model for the AC Optimal Power Flow

Writing the flow equillibrium equations for this node leads to equation A.1 for the active power and equation A.2 for the reactive power.

$$V_i \sum_{k \in \Gamma_i} \left[V_k [g_{ik} \cos(\delta_i - \delta_k) + b_{ik} \sin(\delta_i - \delta_k)] \right] - \sum_{g \in \mathbb{G}_i} p_g + \sum_{l \in \mathbb{L}_i} p_l + P_i = 0 \tag{A.1}$$

$$V_i \sum_{k \in \Gamma_i} \left[V_k [g_{ik} \sin(\delta_i - \delta_k) - b_{ik} \cos(\delta_i - \delta_k)] \right] - \sum_{g \in \mathbb{G}_i} q_g + \sum_{l \in \mathbb{L}_i} q_l + Q_i = 0 \tag{A.2}$$

In the DC-OPF study only the active power is under interest. Reactive Power is considered as an ancilliary service which has to be provided by the ISO and its cost must

be socialized among the market players. Therefore equation A.2 is disregarded. On the other hand three assumptions are introduced

- A plain voltage is assumed i.e. 1 pu on every node,
- The resistance compared with the reactance is negligible i.e. $r_{ik} \ll x_{ik}$, and
- The difference between δ_i and δ_k is very small.

The first assumption rules out V_i and V_k from equation A.1. The second assumption strips the term $g_{ik}\cos(\delta_i - \delta_k)$ off equation A.1. Applying these two assumptions equation A.1 can be expressed using equation A.3

$$\sum_{k\in\Gamma_i}[b_{ik}\sin(\delta_i - \delta_k)] - \sum_{g\in\mathbb{G}_i} p_g + \sum_{l\in\mathbb{L}_i} p_l + P_i = 0 \tag{A.3}$$

Finally, based on the third assumption as $\delta_i \approx \delta_k$ a final assumption is derived as shown by equation A.4.

$$\sin(\delta_i - \delta_k) \approx \delta_i - \delta_k \tag{A.4}$$

Therefore equation A.3 is finally reduced into equation A.5

$$\sum_{k\in\Gamma_i}[b_{ik}(\delta_i - \delta_k)] - \sum_{g\in\mathbb{G}_i} p_g + \sum_{l\in\mathbb{L}_i} p_l + P_i = 0 \tag{A.5}$$

This is the expression used in the DC Optimal Power Flow. A final remark on the notation used throughout this document. The terms which represent produced power are denoted with p and the terms representing consumed power or loads are denoted as q. Therefore in equation A.5, the term p_l is replaced by q_l and P_i is denoted as Q_i. This leads to an equivalent expression given by equation A.6 such as the one used in equation 2.1.

$$\sum_{k\in\Gamma_i}[b_{ik}(\delta_i - \delta_k)] - \sum_{g\in\mathbb{G}_i} p_g + \sum_{l\in\mathbb{L}_i} q_l + Q_i = 0 \tag{A.6}$$

Appendix B

Electrical Power Market DTD

```
<?xml version='1.0' encoding='UTF-8'?>

<!--   Specifies an Electrical Power Market which comprises

        EPMId            - The Id for the EPM
        zones            - Composed by a separable quadratic function such as
        interconnections - These are the lines which connect the diferent zones
        setup            - Additional information to set up the system
-->

<!ELEMENT EPM (EPMId, zones, interconnections?, setup)>
    <!ELEMENT EPMId (#PCDATA) >
    <!ELEMENT zones (zone+)>
        <!-- Defines a zone within the EPM. Its fields are

            zoneId - The zone Id
            buses  - The set of buses located within this zone
            lines  - The set of lines within this zone

        -->
        <!ELEMENT zone (zoneId, buses, lines)>
            <!ELEMENT zoneId (#PCDATA) >
            <!ELEMENT buses (bus+)>
                <!-- Defines an electrical bus and the components attached to it.

                busId           - The bus Id
                delta           - The electrical angle for this bus
                lambda          - The locational marginal price (LMP) at this bus
                location        - The coordinates where this bus is located
                generatedPower  - The power which is being injected in this bus
                fixedLoad       - The non-elastic load attached to this bus (if there is any)
                generator       - The set of generator attached to the bus,
                elasticLoad     - The set of elastic loads

                -->
                <!ELEMENT bus (busId, delta, lambda, location, generatedPower,
                               fixedLoad?, (generator|elasticLoad)*)>
                    <!ELEMENT busId (#PCDATA)>
                    <!ELEMENT delta (#PCDATA)>
                    <!ELEMENT lambda (#PCDATA)>
                    <!ELEMENT location (xLocation, yLocation)>
                        <!ELEMENT xLocation (#PCDATA)>
                        <!ELEMENT yLocation (#PCDATA)>
                    <!ELEMENT generatedPower (#PCDATA)>
                    <!ELEMENT fixedLoad (#PCDATA)>
                    <!-- Defines the atributes of a generator

                      genId           - Generator Id,
                      genType         - The type of generator,
                      actualPower     - The power this generator is actually producing,
                      minimumPower    - The minimum power this generator has to produce,
                      maxumumPower    - The maximum power this generator can produce,
```

Continue in next page

Continued from previous page

```
                quadraticModel  - The mathematical model for the production
                                  function of this generator

                                                                        -->
            <!ELEMENT generator (genId, genType, actualPower, minimumPower,
                                 maximumPower, quadraticModel)>
                <!ELEMENT genId (#PCDATA)>
                <!ELEMENT genType (#PCDATA)>
                <!ELEMENT actualPower (#PCDATA)>
                <!ELEMENT minimumPower (#PCDATA)>
                <!ELEMENT maximumPower (#PCDATA)>
                <!ELEMENT quadraticModel (alphaCoefficient, betaCoefficient,
                                          gammaCoefficient)>
                    <!ELEMENT alphaCoefficient (#PCDATA)>
                    <!ELEMENT betaCoefficient (#PCDATA)>
                    <!ELEMENT gammaCoefficient (#PCDATA)>
            <!--- Defines the atributes of an eleastic load

                loadId              - Load Id,
                actualPowerFlow     - The power this generator is actually producing,
                minimumPower                    - The minimum power this generator has to produce,
                maxumumPower        - The maximum power this generator can produce,
                concaveModel        - The mathematical model for the benefit
                                      function of this load

                                                                        -->
            <!ELEMENT elasticLoad (loadId, actualPower, minimumPower,
                                   maximumPower, concaveModel)>
                <!ELEMENT loadId (#PCDATA)>
                <!--- Defines the atributes for the concave model of the load

                   betaCoefficient:                    The linear coefficient,
                   gammaCoefficient:             The quadratic coefficient,

                -->
                <!ELEMENT concaveModel (betaCoefficient, gammaCoefficient)>

        <!ELEMENT lines (line*)>
            <!--- Defines the atributes of a transmission line

                lineId              - Line Id,
                actualPowerFlow     - The power flowing through this line,
                maxPowerFlow        - The maximum power flow this line can handle,
                fromBus             - The bus where the power flow is coming from,
                toBus               - The bus where the power flow is going to,
                resistance          - This line resistance,
                reactance           - This line reactance

                                                                        -->
            <!ELEMENT line (lineId, actualPowerFlow, maxPowerFlow, fromBus,
                            toBus, resistance, reactance)>
                <!ELEMENT lineId (#PCDATA)>
                <!ELEMENT actualPowerFlow (#PCDATA)>
                <!ELEMENT maxPowerFlow (#PCDATA)>
                <!ELEMENT fromBus (#PCDATA)>
                <!ELEMENT toBus (#PCDATA)>
                <!ELEMENT resistance (#PCDATA)>
                <!ELEMENT reactance (#PCDATA)>
    <!ELEMENT interconnections (interconnection+)>
        <!ELEMENT interconnection (lineId, actualPowerFlow, maxPowerFlow, fromBus,
                                   toBus, resistance, reactance)>
    <!ELEMENT setup (mapFileName, EPMDescription )>
        <!ELEMENT mapFileName (#PCDATA)>
        <!ELEMENT EPMDescription (#PCDATA) >
```

LISTING B.1: DTD to describe Electrical Power Markets

Appendix C

Listing for the one node EPM model

In this part the XML specification model for the one node case is described in listing C.1 whose corrresponding QSP listing is given in listing F.1 in appendix F.

```
<?xml version="1.0"?>
<!DOCTYPE eps SYSTEM "EPM.dtd">
<EPM>
  <EPMId>OneNodeExampleOnScotland</EPMId>
  <zones>
    <zone>
      <zoneId>Zone North </zoneId>
      <buses>
        <bus>
          <busId> Edinburgh </busId>
          <delta> 1.0 </delta>
          <lambda> 1.0 </lambda>
          <location>
            <xLocation> 0.5213483146067416 </xLocation>
            <yLocation> 0.3422018348623853 </yLocation>
          </location>
          <generatedPower> 0.0 </generatedPower>
          <fixedLoad> 850.0 </fixedLoad>
          <generator>
            <genId> G-Edinburgh-1 </genId>
            <genType> Pelton </genType>
            <actualPower> 1.0 </actualPower>
            <minimumPower> 150.0 </minimumPower>
            <maximumPower> 600.0 </maximumPower>
            <quadraticModel>
              <alphaCoefficient> 510.0 </alphaCoefficient>
              <betaCoefficient> 7.92 </betaCoefficient>
              <gammaCoefficient> 0.001562 </gammaCoefficient>
            </quadraticModel>
          </generator>
          <generator>
            <genId> G-Edinburgh-2 </genId>
            <genType> Pelton </genType>
            <actualPower> 1.0 </actualPower>
            <minimumPower> 100.0 </minimumPower>
            <maximumPower> 400.0 </maximumPower>
            <quadraticModel>
              <alphaCoefficient> 310.0 </alphaCoefficient>
              <betaCoefficient> 7.85 </betaCoefficient>
              <gammaCoefficient> 0.00194 </gammaCoefficient>
            </quadraticModel>
          </generator>
          <generator>
            <genId> G-Edinburgh-3 </genId>
```

Continue in next page

Continued from previous page

```
            <genType> Pelton </genType>
            <actualPower> 1.0 </actualPower>
            <minimumPower> 50.0 </minimumPower>
            <maximumPower> 250.0 </maximumPower>
            <quadraticModel>
              <alphaCoefficient> 78.0 </alphaCoefficient>
              <betaCoefficient> 7.97 </betaCoefficient>
              <gammaCoefficient> 0.00482 </gammaCoefficient>
            </quadraticModel>
          </generator>
        </bus>
      </buses>
    </zone>
  </zones>
</EPM>
```

LISTING C.1: XML description for the one node EPMS example based on the EPM DTD in listing B.1

Appendix D

Listing for the two nodes EPM model

In this part the XML specification model for the two nodes case is described in listing D.1 whose corrresponding QSP listing is given in listing G.1 in appendix G.

```
<?xml version="1.0"?>
<!DOCTYPE eps SYSTEM "EPM.dtd">
<EPM>
  <EPMId>OneNodeExampleOnScotland</EPMId>
  <zones>
    <zone>
      <zoneId>Zone North </zoneId>
      <buses>
        <bus>
          <busId> Edinburgh </busId>
          <delta> 1.0 </delta>
          <lambda> 1.0 </lambda>
          <location>
            <xLocation> 0.5213483146067416 </xLocation>
            <yLocation> 0.3422018348623853 </yLocation>
          </location>
          <generatedPower> 0.0 </generatedPower>
          <fixedLoad> 850.0 </fixedLoad>
          <generator>
            <genId> G-Edinburgh-1 </genId>
            <genType> Pelton </genType>
            <actualPower> 1.0 </actualPower>
            <minimumPower> 150.0 </minimumPower>
            <maximumPower> 600.0 </maximumPower>
            <quadraticModel>
              <alphaCoefficient> 510.0 </alphaCoefficient>
              <betaCoefficient> 7.92 </betaCoefficient>
              <gammaCoefficient> 0.001562 </gammaCoefficient>
            </quadraticModel>
          </generator>
          <generator>
            <genId> G-Edinburgh-2 </genId>
            <genType> Pelton </genType>
            <actualPower> 1.0 </actualPower>
            <minimumPower> 100.0 </minimumPower>
            <maximumPower> 400.0 </maximumPower>
            <quadraticModel>
              <alphaCoefficient> 310.0 </alphaCoefficient>
              <betaCoefficient> 7.85 </betaCoefficient>
              <gammaCoefficient> 0.00194 </gammaCoefficient>
            </quadraticModel>
          </generator>
          <generator>
            <genId> G-Edinburgh-3 </genId>
```

Continue in next page

Continued from previous page

```
            <genType> Pelton </genType>
            <actualPower> 1.0 </actualPower>
             <minimumPower> 50.0 </minimumPower>
             <maximumPower> 250.0 </maximumPower>
             <quadraticModel>
               <alphaCoefficient> 78.0 </alphaCoefficient>
               <betaCoefficient> 7.97 </betaCoefficient>
               <gammaCoefficient> 0.00482 </gammaCoefficient>
             </quadraticModel>
           </generator>
         </bus>
       </buses>
     </zone>
   </zones>
</EPM>
```

LISTING D.1: XML description for the two node EPMS example based on the DTD in listing B.1

Appendix E

Quadratic Separable Programming DTD

```
<?xml version='1.0' encoding='UTF-8'?>

<!---   Specifies a Quadratic separable Program whose form is
        Objective Function: Composed by a separable quadratic function such as

                            sum(f(Xi))

                            where each separated function has the form:

                                f(Xi)=ALPHAi+BETAi*Xi+GAMMAi*Xi^2

                            each Xi is called an objectiveVariable

        Constraints:    There are several kinds
                            Equality Constraints
                                sum(Ci*Xi)+Y = RHS
                            Inequality Constraints
                                sum(Ci*Xi)+Y <= RHS
                            Bound Constraints
                                lowerLimit<=Xi
                                Xi<=upperLimit
                        Some Ci's can be zero so that term dissapears
                        Y is called a nonObjectiveVariable

    Elements:
            objectiveVariable+
            nonObjectiveVariable*
            constraint+
            agent*
-->

<!ELEMENT QSP (objectiveVariable+, nonObjectiveVariable*, constraint+, agent*)>

<!---   Specifies variables which appear within the objective function
    Elements:
            varNname-   The name for the variable,
            alpha   -   The constant coefficient,
            beta    -   The linear coefficient,
            gamma   -   The quadratic coefficient,
            lowerLimit  -   The lower boud for this variable,
            upperLimit  -   The upper bound for this variable
-->
<!ELEMENT objectiveVariable (varName, xInit?, alpha?, beta?, gamma, upperLimit?,
                             lowerLimit?)>

<!---   Specifies a NonObjectiveVariable which appear only in the constraints
    Fieds:
            consName    -   The name for the variable,
            xInit?      -   Variable initial value,
```

Continue in next page

Continued from previous page

```
                lowerLimit  -   The lower boud for this variable,
                upperLimit  -   The upper bound for this variable
-->
<!ELEMENT nonObjectiveVariable (varName, xInit?, lowerLimit?, upperLimit?)>

<!---   Specifies a constraint ...+term+...([=]|[<=])RHS
        Fieds:
                consName    -   The name for the constraint,
                type        -   The constarint type i.e. EQUALITY|INEQUALITY,
                term        -   The constraint is formed by one or more terms,
                RHS         -   The constraint Right Hand Side
-->
<!ELEMENT constraint (consName, type, term+, RHS)+>

<!---   Specifies a term coefficient*varName
        Fieds:
                coefficient-   The term's coefficient name,
                varName     -   The term variable constraint type
                                  i.e. EQUALITY|INEQUALITY,
                term        -   The constraint is formed by one or more terms,
                RHS         -   The constraint Right Hand Side
-->
<!ELEMENT term (coefficient, varName)>

<!---   Specifies an agent who has variables and constraints
        Fieds:
                coefficient-   The term coefficientname for the constraint,
                varName     -   The term variableconstrint type
                                  i.e. EQUALITY|INEQUALITY,
                term        -   The constraint is formed by one or more terms,
                RHS         -   The constraint Right Hand Side
-->
<!ELEMENT agent (agentName, varName+, consName+)>

<!-- Name of a variable <String> -->
<!ELEMENT varName (#PCDATA)>

<!-- Initial value <double> -->
<!ELEMENT xInit (#PCDATA)>

<!-- Constant coefficient in a quadratic function <double> -->
<!ELEMENT alpha (#PCDATA)>

<!-- Linear coefficient in a quadratic function <double> -->
<!ELEMENT beta (#PCDATA)>

<!-- Quadratic coefficient in a quadratic function <double> -->
<!ELEMENT gamma (#PCDATA)>

<!-- Lower limit for a variable <double> -->
<!ELEMENT lowerLimit (#PCDATA)>

<!-- Upper limit for a variable <double> -->
<!ELEMENT upperLimit (#PCDATA)>

<!-- Name for a constraint <String> -->
<!ELEMENT consName (#PCDATA)>

<!-- Constraint type <INEQUALITY | EQUALITY> -->
<!ELEMENT type (#PCDATA)>

<!-- A number <double> -->
<!ELEMENT coefficient (#PCDATA)>

<!-- The constant in the Right Hand Side of a constraint <double> -->
<!ELEMENT RHS (#PCDATA)>

<!-- Name for an Agent <String> -->
<!ELEMENT agentName (#PCDATA)>
```

LISTING E.1: DTD to describe Quadratic Separable Programs

Appendix F

Listing for the one node QSP model

In this part the XML specification model for the one node case is described in listing F.1 whose corrresponding EPM listing is given in listing C.1 in appendix C.

```
<?xml version="1.0"?>
<!DOCTYPE QSP SYSTEM "QSP.dtd">
<OneNodeExampleOnScotland-QSP>
    <objectiveVariable>
        <varName> power.G-Edinburgh-1 </varName>
        <alpha> 459.0 </alpha>
        <beta> 6.48 </beta>
        <gamma> 0.00128 </gamma>
        <lowerLimit> 150.0 </lowerLimit>
        <upperLimit> 600.0 </upperLimit>
    </objectiveVariable>
    <objectiveVariable>
        <varName> power.G-Edinburgh-2 </varName>
        <alpha> 310.0 </alpha>
        <beta> 7.85 </beta>
        <gamma> 0.00194 </gamma>
        <lowerLimit> 100.0 </lowerLimit>
        <upperLimit> 400.0 </upperLimit>
    </objectiveVariable>
    <objectiveVariable>
        <varName> power.G-Edinburgh-3 </varName>
        <alpha> 78.0 </alpha>
        <beta> 7.97 </beta>
        <gamma> 0.00482 </gamma>
        <lowerLimit> 50.0 </lowerLimit>
        <upperLimit> 250.0 </upperLimit>
    </objectiveVariable>
    <constraint>
        <consName> powerBalance.Edinburgh</consName>
        <type> EQUALITY </type>
        <term>
            <varName> power.G-Edinburgh-1 </varName>
            <coefficient> 1 </coefficient>
        </term>
        <term>
            <varName> power.G-Edinburgh-2 </varName>
            <coefficient> 1 </coefficient>
        </term>
        <term>
            <varName> power.G-Edinburgh-3 </varName>
            <coefficient> 1 </coefficient>
        </term>
        <RHS> 850.0 </RHS>
    </constraint>
```

Continue in next page

Continued from previous page

```
    <agent>
        <agentName> A. Edinburgh </agentName>
        <varName> power.G-Edinburgh-1 </varName>
        <varName> power.G-Edinburgh-2 </varName>
        <varName> power.G-Edinburgh-3 </varName>
        <consName> powerBalance.Edinburgh </consName>
     </agent>
</OneNodeExampleOnScotland-QSP>
```

LISTING F.1: XML description for the one node QSP example based on the DTD in listing E.1

Appendix G

Listing for the two nodes QSP model

In this part the XML specification model for the one node case is described in listing G.1 whose corrresponding EPM listing is given in listing D.1 in appendix D.

```
<?xml version="1.0"?>
<!DOCTYPE QSP SYSTEM "QSP.dtd">
<TwoNodesExampleOnScotland-QSP>
    <objectiveVariable>
        <varName> power.G-Edinburgh-1 </varName>
        <alpha> 459.0 </alpha>
        <beta> 6.48 </beta>
        <gamma> 0.00128 </gamma>
        <lowerLimit> 150.0 </lowerLimit>
        <upperLimit> 600.0 </upperLimit>
    </objectiveVariable>
    <objectiveVariable>
        <varName> power.G-Edinburgh-2 </varName>
        <alpha> 310.0 </alpha>
        <beta> 7.85 </beta>
        <gamma> 0.00194 </gamma>
        <lowerLimit> 100.0 </lowerLimit>
        <upperLimit> 400.0 </upperLimit>
    </objectiveVariable>
    <objectiveVariable>
        <varName> power.G-Glasgow-1 </varName>
        <alpha> 78.0 </alpha>
        <beta> 7.97 </beta>
        <gamma> 0.00482 </gamma>
        <lowerLimit> 50.0 </lowerLimit>
        <upperLimit> 250.0 </upperLimit>
    </objectiveVariable>
    <nonObjectiveVariable>
        <varName> delta.Glasgow </varName>
    </nonObjectiveVariable>
    <constraint>
        <consName> powerBalance.Edinburgh</consName>
        <type> EQUALITY </type>
        <term>
            <varName> power.G-Edinburgh-1 </varName>
            <coefficient> 1 </coefficient>
        </term>
        <term>
            <varName> power.G-Edinburgh-2 </varName>
            <coefficient> 1 </coefficient>
        </term>
        <term>
            <varName> delta.Glasgow </varName>
            <coefficient> -1.0 </coefficient>
        </term>
```

Continue in next page

Continued from previous page

```
        <RHS> 450.0 </RHS>
    </constraint>
    <constraint>
        <consName> powerBalance.Glasgow</consName>
        <type> EQUALITY </type>
        <term>
            <varName> power.G-Glasgow-1 </varName>
            <coefficient> 1 </coefficient>
        </term>
        <term>
            <varName> delta.Glasgow </varName>
            <coefficient> 1.0 </coefficient>
        </term>
        <RHS> 400.0 </RHS>
    </constraint>
    <agent>
        <agentName> A.Edinburgh </agentName>
        <varName> power.G-Edinburgh-1 </varName>
        <varName> power.G-Edinburgh-2 </varName>
        <consName> powerBalance.Edinburgh </consName>
    </agent>
    <agent>
        <agentName> A.Glasgow </agentName>
        <varName> power.G-Glasgow-1 </varName>
        <varName> delta.Glasgow </varName>
        <consName> powerBalance.Glasgow </consName>
    </agent>
</TwoNodesExampleOnScotland-QSP>
```

Listing G.1: XML description for the two nodes QSP example based on the DTD in listing E.1

Bibliography

E. Acha, C.R. Fuerte-Esquivel, H. Ambriz-Perez, and C. Angeles-Camacho. *FACTS Modelling and Simulation in Power Networks*. John Wiley & Sons, Ltd, England, 2004.

J. A. Aguado, V. H. Quintana, and A. J. Conejo. Optimal power flows of interconnected power systems. *IEEE Power Engineering Society Summer*, 2:814–819, 1999.

J.A. Aguado and V. H. Quintana. Inter-utilities power-exchange coordination: A market-oriented approach. *IEEE Transaction on Power Systems*, 16(3):513–519, August 2001.

A.G. Bakirtzis, P. N. Biskas, N.I. Macheras, and N.K. Pasialis. A decentralized implementation of dc optimal power flow on a network of computers. *IEEE Transaction on Power Systems*, 20(1):25–33, February 2000.

Anastasios G. Bakirtzis and Pandelis N. Biskas. A decentralized solution of the dc-opf of interconnected power systems. *IEEE Transaction on Power Systems*, 18(3): 1007–1013, August 2003.

P.S. Bath. The electricity market place - the pool. *IEE Power System Control and Management Conference*, pages 16–18, April 2006.

Anne Berry and Pinar Heggernes. The minimum degree heuristic and the minimal triangulation process. In *Proceedings of WG 2003*, pages 58–70. Springer Verlag, 2003.

W. Bialek and Q. Zhou. Approximate model of european interconnected system as a benchmark systemto study effects of cross-border trades. *IEEE Transaction on Power Systems*, 20(2):782–788, May 2005.

G.J. Chen, K.K. Li, T.S. Chung, and G.Q. Tang. An efficient two-stage loa flow method for meshed distribution networks. In *Proceedings of the 5th Conference on Advances in Power System, Control, Operation and Management, APSCOM 2000*, pages 537–542, October 2000.

Guy Cohen. Optimization by decomposition and coordination: A unified approach. *IEEE Transactions on Automatic Control*, 2(2):222–232, 1978.

Antonio J. Conejo and Jose A. Aguado. Multi-area coordinated decentralized dc optimal power flow. *IEEE Transaction on Power Systems*, 13(4):1272–1278, November 1998.

Consentec. Analysis of cross-border congestion management methods for the eu internal electricity market. Technical study report, Consentec and Frontier Economics Limited, Aachen, Deutchland, June 2004.

J. Contreras and F.F. Wu. Coalition formation in transmission expansion planning. *IEEE Transaction on Power Systems*, (3):1144–1152, August 1999.

Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Cliff Stein. *Introduction to Algorithms*. MIT Press, 2001.

D. Das, H. S. Nagi, and D. P. Kothari. Novel method for solving radial distribution networks. *IEE Proceedings on Generation, Transmission and Distribution*, 141(4): 291–298, July 1994.

Alexander Eydeland and Krzysztof Wolyniec. *Energy and Power Risk Management - New Developments in Modeling, Pricing and Hedging*. John Wiley & Sons, Inc., Hoboken, New Jersey, 2003.

John R. Gilbert, Cleve Moler, and Robert Schreiber. Sparse matrices in matlab: design and implementation. *SIAM J. Matrix Anal. Appl.*, 13(1):333–356, 1992. ISSN 0895-4798. doi: http://dx.doi.org/10.1137/0613024.

Richard J. Gilbert and Edward P. Kahn. *International Comparisons of Electricity Regulation*. Cambridge University Press, 1996.

S. K. Goswami and S.K. Basu. Direct solution of distribution systems. *IEE Proceedings on Generation, Transmission and Distribution*, 138:78–85, 1991.

Richard Green. Did english generators play cournot? capacity withholding in the electricity pool. CMI Working Paper 41, 2004.

Richard Green. Electricity deregulation in oecd (organization for economic cooperationand development) countries. *Energy*, 31(6):769–787, May 2006.

Etan Z. Gumerman, Ranjit R. Bharvirkar, Kristina H. LaCommare, and Chris Marnay. Evaluation framework and tools for distributed energy resources. Technical Report LBNL-52079, ERNEST ORLANDO LAWRENCE BERKELEY NATIONAL LABORATORY, Lawrence Berkeley National Laboratory, February 2003.

R. Haas and H. Auer. The prerequisites for effective competition in restructured wholesale electricitymarkets. *Energy*, 31(6):857–864, May 2006.

X.S. Han, H.B. Gooi, and D.S. Kirschen. Dynamic economic dispatch: feasible and optimal solutions. In *Power Engineering Society Summer Meeting, 2001. IEEE*, volume 3, page 1704vol.3, 15-19 July 2001. doi: 10.1109/PESS.2001.970332.

S.A. Harp, S. Bruce, F. Wollemberg, and T. Samad. Sepia a simulator for electric power industry agents. *IEEE Control Systems Magazine*, 20(4):53–69, August 2000.

Minghua He, Nicholas R. Jennings, and Ho-Fung Leung. On agent-mediated electronic commerce. *IEEE Transactions on Knowledge and Data Engineering*, 15(4):985–1003, 2003. ISSN 1041-4347. doi: http://doi.ieeecomputersociety.org/10.1109/TKDE.2003.1209014.

W. Hogan. Contracts networks for electricity power transmission. *Harvard Electricity Policy Group, John F. Kennedy School of Government,Harvard University, Cambridge, Mass.*, September 1992.

W. Hogan. Competitive electricity model. *Harvard Electricity Policy Group, John F. Kennedy School of Government,Harvard University, Cambridge, Mass.*, October 1993.

M. Huneault and F.D. Galiana. A survey of the optimal power flow literature. *IEEE Transaction on Power Systems*, 6(2), May 1991.

N. Jennings and M. Wooldridge. Intelligent agents: Theory and practice. *Knowledge Engineering Review*, 10(2):115–152, 1995.

P.L. Joskow and R. Schmalensee. *Markets for Power An Analysis of Electrical Utility Deregulation*. The MIT Press, Cambridge, Mass., 1983.

B. H. Kim and R. Baldwick. Coarse-grained distributed optimal power flow. *IEEE Transaction on Power Systems*, 12(2):932–939, May 1997.

Paul Klemperer. *Auctions: Theory and Practice*. Princeton University Press, 2004.

M. Klush and O. Shehory. Coalition formation among rational agents. In W. van de Velde and J. Perran (Hrsg), editors, *Proc. 7nth European Workshop on Modelling Autonomous Agents in a Multi-Agent World, MAAMAW-96*, volume 1038 of *Lecture Notes in Artificial Intelligence, LNAI Series*, pages 204–217. Springer-Verlag, 1996.

V. Krish and V.C. Ramesh. Intelligent agents for negotiation in market games, part 1: Model. *IEEE Transaction on Power Systems*, (3):1103–1108, August 1998a.

V. Krish and V.C. Ramesh. Intelligent agents for negotiation in market games, part 2: Application. *IEEE Transaction on Power Systems*, (3):1109–1114, August 1998b.

Y.C. Lam and F.F. Wu. Simulating electricity markets with java. *IEEE Transaction on Power Systems*, (3):1010–1020, August 1999.

Robert H. Lasseter. Microgrids. In IEEE, editor, *Power Engineering Society Winter Meeting, 2002. IEEE*, volume 1, pages 305–308, 2002.

Harry M. Markowitz. The elimination form of the inverse and its application to linear programming. *Management Science*, 3(3):255–269, 1957. ISSN 00251909. URL `http://www.jstor.org/stable/2627454`.

S.F. Mekhamer, S.A. Soliman, M.A. Moustafa, and M.E. El-Hawary. Load flow solution of radial distribution feeders: A new contribution. *Electrical power and Energy Systems*, 24:70–707, 2002.

J. Nocedal and S. Wright. *Numerical Optimization.* Springer Verlag, 2nd edition, 2006.

Ofgem. Rises in pool prices in july: A decision document. Technical report, Office of Gas and Electricity Markets, October 1999.

Thomas J. Overbye, Xu Cheng, and Yan Sun. A comparison of the ac and dc power flow models for lmp calculations. In *HICSS '04: Proceedings of the Proceedings of the 37th Annual Hawaii InternationalConference on System Sciences (HICSS'04) - Track 2*, page 20047.1, Washington, DC, USA, 2004. IEEE Computer Society. ISBN 0-7695-2056-1.

K. Purchala, L. Meeus, D. Van Dommelen, and R. Belmans. Usefulness of dc power flow for active power flow analysis. *Power Engineering Society General Meeting, 2005. IEEE*, pages 454–459 Vol. 1, June 2005. doi: 10.1109/PES.2005.1489581.

K. Purchala, D. Van Hertem, J. Verboomen, R. Belmans, and W.L. Kling. Usefulness of dc power flow for active power flow analysis with flow controlling devices. In *The 8th IEE International Conference on AC and DC Power Transmission, 2006. ACDC 2006*, pages 58–62, March 2006.

T. Rahwan and N. R. Jennings. An algorithm for distributing coalitional value calculations among cooperating agents. *Artificial Intelligence Journal*, 171, 2007.

N.S. Rau. *Optimization Principles - Practical Applications to the Operation and markets of the Electric Power Industry.* IEEE Press, Wiley Inter-Science, USA, 2003.

Richard E. Rosenthal. Gams a users guide. GAMS Development Corporation, Washington, DC, USA, 2007.

Hugh Rudnick. Chile: Pioneer in deregulation of the electric power sector. *IEEE Power Engineering Review*, pages 28–30, June 1994.

F. C. Schweppe. *Spot Pricing of Electricity.* Kuwer Academic Publishers, Boston, USA, 1998.

Mohammad Shahidehpour, Hatim Yamin, and Zuyi Li. *Market Operations in Electric power Systems - Forecasting, Scheduling, and Risk Management.* John Wiley & Sons, Inc., 2002.

W. Stoft. *Power System Economics: Designing Markets for Electricity.* IEEE Press, Wiley Inter-Science, USA, 2002.

Gilbert Strang. *Linear Algebra and Its Applications.* Brooks Cole, 4 edition, July 2005.

SUPERGEN(EPSRC). Supergen sustainable power generation and supply - powering the peolpe. Technical report, Engineering and Physical Sciences Research Council.

S. Talukdar and V.C. Ramesh. A multiagent approach for contigency constrained optimal power flows. *IEEE Transaction on Power Systems*, (2):855–861, May 1994.

L. Tesfatsion and D. Koesrindaroto. Testing the reliability of ferc's wholesale power market plattform: An agent-based computational economics approach. In *24th. USAEE/IAEE North American Conference*, Washington, D.C., July 2004.

W. F. Tinney and J. W. Walker. Direct solutions of sparse network equations by optimally ordered triangular factorization. *Proceedings of the IEEE*, 55(11):1801–1809, 1967. URL `http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1447941`.

W. F. Tinney, V. Brandwajn, and S. M. Chan. Sparse vector methods. *IEEE Transaction on Power Systems*, PAS-104(3):295–301, 1985.

L.M. Tolbert, H. Qi, and F.Z. Pendg. Scalable multi-agent system for real-time electric power management. pages 1676–1679, Vancouver, Canada, July 2001.

P. Wei, Y. Yang, J. Yen, and F.F. Wu. A decentralized approach for optimal wholesale cross-border planning using multi-agent technology. *IEEE Transaction on Power Systems*, (4):833–838, November 2001.

J. H. Williams and R. Ghanadan. Electricity reform in developing and transition countries: A reappraisal. *Energy*, 31(6):815–844, May 2006.

Robert Wilson. Architecture of power markets. *Econometrica*, 70(4):1299–1340, July 2002.

B.F. Wollenberg and A.J. Wood. *Power Generation, Operation and Control*. IEEE Press, Wiley Inter-Science, USA, 1996.

M. Wooldridge. *An Introduction to MultiAgent Systems*. John Wiley & Sons, England, 2002.

C.S.K. Yeung, A.S.Y. Poon, and F.F. Wu. Game theoretical multi-agent modelling of coalition formation for multilateral trades. *IEEE Transaction on Power Systems*, (3): 929–934, August 1999.

Fredrik Ygge and Hans Akkermans. Power load management as a computational market. In *Proceedings of the Second International Conference on Multi-Agent Systems (ICMAS)*, pages 393–400. AAAI Press, 1996.

Xu Yi-chong. The myth of the single solution: Electricity reforms and the world bank. *Energy*, 31(6):802–814, May 2006.

H. Zhou, Z. Tu, S. Talukdar, and K.C. Marshall. Wholesale electricity market failure and the new market design. In *North American Association for Computational Social and Organizational Science (NAACSOS)*, Pittsburgh PA, USA, June 29 2004.

K. Zollenkopf. Bifactorization: Basic computational algorithm and programming techniques. In Oxford, editor, *Conference on Large Sets of Sparse Linear Equations*, pages 76–96, 1970.