Ana Maria Marques de Moura Gomes Viana

# Metaheuristics for the Unit Commitment Problem

## The Constraint Oriented Neighbourhoods Search Strategy

Dissertação apresentada à Faculdade de Engenharia da Universidade do Porto
para a obtenção do grau de Doutor em Engenharia Electrotécnica e de Computadores

Orientação: Prof. Doutor Jorge Manuel Pinho de Sousa
Co-orientação: Prof. Doutor Manuel António Cerqueira da Costa Matos

Faculdade de Engenharia da Universidade do Porto
2004

O trabalho de investigação apresentado nesta dissertação foi parcialmente suportado pelo Programa PRODEP III – Acção 5.3 – Formação Avançada no Ensino Superior – Concurso N$^o$ 4/5.3/PRODEP/2000

# Resumo

Nesta tese são explorados dois tópicos de investigação. Por um lado, aborda-se o problema de escalonamento de geradores eléctricos, num processo normalmente referido como "Unit Commitment" (UCP), sendo discutidas formas alternativas de modelização do problema e desenvolvendo-se técnicas de resolução inovadoras para o mesmo. Por outro lado, desenvolvem-se novas estratégias de pesquisa, a usar em meta-heurísticas baseadas em Pesquisa Local, estratégias essas que têm como objectivo a redução do impacto dos parâmetros associados às meta-heurísticas na qualidade das soluções obtidas.

Numa primeira tentativa de resolver eficientemente o problema de Unit Commitment usando meta-heurísticas, é proposta uma nova abordagem em que é usada a metodologia GRASP. Esta abordagem considera uma nova estrutura de representação de soluções que permite uma manipulação mais fácil das restrições do problema. No entanto, obtenção de boas soluções usando a metodologia GRASP obriga, tal como noutras meta-heurísticas, a uma afinação precisa dos seus parâmetros. Como o processo de afinação pode ser complexo, é susceptível de ser realizado de uma forma incorrecta, podendo levar a resultados francamente maus. Por isso mesmo, a necessidade de afinação de parâmetros é apontada como um dos aspectos negativos das meta-heurísticas.

Em consequência, esta tese propõe uma nova estratégia de pesquisa, Vizinhanças Orientadas às Restrições (aqui referida como "Constraint Oriented Neighbourhoods" – CON), que pretende reduzir a influência dos parâmetros das meta-heurísticas nos resultados finais. Para isso, esta estratégia admite que sejam aplicadas à solução actual operações de vizinhança diferentes, durante o processo de construção de uma solução vizinha, dependendo do tipo de restrições que são violadas. Deste modo, evita-se que em iterações sucessivas sejam introduzidas alterações drásticas numa solução, permitindo-se uma pesquisa mais suave do espaço de soluções. A estratégia foi aplicada ao UCP e conseguiu, sistematicamente, obter resultados de grande qualidade, com uma redução drástica dos tempos computacionais, quando comparados com os obtidos por outras metodologias propostas na literatura.

Finalmente, a estratégia CON serviu de base ao desenvolvimento de um novo algoritmo para resolução de problemas multiobjectivo (mCON – “multiobjective Constraint Oriented Neighbourhoods”) que foi usado na resolução de uma versão multiobjectivo do UCP com dois objectivos, ambos a minimizar: custos totais de produção e de emissões. A abordagem mostrou ser efectiva na resolução do problema referido, atingindo resultados melhores do que os obtidos com outras meta-heurísticas multobjectivo propostas na literatura.

# Abstract

This thesis addresses two main topics of research. It tackles the short-term planning problem of scheduling power generators, the Unit Commitment problem (UCP), discussing alternative modelling approaches and developing innovative resolution techniques. Furthermore, it describes the development of new search strategies to be used by Local Search based metaheuristics, aiming at reducing the impact of metaheuristic parameters in the quality of the solutions.

In a first attempt to efficiently solve the Unit Commitment problem with metaheuristics, an innovative GRASP methodology is proposed. The approach is based on a new representation of the problem solutions that allows constraints to be tackled more easily. However, achieving good quality solutions with GRASP (or with other metaheuristics) does require an accurate tuning of the algorithm parameters. As this tuning procedure may be complex and likely to be performed incorrectly, metaheuristics often lead to very bad quality results. This issue is frequently pointed out as a major shortcoming of metaheuristics.

As a consequence, this thesis proposes a new search strategy, Constraint Oriented Neighbourhoods (CON), that aims at reducing the influence of metaheuristic parameters on the final results. In the process of constructing a neighbour solution, this strategy considers that different movements may be applied to the current solution depending on the kind of constraints that are violated. By doing so, in successive iterations, drastic changes in a solution are avoided allowing a smoother search process. The strategy was applied to the UCP and was capable of systematically achieving high quality results, with drastic reductions in computational time, when compared with other methodologies proposed in the literature.

Finally, the CON search strategy has been extended, to be able to tackle multiobjective problems. The new algorithm (mCON – multiobjective Constraint Oriented Neighbourhoods) has been applied to a multiobjective UCP with two objectives to be minimised: total production costs and emissions. The approach proved to be effective, leading to better results than those obtained with other multiobjective metaheuristics proposed in the literature.

# Résumé

Cette thèse adresse deux sujets de recherche principaux. Le premier concerne le problème de ordonnancement de générateurs d' énergie en planification de court terme, et qui est connu comme “Unit Commitment Problem” (UCP). On discute des modèles alternatifs pour ce problème, et développe des techniques de résolution innovatrices. Le deuxième sujet concerne la conception et le développement de nouvelles stratégies de recherche pour des meta-heuristiques basées en “local search”, ayant comme but de réduire l'impacte des paramètres des meta-heuristiques sur la qualité des solutions.

Pour le premier de ces sujets, et en essayant de résoudre l' UCP avec de meta-heuristiques, ce travail propose une méthodologie GRASP innovatrice. L'approche est basée sur une représentation des solutions du problème qui permet une manipulation plus facile des contraintes. Cependant, obtenir de bonnes solutions avec GRASP (ou avec une autre meta-heuristique) demande un calibrage précise des paramètres de l'algorithme. En pratique, cette procédure peut être très complexe et facilement réalisée d'une façon incorrecte, menant a des solutions assez mauvaises. Cette question est souvent considérée comme un des aspects négatifs des meta-heuristiques.

Sur un autre axe, cette thèse propose une nouvelle stratégie de recherche qu'on appelle “Constraint Oriented Neighbourhoods” (CON) qui a été développé avec le but de réduire l' influence des paramètres des meta-heuristiques sur les résultats finaux. Pendant le processus de construction d'une solution voisine, cette stratégie considère que des mouvements différents peuvent être appliqués à la solution actuelle, en fonction du type de contraintes qui sont violées. En faisant ça, sur des itérations successives, on évite des changements drastiques des solutions, en permettant un processus de recherche beaucoup plus douce. La stratégie a été appliquée à l'UCP, en menant systématiquement à de meilleurs solutions, avec des réductions drastiques du temps de computation, si on les compare avec d' autre méthodes proposées dans la littérature.

Finalement, la stratégie CON a servi de base pour le développement d'une approche assez générale pour traiter des problèmes multi-objectif. Un algorithme nouveau a été développé (mCON – "multiobjective Constraint Oriented Neighbourhoods") et il a été appliqué à une version multi-objectif de l' UCP avec deux objectifs à minimiser: les coûts globaux de production et les émissions. L'approche développé a prouvé être effective, en menant à des résultats meilleurs que ceux obtenus avec d'autres meta-heuristiques multi-objectif proposées dans la littérature.

# Agradecimentos

Os meus primeiros agradecimentos vão para os Professores Jorge Pinho de Sousa e Manuel Matos, meus orientadores científicos, a quem agradeço o apoio incondicional, a confiança e o entusiasmo que sempre demonstraram pelo meu trabalho. Eles foram, para mim, os orientadores perfeitos. Não tenho quaisquer dúvidas que se estivesse hoje a começar este trabalho seriam, mais uma vez, os orientadores que eu gostaria de ter.

Não posso também deixar de agradecer ao INESC Porto pelas excelentes condições de trabalho que me proporcionou, disponibilizando todos os meios necessários à realização desta dissertação. Aos meu colegas de gabinete, Andreia, Marta e Zé Raúl, obrigada pelo ambiente amigável e descontraído que criaram, que tão bem me fizeram em alturas de desespero. Foi bom ter alguém com quem partilhar frustrações e alegrias.

Um agradecimento muito especial ao Hugo Ferreira, o meu *guru* em C++. O Hugo foi incansável a tentar resolver o que eu diagnostiquei como sendo um erro de *software*. Não foi por falta de persistência que não o resolveu. Viemos a descobrir, depois de um mês de trabalho, que essa seria uma tarefa impossível – o responsável pelo comportamento anormal do meu software não era qualquer erro, mas um problema de *hardware*...

Não posso esquecer os meu colegas do Grupo de Disciplinas de Ciências Básicas da Electrotecnia do ISEP que, entre eles, arranjarem maneira de eu ter algum tempo extra para concluir a tese. Um agradecimento especial ao Custódio Dias, pela sua preocupação e pelo cuidado que teve em oferecer-me as melhores condições possíveis para concluir o trabalho.

No início deste projecto, tive a oportunidade de visitar a secção de Planeamento da Rede Eléctrica Nacional (REN), onde me explicaram em detalhe como era feito o “Unit Commitment” no sistema português. Esta foi, sem dúvida, uma experiência muito enriquecedora para mim. Ao Eng. Pedro Roldão, ao Dr. António Guerreiro e ao Dr. Rui Coelho, agradeço o tempo que disponibilizaram para responder a todas as minhas perguntas e a forma cuidadosa com que prepararam a minha visita à REN. Agradeço ainda ao Eng. Baptista Gomes, da EDP Produção, pela disponibilidade demonstrada e pela informação importante que disponibilizou para a minha pesquisa.

À Susana, à Teresa, à Helena e, mais uma vez, ao Jorge, obrigada pela vossa amizade! Mais do que um orientador, o Jorge é um bom amigo, com quem tenho podido contar sempre que preciso. O seu apoio constante tem sido precioso e eu tenho de lhe agradecer por isso.

A Helena é a minha "psicóloga" de serviço, com quem partilho os meus medos e o sentimento de ansiedade que este tipo de trabalho provoca. Obrigada Helena, por me transmitires alguma calma e por me mostrares que nem tudo é tão grave quanto pode parecer à primeira vista!

É agora a vez da Teresa. Obrigada Teresa por, lutando contra a minha teimosia, tentares explicar-me que por vezes é preciso parar e descansar, mesmo que tal pareça impossível. Obrigada, pelos bolinhos de Sábado à tarde ("– Fazem-se em 5 minutos. Não dá trabalho nenhum. Além disso é Sábado, não tens onde lanchar..."), pelos muitos jantares de onde saí muito mais bem disposta do que estava quando cheguei, pelos magníficos fins-de-semana que organizaste. Obrigada principalmente por, graças a ti, termos construído esta amizade.

Finalmente a Susana – uma amiga de sempre, uma amiga para sempre. Mesmo usando as palavras mais ricas e elaboradas de que me consiga lembrar, nunca conseguirei exprimir o quão importante é para mim ter a Susana como amiga. Felizmente, entre amigas como nós não são precisas palavras, basta ser-se. Assim nos temos entendido ao longo dos anos, assim espero que continue a ser.

Os meus pais e irmãos continuam a ser "o porto de abrigo em dias de tempestade". É para mim um motivo de orgulho fazer parte de uma família tão especial, que influenciou positivamente a minha forma de estar na vida e, espero, fez de mim uma pessoa melhor. Descrever aquilo que eles significam para mim, seria uma tarefa árdua, se calhar impossível. Por isso, fico por aqui. O facto de lhes dedicar esta dissertação diz, provavelmente, bastante mais do que quaisquer palavras.

# Acknowledgements

My first acknowledgements go to my supervisors, Prof. Jorge Pinho de Sousa and Prof. Manuel Matos, for their continuous support, and enthusiastic attitude towards my research work. For me, they were the perfect supervisors. I have no doubts that if, rather than concluding, I was starting this work today, again they would be the supervisors that I would like to have.

I must also thank INESC Porto for providing me with a stimulating work environment, and with all the necessary facilities to develop my work. Thanks also to my officemates, Andreia, Marta and Zé Raúl, for the friendly environment that they have created, so important in moments of despair. I am happy that I had them to share with the moments of frustration, but also the happiness of the small achievements.

A very special thanks to Hugo Ferreira, my guru in C++. Hugo was tireless at trying to help me to fix what I diagnosed to be a software bug. It was certainly not due to his lack of persistency that we did not succeed. After a full month of work, we found out that this was an impossible task – not a bug, but a CPU fan, was the responsible for the abnormal behaviour of my software...

I cannot also forget the support of my colleagues of “Grupo de Disciplinas de Ciências Básicas da Electrotecnia”, in ISEP, for allowing me to have some extra time to conclude the thesis. A special thanks to Custódio Dias, for his continuous care and concern on trying to provide me with the best conditions so that I could conclude this work.

At the beginning of this project, I had the opportunity of visiting REN (Rede Eléctrica Nacional), where Eng. Pedro Roldão, Dr. António Guerreiro and Dr. Rui Coelho explained me how the Unit Commitment was performed in the Portuguese power production sector. This was certainly a very fruitful experience and I must thank them for having received me there and for having carefully prepared that visit. Thanks also to Eng. Baptista Gomes, from EDP Produção, for providing me with some important information for my research work.

To Susana, Teresa, Helena and, again, Jorge, thank you for your friendship! More than a supervisor, Jorge is a good friend. In several moments, during these years that I have been working with him, he proved that I can count on him whenever I need. His support has been precious and I must thank him for that.

Helena is my private "psychologist", the one I shared fears and anxiety with. Thank you Helena, for making me calm down and for showing me that some problems are not as serious as they may seem at first glance!

Thank you Teresa for combating my stubbornness and trying to make me stop and rest, when (for me) it seemed impossible. Thank you for the Saturday afternoon cookies ("– It's 5 minutes to make them. It's so easy. Besides, it's Saturday and everything is closed..."), for the many dinners that you have prepared, for organising those wonderful "relaxing" weekends (with cookies, cakes, fresh orange juice and whatever one may think of). Mostly, I thank you for building our friendship.

Finally, Susana – a friend from ever, a friend forever. Even the "richest" and the more elaborate words that I can think of would be too poor to express how this friendship is important for me. Fortunately, between friends like us, compliments are not so important. It has been like that for years and, I hope, it will remain this way.

My parents, brother and sister remain "the safe haven in days of tempest". Growing up in such a special family has definitely influenced my way of facing life and, I hope, has made me a better person. To describe what they have done and meant for me over the years is a arduous, probably impossible, task. Thus, I will leave it to this. The fact that this dissertation is dedicated to them says probably more than any words that I may think of.

# Contents

# List of Figures

# List of Tables

# Glossary

AFLC – Average Full Load Cost

ALR – Augmented Lagrangian Relaxation

CCA – Cooperative Coevolutionary Algorithm

CLP – Constraint Logic Programming

CON – Constraint Oriented Neighbourhoods

CPP – Competitive Power Pool

CUF – Commitment Utilisation Factor

DM – Decision Maker

DP – Dynamic Programming

DSB – Demand Side Bidding

EA – Evolutionary Algorithms

EMS – Energy Management Systems

ES – Expert Systems

GA – Genetic Algorithms

GENCO – Generator Company

GPM - General Purpose Movement

GRASP – Greedy Randomised Adaptive Search Procedure

HTC – Hydrothermal Coordination

ISO – Independent System Operator

LR – Lagrangian Relaxation

mCON – Multiobjective Constraint Oriented Neighbourhoods

MCP – Market Clearing Price

MCQ – Market Clearing Quantity

MILP – Mixed Integer Linear Programming

MIP – Mixed Integer Programming

MOCO – Multiobjective Combinatorial Optimisation

MOMH – Multiobjective Metaheuristics

MOSA – Multiobjective Simulated Annealing

NETA – New Electricity Trading Arrangements

NIE – Northern Ireland Electricity

NN – Neural Networks

OR – Operational Research

PE – Potentially Efficient

PSA – Pareto Simulated Annealing

PX – Power Exchanger

RCL – Restricted Candidate List

SA – Simulated Annealing

SASS – Successive Approximation in Solution Space

SDP – Semi-definite Programming

SUC – Sequential Unit Commitment

TS – Tabu Search

TSP – Travelling Salesman Problem

UCP – Unit Commitment Problem

UCTE – Union for the Coordination of Transmission of Electricity

VNS – Variable Neighbourhood Search

# Chapter 1

# Introduction

Operational Research (OR) has a long tradition as a multidisciplinary area, often requiring several subjects of expertise to solve a single problem. It embraces a diversity of disciplines, ranging from Social Sciences, to medical health care, Economy, and several areas of Engineering. This thesis is a reflex of this generic approach, by putting together the two main areas of interest of the author: Operational Research and Electrical Engineering. As a result, it is developed in two parallel, still complementary, lines of research – the design of new search strategies for metaheuristic techniques based on Local Search, and their application as effective optimisation techniques to a problem arising in Power Systems Management – the Unit Commitment Problem (UCP).

## 1.1 Scope and relevance of the theme

### 1.1.1 The Operational Research line of research

Heuristics are an essential tool in applied combinatorial optimisation and for many of the large, and often messy, problems found in practice the only applicable ones to provide good quality solutions within reasonable time.

Until the middle 80's *heuristics* were strictly and unequivocally related to the definition of more or less elaborate "rules of thumb", strongly based on experience, that iteratively build a solution from scratch. But, since then, other heuristics based on a different paradigm have been proposed. Generally referred to as *metaheuristics*, these optimisation tools were originally defined as approximate methods where subordinate heuristics are guided by a generic concept to tackle a problem. The techniques, that include, but are not restricted to, Simulated Annealing, Genetic Algorithms, Tabu Search, GRASP, Variable Neighborhood Search, etc, have already proven to be effective at solving several combinatorial optimisation problems. Furthermore, they present a set of characteristics, namely the ability of easily incorporating variations in a problem's structure, that make them particularly appealing in scenarios that are continuously changing, i.e. most of the real decision environments. This led [Resende and Sousa 2004] to say that "metaheuristics have probably been one of the most stimulating research topics in optimisation for the last two decades", and it is reflected in the ever growing publication of reports, papers and books in this area.

However, this ever increasing development did not correspond to a similar increase in the number of practical (commercial) applications that include metaheuristics. Although other reasons may exist, one may point out the high dependency of metaheuristics performance on parameter tuning as an aspect that makes Decision Makers feel reticent on using them. Therefore, it is essential to develop strategies that reduce the grade of dependency of metaheuristics performance on their parameters, so that they become more appealing as effective optimisation tools to help decision making.

Such an achievement should also be pursued in multiobjective contexts. It is widely recognised that most real-world optimisation problems are multiobjective in nature. Still, it is also true that most problems are generally solved as if they were single objective ones, most probably due to the lack of alternative optimisation tools enabling a simple and effective management of several objectives simultaneously. The arising interest on metaheuristics specially designed to handle several objectives (Multiobjective Metaheuristics), while keeping the appealing characteristics of their single objective counterparts, is surely a topic for research. However, their application in practice is again constrained by the need of correctly tuning the metaheuristics parameters, to achieve good quality results. Therefore, for the same reason stated for single-objective metaheuristics, a special effort should be made on designing methodologies that are less dependent on parameter tuning.

### 1.1.2 The Electrical Engineering line of research

Within Electrical Engineering, the Power Systems sector is an area where OR already has some tradition, with numerous models along with optimisation and simulation methods for decision support. The applications range from short-term production planning to long-term network expansion planning and, more recently, with the liberalisation of the power sector, to the definition of optimal bidding strategies for trading in power pools.

Due to its impact in the economical results of generation companies, a problem that has for long deserved attention within short-term planning is the Unit Commitment Problem. In its basic format, it is the on/off problem of selecting the power generating units to be in service, and of deciding for how long will they remain in that state, for a given planning horizon (lasting from 1 day to 2 weeks, and generally split in periods of 1 hour). The committed units must satisfy the forecasted system load and reserve requirements, at minimum operating cost, subject to a large set of other system and technological constraints.

Due to its combinatorial nature, the problem tends to be solved through more or less elaborate heuristic techniques, the most popular seeming to be those based on Lagrangian Relaxation. More recently, metaheuristics such as Genetic Algorithms, Simulated Annealing and Tabu Search have also been proposed. These approaches have the advantage of allowing a correct modelling of the problem discontinuities that are highly present in this problem, which is *per se* a strong argument for using them. However, although receiving increasing attention, one may say that the application of metaheuristics to the UCP is still in its infancy and further research should be done to achieve better quality results.

In terms of objectives, the problem does not differ from others found in practice and, again, one might think of several other objectives rather than the minimisation of production costs. Nevertheless, except for a couple of papers, no other references are found in the literature on modelling the UCP as a multiobjective problem, the area remaining relatively unexplored. One objective that one may immediately think of, due to the growing concerns on environmental issues and the tight regulations that shall be imposed on emissions in the near future, is the minimisation of emissions from the use of fossil fuels. This issue may become even more important, from an economical point of view if, as expected, the market is such that companies are allowed to trade their emission quotas. Therefore, the theme deserves to be deeper analysed.

## 1.2 Objectives of the thesis

Given the author's decision of working in two complementary areas, the objectives of this thesis are twofold: to develop contributions for enhancing the effective application of metaheuristics, through the development of new search strategies that reduce the influence of parameter tuning on the quality of the final solution, and to propose effective innovative approaches to the Unit Commitment Problem.

**Metaheuristics**

It has for long been considered that, to be successful, a metaheuristic should be as simple as possible, both conceptually and in practice. It should also be effective and, if possible, general purpose, in the extreme case without any problem-dependent knowledge. However, as metaheuristics have become more and more elaborated, the ideal case has been pushed aside in the quest of a greater performance, and problem specific knowledge is now incorporated into metaheuristics [Lourenço et al. 2003].

Within this line of reasoning, and always pursuing the central aim of optimisation techniques – reaching the best achievable results – one aim of this thesis is to propose a new search strategy that, by considering problem specific characteristics, will allow a smoother search process and reduce the impact of parameter tuning in the performance of a metaheuristic.

Reducing the influence of parameter tuning on the quality of the final results is a key issue in metaheuristics. It does not only make them more user friendly but also prevents Decision Makers from inadvertently introducing inefficiencies in the optimisation process, leading to worse quality solutions and an increase in computation time, if the parameters are not correctly chosen. Therefore, this concern should also be present when developing new search strategies that are capable of managing more than one objective simultaneously. Furthermore, and within multiobjective contexts, such strategies should avoid the need to define/update weights usually used to aggregate the objectives in multiobjective metaheuristics, thus reducing the number of decisions that must be made to tune the metaheuristic.

**Unit Commitment Problem**

From this work, several contributions are expected for the Unit Commitment Problem. First, and due to the difficulty of tackling some of the problem constraints, new alternative ways to represent a UCP solution should be tried. Such solution representations should allow an efficient checking of its feasibility, when metaheuristic operators are applied.

A second line of research is on developing robust innovative algorithms to efficiently tackle the problem.

Finally, and given the increasing importance of environmental issues, include those aspects in the base problem as an objective and develop appropriate multiobjective techniques to solve it.

## 1.3 Outline of the thesis

This thesis is organised in 7 chapters. In Chapter 1, the scope and relevance of the work are introduced and its main goals are outlined.

Chapter 2 introduces, in section 2.1, some basic concepts related to Combinatorial Optimisation and discusses the computational complexity of combinatorial optimisation problems. Section 2.2 elaborates on the heuristic paradigms that are available for this kind of problems and proceeds with a section on metaheuristics, the emphasis being given to Local Search based metaheuristics.

Chapter 3 introduces the Unit Commitment Problem and its main variants, and makes a literature review on the main techniques used to solve these problems. A short survey on commercial software packages that are available to solve the problem is also presented.

Chapter 4 starts with a literature review on the application of metaheuristics to the UCP. It proceeds with the presentation of a new solution representation and a GRASP based approach to solve the problem.

For its effective application, it is required that some aspects of metaheuristics are improved, so that Decision Makers view them as reliable and alternative tools for decision support. Chapter 5 introduces a new search strategy – *Constraint Oriented Neighbourhoods* – that will, hopefully, contribute for that purpose. Its aim is to reduce the importance of parameter tuning in Local Search based metaheuristics, keeping the quality standards of the final results. In the same chapter the concept of *Constraint Oriented Neighbourhoods* is applied to the Unit Commitment Problem.

After introducing some basic concepts on Multiobjective Combinatorial Optimisation, and presenting a brief literature review on multiobjective metaheuristics, the concept of *Constraint Oriented Neighbourhoods* is extended in Chapter 6, leading to an innovative approach to tackle multiple objectives simultaneously. The new approach is used to solve a multiobjective Unit Commitment Problem, with two objectives to be minimised: production costs and emissions.

Finally, Chapter 7 draws a set of conclusions, reviews the main contributions of the thesis and gives some guidelines for future work in this area.

# Chapter 2

# Combinatorial Optimisation and Metaheuristics

The increasing dimension and complexity of problems found in practice lead frequently to a situation where exact optimisation algorithms (from now on simply referred to as *exact algorithms*) are not able to find a solution within a reasonable amount of time, i.e. in a way that is useful for the decision making process under consideration. In such cases, approximation algorithms (*heuristics*) are an essential tool and, sometimes, the only applicable one to provide good quality solutions within the time available for making a decision. These techniques are based on the implementation of more or less elaborate "rules of thumb", that try to explore the solution space in an appropriate manner. However, these rules are usually tightly related to specific problem characteristics (constraints or objectives) and, due to that, the heuristic tends to become very problem dependent – small changes in the problem characteristics may result in drastic changes in the heuristic rules. Trying to overcome this problem, increasing attention has been given in the last two decades to *metaheuristics* – general heuristics that can be applied to different optimisation problems, needing relatively few modifications to become adapted to a specific problem.

An area where metaheuristics have received particular attention is in combinatorial optimisation problems. These problems are usually rather complex to solve and, despite the fast evolution of computers' technology, it is still very challenging (if not impossible) to find exact solutions for many such problems. On the other hand, the ever changing scenarios that one must face nowadays, do not make traditional heuristic techniques particularly appealing if they are too problem dependent and the problem characteristics are frequently changing.

In this chapter we start by presenting some basic concepts related to combinatorial optimisation and problem complexity. We then make a brief introduction to heuristic techniques and conclude the chapter with a discussion on current trends in this area and with some final remarks.

## 2.1 Combinatorial Optimisation and computational complexity

Combinatorial optimisation is the process of finding one or more optimal solutions in a well defined discrete problem space, trying to maximise or minimise a given objective function. Such problems occur in almost all fields of management (e.g. finance, scheduling, inventory control), as well as in many engineering subjects (e.g. VLSI-circuitry design and testing, design and analysis of data networks, management of electrical power generation), and can be represented, in very broad terms, in the following way:

$$\min f(X) \tag{2.1}$$

subject to:

$$X \in D$$

where $X$ is a solution of the problem and $D$ is the discrete space of feasible solutions, called the *solution space* (or *decision space*). The objective function, $f$, maps $D$ into $\Re^1$. When all parameters rather then $X$ are know in (2.1) we have a problem *instance*.

Theoretically speaking, the finiteness of $D$ suggests that any given instance could be solved by enumerating and evaluating all possible alternative solutions and picking the best one. However, this approach is not possible for many combinatorial problems, when the size of the search space grows exponentially with the instance size, leading to computation times that are far above those available for decision making. These problems fall into the class of $\mathcal{NP}$-hard problems.

Other classes of problems do however exist, the classification scheme including (but being not restricted to) the well known terminology classes $\mathcal{P}$, $\mathcal{NP}$, $\mathcal{NP}$-complete and $\mathcal{NP}$-hard. The categorisation of problems is done according to the grade of difficulty of resolution of each problem, its *time complexity*, studied within a field of Mathematics called *Theory*

[1]Without any loss of generality in this thesis we will restrict ourselves to minimisation problems.

*of Computational Complexity.* For early work on computational complexity, the reader is addressed to [Garey and Johnson 1979]. A more recent work is that by [Papadimitriou 1994] and a tutorial can be found in [Tovey 2002].

**Time complexity**

In order to compare several alternative algorithms, when their performance is associated to required computer time, one would like to have a measure of efficiency that is independent of the machine used and of particular implementation details. This may be achieved by measuring the number of elementary computer operations it takes to solve the problem in the worst case. This is the *time complexity* of the problem and depends on the size of the instance, the *length of the input.*

**Definition 1** For a problem instance X, the *length of the input* $L = L(X)$ is the length of the binary representation of a "standard" representation of the instance.

Depending on how the number of basic steps increases, with an increase in the length of the input, the corresponding algorithm is labelled either as *polynomial* or *exponential.* The classification of the algorithms is based on the *Big-O* notation.

**Definition 2** For two functions f(t) and g(t), with nonnegative t, we say that *f(t) = O(g(t))* if there is a constant c > 0 such that, for all sufficiently large t, f(t) = cg(t). The function cg(t) is thus an asymptotic upper bound on f.

**Definition 3** Given a problem T, an algorithm A that solves T, and an instance X of T, let $f_A(X)$ be the number of elementary calculations required to run algorithm A, on instance X. $f_A^*(l) = \sup_X\{f_A(X) : L(X) = l\}$ is the *running time* of algorithm A.

**Definition 4** An algorithm A is *polynomial* for a problem T if $f_A^*(l) = O(l^p)$, for some positive integer p. An algorithm A is *exponential* for a problem T if $f_A^*(l) \neq O(l^p)$, for any p.

Although the classical complexity theory mainly relies on analytical approaches, such as worst-case analysis, experimental analysis of algorithms has also been intensively used in some fields, e.g. Operational Research and Artificial Intelligence [Guo 1996]. Experimental (or empirical) analysis of algorithms studies algorithms and data structures by solving a large number of representative instances of various sizes and fitting a function for their running time, in terms of size. One of the main advantages of the approach is that it allows to investigate how algorithmic performance statistically depends on problem characteristics. Still, the measure depends on programming skills and on the patterns of data in the examples solved. Besides, the approach is not suitable for developing a mathematical theory.

**Complexity classes**

Besides studying the time complexity of a problem, the Theory of Computational Complexity defines the rules that separate the problems into those for which one can devise a solution through a polynomial algorithm, the $\mathcal{P}$ problems, and those for which an efficient algorithm has not been found so far, and the time needed to obtain a feasible solution grows exponentially with the problem size, the $\mathcal{NP}$ problems.

As this theory applies only to decision problems (problems whose solution is a *yes-or-no* answer) rather than to optimisation problems, an optimisation problem like that in (2.1) cannot be studied from the point of view of problem complexity, when in that format. So, to devise whether such a problem fits into the $\mathcal{P}$ or the $\mathcal{NP}$ class of problems, it must be replaced by its corresponding decision problem [Wolsey 1998]:

$$\text{For any value K, is there a solution } X \in D \text{ such that } f(X) \leq K? \tag{2.2}$$

**Example 1** *The Bin Packing Problem as an optimisation and as a decision problem*

Given a set S of items, each of them having a specified integral size, and being C the size of the bin:

*i.* *Optimisation problem:* Determine the smallest number of subsets into which one can partition S, such that the total size of the items in each subset is at most C.

*ii.* *Decision problem:* For a given constant K, determine whether S can be partitioned into K subsets, such that the total size of the items in each subset is at most C.

**Definition 5** $\mathcal{NP}$ is the class of decision problems that, whenever the answer to (2.2) is affirmative, one can prove it within polynomial time.

**Example 2** *A $\mathcal{NP}$ problem*

One example of a $\mathcal{NP}$ problem is the *subset sum problem*: given a finite set of integers, determine whether any non empty subset of them adds up to zero. A supposed answer is very easy to verify for correctness (e.g. given the set {-7, -3, -2, 5, 8} and the subset {-3, -2, 5}, the answer is yes because the elements in this subset sum to zero), but no one knows a faster way to solve the problem than to try every single possible subset, which is an exponential procedure.

**Definition 6** $\mathcal{P}$ is the class of decision problems in $\mathcal{NP}$ that can be solved within polynomial time.

From definitions 5 and 6 it is clear that the $\mathcal{NP}$ class contains the $\mathcal{P}$ class, $\mathcal{P} \subseteq \mathcal{NP}$. However, it is not clear whether $\mathcal{P} = \mathcal{NP}$. In other words, if it is always easy to check a solution, should it also be easy to find the solution? Although there is no reason to believe it should be true, there is not also a proof that it is false and this remains one of the most important open questions in Computer Science.

### $\mathcal{NP}$-completeness vs $\mathcal{NP}$-hardness

$\mathcal{NP}$-complete problems can be viewed as the hardest problems in $\mathcal{NP}$. Formally, $\mathcal{NP}$-completeness is defined in terms of *reduction.*

**Definition 7** If Q, R $\in \mathcal{NP}$, and if an instance of Q can be converted in polynomial time to an instance of R, Q is *polynomially reducible* to R.

**Definition 8** The *class of $\mathcal{NP}$-complete problems,* is the subset of problems M $\in \mathcal{NP}$, such that, for all Q $\in \mathcal{NP}$, Q is polynomially reducible to M, i.e. a decision problem is $\mathcal{NP}$-complete if it is $\mathcal{NP}$ and has the property that any $\mathcal{NP}$ problem can be converted into it in polynomial time.

Another key class of problems within this context is the class of $\mathcal{NP}$-hard problems.

**Definition 9** A problem is *$\mathcal{NP}$-hard* if any problem in $\mathcal{NP}$ reduces polynomially to it, but it cannot be proved to be in $\mathcal{NP}$.

Although the theory behind problem complexity applies only to decision problems, some conclusions may be taken from above, concerning the complexity of their corresponding optimisation problems. As an optimisation problem can be transformed into a decision problem, if a decision problem is $\mathcal{NP}$-complete, the corresponding optimisation problem must be at least has hard to solve, otherwise one would fall into a contradictory situation (see Proof 1, below). Therefore, the term $\mathcal{NP}$-hard is used for the optimisation versions of combinatorial problems for which the decision version is known to be $\mathcal{NP}$-complete. In other words, an optimisation problem whose corresponding decision problem is $\mathcal{NP}$-complete, is $\mathcal{NP}$-hard.

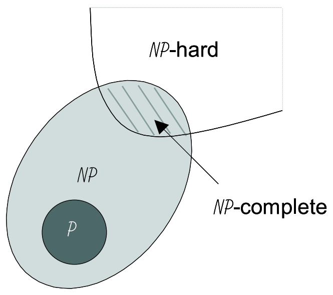


**Figure 2.1:** Problem complexity

A schematic representation of the sets of problems that were introduced in this section is given in Figure 2.1.

**Proof 1** *If a decision problem is $\mathcal{NP}$-complete, the corresponding optimisation problem is $\mathcal{NP}$-hard*

Suppose that OP is an optimisation problem that can be easily solved to optimality, and that the corresponding decision problem, DP, is $\mathcal{NP}$-complete. As the optimisation problem can be solved within polynomial time, we can first solve this problem to optimality and then, given a bound K to the DP, solve the decision problem in polynomial time by comparing K to the know optimum. This contradicts the initial assumption that stated that DP was a $\mathcal{NP}$-complete problem.

In terms of combinatorial optimisation, an important conclusion taken from the $\mathcal{NP}$-hardness of many combinatorial optimisation problems is that we cannot expect an exact algorithm to solve a given instance to optimality in polynomial time. Therefore, proving or knowing that a problem is $\mathcal{NP}$-hard is critical in the choice of a resolution approach. In fact, as soon as a problem is classified as hard to solve, one can immediately focus the attention on developing approximate algorithms – heuristics – that find near optimal solutions within reasonable time, rather then spending time looking for an unreachable optimal solution. In the following sections we discuss such approaches. A special emphasis will be given to metaheuristics based on Local Search.

## 2.2 Heuristic paradigms

Due to the inherent complexity of most combinatorial optimisation problems, exact algorithms need exponential run time to reach the optimum and become useless when the size of problems grows. This gave rise to an increasing attention on the development of heuristic methods that are able of efficiently finding satisfactory solutions (not necessarily optimal), for real size problems, sacrificing the guarantee of finding optimal solutions for the sake of getting good ones in reasonable time.

Several approaches have been proposed for obtaining good quality solutions, subject to given computational time constraints and there are naturally various ways to classify them. According to [Maniezzo and Carbonaro 2001] they can be divided in three main paradigms: *approximation algorithms*, that guarantee that a solution is found within a known gap from optimality; *probabilistic algorithms*, that guarantee that for instances over a given size the probability of getting a bad solution is small; or *heuristic algorithms* (heuristics) that do not offer any guarantees, concerning the quality of the solution, but that somehow represent a good trade-off *quality of the solution/time spent* to solve the problem. According to the same authors, heuristic algorithms can be further divided into three sub-categories: those that concentrate on structural properties of the problem and that use them to define constructive rules; those that focus on the guidance of a constructive or local search algorithm, to avoid local optima; and finally those that focus on incorporating partial results obtained from exact methods into a heuristic framework. In this thesis we are concerned with the second category of heuristics. Even so, a general description of the other categories is also presented below.

### Type I heuristics – focus on solution structure characteristics

#### Constructive heuristics

The heuristics falling into this category are more or less elaborate "rules of thumb" that are based on experience, a kind of shortcuts that we use in everyday decision making, often having an intuitive justification. Constructive heuristics generate solutions from scratch by adding solution components to an initially empty solution, in some order, until the solution is "feasible" or complete. The framework of these heuristics, follows the reasoning implicitly described in example 3.

**Example 3** *A constructive heuristic*

"When going out on vacation, I always had the problem of deciding what to stuff into my knapsack. At the beginning I tended to put everything inside the bag, until there was no more room. However, during my first InterRail, when there was nobody to carry the bag for me, I soon realised that what might seem a light bag at home, would become very heavy after some kilometers walking. So, next year, when preparing again my knapsack, I started devising a way of choosing the things I should bring with me this time. The outcome of my "scientific approach" for the problem was the following: first, I defined the maximum weight that I was able to carry, gave a grade of importance to each of the items that I would like to bring with me and weighted them. My aim was to select the items to stuff into the knapsack in such a way that the sum of their grades of importance was maximised, and the maximum defined weight was not exceeded.

Always keeping in mind that aim, I sorted the items in decreasing order, according to their "grade of importance/weight" ratio and started packing them following that order. Whenever I reached an item whose weight was above the remaining weight capacity, I skipped that item, picking the next one in the list. The process was repeated until the remaining weight capacity was less then the lightest item in the list."

**Local Search heuristics**

Local Search heuristics depart from an initial (and usually feasible) solution, that is frequently obtained following a random reasoning, with no particular concern on its quality. That solution is a starting point for an improvement process that will hopefully lead to a good quality solution. During the improvement process, the Local Search, the solution is subject to successive slight modifications that are accepted if they lead to a better solution. The procedure stops when a "local optimal" solution is reached.

**Example 4** *A local search heuristic*

Let us go back to the packing problem, introduced in example 3. Instead of designing such an elaborate heuristic, I might have considered a different approach, as follows. First, I would introduce as many items as possible in the knapsack regarding only solution feasibility, with no particular care on my objective. I would then pick one of the items that was still lying over my bed, and exchange it with another that was already in the knapsack. If this solution was better, I would keep it and repeat the previous procedure. Otherwise, I would go back to the previous solution, and try a different exchange. When, independently of the choice I made, no improvements were achieved, I would consider the best solution found so far and stop my search.

A major advantage of local search heuristics is that, in their basic versions, they are general purpose methods which can be easily adapted to a concrete optimisation problem. However, they do also present some weak points. Since only improving solutions are accepted, the method stops if a local optimum has been reached. Generally, this solution is not globally optimal and no information on how much the quality of this solution differs from the global optimum is available.

**Type II heuristics – heuristic guidance**

To avoid the drawback presented by local search heuristics, new approaches based in the concepts of *diversification* (or exploration) and *intensification* (or exploitation) of the search space have been developed since the mid 70's [Holland 1975]. When the search enters non-promising areas of the search space, *diversification* allows that promising areas are discovered. When such areas are found, *intensification* carefully seeks for the best solutions within the scope of that area.

Heuristics of type II are commonly referred to as *Metaheuristics* and, depending on how diversification and intensification are achieved, they are are based on *Local Search* procedures (e.g. Simulated Annealing, Tabu Search, Variable Neighbourhood Search), or on *recombination* procedures (e.g. Genetic Algorithms). Metaheuristics based on recombination are beyond the scope of this thesis. Nevertheless, some references to these methods will be given in section 2.3.4.

**Type III heuristics – Mathematical programming contribution**

A current trend in optimisation heuristics for combinatorial problems is on exploiting results obtained through exact methods, with heuristics of type II. Some applications can be found in the literature, with an hybridisation of Branch-and-Bound [Fischetti and Lodi 2003], Lagrangian Relaxation, etc, and metaheuristics. Such approaches have the advantage of defining a lower bound on the value of the objective function, thus controlling the performance of the procedure.

This thesis is concerned with heuristics of type II, specifically with those belonging to the Local Search family.

## 2.3 Metaheuristics

### 2.3.1 General features and scope

Local search in combinatorial optimisation dates back to the late 50's, when the first edge-exchange algorithms for the Travelling Salesman Problem (TSP) were introduced [Bock 1958; Croes 1958]. However, it was only in the last two decades that local search based heuristics have become popular, leading to the development of a new heuristic paradigm, labelled as *metaheuristics*. This sudden interest has several reasons: they are easy and intuitive to understand, flexible, generally easier to implement than constructive heuristics, and have shown to be very valuable to solve large problem instances. Furthermore, in the theoretical side considerable progress has also been made, not only on tackling local search from a complexity point of view [Johnson et al. 1988], but also in developing convergence proofs for some of the methods within this group [van Laarhoven and Aarts 1987; Faigle and Kern 1991, 1992; Granville et al. 1994; Glover and Hanafi 2002].

The term *metaheuristic* was first introduced by Glover in [Glover 1986] to define a top-level general strategy which guides other heuristics towards promising regions of the search space, i.e. regions that are likely to contain high quality solutions. The heuristics guided by the top level strategy may be either very elaborate procedures, or very straightforward moves that transform the current solution in a similar (neighbour) one.

After two decades of existence, these methods are still an important area of research in Combinatorial Optimisation, with an ever growing publication of reports, papers and books. What we can call now "the classical metaheuristics" (Simulated Annealing, Tabu Search and Genetic Algorithms) [Reeves 1995], evolved to more refined methods, and also gave rise to new methodologies like VNS – Variable Neighbourhood Search [Hansen and Mladenović 1997], GRASP – Greedy Randomised Adaptive Search Procedure [Feo and Resende 1995], Ant Colonies [Dorigo and di Caro 1999], etc. The areas of application of these techniques have also increased, and one can find studies in as diverse areas as Telecommunications [Costamagna et al. 1998], Power Engineering [Viana et al. 2003b], Timetabling [Thompson and Dowsland 1998], Scheduling [Jozefowska et al. 2002], or Vehicle Routing [Gendreau et al. 1994]. A recent natural trend is on the design of multiobjective metaheuristics, e.g. [Czyzac and Jaszkiewicz 1998; Serafini 1992; Horn et al. 1994; Hansen 1997], where very promising results are being achieved.

Two main lines of research have been followed: one lying on the concept of Local Search (or neighbourhood search), and a second one based on the concept of Recombination. Starting from an initial solution, a Local Search based procedure moves from one feasible solution to a new solution, by applying some changes to the current solution, until some stopping criteria are met. Recombination refers to the process of combining two or more solutions into a new solution that usually inherits characteristics of its generating solutions. The metaheuristics that embed this concept are usually referred to as population-based metaheuristics and compose the class of *Evolutionary Algorithms* (EA).

### 2.3.2 General issues of Local Search

Local Search, also referred to as *Neighborhood Search* or *Hill Climbing*, is the basis of many heuristic methods for combinatorial optimisation problems. It is based on the iterative exploration of neighbourhoods of solutions, trying to improve the current solution by local changes. The type of local changes that may be applied to a solution is defined by a *neighbourhood structure.*

**Definition 10** For an instance of a combinatorial optimisation problem with feasible solution set D, a *neighbourhood structure* is a mapping $N : D \rightarrow 2^D$, which defines for each feasible solution $X \in D$ a set $N(X) \subseteq D$ of feasible solutions that are in some sense close to X. The set N(X) is the *neighbourhood* of the feasible solution X, and each $X' \in N(X)$ is a *neighbour* of X [Aarts and Lenstra 1997].

The neighbourhood structure can also be represented as a graph, called the neighbourhood graph. The nodes of the graph represent the solutions and two solutions are connected by an edge if they are neighbours and the neighbourhood structure is symmetric. Thus, a Local Search algorithm corresponds to a walk in the neighbourhood graph. In general, the solution found by local search will be a local optimum, rather than a globally optimal solution.

**Definition 11** A solution X is called a *local minimum* of the objective function $f$ with respect to the neighbourhood N, if and only if $f(X) \leq f(Y), \forall\, Y \in N(X)$

**Definition 12** A solution X is a *global optimum* of the objective function $f$, if and only if $f(X) \leq f(Y), \forall\, Y \in D$

A basic Local Search heuristic may begin with an arbitrary solution, and explores the search space by allowing, in each iteration, a small change in the current solution through a neighbourhood operation that leads to a new solution. The optimisation process ends in a local minimum where no further improvements are possible (Figure 2.2).

In between these stages there are many different ways to drive a local search. For example, *best improvement local search* replaces the current solution with the solution from the whole neighbourhood of that solution that improves the most the objective function. Another example is *first improvement local search*, which accepts a better solution as soon as it is found.

The computational complexity of a local search procedure depends on the size of the neighbourhood and also on the time needed to evaluate a move. In general, the larger the

```
Algorithm Local Search

i = 0
Generate initial solution $X_0$
While stopping criterion not met
    Explore neighbourhood of $X_i$ and generate candidate solution Y
    if f (Y) < f ($X_i$)
        $X_{i+1}$ = Y
    else
        $X_{i+1}$ = $X_i$
    i = i + 1
```

**Figure 2.2:** Local Search algorithm

neighbourhood, the more the time needed to search and the better the local minima. As a result, and due to time constraints, instead of exploring the whole neighbourhood of a given approach, one usually searches a limited area called a *sub-neighbourhood* (obtained, e.g. by randomly selecting a reduced number of neighbours). The size of the neighbourhood must therefore be set before using such procedures. Other aspects that must be specified are:

i. *How to generate initial solutions (and how many)*

   A simple and usually effective way of obtaining initial solutions is by using random constructive heuristics. These heuristics differ from those described in section 2.2 in the sense that the selection of solution components, in each iteration, is done randomly. A deterministic selection is also possible but this imposes that the search always starts from the same solution, which may not be interesting. The number of initial solutions that must be obtained, depends on the metaheuristic considered. Simulated Annealing and Tabu Search, for example, do only need a single starting solution, to perform the search. On the other hand, GRASP assumes that several different initial solutions are obtained. In such cases deterministic constructive heuristics cannot be applied.

ii. *The neighbourhood structure of a solution*

   The definition of neighbourhood structures is probably one of the key issues in designing metaheuristics. On the one hand, they should be designed in such a way that the structures common to good solutions are preserved after the neighbourhood operation is applied. On the other hand, they should allow search diversification and, for that purpose, less restrictive moves should also be allowed. In many cases, more than one promising definition of the neighbourhood is possible, to make heuristics more powerful.

That is the case of *variable depth search*, an embedded neighbourhood construction that compounds simple movements to create more complex and powerful ones. That idea also gave rise to the development of metaheuristics such as Variable Neighbourhood Search that successively consider different neighbourhoods during the search process.

iii. *The neighbourhood size of a solution*

As referred before, in theoretical terms, the neighbourhood size is settled when the neighbourhood structure is defined. However, in most cases it becomes impractical to generate those many solutions and the generation is bounded by some upper limit. In such cases, one says that the method is searching a *sub-neighbourhood* of the solution.

iv. *The evaluation function*

When the search is constrained to the set of feasible solutions, a straightforward way of evaluating solutions is through the problem objective function. However, some metaheuristics allow that the search space includes infeasible solutions. In such cases, penalty terms are often added to the original objective function, to evaluate the degree of infeasibility of a solution. Besides, for those problems where solution evaluation is a very hard and time consuming task, that may negatively affect the general performance of the algorithm, one may consider the alternative of making faster approximation evaluations, at the beginning of the search process, to reduce the overhead in computation times.

v. *The move strategy*

A basic move strategy is to choose a better solution in the neighbourhood of the current one, as in *best improvement local search* or in *first improvement local search*. However, it may also be allowed to move to non-improved (or even worse) solutions, in a somehow contained way.

vi. *The stopping criterion*

The stopping criterion depends on the move strategy that is considered. If the strategy is to move always to a better solution, the algorithm usually stops at a local optimum. However, if non-improved solutions are accepted, the algorithm could continue indefinitely, being therefore necessary to define some stopping rule. Frequently, the following rules are considered: stop when the number of moves or the computational time reach a pre-specified limit; stop if the current solution has not been improved after a pre-specified number of consecutive iterations.

### 2.3.3 A short overview of metaheuristics

The number of different metaheuristics described in the literature is getting so large that it becomes difficult to present a complete survey on such methods, always facing the risk of omitting a specific one or, given the similarity of some of them, referring to two different metaheuristics as if they were the same. Such a thorough survey is not the aim of this section and the reader is reported to [Laporte and Osman 1996; Glover and Kochenberger 2003], for additional information on the subject. The purpose of this section is solely to introduce those metaheuristics that have set the ground for the tremendous development of this area, namely Simulated Annealing, Tabu Search and Genetic Algorithms, and those that are somehow related to the work of this thesis, GRASP and VNS.

#### Simulated Annealing

Simulated Annealing (SA) was first proposed in [Kirkpatrick et al. 1983]. It is a randomised local search procedure where, at each iteration, the current solution X is modified by randomly selecting a move that leads to a neighbour solution. If the new solution Y is better, it is automatically accepted and becomes the new current solution. Otherwise, the new solution is accepted according to the Metropolis distribution (equation 2.3), where the probability of acceptance is related to the magnitude of quality reduction and to a parameter called the temperature (T). Basically, a move is more likely to be accepted if the temperature is high and the decrease in quality is low.

The temperature parameter is progressively lowered, according to some pre-defined cooling schedule, and a certain number of iterations are performed at each temperature level. When the temperature is sufficiently low, only improving moves are accepted and the method stops at a local optimum. So, to achieve diversification, the algorithm starts with a high temperature to perform a wide search of the solution space. The temperature is then gradually reduced to focus on a specific region, allowing a correct search intensification.

$$P_{\text{accept}}(X,Y,T) = \begin{cases} 1 & \text{if } f(Y) < f(X) \\ e^{\left(\frac{f(X)-f(Y)}{T}\right)} & \text{otherwise} \end{cases} \tag{2.3}$$

Under certain neighbourhood structures and depending on the way in which the temperature is decreased, it is possible to prove that SA asymptotically converges to the global optimum. The proof is based on Markov chains. If: 1) the neighbourhood is connected, i.e. if it is possible to move from each solution via a sequence of neighboured solutions to any

**Algorithm SA**

```
i = 0
Generate initial solution $X_i$
$X^* = X_i$
Set initial temperature value (T)
Set $n_T$
**While** stopping criterion not met
    n = 0
    **While** $n < n_T$
        Explore neighbourhood of $X_i$ and generate candidate solution Y
        **if** $f(Y) < f(X_i)$
            $X_{i+1} = Y$
        **else**
            Calculate $P_{accept}(X_i, Y, T)$
            **if** $P_{accept}(X_i, Y, T) > random[0,1]$
                $X_{i+1} = Y$
            **else**
                $X_{i+1} = X_i$
        **if** $f(X_{i+1}) < f(X^*)$
            $X^* = X_{i+1}$
        n = n + 1
    Update temperature value (T)
    i = i + 1
**Return** X*
```

**Figure 2.3:** Simulated Annealing

other solution, and 2) the temperature does not decrease too quickly to 0, SA converges to a globally optimal solution with probability 1 [van Laarhoven and Aarts 1987]. However, this result is mainly of theoretical interest as each real implementation of SA is still a heuristic method, since it only runs a finite number of iterations. Furthermore, the above result does not give a rate of convergence. Thus, no bounds on the quality of the solution after a finite number of steps are known.

The SA procedure is outlined in Figure 2.3. X* stands for the best solution found so far, and $n_T$ stands for the number of iterations that shall be performed at the same temperature level.

The reader is addressed to [Vidal 1993; Reeves 1995; Rayward-Smith et al. 1996] or [Azencott 1992; Aarts and Lenstra 1997; Aarts and Ten Eikelder 2002; Henderson et al. 2003], for additional information on this algorithm.

### Tabu Search

Tabu Search (TS) ideas were introduced by [Glover 1986] and also by a parallel and independent work developed by [Hansen 1986] named as *Steepest Ascent/Mildest Descent* method. It is a deterministic local search strategy that, in each iteration, moves to the best admissible neighbour, even if this leads to a solution that is worse than the current one. As this acceptance criterion may lead to cycling (i.e. returning to solutions recently visited), in order to prevent that, a list of forbidden moves (the *Tabu List*), is considered. The Tabu List stores the last $k$ moves (or some attributes of the moves), where $k$ is a parameter of the method. Whenever a new move is accepted as the new current solution, the oldest one is discarded.

Storing attributes rather than the complete solutions may prevent the method from achieving some interesting solutions. An aspiration criterion is normally used to avoid this problem. The most straightforward aspiration criterion considers that if for a solution X a move leads to a better solution, the new solution is accepted as the new current one, regardless of its tabu status.

Like in Simulated Annealing, a Tabu Search procedure may stop either when the number of iterations (or CPU time) reaches a given value, or when a solution is not improved after a pre-specified number of consecutive iterations. An outline of this procedure is given in Figure 2.4.

```
Algorithm TS

i = 0
Generate initial solution $X_i$
$X^* = X_i$
Initialise Tabu List
Set aspiration criterion
While stopping criterion not met
    Generate $n$ neighbours of $X_i$
    Choose best neighbour of $X_i$, Y, that is not in the Tabu List
    or that reaches the aspiration criterion
    if $f(Y) < f(X^*)$
        $X^* = Y$
    Update Tabu List
    i = i + 1
Return $X^*$
```

**Figure 2.4:** Tabu Search

Starting from the simple Tabu Search method just described, a number of developments and refinements have been proposed over the years. The reader is reported to [Glover 1986; Goldberg 1989; Glover 1990; Glover et al. 1993; Glover and Laguna 1997, 2002; Gendreau 2002, 2003], for additional information on these developments.

### Genetic Algorithms

Genetic Algorithms (GA) were first introduced by [Holland 1975]. The main point of distinction between these algorithms and others previously reported is that they are population-based algorithms, i.e. they consider several solutions in parallel and exchange "information" between each solution, aiming at obtaining better quality results. The algorithms are inspired by models of the natural evolution of species and their reasoning relies on the principle of natural selection which favours stronger individuals that are more prone at surviving and reproducing.

Three main operators are used to correctly exploit the space: *selection*, *crossover* and *mutation*. Selection gives to individuals (solutions) with a higher fitness score, a higher priority of being chosen for the next generation and for the application of the remaining operators. The genetic material of two individuals, also called parents, is recombined by means of a crossover operator to generate new individuals, called *offsprings*. The idea of crossover is to exchange useful information between two individuals and in this way to generate a hopefully better offspring. Mutation is understood as a background operator which introduces small random modifications to an individual.

A sketch of a GA algorithm is presented in Figure 2.5 (note however that numerous variations of this basic scheme can be considered). The algorithm considers a set $X_i$ of N individuals. In each iteration 2m elements are selected from $X_i$ (according to their fitness score), pairwise crossover is applied and the offspring solutions are stored in $Z_i$. Mutation is then applied to each element of this set, with a given probability. Finally, 2m elements are selected from $X_i$ (a higher probability of selection being given to worse elements) and replaced by $Z_i$. At the end, the best solution found so far is updated.

For additional information on GA the reader is addressed to [Goldberg 1989], [Michalewicz 1994], or [Reeves 1995].

### GRASP

The first references to GRASP (Greedy Randomised Adaptive Search Procedure) appear in [Feo and Resende 1989], motivated by the work of [Hart and Shogan 1987]. GRASP It

```
Algorithm GA

i = 0
Generate initial population of solutions $X_i = \{x_{i1}, x_{i2}, ..., x_{iN}\}$
Set $F^* = \min \{F(x_{ij}), j = 1, ..., n\}$
Set $x^* = \arg\min \{F(x_{ij}), j = 1, ..., n\}$
While stopping criterion not met
    $Y_i$ = Select 2m elements from $X_i$
    $Z_i = \{\}$
    For k = 1 to m
        $Z_i = Z_i \cup$ Crossover($y_{i,2k-1}, y_{i,2k}$)
    Mutation($Z_i$)
    Replace($X_i, Z_i$)
    if ($\min \{F(x_{ij}), j = 1, ..., n\} < F^*$)
        $F^* = \min \{F(x_{ij}), j = 1, ..., n\}$
        $x^* = \arg\min \{F(x_{ij}), j = 1, ..., n\}$
    i = i + 1
Return $x^*$
```

**Figure 2.5:** Genetic Algorithms

is a multi-start metaheuristic composed of two main phases: a Construction Phase and a Local Search Phase. In the Construction Phase, an initial feasible solution is iteratively built following an adaptive reasoning, i.e. the decisions taken in previous iterations influence the decision taken in the current iteration. Then, the neighbourhood of that solution is explored, using a Local Search procedure, and if the solution (X) that is obtained is better than the best solution found in previous runs of the algorithm (X*) it is kept. The procedure is outlined in Figure 2.6, where $n_S$ stands for the number of initial solutions that shall be generated.

To obtain an initial feasible solution, at each iteration of the Construction Phase all elements (decisions that can be taken) in a set of possible candidates are ranked, according to a *greedy function* that evaluates the contribution to the objective function obtained by choosing that particular element. If the elements in the ranked list reach a given threshold, they are accepted for future decisions and stored in a *Restricted Candidate List* (RCL). The decisions taken in each iteration of the Construction Phase are then randomly selected among those in the RCL. By following this reasoning, at each GRASP iteration a different initial solution is obtained and, hopefully, different regions of the search space are explored by the Local Search procedure. Finally, at the end of each iteration, the *greedy function* is adapted, so that in the following iterations it will take into account the decisions previously taken.

```
Algorithm GRASP

n = 0
Set $n_S$
While n < $n_S$
    Generate new solution $X_n$
    X = Local Search($X_n$)
    if n = 0
        X* = X
    else
        if f(X) < f(X*)
            X* = X
    n =n + 1
Return X*
```

**Figure 2.6:** Greedy Randomised Adaptive Search Procedure

More details on GRASP will be given in Chapter 4, where it is applied to the Unit Commitment problem. Even so, for further information on this procedure, the reader is addressed to [Festa and Resende 2001; Pitsoulis and Resende 2002; Resende and Ribeiro 2003].

### Variable Neighbourhood Search (VNS)

VNS was presented in the late 90's by [Hansen and Mladenović 1997]. The aim of this metaheuristic is to avoid poor local optima by defining alternative neighbourhood movements ($N_k$) that are systematically changed to explore an increasingly larger region of the solution space.

Given a set of neighbourhood structures, K, a solution is randomly generated in the first neighbourhood of the current solution, from which a local descent is performed. If the local optimum obtained is not better than the current best one, the procedure is repeated with the next neighbourhood. The search restarts from the first neighbourhood when a better solution is found (Figure 2.7).

Additional information on VNS can be found in [Hansen and Mladenović 1999, 2001a,b; Hansen et al. 2001; Hansen and Mladenović 2002, 2003].

**Algorithm VNS**

```
Set K
Generate initial solution X
For k = 0 to K
    Set $N_k$
While k < K
    Y = Local Search(X,$N_k$)
    if f (Y) < f (X)
        X = Y
        k = 0
    else
        k = k + 1
Return X
```

**Figure 2.7:** Variable Neighbourhood Search

### 2.3.4 Classification of metaheuristics

Given the ever increasing number of metaheuristics, it would clearly be interesting to classify them according to their main characteristics. Needless to say, this is a difficult and non-consensual task because of different interpretations about the essential nature of the methods [Glover and Laguna 1997]. Nevertheless, some authors made an attempt to develop a taxonomy for metaheuristics based on the basic Local Search algorithm [Vaessens et al. 1992], for hybrid metaheuristics [Talbi 2002] or even for a specific metaheuristic [Crainic et al. 1997].

[Glover and Laguna 1997] propose a classification of metaheuristics based on three basic design choices: the use (or not) of memory, the kind of neighbourhood exploration used, and the number of current solutions carried from an iteration to the next one. [Hansen 1998] made a similar attempt and did also classify metaheuristics according to them being based on Local Search or on Recombination. For Local Search based metaheuristics, a distinction was proposed between those that rely on continuous local search and those relying on re-start or multi-start (once a region has been extensively explored, the search is restarted from a new solution). Finally, he made a distinction between those that use only one solution and those that are population based.

*Continuous vs multi-start*

Metaheuristics such as Simulated Annealing depart from a single initial solution and perform their search procedure once, until a stopping criterion is reached, following one single search trajectory corresponding to a connected walk on the neighborhood graph. In contrast, multi-start local search methods repeatedly apply a local search from different initial solutions. According to some authors, this approach may be particularly appealing for problems where it is more effective to construct solutions rather than to apply a local search procedure [Martí 2003].

*Memory usage*
The use of memory is a way of providing learning mechanisms, based on the search experience, to influence the future search direction. Memory is explicitly used in Tabu Search: short-term memory is used to forbid revisiting recently found solutions and to avoid cycling, while long-term memory is used for diversification and intensification purposes. On the contrary, Simulated Annealing and GRASP do not use memory functions to influence the future search direction and therefore are memoryless algorithms.

*Single vs multiple neighbourhood definitions*
Most local search algorithms are based on a single neighborhood structure (e.g. Simulated Annealing, Tabu Search). However, more recently, some methods that consider more than one neighbourhood structure have been developed. In such cases, the local search starts with a neighbourhood, until a local optimum is reached. By then, the neighbourhood changes and the search proceeds.

In this work, we present a classification that considers both the issues referred in [Glover and Laguna 1997] and in [Hansen 1998]. Given the increasing relevance of the number of neighbourhood structures defined, we do also classify metaheuristics according to that number. In Table 2.1 we summarise the classification of some methods according to these criteria (note that we do not suggest that all implementations of these algorithms correspond to this classification, but it rather gives an indication of their distinguishing features while in their "standard" format). The table is by no means exhaustive, its only aim being to exemplify a possible classification of metaheuristics.

The reader can find general information and further references on the methods referred to in Table 2.1 in [Glover and Kochenberger 2003]. Additional info on the metaheuristics that are not referred in that table, and that fall into the set of Evolutionary Algorithms, can also be found in the same work or in, e.g. [Goldberg 1989] or [Michalewicz 1994].

| Metaheuristics | | Generating solutions | Memory usage | Neighbourhoods |
|---|---|---|---|---|
| Genetic Algorithms | R | n | P | - |
| Simulated Annealing | CLS | 1 | N | 1 |
| Tabu Search | CLS | 1 | Y | 1 |
| Greedy Randomized Adaptive Search (GRASP) | MSLS | n | N | 1 |
| Scatter Search | R | n | Y | - |
| Variable Neighbourhood Search (VNS) | CLS | 1 | N | n |
| Ant Colony Optimization (ANTS) | R | n | Y | - |
| Iterated Local Search (ILS) | MSLS | 1 | P | n |
| Hyper-heuristics | CLS | 1 | P | n |

**Table 2.1:** Classification of metaheuristics (in the second column, R stands for Recombination, CLS for Continuous Local Search and MSLS for Multi-Start Local Search; the third column denotes the number of generating solutions ($n$ meaning more than 1); the fourth column refers to memory usage (Y stands for Yes, N for No and P for Partially); column five refers to the number of neighbourhoods used by each metaheuristic.

## 2.4 Current trends in metaheuristics

Although the results presented in the literature for metaheuristic applications have in general been quite successful, the impact they had until now in real organisations for operations and services management has not been as strong as it might be expected, and the number of practical applications is rather deceptive [Burke et al. 2003]. It is probably with the objective of inverting such a tendency that many of the current research and developments in metaheuristics aim at making metaheuristics more appealing for a real application. We highlight here three of the major lines in current metaheuristics research: reduction of manual parameter tuning, development of reusable software classes, and multiobjective methods. These topics have strongly influenced the work presented in this thesis.

### Parameter tuning

It is recognised that a main reason for the end-user reluctance in using metaheuristics has to do with the need of correctly tuning a set of parameters, as a pre-condition for metaheuristics to perform well. Parameter tuning is by no way an easy and straightforward process and implies a reasonable knowledge of the structure of the metaheuristics and of the role that each parameter plays in the performance of a particular algorithm. Thus, it is desirable to develop processes that are capable of automatically tuning the parameters or that, at least, reduce their impact on the algorithms behaviour. This is important not only to increase the grade of acceptability of metaheuristics, but also as a way of preventing unexperienced users from introducing inefficiencies in the optimisation process due to a wrong tuning.

Several lines of research have been followed for that purpose. Some do directly work with metaheuristic parameters, as is the case of reactive implementations [Battiti 1996]. In these approaches some history-based feedback is integrated in local search and the parameters are automatically adjusted during the search, avoiding a manual tuning. A second line of research is on developing approaches that are less dependent on parameter tuning. That seems to be the case when hybrid and parallel implementations are used. Hybrid implementations aim at exploiting the strengths of different methods by designing a new metaheuristic that combines various algorithmic principles of the original methods. Parallel implementations are not only an effective alternative to speed up the search but, as threads exchange "good" information about their current solutions, they tend to produce better quality solutions [Ribeiro et al. 2002].

**Development of reusable software**

There are several strong arguments in favour of reusable software components. Economically speaking, it is particularly appealing due to the inherent reduction in software development costs, and to provide a fast answer to a client specific requirement. From a scientific perspective, several advantages may also be pointed out: it allows a faster and more precise comparison of algorithms' efficiency, it simplifies the task of developing new methods, etc.

It was within this context that several "frameworks" for metaheuristics have been developed recently [Fink et al. 2003]. A framework is a special kind of software library, usually developed under an Object Oriented environment, which consists of a hierarchy of abstract classes representing a partial implementation of the program. For a specific optimisation problem, the classes are instantiated and adapted to the characteristics of that problem.

The modular structure of metaheuristics makes them particularly appropriate to share common modules of software. Metaheuristics frameworks explore this characteristic and develop software libraries in such a way that, to implement a specific metaheuristic, the user only defines suitable derived classes, related to that method. The advantages of the approach are obvious: faster development, easier and more effective comparison of several methodologies and easier development of new methodologies, by using previously developed software modules that are common to other metaheuristics.

**Multiobjective metaheuristics**

An increasing attention has been given to the development of new metaheuristics that, inheriting the advantages of their single objective counterparts, are capable of simultaneously tackling several objectives.

A major reason for such an interest has to do with the lack of powerful alternatives to solve multiobjective combinatorial problems: exact approaches are not an alternative for larger problems and developing specialised multiobjective heuristics is difficult and usually too much dependent on the specific problem under consideration. As a result, most approaches tend to aggregate the objectives in a unique overall objective function. However, this requires extensive *a priori* preference information for defining weights, thresholds, marginal benefits, etc, which is not necessarily available.

A way to go may be on developing multiobjective metaheuristics (MOMH). Given the relative success of metaheuristics at solving single objective combinatorial problems, since the 90's a considerable effort has been made at developing MOMH. They inherit parts of the basic structure of their single objective versions, and are extended and embed some new features that make them suitable for dealing with the multiobjective paradigm.

## 2.5 Concluding remarks

It is clear that metaheuristics play an important role within search techniques to solve combinatorial optimisation problems, a huge number of publications showing that they are able of efficiently finding good quality results, faster than other optimisation techniques, in as diverse areas as Scheduling, Telecommunications or Power Engineering. Even so, it is still difficult to find commercial software applications that include metaheuristic techniques.

In an attempt to answer to some of the "clients"' critical and more challenging requirements, several lines of research are currently being explored. In this chapter we have referred to three of them: parameter tuning, reusable software and multiobjective methods. These issues are therefore some of the essential topics covered in the remaining of this thesis.

# Chapter 3

# The Unit Commitment Problem

Power Systems optimisation models are widely used in practice, ranging from expansion planning studies to real-time operation, and passing by short-term planning.

A central problem arising in short-term planning is the Unit Commitment Problem (UCP). It is the problem of deciding which electric generators must be committed/decommitted over a planning horizon (lasting from 1 day to 2 weeks, generally split in periods of 1 hour), and the production levels at which they must be operating, so that costs are minimised. The committed units must satisfy the forecasted system load and reserve requirements, subject to a large set of other system and technological constraints. The classical thermal UCP model considers that the power units are centrally managed in a global way, and aims at minimising the total production cost, over the planning horizon, expressed as the sum of fuel, start-up and shut-down costs.

Being a hard and challenging problem (it belongs to the set of $\mathcal{NP}$-hard problems), and due to its economical importance (large operational costs are involved), the problem has for long been a matter of concern for power generator companies (GENCO). In spite of that, trying to give an answer to GENCO requirements, research has focused on developing faster and more effective optimisation tools, and on improving the base model by including additional information that has recently become important for a more effective management of power production. These concerns have naturally led to the design of several variants of the classical thermal formulation and models considering fuel and/or multi-area constraints have been studied. The increasing importance of environmental issues, and the intention expressed by several countries of reducing pollutants emissions, also gave rise to models that include emissions constraints (or objectives aiming at minimising the emissions).

Another issue of current intensive discussion has to do with the deregulation of the power industry sector. As a result of deregulation, the restructured power companies have additional needs that lead to a new set of modelling requirements and market design challenges.

Other variants to the base problem handle different sources of energy rather than thermal. In such cases, one must consider a coordination mechanism between all the sources to obtain a global optimisation of the system. Due to the strategic importance of water resources a special emphasis is given in this chapter to hydrothermal coordination.

The chapter starts by introducing the base thermal UCP problem, its most well-known variants (including hydrothermal coordination) and the methodologies used to solve such problems. Given the increasing impact of deregulation in these models, the chapter proceeds with a reference to some important issues within deregulated environments. A reference is then made to available commercial software to tackle the UCP and, finally, some considerations on the current issues of discussion in this area are presented.

## 3.1 Basic thermal problem

### 3.1.1 Problem features and resolution approaches

The UCP is the on/off problem of selecting the power generating units to be in service, and deciding for how long will they remain in that state, for a given planning horizon. The committed units must satisfy the forecasted system load and reserve requirements, at minimum operating cost, subject to a large set of other constraints. Given that operating costs depend on the load assigned to each generator, the problem of committing units is directly connected to the additional problem of (roughly) assigning the load demand to the units that are on (*Pre-Dispatch* problem).

This leads to a two-phase resolution method (Figure 3.1). First, for each period of time, it must be decided which units will be on/off. Secondly, the problem is decoupled into T subproblems, T being the size of the planning horizon, and for each subproblem (i.e. for each period of time) the production level of each unit that is on is calculated. The definition of the production level of each unit is a non-linear problem, that can be easily solved by using, e.g. the $\lambda$-iteration method based on the Kuhn-Tucker conditions [Wood and Wollenberg 1996]. However, the on/off decision problem is a combinatorial, non-linear and non-convex optimisation problem [Rudolf and Bayrleithner 1999], that is $\mathcal{NP}$-hard.

Being a combinatorial true multi-period problem (due to important start-up and shut-down costs) the UCP is in general very hard to solve, as it is not possible to perform a

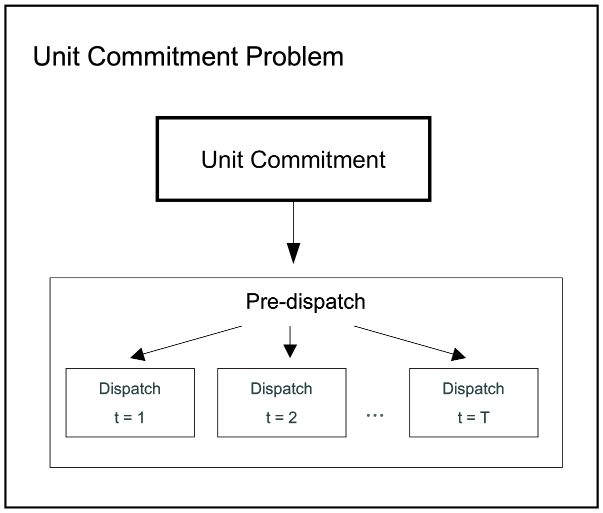


**Figure 3.1:** The Unit Commitment Problem

separate optimisation for each time interval. Except for very small size problems, exact methods such as Dynamic Programming and Branch-and-Bound proved to be inefficient and, in general, unable to find a solution within useful time. Thus, during the last decades, research efforts have concentrated on developing heuristic approaches, capable of efficiently finding satisfactory (not necessarily optimal) solutions for real size problems. Several heuristic approaches based on exact methods have been used, e.g. Branch-and-Bound [Cohen and Yoshimura 1983], as well as methods based on priority-lists [Lee 1988] and Lagrangian Relaxation [Bertsekas et al. 1983; Merlin and Sandrin 1983; Aoki et al. 1987] or, more recently, Neural Networks [Sasaki et al. 1992; Ouyang and Shahidehpour 1992; Huang and Huang 1997] and metaheuristics (e.g. Genetic Algorithms [Dasgupta and McGrevor 1994; Kazarlis et al. 1996], Simulated Annealing [Zhuang and Galiana 1990; Mantawy et al. 1998a; Yin Wa Wong 1998] or Tabu Search [Mantawy et al. 1998b]).

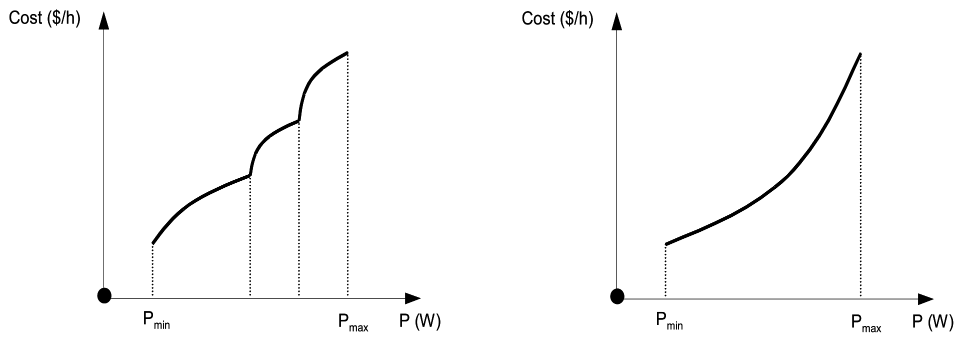
### 3.1.2 Objective and constraints

Usually, the basic thermal UCP is modelled as a single objective problem that aims at minimising total operation costs. Other objectives such as reduction of emissions or maximisation of feasibility and system security, though less frequent, are also referred in the literature [Sen and Kothari 1998a]. In this section, unless otherwise stated, the minimisation of total operation costs, expressed as the sum of fuel, start-up and shut-down costs is considered.

### Fuel costs

Fuel costs are related to the fuel consumption of each unit and depend on its production level in each period of time, and on the type of unit under study (coal, fuel-oil, nuclear, etc).

Although the functions that describe these costs are, in general, non-continuous and non-convex (see Figure 3.2 a)), as the non-convexity of such functions prevents conventional optimisation techniques from being used, approximate polynomial functions, as that depicted in Figure 3.2 b), are generally used in practice.



a) Exact b) Polynomial approximation

**Figure 3.2:** Fuel cost function

### Start-up costs

Every time a unit is shut-down, bringing it back into operation leads to an extra cost due to fuel waste, extra maintenance and to the additional feed water and energy that are needed for heating. Besides, frequent start-ups are undesirable from a social as well as a technical point of view: not only are they stressful to operators, but thermal units also undergo a heating and cooling cycle, coupled with pressurisation and decompression of boilers, turbine chambers, etc, that lead to a reduction in the effective life of the generating units. Furthermore, the emission of pollutants like $CO_2$, $SO_x$ and $NO_x$ is particularly high during the transient period of start-up and shut-down.

Start-up cost functions ($S_x$) should, in a certain way, reflect these concerns in the decision process. The costs depend on the latest period the unit was operating and, for steam units, they also depend on whether the boiler was kept hot, or not, while off [Matos 1999]. If it was not kept hot (i.e. if it has gone through a *cooling* process), start-up costs can be modelled

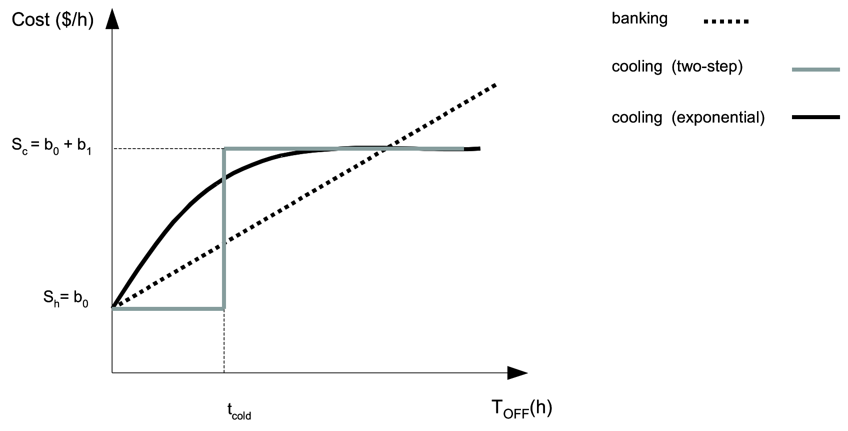


**Figure 3.3:** Start-up cost functions

by an exponential function as that in expression (3.1), where $b_0$ ($) is the fixed start-up cost and $b_1$ ($) is the cold start-up cost. $\tau$ is the cooling constant and OFF (h) the number of consecutive periods that the unit remained off.

$$S_x = b_1 \left(1 - e^{\left(-\frac{\mathrm{OFF}}{\tau}\right)}\right) + b_0 \tag{3.1}$$

As an alternative, several authors do also consider a two step function, as that in expression (3.2).

$$S_x = \begin{cases} S_h & \text{if OFF} \leq t^{cold} \\ S_c & \text{otherwise} \end{cases} \tag{3.2}$$

In expression (3.2), $S_h$ and $S_c$ are the costs incurred for a hot and a cold start-up, respectively. $t^{cold}$ is a unit parameter that indicates the number of hours that the boiler needs to cool down.

If the boiler temperature and pressure levels are maintained (*banking*), start-up costs can be represented by a linear function as that depicted in expression (3.3), where C ($/h) is the cost incurred by fuel consumption, to keep the boiler warm. A graphical representation of these alternatives is given in Figure 3.3.

$$S_x = b_0 + C \times t \tag{3.3}$$

### Shut-down costs

Shut-down costs are typically represented by a constant [Orero and Irving 1999]. In a certain way they measure the labour cost of decoupling units and are usually much lower than start-up costs.

### Constraints

The following constraints, modelling system and technical aspects of the generating units are considered.

*System power balance demand*
These constraints state that, in each period, the committed units must satisfy the total load demand.

*Spinning reserve requirements*
Reserve constraints are related to reliability and quality of service. They aim at supplying surplus power in the event of a contingency (e.g. due to an unit failure), or to compensate for possible differences between forecasted and current load demand. If such an event occurs, a quick system response can only be obtained by using units that are synchronised with the electrical power grid, i.e. that do not need any preliminary operations to be connected to the grid. The *spinning reserve* refers to the power that the generating system should be able of quickly supplying and is given by the difference between the total amount of quick generation available in the system, and the current load demand. According to the Union for the Coordination of Transmission of Electricity – UCTE – system deviations must be fully deployed within 15 minutes.

*Unit minimum up and down times*
If a unit is off, it must remain off for at least $\mathrm{T}^{off}$ periods of time. In the same way, if a unit is on it must remain on for at least $\mathrm{T}^{on}$ periods of time. $\mathrm{T}^{off}$ and $\mathrm{T}^{on}$ are an unit minimum down and up time, respectively.

*Unit generation limits*
Thermal units are not technically capable of producing below a minimum production level, nor above a maximum. According to [Wood and Wollenberg 1996] the minimum production level has to do with fuel combustion stability and steam generator design constraints. This value is typically between 10% and 30% of the maximum production level, for gas or petrol

supplied units, and between 20% and 50% of the maximum production level, for carbon supplied units. Maximum production levels are bounded by the unit design characteristics.

*Unit ramp rate limits*

Thermal units cannot drastically change their production levels in consecutive periods of time. Minimum up (down) rate constraints specify the maximum increase (decrease) that a unit may have in its production level, in consecutive periods of time.

### 3.1.3 Mathematical model

Even if we are not going to use any optimisation package, this section presents a mathematical model of the UCP, to clarify and precise the decision problem under consideration. First, the notation that will be used along the text is introduced. It is followed by the objective function and problem constraints. The model considers that the planning horizon is split into periods of 1 hour.

#### Notation

*Indexes*

$t$ – time periods ($t = 1,\dots,T$)

$i$ – thermal units ($i = 1,\dots,I$)

*Decision variables*

$$u_{it} = \begin{cases} 1 & \text{if } i \text{ is on in period } t \\ 0 & \text{otherwise} \end{cases}$$

$P_{it}$ – production level of unit $i$, in period $t$ (MW)

*Auxiliary variables*

$OFF_{it}$ – number of consecutive periods for which unit $i$ has been off, immediately before period $t$

$ON_{it}$ – number of consecutive periods for which unit $i$ has been on, immediately before period $t$

*Problem parameters*

$\underline{P}_i$, $\overline{P}_i$ – minimum and maximum production level of unit $i$ (MW)

$T_i^{on}$, $T_i^{off}$ – minimum up and down time of unit $i$ (h)

$r_{it}^{up}$ – maximum up rate of unit $i$, in period $t$ (MW/h)

$r_{it}^{down}$ – maximum down rate of unit $i$, in period $t$ (MW/h)

$P_t^d$ – system load requirements in period $t$ (MW)

$R_t$ – spinning reserve requirements in period $t$ (MW)

$a_i$, $b_i$, $c_i$ – fuel cost parameters (measured in \$/MW$^2$h, \$/MWh and \$/h, respectively - see below expression (3.5))

$S_h$, $S_c$ – costs incurred for a hot and a cold start-up, respectively, when a two-step function is considered (\$)

$t_i^{cold}$ – number of hours that the boiler of unit $i$ needs to cool down (h)

*Unit initial state*

$$u_{i0} = \begin{cases} 1 & \text{if } i \text{ was on for } t < 1 \\ \\ 0 & otherwise \end{cases}$$

$OFF_{i0}$ – number of consecutive periods for which unit $i$ has been off, immediately before $t$ $= 1$

$ON_{i0}$ – number of consecutive periods for which unit $i$ has been on, immediately before $t =$ 1

*Production costs*

*$F(P_{it})$* – fuel cost of unit $i$, in period $t$ (\$/h)

*$SC(OFF_{it}, u_{it})$* – start-up cost of unit $i$, in period $t$ (\$)

$DC_i$ – shut-down cost of unit $i$ (\$)

**Objective function**

The objective is to minimise the total operating costs:

$$min \sum_{t=1}^{T} \sum_{i=1}^{I} F(P_{it}) + SC(OFF_{it}, u_{it}) + DC_i \tag{3.4}$$

the fuel cost being given by expression (3.5). Start-up costs are given by expression (3.6), where $\mathrm{S}_x$ is modelled either by a two-step (3.7) or an exponential function (3.8).

$$F(P_{it}) = a_i P_{it}^2 + b_i P_{it} + c_i \tag{3.5}$$

$$SC\,(OFF_{it}, u_{it}) = u_{it} \times (1 - u_{i,(t-1)}) \times S_x(OFF_{it}) \tag{3.6}$$

$$S_x = \begin{cases} S_h & \text{if } \mathrm{OFF}_{it} \leq t_i^{cold} \\ S_c & \text{otherwise} \end{cases} \tag{3.7}$$

$$S_x = b_1 \left(1 - e^{\left(-\frac{\mathrm{OFF}}{\tau}\right)}\right) + b_0 \tag{3.8}$$

Shut-down costs are, in this work, considered to be constant.

**Constraints**

Constraints related to system and technical characteristics are:

$$\sum_{i=1}^{I} P_{it} = P_t^d \qquad t = 1\,, \ldots, \,T \tag{3.9}$$

$$\sum_{i=1}^{I} \overline{P}_i u_{it} \geq P_t^d + R_t \qquad t = 1\,, \ldots, \,T \tag{3.10}$$

$$\underline{P}_i u_{it} \leq P_{it} \leq \overline{P}_i u_{it} \qquad i = 1\,, \ldots, \,I \quad t = 1\,, \ldots, \,T \tag{3.11}$$

$$(u_{it} - u_{i,(t-1)}) \times (OFF_{i,(t-1)} - T_i^{off}) \leq 0 \qquad i = 1\,, \ldots, \,I \tag{3.12}$$

with

$$OFF_{it} = (1 + OFF_{i,(t-1)}) \times (1 - u_{it}) \tag{3.13}$$

$$(u_{it} - u_{i,(t-1)}) \times (ON_{i,(t-1)} - T_i^{on}) \leq 0 \qquad i = 1\,, \ldots, \,I \tag{3.14}$$

with

$$ON_{it} = (1 + ON_{i,(t-1)}) \times u_{it} \tag{3.15}$$

$$r_{it}^{up} \geq P_{it} - P_{i(t-1)} \qquad i = 1, \ldots, I \quad t = 1, \ldots, T \tag{3.16}$$

$$r_{it}^{down} \geq P_{i(t-1)} - P_{it} \qquad i = 1, \ldots, I \quad t = 1, \ldots, T \tag{3.17}$$

Load demand constraints are given by expression (3.9) and the spinning reserve by expression (3.10). The technical constraints of each unit are given by expression (3.11) to (3.17). Expression (3.11) defines bounds on generation, while expressions (3.12) and (3.14) constraint minimum down and up time of units, respectively. Expressions (3.16) and (3.17) specify the minimum up and down rates.

**Pre-Dispatch problem**

If the state of each unit is defined, i.e. when the values of $u_{it}$ are fixed, it is possible to calculate the cost incurred by starting and shutting down units. However, to obtain the total operating costs it is still necessary to settle the economical production levels of those units that will be on, so that fuel costs are minimised (expression (3.18)). This problem is known as *Pre-Dispatch*.

In each time interval the production levels are constrained by the forecasted demand (expression (3.19)) and by the power production limits of each unit (expression (3.20)).

$$min \sum_{i=1}^{I} F(P_i) \tag{3.18}$$

$$\sum_{i=1}^{I} P_i = P^d \tag{3.19}$$

$$\underline{P}_i \leq P_i \leq \overline{P}_i \qquad i = 1, \ldots, I \tag{3.20}$$

As referred in [Wood and Wollenberg 1996], the optimality conditions for this problem can be derived by formulating the Lagrangian function $\mathcal{L}$:

$$\mathcal{L} = \sum_{i=1}^{I} F(P_i) + \lambda(P^d - \sum_{i=1}^{I} P_i) \tag{3.21}$$

for which the Kuhn-Tucker conditions [Kuhn and Tucker 1951], that state when the optimum has been reached, are:

$$\lambda = \frac{dF(P_i)}{dP_i} \tag{3.22}$$

i.e. the necessary condition for the existence of a minimum cost operating state is that the incremental cost rate of all units, $\lambda$, is the same. Thus the knowledge of $\lambda$ uniquely specifies the generation level of each unit. That is:

$$P_i = \frac{\lambda - b}{2c} \tag{3.23}$$

Necessarily, expressions (3.19) and (3.20) must hold for that value of $\lambda$.

### 3.1.4 Variants to the base problem

The base thermal UCP consists in the minimisation of total system costs, subject to system and generating units constraints, but other models including more information can be found in the literature. This section introduces some of those models (that tackle emissions, fuel and multi-area constraints), providing the reader with references to different modelling alternatives for each of the three variants. The same notation of section 3.1.3 will be used, whenever it is possible. When necessary, additional notation will be introduced.

#### Emission Constrained Unit Commitment

Due to the emerging importance of environmental issues, the environmental effect of thermal power generation became a major concern in many countries, leading to the development of new and more comprehensive models. Even so, there is still a lack of references in the literature concerning the inclusion of emission issues in Unit Commitment problems, when compared to real time unit dispatch. According to [Talaq et al. 1994], the issue has not received the attention that it deserves partly due to the complexity of the required start-up environmental models.

Among the few works on the subject, there are two distinct lines of research to handle environmental concerns: one where emissions are explicitly stated and modelled as constraints [Gjengedal 1996; Manzanedo et al. 2001; Wang et al. 1995], and another where they are somehow included in the objective function [Srinivasan and Tettamanzi 1997; Kuloor et al. 1992].

In the first line of research we highlight the model presented in [Gjengedal 1996], that seems particularly accurate. As fossil fired power plants pollute the air differently during normal, start-up and shut-down operations, this model evaluates emissions with different functions, depending on whether the units are in steady-state operation or in transition states. Except for the case of $NO_x$, the functions that measure emissions are, in general, similar to those related to operating costs. $SO_2$ and $CO_2$ emissions are directly related to the fuel consumed and can be represented as a function of the unit fuel input/output equation and an emission factor. Start-up emissions depend on the latest period that a unit was operating and may be represented by a two-step or an exponential function similar to (3.7) or (3.8). Shut-down emissions are generally represented by a constant. Finally, $NO_x$ emissions are combustion-process dependent and, in general, cannot be described using the unit fuel input/output equation. However, they can be expressed as a function of the power output similar to the input/output curve.

Emission constraints may limit a single unit, a group of units or the entire system. The emission limits may be given for a single period of time or for a number of intervals. In [Gjengedal 1996] each pollutant has an emission limit that is imposed on the entire system, over the entire planning horizon. In [Manzanedo et al. 2001], the authors consider that there is a maximum allowed value for $SO_2$ and $NO_x$ emissions, over the entire planning horizon. The emissions of $SO_2$ and $CO_2$ are approximated using the heat rate function of the unit, and the $NO_x$ emissions are approximated by a quadratic function of the power output. An environmental tax is assigned to each pollutant. Finally, the model presented in [Wang et al. 1995] does not consider transition states when evaluating emissions, nor differentiates between different kinds of pollutants. The authors do only define an emission function for each unit and an emission limit, for system emission output, for the entire planning horizon.

In a parallel line of research, [Kuloor et al. 1992] and [Srinivasan and Tettamanzi 1997] model emissions as an objective (or part of an objective) to be minimised.

The work by [Kuloor et al. 1992] considers two objective functions: the traditional total operating cost function, and the total emissions of the system. However, the two functions are aggregated into a single one, and the problem is tackled accordingly. A constraint concerning emission limits is also included in this model.

In [Srinivasan and Tettamanzi 1997], the aim is also to minimise total operating costs and emissions and the problem is solved as a truly multiobjective problem, without aggregating the two objectives. $SO_2$ emissions are measured as a function of an unit fuel input/output equation, as in [Gjengedal 1996], regarding also the transition effects. $CO_2$, $NO_x$ and particulate emissions are evaluated by other second order functions, that are not strictly related to the input/output equation.

**Fuel Constrained Unit Commitment**

In a generator company where thermal production is dominant, fuel costs are an important slice in the total operating costs and, therefore, a correct management of fuel becomes a major point of concern in daily operation planning. Fuel management may be constrained by several factors, e.g. fuel contracts, that usually specify a minimum/maximum consumption requirement for a given time duration, congestion in the fuel delivery system, limited storage facilities, etc, and naturally influences the production decisions involved in the UCP (e.g. the decision of keeping a unit on over the entire planning horizon may not be possible in case there is not enough fuel available). The inclusion of fuel management in short-term production planning is addressed by the *Fuel Constrained Unit Commitment Problem*.

In its general form, the problem is rather complex. However, several authors consider some assumptions that drastically reduce the level of complexity [Lee 1991]. They are: 1) each thermal unit can be supplied by only one type of fuel, and/or 2) each thermal unit can burn up to two types of fuel, one constrained and the other unconstrained. In such cases, there is no correlation of fuels and the fuel coordination problem no longer exists. In [Aoki et al. 1987], [Aoki et al. 1989], [Lee 1989] and [Tong and Shahidehpour 1990] the simplification assumptions prevail when defining the mathematical model.

Lee in [Lee 1989] addresses a model developed for Oklahoma Gas & Electric Co (OG&E), that considers constraints on gas consumption and gas delivery. Gas is traded by take-or-pay contracts where a minimum trading quantity (of fuel or other resource) is defined for each period. In this type of contract, if the consumption of that resource does not reach the minimum quantity, the defined amount still has to be paid. Therefore, some constraints define a minimum level of gas consumption among all gas-fired units, for each period of time. A target for gas consumption over the entire Unit Commitment horizon is also defined. Constraints on gas delivery include minimum hourly gas take, to prevent the build-up of pipeline pressure (that can cause shut-down of gas wells), maximum hourly gas deliverability and maximum daily gas take turn-up ratio, reflecting the dynamic characteristics of the gas delivery system.

Take-or-pay contracts are also considered in [Wong and Wong 1996], but they are modelled in a different way from that proposed in [Lee 1989]. They are included in the objective function, by making a distinction between non fuel constrained and fuel constrained units.

Non fuel constrained operating costs are evaluated through the traditional function introduced in section 3.1.3 and the cost of running the fuel constrained units is given by expression (3.24), where the following additional notation is used:

$C$ – number of fuel constrained units

$C_i$ – contracted take-or-pay fuel cost of unit $i$

$n_t$ – number of hours in interval $t$

$n_t f_{it}(P_{it})$ – cost of non-fuel constrained unit $i$, in period $t$

$$F_C = \sum_{i=1}^{C} \left\{ \max \left( C_i, \sum_{t=1}^{T} n_t f_{it}(P_{it}) \right) \right\} \tag{3.24}$$

By doing so, when determining the most economical generation schedule, if the amount of fuel consumed by a fuel-constrained unit is above the minimum amount stipulated in the contract, the unit is tackled as a standard non-constrained unit.

In [Aoki et al. 1987] and [Aoki et al. 1989], the evaluation of fuel consumption differs, depending on whether the generator is in steady-state operation or starting-up. The concept of "contract" is not present in their work and the total fuel consumption, over the entire planning horizon, must be strictly equal to a given value. A similar modelling approach is followed by [Tong and Shahidehpour 1990] where, instead of fuel quantities, prices are used, accordingly.

In contrast with the previous models, some authors studied the problem where multiple types of fuel can supply the same unit. In [Cohen and Wan 1987] and in [Lee 1991], each contract for fuel supply has a lower and upper limit, the total fuel consumption over the planning horizon being expressed as a function of generation levels that cannot go above or below those limits. In [Vemuri and Lemonidis 1992], minimum and maximum limits on contracts are also specified, and there are no restrictions on the number of constrained fuels for each unit. Moreover, the model considers that a contract can supply several units and a unit can be supplied by several contracts. The model differs from the previous one because minimum and maximum limits on contracts are defined on a hourly, daily and planning horizon basis.

### Multi-area Unit Commitment

The objective of a multi-area UCP is to determine the optimal (or near optimal) commitment strategy for units located in distinct areas, connected through tie-lines. A main specific feature of this UCP variant is that the import and export of power from/to each area is allowed and, as a consequence, the local generation (i.e. the generation within a single area) is not necessarily balanced with the local demand. Still, the total system generation must be equal to the total load demand.

This is probably the variant of the UC problem that introduces more changes in the base model. As so, the mathematical model for the multi-area UC is described in this section. We consider the model presented in [Ouyang and Shahidehpour 1991] that covers typical constraints encountered in the operation of the North American utility industry. The following additional notation is necessary to introduce the model:

$a$ – area identification ($a = 1, \dots, A$)
$I_a$ – number of units in area $a$
$D_{at}$ – total load demand in area $a$, in period $t$
$G_{at}$ – total import power to area $a$, in period $t$
$\overline{G}_a$ – maximum import power to area $a$
$H_{at}$ – total export power from area $a$, in period $t$
$\overline{H}_a$ – maximum export power from area $a$
$I_a$ – total number of units in area $a$
$P_{at}$ – power generation in area $a$, in period $t$
$\underline{P}_{ia}$ – lower limit of unit $i$ in area $a$
$\overline{P}_{ia}$ – upper limit of unit $i$ in area $a$
$P_{iat}$ – power generation of unit $i$, in area $a$, in period $t$
$R_{at}$ – spinning reserve of area $a$, in period $t$
$S_{at}$ – total commitment capacity, for area $a$, in period $t$
$D_t$ – total system demand, in period $t$
$W_t$ – net power exchange with outside systems
$u_{iat}$ – state of unit $i$, in area $a$, in period $t$

The new/adapted constraints, associated to the multi-area problem are the following:

$$\sum_{a=1}^{A} P_{at} = \sum_{a=1}^{A} D_{at} + W_t \qquad t = 1, \dots, T \tag{3.25}$$

where,

$$P_{at} = \sum_{i=1}^{I_a} P_{iat} \tag{3.26}$$

$$\sum_{i=1}^{I_a} \overline{P}_{ia} u_{iat} \geq D_{at} + R_{at} + H_{at} - G_{at} \qquad t = 1, \ldots, T \qquad a = 1, \ldots, A \tag{3.27}$$

$$\sum_{i=1}^{I_a} P_{iat} \leq D_{at} + \overline{H}_a \qquad a = 1, \ldots, A \tag{3.28}$$

$$\sum_{i=1}^{I_a} P_{iat} \geq D_{at} - \overline{G}_a \qquad a = 1, \ldots, A \tag{3.29}$$

$$\sum_{a=1}^{A} H_{at} - \sum_{a=1}^{A} G_{at} + W_t = 0 \qquad t = 1, \ldots, T \tag{3.30}$$

$$\sum_{i=1}^{I_a} P_{iat} \leq \sum_{i=1}^{I_a} \overline{P}_{ia} - R_{at} \qquad a = 1, \ldots, A \quad t = 1, \ldots, T \tag{3.31}$$

$$\sum_{i=1}^{I_a} P_{iat} \geq \sum_{i=1}^{I_a} \underline{P}_{ia} - R_{at} \qquad a = 1, \ldots, A \quad t = 1, \ldots, T \tag{3.32}$$

System power balance is represented by equation (3.25) and the spinning reserve constraints in each area by equation (3.27). Power generation upper and lower limits due to tie-line constraints are represented by equations (3.28) and (3.29), respectively. Import/export balance and area generation limits are modelled by equations (3.30) to (3.32).

The objective function is to minimise the entire operating cost, as follows:

$$\min \sum_{a=1}^{A} \sum_{t=1}^{T} \sum_{i=1}^{I_a} [F(P_{iat}) + SC(OFF_{iat}, u_{iat})] \tag{3.33}$$

Other multi-area UC models are presented in, e.g. [Lee and Feng 1992], [Wang and Shahidehpour 1992] and [Lee et al. 1994].

### “Hybrid” models

Naturally, there is not a strict border between the sets of problems just described and it is possible to find research work where two, or more, of those subjects are considered in a single formulation.

In [Bai and Shahidehpour 1997], for example, the authors introduce a UCP with fuel and emission constraints. [Al-Agtash and Su 1998] present a hydrothermal problem with environment concerns, where fly ash emission of $NO_x$ and $SO_2$ are constrained to an upper value. The model presented in [Kuloor et al. 1992] tackles the thermal UCP, considering emission, fuel and multi-area constraints, its emission concerns having been described in section 3.1.4. Fuel consumption is limited to a minimum and maximum value and and multi-area constraints limit inter-area power exchange and influence each area's spinning reserve requirements.

### 3.1.5 Solution methodologies

The first studies on the UCP date back to the 40's [Li et al. 1997b] and, since, then several methodologies ranging from very straightforward to more complicated methods have been proposed. Initially, methods for unit commitment based on relative cost priority lists dominated the power industry in this area. During the 60's and 70's Dynamic Programming (DP) approaches have been suggested and, since the 70's Lagrangian Relaxation (LR) based unit commitment methods have been successfully applied in Energy Management Systems (EMS). Later, methods based on Artificial Intelligence and on Metaheuristics have also been proposed.

In this section we give an overview of solution methodologies, from the first attempts to the state-of-the-art techniques. Due to their historical importance, Dynamic Programming and Priority List based methods are introduced first. The section proceeds with a reference to Lagrangian Relaxation, to constructive methods and to Metaheuristics and Evolutionary Algorithms. It concludes with a summary of other techniques that, though not so frequent, have received some attention in the literature. Additional information on the UCP and on its solution methodologies can be found in the surveys by [Sheblé and Fahd 1994], [Sen and Kothari 1998a] and [Yamin 2004].

#### Priority List based methods

Priority List based methods date back to the late 60's, early 70's, one of the early procedures being reported in [Happ et al. 1971]. It ranks all units in the system according to a merit function and, based on this ranking and for each time interval, the units are switched on (or off) until load and spinning reserve constraints are fulfilled. The commitment priority order may be determined by, e.g. the Average Full Load Cost (AFLC) (expression (3.34)) or, alternatively, by the Commitment Utilisation Factor (CUF) of each unit (expression (3.35)).

$$\text{AFLC} = \frac{\text{working cost at maximum production level}}{\text{maximum production level}} \tag{3.34}$$

$$\text{CUF} = \frac{\text{load - reserve requirements}}{\text{total commited output}} \tag{3.35}$$

Although these methods are particularly appealing, due to their implementation simplicity, the quality of the solutions is normally weak. This is due to the difficulty in correctly ascertaining the relative efficiency of the units, as important system information is neglected in these functions. Even so, this is still a popular approach in commercial software packages and also in market environments, where suppliers are ordered according to their bid prices and selected, in that order, until load demand is satisfied.

**Dynamic Programming**

Dynamic Programming was first used in the 60's to solve the UCP. In the earlier attempts, the commitment of generating units was determined independently for every time period [Lowery 1966]. For each time period, a stage was assigned to different output levels (states) of a generating unit and the total number of stages was equal to the number of units in the system. For every unit, the start-up cost was assumed to be constant and the total cost of every output level was equal to the sum of production and start-up costs. Such an approach presented a major limitation as it could not take into account the coupling of adjacent time periods, and therefore the time dependent start-up cost was not correctly modelled. Moreover, it could not appropriately handle the minimum up and down time constraints, unless some heuristic procedures were included.

Later, [Pang and Chen 1976] developed an algorithm where each stage represented a particular time period and, in every stage, the corresponding states represented different combinations of commitment states (on/off) for the generating units in that specific period. This approach is particularly interesting, because it maintains solution feasibility. However, it suffers from the "curse of dimensionality" and is not directly applicable to real size problems in its standard form [Hobbs et al. 1988]. To reduce problem dimensionality, different strategies that truncate the hourly state space have been developed, e.g. DP-Sequential Combination, DP-Truncated Combination, etc. Additional heuristic procedures have also been combined with DP to achieve a further reduction of the searching space and to speed up the execution. In [Ouyang and Shahidehpour 1992], for example, DP is combined with neural networks.

The results obtained are, naturally, suboptimal but the computation times are considerably smaller.

**Lagrangian Relaxation**

Lagrangian Relaxation was first used to solve the UCP nearly three decades ago by [Muckstadt and Koenig 1977] and since then it has been widely applied to the problem, as it can be confirmed by the huge number of publications in this area, e.g. [Bertsekas et al. 1983; Merlin and Sandrin 1983; Aoki et al. 1987; Bard 1988; Zhuang and Galiana 1988; Aoki et al. 1989; Tong and Shahidehpour 1990; Baldick 1995; Gjengedal 1996; Takriti and Birge 2000; Borghetti et al. 2001b]. The basic idea of the approach is to relax system constraints by using Lagrangian multipliers. The resulting problem is dualised and decomposed into a set of smaller problems, one for each generator, that can be solved more easily. Once the values of multipliers have been fixed, each separated subproblem is solved with the constraints that represent the operating characteristics of the corresponding unit. The entire solution procedure is an iterative process that successively solves subproblems and adjusts the multipliers, according to the extent of violation of system constraints.

According to [Sen and Kothari 1998a], the main advantages of the approach are the possibility of decomposing the large scale UCP into small subproblems, the relative easiness to incorporate other constraints, its efficiency and its systematic implementation. A major drawback is the difficulty in obtaining feasible solutions, due to the dual nature of the algorithm. To overcome this difficulty, an Augmented Lagrangian Relaxation (ALR) has been developed and used in e.g. [Wang et al. 1995], [Batut and Renaud 1992] and [Beltran and Heredia 2002]. ALR is a combination of penalty and LR methods where quadratic penalty terms, associated to power demand, are added to the objective function. A major advantage of this approach is that it may obtain a feasible primal solution in cases where the classical Lagrangian Relaxation presents a duality gap. Furthermore, the dual function associated to the Augmented Lagrangian function is differentiable in cases where the LR presents a non-differentiable dual function. However, when solving non-convex problems, as it is the case with the UCP, the ALR method may reach a local optimum, and give no information on its relative quality.

Lately, LR has also been successfully used in conjunction with other methodologies. In [Cheng et al. 2000] a joint approach of Lagrangian Relaxation and Genetic Algorithms (GA) is presented. GA are incorporated in LR, to update the Lagrangian multipliers and improve the performance of LR. In [Valenzuela and Smith 2002] LR is used to obtain good initial solutions to build the initial population of a memetic algorithm.

Within the framework of deregulated energy markets, the LR technique has been receiving particular attention, as it gives “price signals” to the generator companies.

### Constructive heuristics

Several constructive heuristics, rather than the priority list based ones, have been designed to solve the UCP. One such heuristic is the Unit Decommitment method, presented in [Tseng and Oren 1997]. It serves as a post-processing tool to improve the solution quality of other methods used for the UCP and works as follows: given an initial feasible solution, where all the available units are committed over the planning horizon, the method uses DP to determine an “optimal” strategy to decommit overcommitted units, according to some specified economic criteria. The decommitment process is concluded when no further reduction in the total cost is possible, or the unit schedules of two consecutive iterations over the time period remain unchanged, without any violation of the spinning reserve constraint. In [Tseng et al. 2000] the authors extend the method to a more general formulation. They also show that it may be viewed as an approximate implementation of the LR approach and that the number of iterations that it requires is bounded by the number of units.

Successive Approximation in Solution Space (SASS) is presented in [Sheblé and Grigsby 1986]. The algorithm is strongly based on the fundamental concepts of Decision Analysis and Dynamic Programming.

The approach presented in [Sen and Kothari 1998b] reduces the number of units in the power system to the lowest possible number, according to their fuel/generation cost characteristics. The reduced system is solved through modified DP.

Other problem-specific heuristics can be found in, e.g. [Tong et al. 1991], [Lee 1988] and [Sheblé 1990].

### Metaheuristics and Evolutionary Algorithms

More recently, Metaheuristics and Evolutionary Algorithms have also been regarded as interesting tools to tackle the UCP, a literature review showing that approaches based on GA prevail: [Sheblé and Fahd 1994; Ma et al. 1995; Kazarlis et al. 1996; Maifeld and Sheblé 1996; Sheblé et al. 1996; Yang et al. 1996; Swarup and Yamashiro 2003]. Nevertheless, other approaches based on, e.g. Simulated Annealing, Tabu Search or hybridisations of these methods, have also been developed.

Mainly to reduce the computational time of GA approaches, some authors have designed additional tools to include in the base algorithm. [Orero and Irving 1997a,b] consider a hybrid GA approach where the solution obtained with a priority list method is part of its initial population. By doing so, they reach better results than the standard GA method presented in [Orero and Irving 1996].

In [Valenzuela and Smith 2002], the authors study a GA with Local Search. In each generation, Local Search with two distinct neighbourhood operators is applied to the best solution of the new generation if, and only if, the solution is better than the best solution found so far. A modified version, where LR is first used to obtain an initial solution, is compared to the base version leading to better results for large size instances.

[Yang et al. 1997] increases the search speed through a parallel implementation.

Other Evolutionary and Immune Algorithm approaches can be found in [Duo et al. 1999] and in [Huang 1999], respectively.

[Mantawy et al. 1999] develop a hybrid method, the core of the algorithm being based on GA. Tabu Search is used to generate offsprings in the reproduction phase of GA, and Simulated Annealing is used to accelerate its convergence.

Concerning Local Search based metaheuristics, the work by [Zhuang and Galiana 1990] was probably the first attempt to solve the UCP through Simulated Annealing (SA). Other approaches are those by [Mantawy et al. 1998a] and by [Yin Wa Wong 1998]. Tabu Search approaches can be found in, e.g. [Mantawy et al. 1998b] and [Borghetti et al. 2001b]. An implementation based on GRASP is presented in [Viana et al. 2003b].

Being a central point of this thesis, the application of metaheuristics to the UCP will be further developed in Chapter 4.

### Other approaches

Other paradigms such as Neural Networks, Expert Systems, Fuzzy Logic and Constraint Logic Programming have also been considered when solving the UCP.

Neural Networks (NN) are inspired in neuronal behaviour and have the capability of adaption and generalisation under changing conditions. A Neural Network approach is proposed in [Sasaki et al. 1992] and in [Ouyang and Shahidehpour 1992]. For the latter there is a pre-processor stage, where a load pattern matching scheme is performed to choose the optimal solution in the database that is closer to the given load profile. The solution feasibility is recovered in an intermediate step and, in a post-processing stage, a trained NN performs adjustments to refine the final solution quality. The estimation of NN parameters is based on a database holding typical load curves and corresponding UC schedules. The pattern of the current load curve is compared to the information in the database to select the most economical UC schedule. If the solution is not feasible for the entire period, it can be used as the starting point for a near optimal solution.

Several Expert Systems (ES) have also been developed to assist operators in scheduling thermal generating units. In [Tong et al. 1991] the authors present an algorithm that uses priority list based heuristics, in the form of reference rules, to find a suboptimal schedule for a given load pattern. Afterwards, an ES tries to improve that solution. Other ES approaches can be found in [Mokhtari et al. 1988], [Ouyang and Shahidehpour 1990], [Kothari and Ahmad 1993] or [Li et al. 1993].

Constraint Satisfaction Techniques reduce the search space and enhance the efficiency of logic programming. In [Huang et al. 1998] Constraint Logic Programming (CLP) is used to solve the problem. The method combines constraint satisfaction (that includes backtracking, forward checking and looking ahead tools) and Branch-and-Bound search techniques, with logic programming.

As a final reference, Fuzzy Logic based approaches are presented in [Saneifard et al. 1997] and in [Mantawy and Abdel-Magid 2002].

## 3.2 Hydrothermal coordination

### 3.2.1 Context

Other sources of energy rather than thermal are available in practice, requiring higher levels of management and coordination, to minimise global production costs. A major source of energy is that stored in hydraulically coupled river basins where, as the energy that is stored is finite (it depends on the water stored in each reservoir, and also on the topology of the hydro system) water usage should be managed in an effective way, to produce as much energy as possible with the available resources.

If considered alone, the aim of hydro-scheduling is to manage water usage, by determining the hourly production of each hydro unit, water flows through generating stations, reservoir releases and storage levels, so that the energy production from hydro resources meets the demand. However, when belonging to a “hybrid” production system, where more than one source of energy is available, the hydro system must be coordinated with the other systems, to achieve a good global management of resources.

The hydrothermal coordination (HTC) problem arises when the production system comprises hydro and thermal units and its aim is to determine the commitment of thermal units and their generation levels, as well as the hydro schedules, with the objective of minimising the total operating costs, satisfying constraints from the hydro system, the thermal system and the power system network. The characteristics and limitations of thermal units have already been extensively described in section 3.1: they have high operation costs, they have pollutant emissions that must be controlled, their power output cannot change quickly, etc. However, they present the major advantage of being fed by primary sources of energy that can be considered limitless. On the other hand, hydro units have low operation costs, they are emission free and their production levels may change quickly. But water resources are limited and the decisions made in a given moment may have drastic consequences in the future. If, for example, the hydro energy that is stored today is used, and a drought occurs, it may be necessary to use expensive thermal energy in the future, or even interrupt the energy supply. On the other extreme, if reservoir levels are kept high through a more intensive use of thermal generation, and high water inflows occur in the future, there can be a spillage in the system (the spillage is the water that passes the dam without being turbinated), which represents a waste of energy and, as a consequence, an increase in operation costs. Therefore to achieve an efficient usage of the whole resources, a strong coordination between the thermal and hydro plants is required to satisfy the customer demand at low costs, while operational and security constraints are met.

The important role that this problem plays in practice is reflected in the amazing number of publications concerning the subject, where different models and optimisation techniques are proposed. Being $\mathcal{NP}$-hard, real size problems are not solvable through exact techniques and, as a result, heuristic based approaches have been proposed [Al-Agtash and Su 1998; Bai and Shahidehpour 1996; Chen and Chang 1996; Li et al. 1997a; Orero and Irving 1998; Rudolf and Bayrleithner 1999]. Although following different paradigms, they have in common the fact of being often based on decomposition methods, the overall problem solution being obtained by a proper coordination of the solution of the thermal and hydro scheduling subproblems. Usually, the hydro scheduling subproblem is solved first based on assumed thermal marginal costs. Then, for the remaining demand (the total load demand minus the hydro production) the thermal subproblem is solved and, based on the thermal marginal costs obtained, the hydro problem is solved again. The process is repeated until the marginal costs or hydro generation converge [Ferreira 1994].

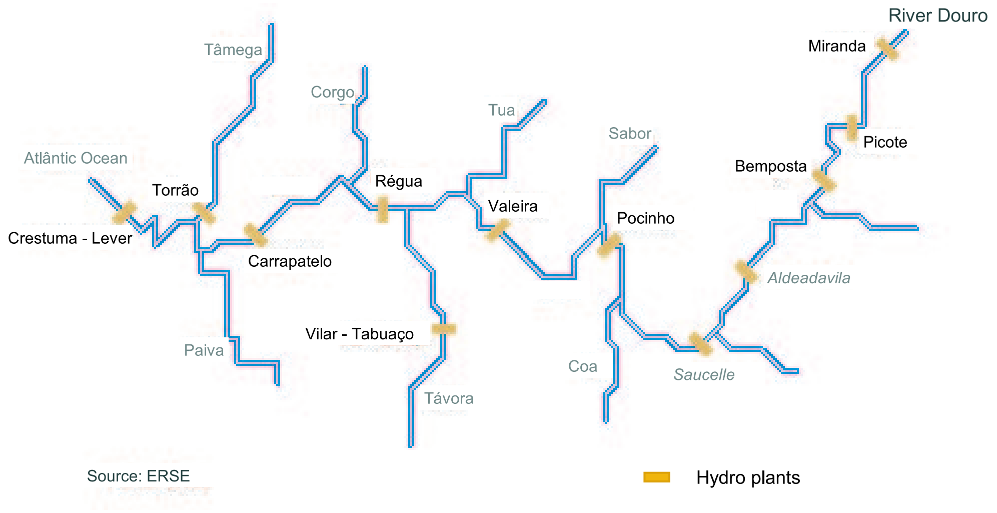


**Figure 3.4:** Plan of Douro hydro system

### 3.2.2 Brief introduction to hydro systems

A hydro production system has a natural network structure, where the arcs represent water flows, and the nodes the hydro plants and their associated reservoirs. As an example, the Douro System in the Portuguese hydro production system, presented in Figure 3.4, might well be represented by the network in Figure 3.5 (the Aldeadavila and Saucelle hydro plants were excluded from this representation because they do not belong to the Portuguese production system). Such a representation allows a faster and more straightforward analysis of the system. We can for example notice that, for a given period of time, the water that can be discharged in Carrapatelo depends on its own discharges, on previous discharges in Régua, on inflows from River Corgo and on natural inflows (e.g. rain). Similarly, the water that can be discharged in Crestuma depends on its own discharges, on the discharges in Carrapatelo and Torrão, on the inflows from River Paiva and, again, on other natural inflows.

The complexity of the system may strongly vary, according to the number of rivers and to the way they are interconnected. The number, location and type of each plant do also affect system complexity. To take advantage of the potential energy of water plants, isolated hydro plants are usually hydraulically coupled, forming a cascade system (Figure 3.6). Hence, the water discharged in one station travels along the river to the next downstream station and one typically has to consider the whole river system when planning production in hydro stations, thus increasing the complexity of the problem.

Different plant types may exist within a system. They are divided in three sets:

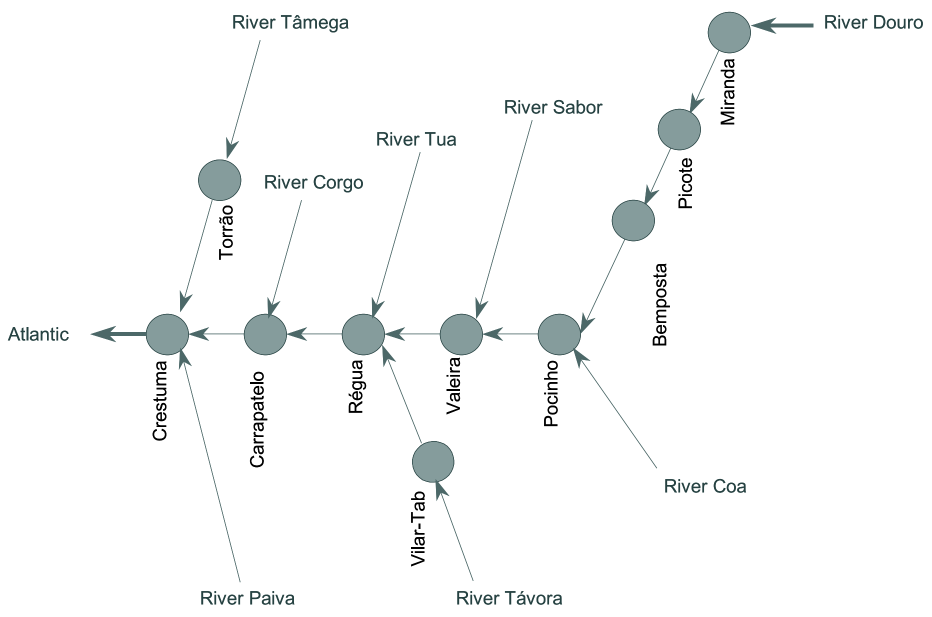


**Figure 3.5:** Network of Douro hydro system

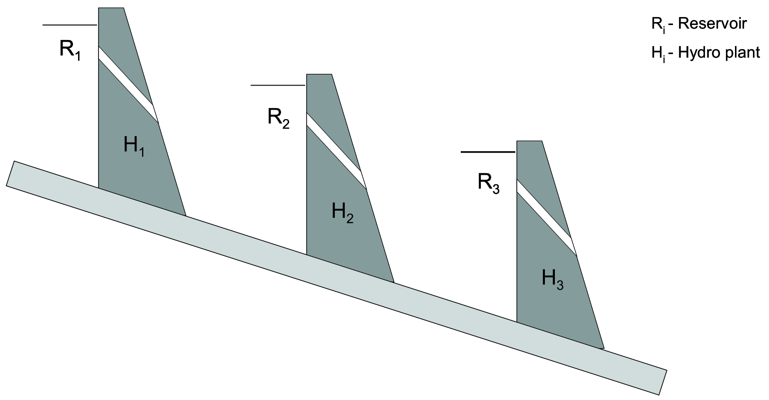


**Figure 3.6:** A cascade system

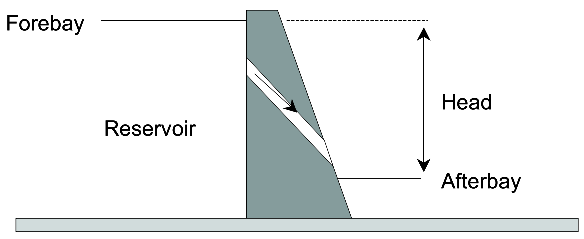


**Figure 3.7:** Head of the dam

i. *Run-of-the-river type* – when there is no reservoir associated and the water is immediately turbinated.

ii. *Storage type* – when the water may be stored in a reservoir for a considerable amount of time, being turbinated when necessary. An incorrect management of water or unexpected inflows, may lead to the decision of spillage (i.e. wasting water without producing energy), if the water stored in the reservoirs reaches their maximum storage capacity.

iii. *Pump-storage type* – this type of hydro plant is similar to Storage plants. However, it has additional mechanisms that allow water to be pumped up to fill a higher level reservoir.

Concerning the power output of a unit, strictly speaking, it is proportional to the throughput of discharged water and to the head of the dam, i.e. the difference in elevation between the upstream and downstream water surfaces (see Figure 3.7). Thus, depending on the current head of a reservoir, one should pick different input/output characteristics (Figure 3.8) to calculate hydro production levels. Despite that, it has been generally accepted that, when the reservoirs are sufficiently large, one can neglect the variation of their volume for short periods of time and, consequently, consider that the power output is proportional only to the throughput of discharged water.

### 3.2.3 Objective function and constraints

#### Objective function

Although the definition of a base thermal UCP model is rather consensual, the definition of a "base" hydrothermal UCP model is not as straightforward, a literature review clearly showing that different authors consider different models, and that there is not a prevailing

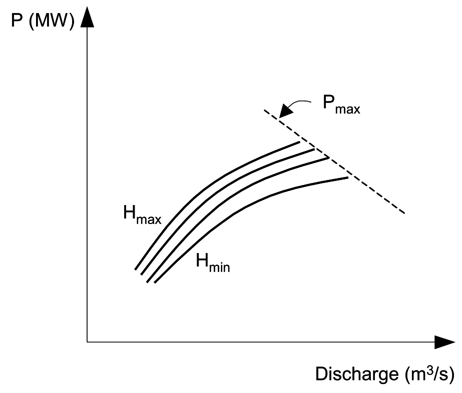


**Figure 3.8:** Typical input/output characteristic for a variable-head hydro plant

model. The main points of distinction between these models have to do with the inclusion (or not) of hydro costs in the objective function, with the types of hydro units that are modelled and with the inclusion (or not) of water travelling times between reservoirs.

Concerning the objective function, although it has generally been assumed that hydro production costs are negligible, when compared to thermal costs, some authors claim that they should also be evaluated. The way in which these costs are measured does however differ. In [Guan et al. 1997] and [Ni et al. 1999] hydro start-up costs (as a function of the time that the unit was off) are included in the objective function. Oliveira et al. in [Oliveira et al. 1993] measure hydro costs as a function of water discharges and Li et al. in [Li et al. 1998] consider that the two costs (start-up and operation costs) should be modelled.

In [Rajaković and Ružić 1993], [Ružić et al. 1996] and [Ružić and Rajaković 1998] the authors evaluate the usage in excess of available water. According to them, the reasoning behind the definition of this evaluation criterion is that if final reservoir levels are violated, that will be reflected in additional thermal costs, in the next optimisation period. Due to that, the penalty term associated with the excessive usage of water in the current period is equal to the additional thermal costs in the next period. These costs depend on the expected system marginal cost (the cost of providing an additional kilowatt-hour of energy output above any energy currently being produced), and on the amount of the additional thermal energy that is needed in the next optimisation period.

Several other authors [Rudolf and Bayrleithner 1999; Redondo and Conejo 1999; Chen and Chang 1996; Guan et al. 1994; Yan et al. 1993; Zhang et al. 1999; Wu et al. 2000; Wang and Shahidehpour 1993; Orero and Irving 1998; Luh et al. 1998; Werner and Verstege 1999] do however neglect hydro costs, the objective function measuring only the thermal operation costs. Even so, one should not forget that when intermediate and final limits for reservoir levels are defined, they were previously settled in long or medium term planning, based on the cost of opportunity of water. Therefore, when one moves to short-term planning, this cost is implicitly reflected in the volume targets.

Concerning hydro transition costs, i.e. hydro start-up and shut-down costs, as far as the author knows, there has not been much work on evaluating its influence on the performance of a hydrothermal system. Although the work presented by [Eliasson 2002] suggests that they should not be neglected, further research is still necessary to reach more robust conclusions.

**Constraints**

The model constraints usually depend on the type of hydro units that are modelled, and on whether water travelling times are considered or not. The models presented in [Oliveira et al. 1993], [Rakić and Marković 1994], [Guan et al. 1994], [Ni et al. 1999], [Zhang et al. 1999], [Guan et al. 1997] consider the existence of different types of hydro units, with different capabilities. In contrast, [Gollmer et al. 1999], [Bai and Shahidehpour 1996], [Wong and Wong 1994], [Al-Agtash and Su 1998], [Fuentes and Quintana 2002], [Ružić and Rajaković 1998], [Rajaković and Ružić 1993] and [Yan et al. 1993] do only work with one type of units. Water travelling times are considered in [Ramos et al. 2001], [Oliveira et al. 1993], [Li et al. 1997a], [Bai and Shahidehpour 1996] and [Chang et al. 2001], assuming that the travelling time of water, from a unit to its successors, is a constant. This is not true in reality since larger water flows usually imply shorter travelling times but, as taking into account real times would complicate the model severely, the approximation has been considered as acceptable. The work by [Rudolf and Bayrleithner 1999], [Gollmer et al. 1999] and [Wong and Wong 1994] consider the extreme case of water travelling times being equal to one period.

Some models do also address dynamic hydro constraints such as units minimum up and down times and plants ramp rate constraints [Li et al. 1997a], to prevent frequent switching of hydro units, which is undesirable because of mechanical stress.

In [Wu et al. 1991], Wu et al. refer some environmental concerns related to the hydro system, namely: 1) backwater curve – the head-pond elevation should be lowered during the hours of high inflows, to prevent upstream flooding, 2) maximum head-pond change rate – to prevent the reservoir from erosion, or for *aqua culture* reasons, the rate of change of head-pond elevation between two consecutive hours should be limited within a specified range.

The issues that usually prevail in constraint modelling are the time coupling effect of the hydro system, where the water flow in an earlier time interval affects the discharge capabilities at later periods of time, the cascade nature of the hydraulic system, the reservoir inflows, the physical limitations on the reservoir storage and discharge rates. A short description of these constraints is given below.

*Minimum and maximum operating levels for generation and pumping*
Due to technical limitations, the power plant cannot produce/pump neither below nor above a production/pumping level.

*Water spillage limits*
In extreme cases (danger of floods, for example) one may decide to spill water from the reservoirs, in a controlled way. The amount of water that may be spilled is limited to a maximum volume.

*Reservoir intermediate and final volumes*
To avoid that one depletes the reservoirs, minimum reservoir levels must be defined. Besides, the physical limitations of the reservoir require that the volume of water stored does not go above a certain value.

*Water dynamic balance*
These constraints, also referred to as "continuity equations", define the volume of water in each reservoir, in each period of time. They depend on the discharge/spilling levels from the current reservoir, on upstream reservoirs discharges/spillages (and on water travelling times from those reservoirs), on inflows and on pumping.

*Load and reserve requirements*
The load and reserve constraints introduced in section 3.1.2 need to be adapted to include hydro production.

In the following section we propose a model that is as general as possible and that tries to encompass the concerns embedded in the generality of the models proposed until now. The base thermal UCP model has already been extensively described in previous sections of this chapter and, unless strictly necessary, no further considerations will be done on the subject.

### 3.2.4 Mathematical model

The model presented in this section is based on the following assumptions: only the thermal costs are considered in the objective function, hydro costs being implicitly modelled in reservoir limits; the system comprises must-run, storage and pump-storage units; and water travelling times between units differ but do not vary with the volume of the water discharged. Head effects as well as losses due to friction effects have been neglected. Therefore, the power produced by a hydro unit $h$ is modelled as a linear function – expression (3.36). $\rho_h$ is a constant and $d_h$ is the volume of water discharged. Finally, hydro generation is modelled at the reservoir level, each reservoir being labelled with the same number of the corresponding storage or pump-storage unit.

$$P_h = \rho_h d_h \tag{3.36}$$

#### Additional notation

*Indexes*

$k$ – must-run units ($k$ = *1, ..., K*)

$p$ – pump-storage units ($p$ = *1, ..., P*)

$s$ – storage units ($s$ = *1, ..., S*)

$\Omega_h^U$ – set of hydro units located immediately before (upstream) hydro unit $h$

$\Omega_h^D$ – set of hydro units located immediately after (downstream) hydro unit $h$

$\Omega_h^P$ – set of units that pump water to the reservoir associated to hydro unit $h$

*Decision variables*

$d_{kt}$ – discharge of must-run unit $k$, in period $t$

$d_{pt}$ – discharge of pump-storage unit $p$, in period $t$

$d_{st}$ – discharge of storage unit $s$, in period $t$

$sh_{kt}$ – spillage of must-run unit $k$, in period $t$

$sh_{pt}$ – spillage of pump-storage unit $p$, in period $t$

$sh_{st}$ – spillage of storage unit $s$, in period $t$

$p_{pt}$ – water pumped from unit $p$, in period $t$

$v_{pt}$ – volume of reservoir associated to pump-storage unit $p$, in period $t$

$v_{st}^S$ – volume of reservoir associated to storage unit $s$, in period $t$

$g_{pt}$ – inflow to reservoir associated to pump-storage unit $p$, in period $t$

$g^S_{st}$ – inflow to reservoir associated to storage unit $s$, in period $t$

*Auxiliary variables*

$$\gamma_{pt} = \begin{cases} 1 & \text{if pump-storage unit } p \text{ is generating in period } t \\ 0 & \text{otherwise} \end{cases}$$

$$\nu_{pt} = \begin{cases} 1 & \text{if pump-storage unit } p \text{ is pumping in period } t \\ 0 & \text{otherwise} \end{cases}$$

$$\gamma_{kt} = \begin{cases} 1 & \text{if must-run unit } k \text{ is generating in period } t \\ 0 & \text{otherwise} \end{cases}$$

$$\gamma_{st} = \begin{cases} 1 & \text{if storage unit } s \text{ is generating in period } t \\ 0 & \text{otherwise} \end{cases}$$

*Problem parameters*

$\underline{v}_p, \overline{v}_p$ – minimum and maximum levels of the reservoir associated to pump-storage unit $p$

$\underline{v}^S_s, \overline{v}^S_s$ – minimum and maximum levels of the reservoir associated to storage unit $s$

$V_{p0}$, $V^S_{s0}$ – initial volume of the reservoirs associated to pump-storage unit $p$ and to storage unit $s$, respectively

$\underline{d}_k, \overline{d}_k$ – minimum and maximum discharge levels of must-run unit

$\underline{d}_p, \overline{d}_p$ – minimum and maximum discharge levels of pump-storage unit $p$

$\underline{d}_s, \overline{d}_s$ – minimum and maximum discharge levels of storage unit $s$

$\underline{p}_p, \overline{p}_p$ – minimum and maximum pumping level of pump-storage unit $p$

$\overline{s}_p$ – maximum spillage level of pump-storage unit $p$

$\overline{s}^S_s$ – maximum spillage level of storage unit $s$

$\overline{s}^K_k$ – maximum spillage level of must-run unit $k$

$\Theta_p$ – the inverse of the efficiency (the fraction of the input energy that can be converted into useful output) of the pumping process

**Constraints**

In this model, load demand constraints are given by expression (3.37) and the spinning reserve by expression (3.38). The water dynamic balance of each reservoir is given by expressions (3.39) and (3.40). Expressions (3.41) and (3.42) define bounds on reservoir limits, while expressions (3.43) to (3.45) define bounds on turbination levels. Bounds on spillage limits are set in expressions (3.46) to (3.48) and bounds on pump limits are specified in expression (3.49). Finally, expression (3.50) to (3.53) set the reservoirs initial and final volumes.

$$\sum_{i=1}^{I} P_{it} + \sum_{k=1}^{K} \rho_k d_{kt} + \sum_{p=1}^{P} \rho_p d_{pt} + \sum_{s=1}^{S} \rho_s d_{st} - \sum_{p=1}^{P} \Theta_p p_{pt} = P_t^d \qquad t = 1, \ldots, T \tag{3.37}$$

$$\sum_{i=1}^{I} \overline{P}_i u_{it} + \sum_{k=1}^{K} \rho_k \overline{d}_k \gamma_{kt} + \sum_{p=1}^{P} \rho_p \overline{d}_p \gamma_{pt} + \sum_{s=1}^{S} \rho_s \overline{d}_s \gamma_{st} - \sum_{p=1}^{P} \Theta_p p_{pt} \nu_{pt} \geq P_t^d + R_t \qquad t = 1, \ldots, T \tag{3.38}$$

$$\begin{aligned} v_{p,t+1} = v_{pt} - d_{pt} - sh_{pt} + g_{pt} + p_{pt} + \sum_{h \in \Omega_p^P} p_{h,t-\tau_h} + \sum_{h \in \Omega_p^U} (d_{h,t-\tau_h} \\ + sh_{h,t-\tau_h}) \qquad p = 1, \ldots, P \quad t = 1, \ldots, T \end{aligned} \tag{3.39}$$

$$\begin{aligned} v^S_{s,t+1} = v^S_{st} - d_{st} - sh_{st} + g_{st} + \sum_{h \in \Omega_s^P} p_{h,t-\tau_h} + \sum_{h \in \Omega_s^U} (d_{h,t-\tau_h} \\ + sh_{h,t-\tau_h}) \qquad s = 1, \ldots, S \quad t = 1, \ldots, T \end{aligned} \tag{3.40}$$

$$\underline{v}_{pt} \leq v_{pt} \leq \overline{v}_{pt} \qquad p = 1, \ldots, P \quad t = 1, \ldots, T \tag{3.41}$$

$$\underline{v}^S_{st} \leq v^S_{st} \leq \overline{v}^S_{st} \qquad s = 1, \ldots, S \quad t = 1, \ldots, T \tag{3.42}$$

$$\underline{d}_{kt} \gamma_{kt} \leq d_{kt} \leq \overline{d}_{kt} \gamma_{kt} \qquad k = 1, \ldots, K \quad t = 1, \ldots, T \tag{3.43}$$

$$\underline{d}_{pt} \gamma_{pt} \leq d_{pt} \leq \overline{d}_{pt} \gamma_{pt} \qquad p = 1, \ldots, P \quad t = 1, \ldots, T \tag{3.44}$$

$$\underline{d}_{st} \gamma_{st} \leq d_{st} \leq \overline{d}_{st} \gamma_{st} \qquad s = 1, \ldots, S \quad t = 1, \ldots, T \tag{3.45}$$

$$0 \leq sh_{pt} \leq \overline{d}_{pt} \qquad p = 1, \ldots, P \quad t = 1, \ldots, T \tag{3.46}$$

$$0 \leq sh_{st} \leq \overline{d}_{st} \qquad s = 1, \ldots, S \quad t = 1, \ldots, T \tag{3.47}$$

$$0 \le sh_{kt} \le \overline{d}_{kt} \qquad k = 1, \dots, K \quad t = 1, \dots, T \tag{3.48}$$

$$\underline{p}_{pt}\nu_{pt} \le p_{pt} \le \overline{p}_{pt}\nu_{pt} \qquad p = 1, \dots, P \quad t = 1, \dots, T \tag{3.49}$$

$$v_{p0} = V_{p0} \qquad p = 1, \dots, P \tag{3.50}$$

$$v^S_{s0} = V^S_{s0} \qquad s = 1, \dots, S \tag{3.51}$$

$$\underline{v}_{pT} \le v_{p,T+1} \le \overline{v}_{pT} \qquad p = 1, \dots, P \tag{3.52}$$

$$\underline{v}^S_{sT} \le v^S_{s,T+1} \le \overline{v}^S_{sT} \qquad s = 1, \dots, S \tag{3.53}$$

Oliveira et al. refer in [Oliveira et al. 1990] that, for particular values of demand and reserve, generation and pumping can (theoretically) occur simultaneously. Therefore, equation (3.54) must be added to the formulation, to prevent simultaneous generation and pumping (see also [Chang et al. 2001]).

$$0 \le \gamma_{pt} + \nu_{pt} \le 1 \qquad p = 1, \dots, P \quad t = 1, \dots, T \tag{3.54}$$

### 3.2.5 Solution methodologies

Most of the methodologies that are currently used to solve the HTC problem are based on decomposition approaches that involve a hydro and a thermal subproblem, the overall problem solution being obtained by a proper coordination, within an iterative procedure, of the two subproblem solutions. A common approach is to commit the hydro units first and then, for the non-fulfilled demand (the total load demand minus the hydro production) the thermal subproblem is solved (Figure 3.9). The process is repeated until the thermal marginal costs or hydro generation converge.

As a result of this decomposition, the techniques referred in section 3.1.5 are also valid and useful for the resolution of the thermal subproblem: e.g. Lagrangian Relaxation [Ružić and Rajaković 1998; Gollmer et al. 1999; Ni et al. 1999], Augmented Lagrangian Relaxation [Al-Agtash and Su 1998], Genetic Algorithms [Chen and Chang 1996; Orero and Irving

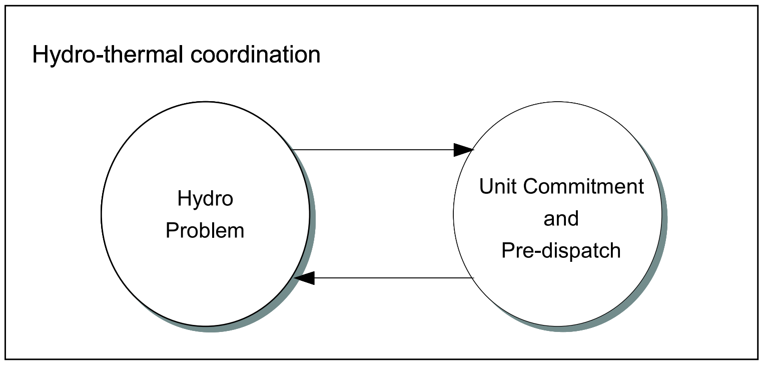


**Figure 3.9:** A decomposition approach for hydrothermal coordination

1998; Rudolf and Bayrleithner 1999], Tabu Search and Benders Decomposition [Bai and Shahidehpour 1996], Simulated Annealing [Wong and Wong 1994], MILP [Chang et al. 2001], Neural Networks [Nayak and Sharma 2000], Branch-and-Bound [Oliveira et al. 1993].

An approach based on LR is presented in [Ni et al. 1999]. The paper deals with hydrothermal power systems with cascade and head-dependent reservoirs. Due to head-dependent water-power conversion, the objective function is no longer stage additive with respect to water discharge. Therefore, it is no longer stage decomposable in the dual problem and additional difficulties arise. To overcome these difficulties, another set of multipliers is used to relax the constraints for minimum generation of individual hydro units. By doing so, a river catchment subproblem can be decomposed into two sets of problems: a continuous subproblem, to define the generation levels of all units, and several integer problems, one for each unit, to determine the hydro commitment states. The continuous problem is solved by a non-linear network flow algorithm and the integer subproblems are solved by DP.

Li et al. in [Li et al. 1998] combine LR with Sequential Unit Commitment (SUC) [Lee 1988] and Unit Decommitment [Tseng and Oren 1997]. In a first stage, LR is used to obtain a near optimal feasible solution. Then, SUC is used to obtain a feasible primal solution: given the Lagrange multipliers, associated to the dual solution, the SUC algorithm sequentially selects the most advantageous units to be committed, according to a unit average spinning cost index, for all periods of time where there is a deficit of spinning reserve. At the end, if possible, the Unit Decommitment method decommits overcommited units.

Fuentes and Quintana in [Fuentes and Quintana 2002] present an approach based on Semi-definite Programming (SDP). The technique can only be used for convex problems and, in such a case, it guarantees the convergence to an optimal solution in polynomial time, if Interior Point methods are used. To achieve the convexity shape, the integrality constraints of the problem are modelled as quadratic equations, whose roots are the desired integer values (for the HTC problem, the integrality constraints $u_{it} \in \{0, 1\}$ are replaced by $u_{it}^2 - u_{it} = 0$). The model is then transformed into a SDP, and solved with the SDP algorithm presented in the same paper. The solution obtained is mapped into a feasible solution, using a heuristic based on its closeness to an integer solution, to the time constraints and to demand and reserve constraints. It ensures that there is enough production or, in the case of a single committed plant, that its minimum power is less than the demand for the corresponding time period.

In [Li et al. 1997a], the thermal problem is solved through LR. The hydro system is divided into watersheds, that are further divided into reservoirs. Watersheds are optimised by Network Flow Programming and the resolution of the problem, at the reservoir level, is done through a priority list based DP. The process follows an iterative approach to update the Lagrange multipliers and to improve convergence. The integration of the hydro unit commitment with the existing hydrothermal optimisation package greatly improved the quality of its solution at the Pacific Gas and Electric Company (PG&E).

## 3.3 From regulated to deregulated markets

The power industry is presently undergoing a radical restructuring, involving a change from vertically integrated centralised management to a company level management, each company having to trade its energy by making bids within a Competitive Power Pool (CPP) or, alternatively, through contracts with other entities. Because of these changes, the concepts and principles of power markets are currently under extensive study – the new conjecture leads to the emergence of new optimisation requirements and several models have been built trying to capture the new reality of competitive markets. As the performance of each company may be highly affected by the decisions of other market participants, these models are inevitably associated to more or less elaborate strategic studies.

This section presents UCP models within deregulated markets and discusses the important role that the base UCP still plays within those environments. Although the topic "Markets of Energy" is too vast and out of the scope of this thesis, for the sake of completeness, we introduce here some concepts arising in this environment. The reader is addressed to [Sheblé 1999] for additional information and further references on the subject.

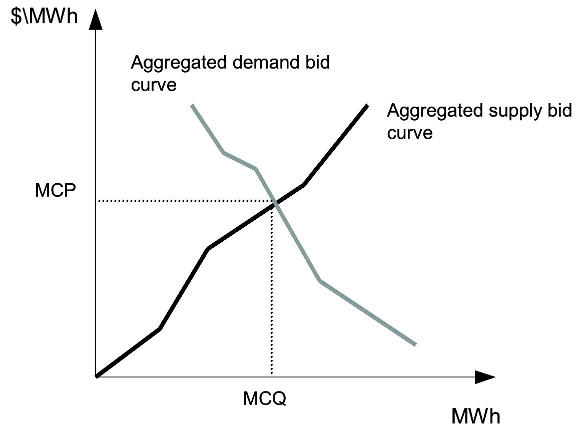


**Figure 3.10:** Market clearing price

### 3.3.1 Key concepts

In deregulated markets, energy may be traded in several ways, the two main choices being directly, through bilateral contracts between supplier and consumer, or within a power pool, where the market participants present their bids – suppliers indicate their supply capacities and bid price and consumers indicate their needs and, if allowed, the price that they are willing to pay. At the end, a third entity matches the two types of bids and, by doing so, settles the trading price of electricity – the *Market Clearing Price*. Market Clearing Prices (MCP) and quantities (MCQ) are the outcome of the bid acceptance process (Figure 3.10). Regardless of their asking prices, all selected bidders are paid the MCP in a way to force the bidders to price energy close to their marginal cost.

The trading process and rules, as well as the entities involved in the transaction, may vary resulting in different types of Markets of Energy, differentiated by the number and type of products traded, the bidding and scheduling process, and the market clearing and settlement rules. For simplification purposes, and without loss of generality, we will consider in the remaining of this text four entities: the generation companies (GENCO), the consumers, the Power Exchanger (PX) and an independent entity, the Independent System Operator (ISO).

The primary function of a PX is to provide a forum to match electric energy supply and demand in the energy markets. The roles and responsibilities of the ISO are yet not clear and are diverse in different markets [Sheblé 1999]. Even so, it may be viewed as an entity that coordinates the market players to provide a reliable power system operation. Several other entities may be involved in the trading process but their role is not essential for our purposes.

*The number and type of products traded*
In Markets of Energy one must provide costumers with other services, rather than the energy required, usually related to reliability and quality of service. Those services, *ancillary services* (or supportive services, according to [Sheblé 1999]) may be traded within the same or in different pools.

*The bidding process*
At the beginning, Markets of Energy only allowed supply-side bidding for energy and certain ancillary services, but later demand-side bidding was also allowed in some markets. The structure of the bids may vary in the number of cost components and in the technical parameters that are provided. When including several price components (e.g. start-up costs, minimum-load and energy bid prices), they are called *multi-part bids*. A bid with a single price component is a *single-part bid*. Still, in both cases, it may include several energy price segments depending on the amount of energy.

*The scheduling process*
When the market is based on single-part bids, a simple clearing process based on the intersection of supply and demand bid curves may be sufficient to determine the winning bids and production schedules for each hour. However, if it is based on multi-part bids, a unit commitment software may be needed.

*Market Clearing and Settlement Systems*
These are the procedures that determine the quantities to be produced and consumed, who pays, and who gets paid. When multiple markets are considered (e.g. energy, ancillary services and transmission products), two basic ways are provided to clear each market – sequential or simultaneous. In general, *sequential auction markets* clear each product separately in a sequence, even though several of the products may represent alternative uses of the same generator. In contrast, a *simultaneous auction* clears the relevant markets at the same time, minimising the joint bid cost of providing energy and ancillary services.

### 3.3.2 The role of the base UC model

Although market conditions are rapidly changing, the base UCP introduced in section 3.1 remains very important for those companies that own several generators. Before submitting a bid, for example, the company will have to decide on the amount of energy that it will be interested to bid and on the price at which it will sell that energy. This may be done by performing several UC optimisations, for different load curves (representing different supplying alternatives), and choosing the one that leads to better economical results.

From another perspective, to improve its competitiveness, a company may also try to simulate the behaviour of other bidders and anticipate their response (by predicting the schedule that supports their bids) within a specific context. This behaviour does naturally involve some risk because the information that one uses may not be sufficiently accurate. It may even happen that the other bidders act pretending that their generators have a set of characteristics that they do not have, to induce competitors to act in a specific way.

Finally, the base UCP may be solved by a GENCO after auctions close, when the company aggregates its power awards, considers the aggregate result as system demand, and performs a traditional UC or hydrothermal scheduling to meet its obligations at minimum cost over the bidding horizon.

The model may also be considered to conduct centralised power pool auctions. As referred in [Valenzuela and Mazumdar 2001] it still applies for the commitment decisions made by the ISO in several regions of the United States where deregulation is implemented. As they refer, the role of the ISO resembles very much the operation of a GENCO under regulation: a non-profit entity whose economic objective is to maximise social welfare, obtained by minimising the costs of reliably supplying load demand. So, if the ISO has all the system operational parameters of each generator, it sets the MCP by performing the UC for the whole power system in the market, based on the power-price bid curves received. This procedure was also used in the former England and Wales pool, lately substituted by NETA (New Electricity Trading Arrangements). Suppliers submitted multi-part bids, each bid consisting of a cost function and a set of parameters related to the generators, and the power pool solved the UCP in a centralised way, determining the system marginal price (the maximum average cost among the scheduled generators). Some entities do however claim that such a procedure is questionable as it may foment a non transparent behaviour of GENCO. In [Madrigal and Quintana 2001] it is referred that, for the England and Wales pool, participants were able of strategically choosing the parameters submitted in their cost functions, to drive up the system marginal price beyond competitive levels. This may also lead to an unequitable set of opportunities for each competitor, due to the sub-optimality of the solution. Therefore, conflicts of interest may arise if a particular supplier is favoured by the parameters setting, being dispatched at its maximum profit, while others are not. Still, this is the approach that leads to lower consumer payments which is, *per se*, a strong argument for using the model.

Finally, one should not forget that deregulation is not present in every country and that the base model remains valid for those markets with a vertical organisation.

### 3.3.3 New UC models

For many years, most electric power utilities have developed and used optimisation packages to solve the cost-based unit commitment problem, solely considering their specific park of generating units over a daily or weekly horizon. However, within the framework of some competitive electricity markets, commitment decisions are made by a schedule coordinator (the PX) based on the bids presented by the market participants (GENCO and consumers). Therefore, understanding how market participants bid into the electricity market is of fundamental importance for each player. Their expected profit depends upon the joint actions of the others and, in spite of that, an effective decision-making requires that each participant evaluates the effects not only of their own actions, but also of the actions undertaken by the others.

Several new UC models have been studied lately to respond to the new challenges that competitive markets are creating. These models are inherently connected to other strategic studies, related to the bidding process and, depending on the characteristics of the energy market, can vary in form, according to the type of auction mechanism that is implemented, the kind of energy transactions that are allowed, etc. Even so, they do in general share some common issues, namely: 1) the objective, that is no longer to minimise operating costs but rather to maximise the company's profit, and 2) the non-obligation of serving, as utilities may now choose to generate less than the total consumer's demand, if that decision is more profitable for them.

In this section we refer to some of the issues that have received particular attention when developping/adapting UC models to deregulated markets. They are: the concern on integrating bidding and scheduling, consumer payment minimisation, the inclusion of bilateral contracts and self-commitment.

#### Integrating bidding and scheduling

Although the price of electricity in regulated markets is pre-determined and fixed, when it comes to deregulated environments, it is no longer pre-determined and is set by the bids of each participant. As making a bid implies defining an operation schedule for generating units, integrated bidding and scheduling models have been developed lately, to provide the Decision Maker with a supporting tool that tries to anticipate MCP values. They may consider the influence of the bids of other GENCO on the company's performance, by simulating some of their possible behaviours or, alternatively, and supposing that one knows in advance the competitors' bids, simulate the results that one may obtain by making a specific bid.

Guan et al. in [Guan et al. 2001] present a model of integrated bidding and scheduling, within a perfect market (i.e. when the MCP is not affected by any single bid). The aim is to select bid curves for individual units, maximising the profit and reducing the risk. Due to minimum up and down time constraints, the bid curves (and, consequently, the MCP) are constrained.

In [Li et al. 1999] the bidding strategy aims at determining the optimal bid curve for the generation supplier, so that maximum return is reached satisfying some given goals for revenue adequacy. These constraints are necessary, since the maximum return criterion does not imply that revenue adequacy is guaranteed for every supplier. After some bid price scenarios are built, one has to determine the suppliers' bid curve, for each scenario, through a centralised UC, finding a "reliable" MCP. The bid curve that maximises the objective is the one selected.

Integrated bidding and scheduling is also studied by [Hao 2000], [Ni et al. 1999], [Borghetti et al. 2003] and [Baíllo et al. 2001].

Hao in [Hao 2000] studied the bidding strategies in a clearing price auction and drew the conclusion that, for this type of auctions, the market participants have incentives to mark up their bids above their production cost, the amount of mark up depending on the probability of winning the bid.

Ni et al. in [Ni et al. 1999] present a unified optimisation algorithm for the bidding strategy problem given a mix of hydro, thermal and pumped storage units. Their algorithm manages bidding risk and self-scheduling requirements.

Borghetti et al. in [Borghetti et al. 2003] investigates how traditional cost-based unit commitment tools, already available to generating companies, can be used in the competitive electricity market to assist bidding strategy decisions in a day-ahead electricity pool market.

In [Baíllo et al. 2001] the set of constraints representing the generation facilities in a traditional UC are replaced by a set of hourly constraints which define the firm's hourly revenue as a function of its energy output. In their example, the authors include a set of hourly minimum-market-share constraints to obtain a generation schedule similar to the traditional one. If no strategic constraints were used, the model would blindly follow all the short-term opportunities and the resulting operation would lead to an extremely inefficient dynamic performance of the generating units.

Other work on bidding and scheduling is that by [Richter Jr and Sheblé 1998] and [Richter Jr et al. 1999].

**Demand-Side Bidding (DSB)**

In some markets, the consumers do not directly influence the market price definition as they are not allowed to submit bids [Gross and Finlay 2000]. There are however other markets where, to play a proactive rule in the price determination process, consumers are allowed to submit bids not only for load requirements but also for load reduction in specific periods. Large industrial consumers, for example, may directly offer to the pool their ability to reduce load and receive a payment for that reduction.

As a natural evolution of the work presented in [Gross and Finlay 2000], a model that considers explicit DSB with load reduction is presented in [Borghetti et al. 2001b]. The supply-side must still specify the technical characteristics of each bidding unit (e.g. minimum and maximum output, minimum up and down time, etc.), the bid price and offered capacity, while the demand-side must specify the subset of the load reduction periods settled by the operator (the subset of the periods in which a bidder may undertake load reduction); the minimum and maximum demand that can be reduced by a bid, the subset of the load recovery periods settled by the operator (the subset in which a bidder may undertake a load recovery), and load recovery within a period, that is related to all the load reductions in each reduction period.

**Consumer payment minimisation**

The solution that minimises a generator's cost (the sum of the products of the hourly load demand and the hourly MCP, over the entire scheduling period) may not result in minimum cost to consumers. This is discussed in [Jacobs 1997] that shows that, under uniform pricing rules, minimising generation costs and consumer payments is different.

Hao et al. in [Hao et al. 1998] present an approach for calculating optimal generation schedules that minimise energy payments by power pool consumers, within the framework of centralised optimisation. The model assumes the use of a uniform pricing rule, i.e. all participants are paid the same, and considers an additional constraint, here referred to as *payment adequacy constraint*, to ensure that all units winning in the auction will recover their start-up, no load and energy production costs, as implied by their bids.

Ren and Galiana in [Ren and Galiana 2002] discuss pricing methods with the objective of scheduling generators in such a way that the costumers obtain the lowest price for electricity, while the GENCO do at least cover their offered costs.

### Self-commitment

In some energy markets, instead of (or in addition to) a centralised UC, utilities will have to make independent UC decisions, having to submit independent hourly bids for each generating unit and, define their bidding price. This leads to the need of developing methodologies to define adequate bidding prices and quantities for each separate unit, i.e. for self-committing the units.

Considerable work in this direction has been presented lately, e.g. [García et al. 1999], [Galiana et al. 2001], [Rajaraman et al. 2001].

Rajaraman et al. in [Rajaraman et al. 2001] focus on finding the best self-commitment policy, in presence of exogenous energy and reserve price uncertainty and market multiplicity (day and hour-ahead markets).

Galiana et al. in [Galiana et al. 2001] present an alternative to the centralised power pool – an auction mechanism that allows each independent participant to self-commit and dispatch based on its own profit evaluation. This alternative is based on the assumption of *profit optimality*, i.e. all competing participants are free to maximise profit, subject only to market prices. Under these conditions, the UCP primal and dual forms are equivalent and lead to identical solutions.

Li et al. in [Li et al. 1999] introduce a market model that uses self-commitment to determine generator bids over a fixed time period. Then the usual market resolution process resolves the bids to determine market prices and generator bids are recalculated for the same time period, using the new prices.

In [Xu and Christie 2001], self-commitment price-taking is performed. The authors try to maximise the future profit of one unit, based on the predicted future price of energy.

Galiana et al. in [Galiana et al. 2001] present a self-commitment UC model, based on nodal prices. Besides the standard technical constraints of each unit, power balance and power flow constraints are also included in the model.

Finally, the work presented in [García et al. 1999] describes the self-commitment problem in the Spanish market.

### Bilateral contracts

According to some authors, the pool-bilateral model followed by some markets of energy is advantageous because it introduces more opportunities for competition. In such models, two types of power generation and demand are considered: one related to bilateral transactions and another for trading in the pool.

Valenzuela and Mazumdar in [Valenzuela and Mazumdar 2001] consider bilateral contracts and transactions in the power pool and assume that the GENCO is a price-taker (i.e. if at a particular hour the power supplier decides to switch on one unit, it will be willing to take the price due at that hour).

## 3.4 Commercial software packages

A correct usage of appropriate optimisation tools to support Unit Commitment/HTC decisions may, among other benefits, drastically reduce production costs. As in practice these costs may be extremely high [Ikura et al. 1986; Erwin et al. 1991; Johnson et al. 1998], it is therefore natural that GENCO continuously invest in more sophisticated and complete Energy Management Systems (EMS).

The purpose of this section is to describe some commercially available UCP/HTC modules, embedded in EMS. It mainly focus on the model characteristics and optimisation methodologies that are used. Software applications that do not properly tackle the thermal UCP (e.g. Powel Solution) are excluded from this survey. We should also add that the information reported here is solely supported by public documents available in the Internet.

**Plexos**

[http://www.plexos.info]

Plexos is a simulation tool for electricity power markets, developed by Drayton Analytics and by Elan Energy Consulting. Its UC module allows self-commitment and centralised UC. The properties that are available to define thermal generators' constraints are minimum and maximum production levels, ramp rates, minimum up and down times, start-cost and fuel-at-start, i.e. the quantity of a given fuel that is consumed each time a generator starts operating.

The hydro module allows the definition of cascade systems and the inclusion of all types of units (run-of-the-river, pump-storage and storage). Water travelling times are also specified.

The centralised UCP is solved through a Mixed Integer Linear Programming (MILP) model but, concerning the coordination between the two systems, no information has been obtained. The software does also include specific modules for market environments, namely Dynamic Bidding and Gaming, and Demand Side Participation.

### EnWorkz Commit

[http://enworkz.com/commit.htm]

EnWorkz Commit, developed by EnWorkz Inc, was designed for being used both in regulated and deregulated markets, and for merchant generation as well as industrial co-generation. It can schedule generation assets to maximise profits from the sale of electricity, or to minimise the cost of satisfying load demand, and can also perform a hybrid optimisation, where profits from spot market transactions are maximised subject to the load constraints.

The core of the system is a proprietary technology for stochastic Dynamic Programming, which is used to calculate a profit-maximising generator schedule, given an uncertain forecast of energy and ancillary services prices. This technology is combined with Lagrangian Relaxation to optimise multiple-generator schedules, based on minimum cost.

### PowrSym3

[http://www.osa.comax.com]

PowrSym3 is a multi-area, Monte Carlo production cost simulation model, developed by Operation Simulation Associates. It considers emission costs, start-up costs and minimum up and down times. It can also solve multi-fuel, multi-station contracts. Each fuel may have hourly, daily, and weekly minimum and maximum requirements, all of which may vary with time. This solution is achieved with a network model directly integrated into the unit commitment and dispatch algorithm. Issues concerning multi-area unit commitment are also tackled by the software. The optimisation process can be done through a typically rule based commitment logic, or a “windowed” DP.

PowrSym3 and earlier versions of Powrsym are in use by several utilities in the USA, Europe, and Australia.

### Power Optimisation

[http://www.powerop.co.uk]

Power Optimisation is a UC software based on a proprietary multi-stage version of MILP. For its UC module, the objective is to minimise the total fuel costs over the study period, satisfying the appropriate constraints. Alternatively, the objective can be to maximise profit, defined as the total revenue from electricity sales minus fuel costs.

The software can model the operation of both thermal and hydro generating units. It can also model and optimise the use of different fuels and/or gas contracts in the same generating unit or group of units. Thermal units can include dual-fired generating units, which can use one of two alternative fuel types. Start-up costs are temperature-dependent, determined by the time that the unit has been off. There can be upper limits on the numbers of starts of each generating unit over 24 hours and each generating unit can also have minimum on and off times.

The hydro units can include conventional hydro and pumped storage units that are subject to energy limits, in the form of minimum and maximum values for the total energy generated over the study period. As an alternative to specifying energy limits, the user can specify non-zero values for the incremental costs of the hydro plant, to represent the long-term value of the water in the reservoirs. The software can also optimise the use of pumped storage units, taking into account the energy lost in the pumping cycle. It optimises the times and amounts to pump and/or generate from the pumped storage units, and when the pumped storage units should be off. The water stored in the upper reservoir of the pumped storage units can be kept between specified upper and lower limits, to prevent spillage or drainage of the reservoir, allowing for any inflow of water into the reservoir. Additionally, the user can specify minimum, maximum and target values for the reservoir level at the end of the study period.

An interesting feature of this software is an option to minimise changes from a previously calculated schedule, while taking account of any changes in demand and plant availability that have occurred since the previous run of the software. This is naturally useful if the input data changes suddenly and the system operator does not wish to make significant changes to the instructions previously issued to the power stations.

One version of this unit commitment software, developed for Northern Ireland Electricity (NIE), has been used there since December 1996 to schedule the generating units in the Northern Ireland power system. Another version is being used for self-scheduling by generating companies, under NETA, in the electricity market of England and Wales.

**GenTrader**

[http://www.powercosts.com]

The GenTrader software, developed by Power Costs Inc, uses the *Sequential Bidding Commitment Algorithm* proposed by Lee in [Lee 1988], to solve the UCP, considering unit must-run and must-off constraints, minimum up and down times, initial operating status, ramp-rates and crew constraints.

According to the company, there is no other technology currently available in the market that can deal with multiple competing fuel constraints, and achieve convergence in the presence of extensive transaction and unit commitment constraints. The fuel constrained algorithm uses a proprietary non-linear optimisation algorithm, rather than the conventional network flow algorithm usually applied to fuel constraint problems. This overcomes the shortcomings associated with network flow fuel coordination algorithms, such as non-smooth (and impractical) solutions and computational instability.

A hydro module allows the user to coordinate hydro and thermal resource operation, to minimise cost or maximise profit. The modeling components of pumped storage are, for each reservoir, initial, maximum and final reservoir volumes, and hourly inflow. For each unit one may specify reservoir assignment pumping high and low limits, generating high and low limits, must-on/off periods, fixed and variable operation and maintenance, pumping efficiency, and forced outage rate.

GenTrader is used by some energy companies participating in the deregulated Texan electricity market, e.g. Reliant Energy, TXU and Calpine Corp.

### UPLAN-MAM

[http://www.energyonline.com]

UPLAN Multi-Area Model was developed by LCG Consulting to simulate the electricity markets and the physical dispatch of energy, both for competitive and regulated environments.

The Unit Commitment module aims at minimising total capacity costs taking into account start-up costs, ramp rates and spinning and non-spinning requirements. Unit commitment and dispatch algorithms use a large-scale linear program with mixed integer capabilities, to maximise producers' and consumers' surpluses. The combined Unit Commitment and the multi-area production cost programs generate a production schedule for each generator for each hour, sufficient to meet the demand bids, clear the market and minimise the sum of the start-up, no-load and the incremental energy bids.

A hydro scheduling model is used for managing reservoirs and scheduling of daily and hourly dispatch, to maximise market profits. The optimisation maximises the overall generator profits from serving the demand with the available hydro and thermal generating resources.

**Spectrum PowerCC Generation Management**

[http://www.siemens.com]

Spectrum PowerCC GM is a product developed by SIEMENS AG. In this software both the thermal Unit Commitment and the HTC can be performed in two different ways: 1) to find the optimal commitment of generators to meet the load, based on forecasted load and planned transactions; or 2) to achieve maximum profit maximisation. HTC is performed through a decomposition approach, as explained in section 3.2.5 and the supporting optimisation technique is Lagrangian Relaxation.

There is also a "Trade Optimising Scheduler" that defines the bids of energy and reserve, given a forecast market clearing price for each hour of the planning period, aiming at achieving maximum profit for a generation company.

Generation companies currently using Spectrum PowerCC GM are EW Obwalden, in Switzerland; EDP, in Portugal; SaarEnergie, in Germany; Power IT, in Finland; Fuji Matsumoto Factory, in Japan; and Alliance RTO (ASP), in USA.

**e-terracommit**

[http://www.esca.com]

e-terracommit was developed by ALSTOM Energy Management & Markets. It extends the classical UCP, to analyse transaction and generation resources concurrently, and to take into account the impact of transmission routing and wheeling costs. The software models both thermal and hydro units and, according to the documentation, hydrothermal coordination is performed through a full optimisation-based dispatch, without problem decomposition. It is also capable of studying multi-area UC with energy and security constraints, fuel contracts and scheduling emissions constraints.

## 3.5 Requirements and research trends

### 3.5.1 Research vs Practice: is there a gap?

In this section we give a brief comparative overview on the relative developments of research and commercially available solutions for the UC/HTC problems.

From the point of view of problem modelling, industrial software is in general far more complete than research applications. The former tends to consider broad models, that tackle simultaneously several types of constraints related to, e.g. fuel management, environment concerns and several market issues. The latter tends to focus on niches of the problem and study, for that specific niche, new methods of resolution far more elaborated than those found in commercial software. However, due to the limitations of the techniques used by commercially available software, rough simplifications are often assumed. Non-linear functions (e.g. Input/Output curves) are frequently represented by points saved in look-up tables, linearised or, following a more elaborate approach, linearised around the solution of the last iteration. To reduce problem complexity it is also a common procedure to relax some constraints that are difficult to handle, e.g. up/down ramp rates, by introducing slack variables that are affected by a penalty value in the objective function.

In terms of optimisation techniques, these software applications present a wide variety of offers, the more traditional ones prevailing: Dynamic Programming, Lagrangian Relaxation, heuristics based on lists of priorities and Mixed Integer Programming. More "advanced" techniques, such as metaheuristics, fuzzy logic or Constraint Logic Programming based approaches, are not available in the market.

As a conclusion, we would say that, as in most other areas, a gap between research and practice does naturally exist in this area of study. Even so, it is our opinion that there is also a bridge linking the two sides, with research efforts trying to smooth the weaknesses of commercial approaches, by studying the influence of arising issues in problem modelling, and by developing effective, efficient and reliable methodologies that allow an improvement in modelling accuracy. In what concerns this last issue, metaheuristic approaches as those proposed in Chapters 4 to 6 present a high potential of application within industrial environments, due to their ability of dealing with nonlinearities and of easily incorporating new problem specifications in the optimisation process.

### 3.5.2 Trend lines for further research

Research work on the UCP is clearly alive and there is still a long way to go, as several improvements may be done in optimisation techniques and problem modelling to reach more effective results in the everyday planning of energy production.

Problem modelling is a central point of current research, with the new paradigms and challenges introduced by the market restructuring process.

In what concerns optimisation techniques, an area for further research is on developing techniques that are capable of correctly tackling all the non-linearities that this problem presents and also on developing techniques capable of correctly handling more than one objective, for the same problem. We will highlight here three topics that, in our opinion, are extremely important and deserve further attention and research: strategy and multiobjective modelling, and optimisation tools.

**Strategy and multiobjective modelling**

The power industry restructuring is an ever evolving process, far too complex and involving many strategic agents. As the behaviour and decisions of each of these market participants are uncontrollable and may strongly affect the performance of any single agent, such an agent may therefore need to develop its own strategies to reduce the impact of the other market participants on its own performance. Some of the models described in section 3.3.3 do already reflect concerns on studying alternatives, given the predicted behaviour of competitors. Even so, they should evolve to a higher stage, where both tactical and strategic decisions are also made and should probably be interconnected with elaborated strategy methodologies (e.g. Game Theory).

Concerning the number of objectives, except for one or two situations, the UCP has until now been modelled as a single objective problem. If that might be acceptable in the past, and represented the real problem in a satisfactory way, nowadays several other objectives are also important for a GENCO and should, therefore, be considered when making the operational production decisions. With the imposition of emission allowances, for example, a GENCO does not only want to minimise operating costs (or maximise revenue) but also to minimise emissions, as that may be economically interesting if a company is allowed to trade allowances in the market. However, as the cheaper generating units are typically more pollutant than the more expensive ones, the two objectives are conflicting, this complexity requiring a real multiobjective model.

Several other examples of multiobjective problems may be given. Within energy markets, for example, sellers and buyers are competitive participants with contradictory objectives, sellers seeking for the maximisation of their profit, and buyers trying to buy electricity at the lowest possible cost. Another example are those markets where different commodities are traded in different auctions, each auction having its own objectives and operation rules, that may interact with those of other auctions. The same reasoning applies if the influence of intra-day markets is considered in the evaluation of the overall performance of a company. One may find it strategically advisable to reduce the offers (and consequently the income) in the dairy market, to be able to bid in intra-day markets at a more advantageous price. The applications are therefore vast and the subject deserves further attention.

**Optimisation tools**

A major problem of the conventional optimisation tools provided by commercial software packages is that they require that the highly non-linear and non-convex information related to the hydro and thermal plants is represented by piecewise linear or polynomial approximations of monotonically increasing nature. However, such an approximation may lead to a suboptimal solution, resulting in a considerable loss of revenue over time. Thus the solution for such a problem demands more robust and versatile techniques.

Metaheuristics may prove to be an interesting choice as they do not place any restrictions on the shape of the cost curves and on other problem nonlinearities. Still, a major effort has to be done to make them more user friendly and build the confidence of Decision Makers (DM).

Furthermore, some attention should also be given to the development of techniques capable of "optimising" more then one objective simultaneously. Given the potential that multiobjective metaheuristics have already proven to have at solving some combinatorial optimisation problems, they should also be considered as a topic for further research.

## 3.6 Concluding remarks

This chapter reviewed activity on modelling the UC/HTC problems and on the optimisation techniques that have been designed to tackle them. Concerning problem modelling, the recent restructuring of the power industry led to the development of new models, that try to better capture the new reality introduced by Markets of Energy. Even so, the base problem is still of much relevance in practice, and it is therefore pertinent to concentrate efforts on developping/improving resolution techniques that are capable of cooping with the strong non-linearities and non-convexities that the problem presents.

We suggest here three areas for further research: modelling strategic behaviours, modelling conflicting objectives, and improving the quality of those techniques that are capable of dealing with the non-linearities of the problem. The last two topics will be further developed in this thesis.

Chapter 4

# Applying GRASP to the UCP

Several alternative tools have already been proposed to solve the UCP, Lagrangian Relaxation and Priority-List based methods being among the most used ones. More recently, metaheuristics have also proven to be effective techniques to handle this problem and one can now find in the literature several approaches based on SA, TS or GA to tackle it.

Following this line of reasoning, this chapter proposes an innovative approach based on GRASP, a metaheuristic that has been used with success in several areas (e.g. Routing [Argüello et al. 1997], Manufacturing [Ríos-Mercado and Bard 1998], Telecommunications [Piñana et al. 2004], Electrical Power Systems [Bahiense et al. 2001], etc) but that, as far as the author knows, had never been applied to the UCP.

The chapter is structured as follows. First, a general description on the solution coding and neighbourhood structures that have been considered in Local Search based metaheuristics for solving the UCP is presented. The chapter proceeds with a survey on the metaheuristics already used to solve the problem. Although being out of the scope of this thesis, GA and other Evolutionary Algorithms approaches are also briefly referred. Finally, a new solution coding is suggested, the GRASP approach proposed is described and the reasoning behind the decisions made when designing the algorithm are explained. The chapter concludes with a discussion on the computational results obtained by applying the new algorithm to a set of instances from the literature.

## 4.1 Metaheuristics for the UCP

Following the same reasoning of other approaches, metaheuristics solve the UCP through a two-stage process (see Figure 4.1). In the upper-stage the metaheuristics' operators generate a new solution for the combinatorial problem, by defining the state of each unit in each period (binary decisions). However, for evaluation purposes, the production levels for the new solution must also be computed. This is done in a "lower level", by solving the Pre-Dispatch problem through an appropriate methodology, e.g. the $\lambda$-iteration method based on the Kuhn-Tucker conditions [Wood and Wollenberg 1996]. After being evaluated, the new solution is accepted (or not) according to the acceptance rules of the metaheuristic considered in each specific implementation. The iterative process is repeated until a stopping criterion is reached.

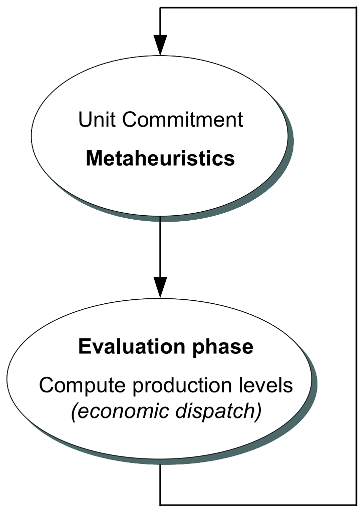


**Figure 4.1:** Metaheuristics: a two-level approach

As metaheuristics are involved, the implementation of this approach requires the definition of how to represent a solution and how to generate a new solution from the current one (i.e. a neighbourhood structure). These specifications may vary and one can find some alternatives in the literature. A summary on the traditional specifications is supplied below.

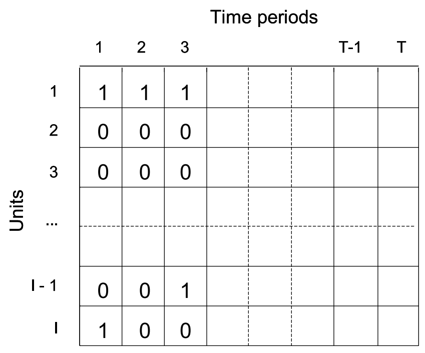


**Figure 4.2:** Typical solution representation

### 4.1.1 Solution representations

Typically a Unit Commitment solution is represented by a binary matrix, where each row represents the operation schedule of one unit over the planning horizon (Figure 4.2). This representation is particularly convenient as it directly reflects the state of each unit in each period of time. However, it does not allow an easy handling of some of the problem constraints.

Other schemes that try to capture specific aspects of the problem are proposed in the literature. By representing these aspects in the solution coding, it becomes easier to tackle some of the problem constraints. The solution representation used by [Rudolf and Bayrleithner 1999], [Cheng et al. 2002] and [Yang et al. 1996], for example, integrates minimum up and down constraints in the solution coding. Each row of the matrix that represents the solution is divided into several substrings ($S_{ij}$), each of them having a leading bit that specifies the state of that unit (Figure 4.3). The remaining bits indicate the number of periods for which the unit stands in that state, minus the corresponding minimum up (or down) time. The length of each substring, $n_i$, is given by expression (4.1), where $n_{imax}$ = max(minimum up time$_i$, minimum down time$_i$).

$$n_i = \begin{cases} \lfloor \log_2(n_{imax)} \rfloor & \text{if } n_{imax} > 1 \\ 1 \quad \text{otherwise} \end{cases} \tag{4.1}$$

Supposing that for a given unit the minimum up and down times are 2 and 4 periods, respectively, the substring length of that unit is 3 bits. Supposing also that its current schedule is the one depicted in Figure 4.3, substring $S_{11}$ indicates that the unit is on in the first time interval (the leading bit is 1) and stays in that state for 3 time periods, i.e. 1

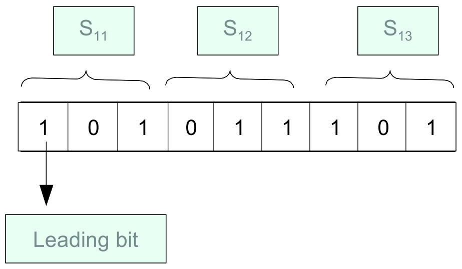


**Figure 4.3:** Solution representation proposed in [Rudolf and Bayrleithner 1999]

time period indicated by the two "non-leading" bits in $S_{11}$, plus the minimum up time. It then changes its state to 0 ($S_{12}$) and remains off for 7 time periods. Finally, the unit is switched on and remains on for 3 time periods. Thus, the actual unit schedule (in the basic representation) is 1110000000111.

### 4.1.2 Neighbourhood structures

A neighbourhood structure that is often used when solving the UCP can be described as follows: 1) randomly choose a unit $i$, a period $t$ and a direction $dir$ (indicating whether changes will be done before or after $t$) and 2) turn on (or off) $i$, for a time interval containing $t$. This movement alone does frequently lead to infeasible solutions. So, when only feasible solutions are accepted during the search process, a feasibility recovering procedure must also be designed. This general neighbourhood structure is used by [Mantawy et al. 1998a,b], [Zhuang and Galiana 1990] and [Purushothama and Jenkins 2003], with various recovering feasibility rules.

Different neighbourhood movements are proposed by [Bai and Shahidehpour 1996] and by [Yin Wa Wong 1998].

Bai and Shahidehpour, in [Bai and Shahidehpour 1996], search for three specific unit states and apply the neighbourhood movements accordingly. The unit states are associated to: 1) peak time periods, $N_1(s)$ (e.g. s = $00000\underline{111}00000$), 2) off-peak time periods, $N_2(s)$ (e.g. s = $11111\underline{000}11111$) and 3) time periods adjacent to state transition periods, $N_3(s)$ (e.g. s = $1111\underline{10}00000\underline{01}111$). In each case, the underlined states are selected as candidates for change. A higher priority of selection is given to units with lower minimum up and down times and to units whose state changes more frequently.

Yin Wa Wong, in [Yin Wa Wong 1998], does also consider alternative movements that are randomly selected and applied to each unit in the system. The alternatives are: no changes, delay unit on-line time by one period, advance unit on-line time by one period, delay unit off-line time by one period and advance unit off-line time by one period. Depending on the neighbourhood that is selected the (possible) infeasibility of the resulting solution will be due to different constraints. Accordingly, different recovering mechanism will be used.

The following sections survey the metaheuristic approaches proposed in the literature and present the distinguishing aspects of each specific approach. Although Evolutionary Algorithms are out of the scope of this thesis, they are also included in this survey, due to the considerable attention that they have recently received.

### 4.1.3 Tabu Search

Tabu Search (TS) applications to the UCP are mostly found within hybrid environments. Even so, some "pure" TS approaches can be found in the literature (e.g. [Mantawy et al. 1998b] and [Borghetti et al. 2001a]).

As soon as an initial (feasible) solution is obtained, an iterative procedure composed of three main steps, and following the general metaheuristics' framework, is considered: 1) generate random feasible solutions through a general neighbourhood movement as that described in section 4.1.2, 2) evaluate each solution by performing the pre-dispatch and, 3) select the new solution, according to the Tabu Search rules.

Four types of Tabu List are studied in [Mantawy et al. 1998b]. The first list records pairs *(unit, time period)*, whenever a unit and a specific period of time are selected as a departure point to generate a new solution. The second list is similar but, in addition, records the unit state at that time period. Each entry of the third type of Tabu List records the number of periods that the unit remained on and does not allow changes that lead to a solution with an equal number of on periods. The fourth list stores the points of state transition of each unit (i.e. it indirectly stores the schedule of each unit) and movements that lead to the previous schedule are forbidden.

In [Borghetti et al. 2001a] a unique Tabu List is maintained. It stores the unit $i$, and time period indices $t$ and $k$, corresponding to the start and end of the considered sequence of 0's or 1's for that unit. A move is forbidden if it involves a vector $i$, $j$, $k$ that is currently in the Tabu List.

Hybrid implementations where Tabu Search is used are those by [Bai and Shahidehpour 1996] and [Mantawy et al. 1999]. In [Bai and Shahidehpour 1996] TS is used with the Benders Decomposition method [Gofferion 1972], that excludes sub-optimal solutions from the search process and consequently improves the search speed. In [Mantawy et al. 1999] the concepts related to Tabu Search are incorporated in the reproduction phase of a Genetic Algorithm. A Tabu List is associated to each unit, each entry recording the equivalent decimal number of the binary representation of a specific unit's schedule. Genetic operations that lead to schedules that are in the Tabu List are forbidden.

### 4.1.4 Simulated Annealing

The first attempt to solve the UCP with Simulated Annealing (SA) is that described in [Zhuang and Galiana 1990]. In that approach, constraints are divided into "easy" and "difficult" ones. Schedules satisfying the "easy" subset of constraints are regarded as feasible, but any violation of the "difficult" constraints is penalised in the objective function.

Later, inspired in this work, [Mantawy et al. 1998a] proposed another approach that, according to the authors, differs from the previous one in several points, namely: it starts with a random solution, instead of a deterministic one obtained through a priority list based method; and only feasible trial solutions are accepted for further exploration. By doing so they obtain better results and a reduction in CPU time. A similar approach is followed by [Yin Wa Wong 1998].

Hybrid applications of SA with other techniques can be found in, e.g. [Purushothama and Jenkins 2003], [Mantawy et al. 1999] and [Cheng et al. 2002]. Trying to improve convergence and solution robustness, [Purushothama and Jenkins 2003] describe a hybrid algorithm where SA is combined with Local Search. Local Search is carried out in the neighbourhood of the best solution found after a number of iterations at the same temperature level, and is based on a priority-based decommitment procedure. In the decommitment process, all units are considered exactly once for decommitment in each hour, in descending order of operating cost. When the Local Search is concluded, the temperature is lowered and the procedure returns to the basic SA algorithm.

Wong and Wong in [Wong and Wong 1995] developed GAA2, a hybrid GA/SA approach, where the generation of new solutions is based on a Genetic Algorithm with a population size of 2, to minimise memory requirements. To prevent premature convergence and the adverse effect of mutation, the SA probability of acceptance is taken as the probability of replacing a chromosome by a weaker one, in the crossover and mutation operations. Besides, to avoid a quick convergence to a local optimum, if one of the chromosomes is fitter than the other, the fittest chromosome generated so far is still considered for replacement, allowing that more diversity is maintained, even for a population with a small size. However, the fittest chromosome is stored and may be reintroduced into the population at the end of each generation in a probabilistic manner. When compared to other approaches, GAA2 showed to have a good performance in terms of solution quality, computational speed and memory requirements. The problem of premature convergence present in basic forms of GA is also overcome.

In the same line of reasoning [Mantawy et al. 1999] incorporates SA concepts into a GA. SA is explored to improve the convergence of the GA by testing the population members of the GA after each generation, allowing the acceptance of almost any solution at the beginning of the search process, but only of good solutions as the number of generations increases.

Finally, the work by [Cheng et al. 2002] is somehow complementary to the two previous ones. It incorporates GA concepts into SA, and creates an Annealing-Genetic algorithm (AG), a two-stage cycle where in the first stage the search is processed by a SA and in the second stage evolves through genetic operators.

### 4.1.5 Evolutionary Algorithms

Evolutionary Algorithms (EA) have also been considered for solving the UCP, several interesting operators having been proposed.

Sheblé and Fahd in [Sheblé and Fahd 1994], and Maifeld and Sheblé in [Maifeld and Sheblé 1996] designed crossover operators in such a way that their use does not imply the recalculation of the production levels of units. As a result, the pre-dispatch algorithm is only used with the initialisation and mutation routines. Figure 4.4 illustrates how this is accomplished, for a 3–unit, 4–hour horizon ($C_{lt}$ represents the cost of producing the desired output in period $t$, when a specific set of units is on, in solution $l$). Applying conventional crossover (with t = 2) to the parents at the top of the figure, leads to the offsprings at the bottom. As the load remains stable from the parent solutions to their offsprings, the production level of each unit does also remain the same. However, transition costs may have to be recalculated.

Maifeld and Sheblé in [Maifeld and Sheblé 1996] extend the concepts developed in [Sheblé and Fahd 1994] and define three types of mutation operators: the *turn-off generator mutation* that randomly chooses a unit to be switched off in a specific scheduling period; the *intelligent mutation I* that starts by looking for 01 and 10 combinations and randomly changes the combination to 00 or 11 (this operator is applied to half of the newly created population and the unit where the operator is applied to is chosen randomly); and the *intelligent mutation II* that does also search for 01 and 10 combinations and checks which state would be cheaper (00, 11 or remain the same). Again, the operator is applied to half of the newly created

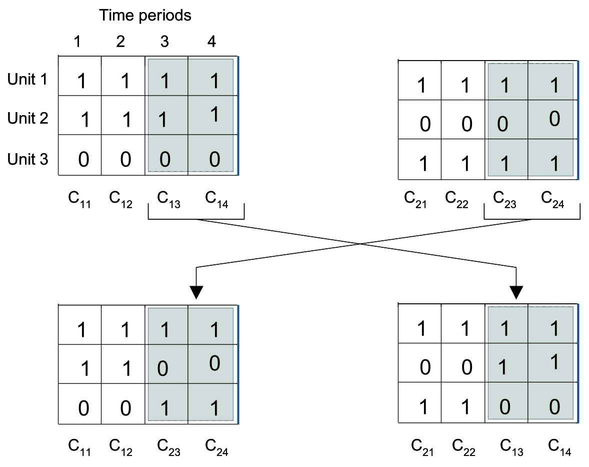


**Figure 4.4:** Crossover operator in [Sheblé and Fahd 1994]

population, and the unit the operator is applied to is chosen randomly.

Taking into account problem-specific knowledge, and having found that most offspring solutions did not satisfy reserve constraints or led to overloaded results, [Ma et al. 1995] proposed a knowledge augmented mutation-like operator – *forced mutation* – to recover solution feasibility when those constraints are not satisfied. It works as follows: if the committed units do not satisfy reserve constraints, a unit randomly chosen from the remaining down units, is forced to be on, and if the committed units represent a system overload, a unit randomly chosen from the on units is forced to be off.

Trying to improve GA convergence [Kazarlis et al. 1996] proposed several additional operators that act on blocks rather than on bits. The *Swap-window operator* randomly selects two units, a time window of width $w$ and a random window position. The bits of the two units included in the window are then exchanged. The *Window-mutation operator* randomly selects an unit and a time window of width $w$. Then all the bits in the time window are mutated. An additional set of operators was also implemented to apply hill-climbing to the best solution obtained in each generation. The *Swap-mutation operator* performs, for every hour of the scheduling horizon, one of the two following operations: 1) it selects two arbitrary units and exchanges their bits for that specific hour or, 2) it selects a single unit and flips its corresponding bit for that hour. The *Swap-window hill-climbing operator* selects two arbitrary units and a time window of width $w$. The time window starts at the first hour of the planning horizon and the bits of the two units are exchanged. The new solution is evaluated and, if better, is kept. Otherwise it is restored to its initial sate. Then the window is shifted one hour and the procedure is repeated until the window reaches the last hour of the scheduling horizon.

Swarup and Yamashiro in [Swarup and Yamashiro 2002] give a special attention to the development of problem specific operators that handle more appropriately time dependent constraints, namely minimum up and down time constraints. Two types of problem specific operators are defined: the *bit change operator* modifies the bit positions to a more similar pattern (e.g. change sequence 010 to 000 or 101 to 111) with a certain probability, and the *minimum up/down operator* rearranges the bits to satisfy the minimum up and down time constraints.

Hybrid methods that join EA with other techniques are also popular. They aim at avoiding premature convergence and at improving the convergence speed, and are claimed to accommodate more complicated constraints and to reach better quality solutions.

Huang and Huang in [Huang and Huang 1997] present a GA based Neural Network, post-processed by Dynamic Programming. GA are used to initialise the Neural Network and DP processes states that remain uncertain after neural computations.

Orero and Irving in [Orero and Irving 1997b] present a GA with a priority list based technique. The solutions produced by the priority list based heuristic are used to obtain an initial population. If elitism is applied, the hybrid algorithm is guaranteed to do no worse than the priority list method. The same authors present in [Orero and Irving 1997a] a combined LR-GA approach that can be implemented in alternative ways: either the GA acts as the main solution method and incorporates LR (the LR unit commitment results providing part of the initial GA population), or LR uses the GA solution to estimate the initial values of the LR multipliers or to update those multipliers during the search process. The latter is also used in [Cheng et al. 2000], where GA are used to update the multipliers of LR.

Mashhadi et al. in [Mashhadi et al. 2003] attempts to improve convergence speed, lead to an approach where local and global optimisation are considered simultaneously. This is achieved by implementing a new genetic operator that may be applied to a unit, with a given priority. Convergence speed may also be improved through parallel implementations as that proposed by [Yang et al. 1997].

Valenzuela and Smith in [Valenzuela and Smith 2002] propose a seeded Memetic Algorithm (a GA combined with Local Search), Local Search being applied only if the solution is better than the best solution found so far. The initial population is seeded with solutions obtained with LR.

Chen and Wang in [Chen and Wang 2002] propose a Cooperative Coevolutionary Algorithm (CCA). A CCA consists on a collection of independent sub-populations, each attempting to evolve sub-components (species) that are useful as modules for achieving more complex structures. For the UCP, the system constraints of the primal problem are relaxed by a Lagrangian function. Then, a two-level optimisation algorithm is formed. The lower-level problems solve the optimal commitment of each individual unit with GA. The high-level problem optimises the Lagrangian multipliers through a sub-gradient based stochastic optimisation method.

Additional references to EA approaches are [Yang et al. 1996], [Sheblé et al. 1996], [Rudolf and Bayrleithner 1999], [Juste et al. 1999], [Wu et al. 2000], [Padhy 2000], [Senjyu et al. 2002] and [Swarup and Yamashiro 2003].

## 4.2 A GRASP approach for the UCP

As referred in Chapter 2, GRASP is a multi-start metaheuristic composed of two main phases: a Construction phase and a Local Search phase. In each iteration, within the Construction phase, a random initial feasible solution is iteratively built following an adaptive reasoning. Then, the neighbourhood of that solution is explored, using a Local Search procedure, and if the best solution found is better than the best solution found in previous runs of the algorithm, it is kept. The process (Construction plus Local Search) is repeated until a number of pre-defined algorithm re-starts is performed.

The Construction phase is by itself an iterative process that, in each iteration, considers the following constructive procedure. All elements in a set of candidates are ranked, according to a *greedy function* that evaluates the contribution to the objective function obtained by adding that particular element to the solution under construction (this *greedy function* varies according to the problem being studied and is therefore a point for further analysis in this section). If the elements in the ranked list reach a given threshold, they are accepted for future decisions and stored in a *Restricted Candidate List* (RCL). The element(s) chosen in each iteration of the Construction phase is (are) then randomly selected among those in the RCL. By doing so, in each GRASP iteration a different initial solution is obtained and, hopefully, different regions of the search space are explored by the Local Search procedure.

In this section, we present an application of GRASP to the UCP and describe the details concerning its implementation. We propose a new solution representation that is better at handling minimum up and down time constraints, and describe the Construction and Local

Search phases. At the end we present the computational results obtained by applying this approach to a set of instances from the literature and compare them with results obtained with other approaches.

### 4.2.1 Solution representation

A UCP solution is usually represented by a binary matrix (see section 4.2), where each column represents the operation schedule of one unit over the planning horizon [Mantawy et al. 1998b; Yin Wa Wong 1998] and each element of a column represents the state of that unit, in a given period of time. As referred in section 4.1.1, this scheme is particularly interesting because it is a direct natural representation of the problem solution. However, it prevents time dependent constraints, such as minimum up/down time constraints, from being efficiently handled.

In this work a new solution representation that tries to better handle these constraints is proposed. It is inspired in [Rudolf and Bayrleithner 1999], and is obtained by first representing a solution in its traditional form, with a vector associated to each production unit. Then, an intermediate (integer) solution coding is built, the first element representing the initial state of each unit and the following elements representing the number of consecutive periods that a unit remains in a given state. Finally, for each element in each column (except for the first one) the minimum up/down time values ($T_i^{on}/T_i^{off}$), that force unit $i$ to remain in a given state for a certain period of time, is subtracted. This leads to the final representation that is particularly suitable for checking the minimum up/down time constraints. A sufficient condition for these constraints to be verified is that each value in the new representation is larger than or equal to zero. Non-zero values indicate the maximum number of periods for which a unit's current state may change.

**Example 5** *Proposed solution representation*

Consider the binary schedule of a unit, presented in Figure 4.5 (1's meaning that the unit is on; 0's meaning that it is off). Its initial state is on (the leading bit in the second representation), and it remains in that state for 3 periods. Then it is switched off for 4 periods and again switched on for a single period. Finally, it is switched off again, until the end of the scheduling horizon. Supposing that the unit's minimum up and down times are 1 and 2 periods, respectively, those values are subtracted from the previous values, leading to the final representation.

Although being appropriate for an efficient management of minimum up/down time con-

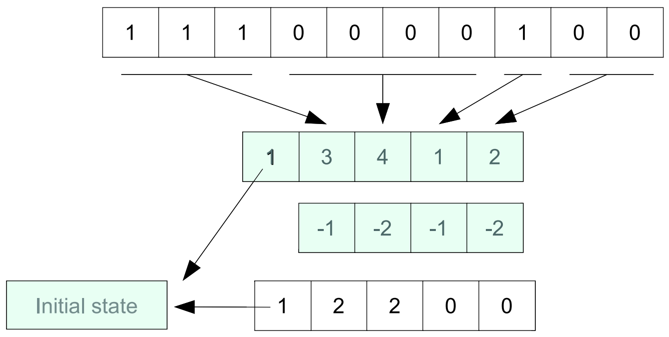


**Figure 4.5:** Proposed solution representation

straints, this representation is not suitable to handle other constraints, such as load and reserve constraints, a binary representation being more appropriate for the later. Therefore, in this work we have considered the two solution representations working in parallel, in complementary ways: the binary representation is used to check load and reserve constraints and the integer representation is used to check multi-period constraints, such as minimum up/down time constraints.

### 4.2.2 Construction Phase

The Construction phase procedure for the UCP is composed of three main steps. For each period $t$, one selects, from the set of all units that are available, those that if switched on in period $t$ satisfy the technical constraints (Filter A, in Figure 4.6). The impact in the objective function, evaluated by the greedy function *gf(i, t)*, of switching on (individually) a unit $i$ belonging to the intermediate list, in period $t$ is then measured and the units for which *gf(i, t)* is within a pre-defined interval are selected (Filter $\alpha$, in Figure 4.6), forming the Restrict Candidate List (RCL) for period $t$. The units that reach the established threshold are those in the interval given by expression (4.2), where $\alpha$ is a pre-defined parameter that controls the RCL size, $\underline{gf_t}$ is a lower bound of the greedy function obtained in period $t$ (i.e. $\underline{gf_t} = \min(gf(i,t))$) and $\overline{gf_t}$ an upper bound. One should notice that when $\alpha = 0$, GRASP can be viewed as a normal greedy algorithm and, when $\alpha = 1$, it becomes a random walk algorithm. To conclude an iteration of the Construction phase, units are randomly picked from the RCL, until load and reserve requirements are met.

$$\underline{gf_t} \leq gf(i) \leq \underline{gf_t} + \alpha\,(\overline{gf_t} - \underline{gf_t}) \tag{4.2}$$

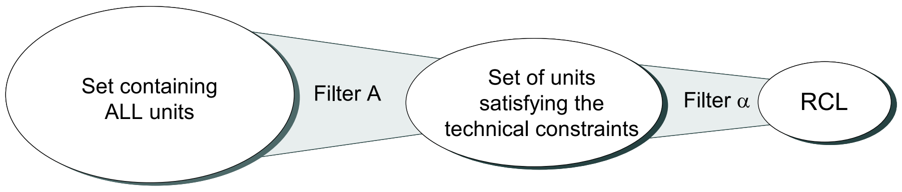


**Figure 4.6:** Construction Phase main steps

The Construction phase procedure is outlined in Figure 4.7 (T represents the planning horizon). In each iteration, the procedure starts by switching on those units that, due to some constraint, must be on in period $t$ (line 2). In line 3 the sum of the maximum production capacity of those units whose state had to be fixed to on is computed. If the sum, prod(t), verifies the reserve requirements, no further calculations are needed for that period. Otherwise, based on the RCL (line 5), other units are set to on, until load and reserve requirements are met. Finally, the operating levels that lead to a minimum operating cost, for period $t$, are computed (line 6) and the current solution cost is obtained (line 7).

**Algorithm Construction UCP**

```
1   For t = 1 to T
2       SwitchOnUnits(mustRun, minUpTime)
3       prod(t) = CheckCurrentProduction($P_{max}$)
4       if the reserve requirements are not verified
5           MakeRCL(Solution)
6       CalculateDispatchValues(Solution)
7       CalculateCurrentCost(Solution)
```

**Figure 4.7:** Algorithm Construction Phase

### Building the Restricted Candidate List

The elements to be stored in the RCL are the units that can be switched on in the period being analysed, the MakeRCL Algorithm (Figure 4.8) guaranteing that the solution will remain feasible.

First, an intermediate list (RCLCandidates) that contains only the units that, if switched on in period $t$, will verify the minimum down-time constraints, is built. If those units are not able to satisfy the demand and reserve requirements on their own, a second list (RCLNoCandidates), storing the remaining units of the system that are off in period $t$, is built. The units in RCLCandidates are then ranked, according to a *greedy function*, and those that reach a certain threshold are stored in the RCL (RCL = Achieve(RCLCandidates)). After constructing the RCL, the RCLCandidates list is updated and the value of $\alpha$ is incremented. When necessary, the same process is applied to RCLNoCandidates. However, as the elements in this list do not verify the minimum down time constraints, if they are chosen, they must be switched on since the last period they were on until period $t$ is reached.

Expression (4.3) defines the *greedy function* $gf(i,t)$ for unit $i$, in a generic period $t$. $FuelCost(P^i_{max})$ represents the fuel cost of unit $i$, when it is operating at its maximum production level ($P^i_{max}$), $SC(i,t)$ represents the start-up costs of unit $i$ (if it was off in period $t$ - 1), and $DC(i,t)$ represents the shut-down costs of unit $i$ (if it was on in period $t$ - 1). As the decision of switching unit $i$ on suppresses the shut-down cost in period $t$, this cost comes with a minus sign in expression (4.3).

$$gf(i,t) = \frac{FuelCost(P^i_{max}) + SC(i,t) - DC(i,t)}{P^i_{max}} \quad \forall t \tag{4.3}$$

**Example 6** *Building an initial solution for the UCP*

Consider the following 7–unit, T–period illustrative example, where the minimum down time of units 3 and 7 is considered to be 4 periods (Figure 4.9). The state of each unit, until t = 3, has already been fixed and the construction phase is now in its 4th iteration. When deciding which units will be on in t = 4, due to minimum down time constraints, units 3 and 7 cannot be committed in that period. Therefore they are excluded from the initial list, and their state is set to off. Each of the remaining units is then evaluated with the *greedy function* and, for a pre-defined value of $\alpha$ (set here to 0.85), the units that are candidate for being committed are units 1, 2, 5 and 6. Units are then randomly picked from that set, until load and reserve constraints are verified. In this example we have considered that after units 2 and 5 are selected, those constraints are verified. The units that remain unsettled, either in the RCL, or not, are set to off.

### 4.2.3 Local Search Phase

To explore the solution space with a Local Search procedure, a neighbourhood structure has been developed. The way it works is presented in Algorithm Neighbourhood Structure in

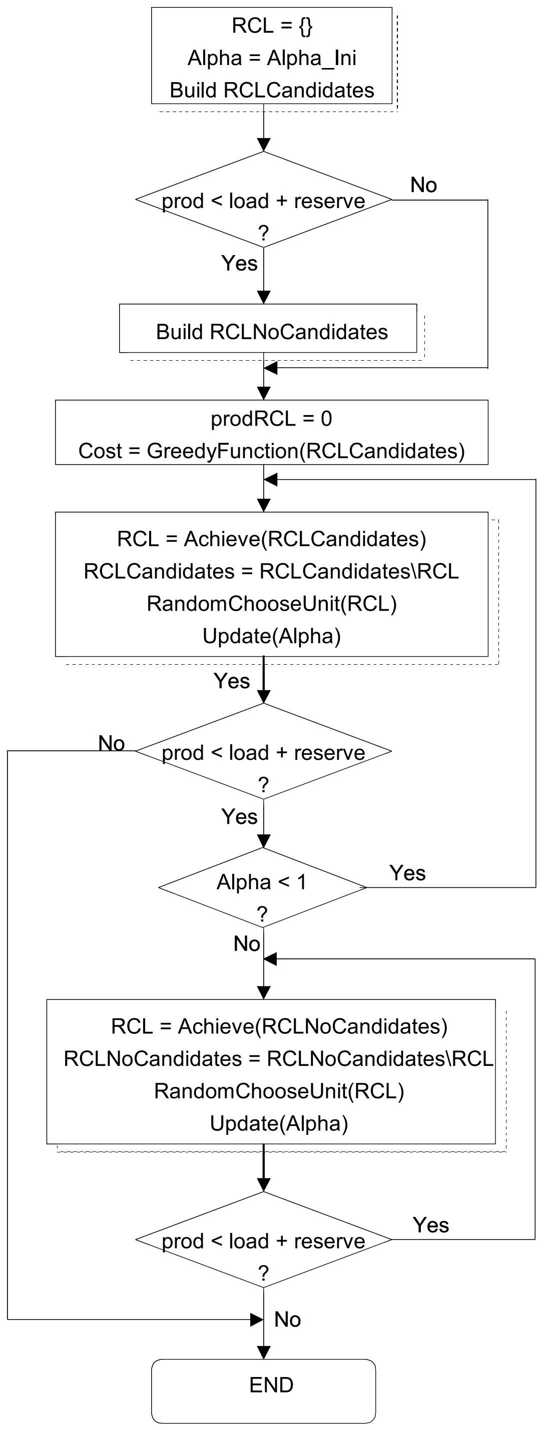


**Figure 4.8:** Flowchart for the MakeRCL procedure

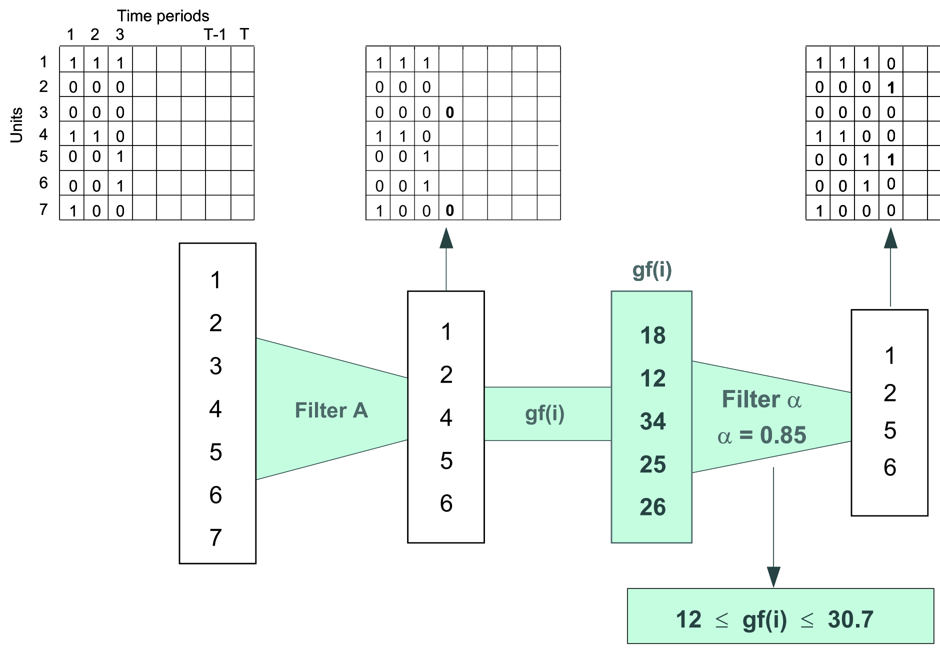


**Figure 4.9:** Building an initial solution for the UCP

Figure 4.10 and consists in randomly choosing one unit $i$, a period $t$ and a direction $dir$, which indicates if the changes will be done to the left (before) or to the right (after) of $t$. Depending on the state of unit $i$ in period $t$, this change will mean a shut-down or a start-up for a certain amount of time, decided by ShutDownInterval($i$, $t$, $dir$) or by StartUpInterval($i$, $t$, $dir$), respectively. The undertaken decision guarantees that minimum up and down time constraints are verified.

The rules used in the ShutDownInterval($i$, $t$, $dir$) procedure, for units that are always on, over the entire planning horizon, are described in Figure 4.11. They should be read as follows: **if** State(...) **or** State(...) **then** Down(...). State(a, b, c, d) represents the state being studied. $a \in$ {ON, OFF} indicates if a unit is always on or off. $b \in$ {Left, Right} is the direction of search. $c \in$ {ON, OFF} gives the initial state of unit $i$ and $d$ is a specific time period or time interval. Down($t_a$, $t_b$) is the interval for which the unit is switched off. UT(i) and DT(i) stand for the minimum up and down times of unit $i$, and per_ini(i) stands for the number of periods that unit $i$ has been on/off for t < 1 (i.e. for the periods before the planning horizon).

**Example 7** *Analysing shut-down rules*

Consider the first rule in Figure 4.11. If the unit has been on for an amount of time that

```
Algorithm NeighbStructure

i  = ChooseAtRandomUnit()
t  = ChooseAtRandomTime()
dir = ChooseAtRandomDirection()
If unit i is ON in period t
    ShutDownInterval(i, t, dir)
else
    StartUpInterval(i, t, dir)
```

**Figure 4.10:** Algorithm Neighbourhood Structure

is above its minimum up time, t1 is set to 0. Otherwise, t1 is set to the number of periods necessary to reach the minimum up time. The shutting down rule is applied in any of the following situations: 1) if the unit is always on, the search direction is Left, its initial state is also on and t is larger or equal to the unit's minimum down time plus t1 or, 2) if the unit is always on, the search direction is Left, its initial state is off and t is larger or equal to the unit's minimum down time plus its minimum up time. In such cases the unit is switched off from t - DT(i) + 1 to t.

The shutting-down rules for those units that change their state at least once over the planning horizon are the following: if the interval containing $t$ is equal to UT(i), the unit is switched off for all periods in that interval. Otherwise, it is switched off for a number of periods chosen at random in the interval [1, SumUp - UT(i)], where SumUp represents the number of consecutive periods for which the unit is on.

Furthermore, to guarantee that load and reserve requirements are still satisfied, we sometimes need to restore the feasibility of the solution. The feasibility recovery procedure is based on StartUpInterval($i$, $t$, $dir$) and switches on units, until feasibility is reached. With an exception for the recovery procedure, which is not needed in StartUpInterval($i$, $t$, $dir$), the same reasoning is followed for defining the start-up rules.

### Additional simplification

In what concerns StartUpInterval($i$, $t$, $dir$), a variant to the basic procedure has also been considered in this work. An "add-in", to check if there is an over-commitment of units, was included in the basic procedure. After each Start-Up operation, this additional simplification procedure will check, for each period of time, if the spinning reserve exceeds the required value. If this is the case, a greedy procedure is started: choose the period with the maximum

**Shut-down rules**

```
                    I
t1 = max(UT(i) - per_ini(i), 0)
State(ON, Left, ON, t ≥ DT(i) + t1)
State(ON, Left, OFF, t ≥ UT(i) + DT(i))
Down(t - DT(i) + 1, t)

                    II
t1 = max(UT(i) - per_ini(i), 0)
State(ON, Left, ON, t <  DT(i) + t1)
Down(t - t1, t - t1 + DT(i) - 1)

                    III
State(ON, Left, OFF, t = 0)
State(ON, Right, OFF, t = 0)
State(ON, Right, ON, t > 0 and t ≥ UT(i) - per_ini(i))
State(ON, Right, OFF, t ≥ UT(i))
Down(t, t + DT(i) - 1)}

                    IV
t1 = max(UT(i), t  - DT(i) + 1)
State(ON, Left, OFF, t > 0 and t < UT(i) + DT(i))
t1 = max(UT(i) - per_ini(i), 0)
State(ON, Right, ON, t < t1)
Down(t1, t1  + DT(i)  - 1)

                    V
State(ON, Right, OFF, t ≠ 0 and t < UT(i))
Down(UT(i), UT(i) + DT(i) - 1 )
```

**Figure 4.11:** Shutting-down Rules

excess of spinning reserve and check if there are any units that can be switched off, without loss of solution feasibility. In case they exist, they are switched off. Otherwise, the next period with maximum excess is chosen.

## 4.3 Computational study

This section describes the experiments carried out in order to assess the performance of the proposed GRASP approach. The description is followed by a presentation, discussion and comparison of the results obtained.

The code was developed in C++ and the computational tests were performed on a 500MHz Pentium III PC. However, as other computational results will be reported in this thesis, and they were performed in different machines, all results will be reported to a 800MHz Pentium III PC, the machine that was used for the computational studies in the remaining of this thesis. The SPECfp95 will be used to make CPU times comparable. This computer benchmark, provided by the Standard Performance Evaluation Corporation – SPEC – allows the relative comparison of CPU times obtained in different computer systems. A new benchmark index – SPECfp2000 – is also provided by SPEC. However, as there was no information available for one of the machines (the HP Apollo 9000-720), it could not be considered in this work.

### 4.3.1 Problem instances

The selection of the problem instances to be used in the computational analysis was based on the two following criteria: they should have diversified sizes, ranging from small to real-size problems and, for benchmarking purposes, they should also have been used with other methodologies. Following these guidelines, our choice was constrained to the problem instances presented in [Kazarlis et al. 1996] and to the ones presented in [Bard 1988]. However, as it was not possible to obtain detailed data to reproduce the larger instances contained in [Bard 1988], these instances were discarded, and tests have only been performed on the instances used in [Kazarlis et al. 1996]. They consider a 24-hour planning horizon and the number of units varies from 10 to 100 units. The instances are fully presented in Appendix A.

These instances have also been used by, e.g. [Feltenmark 1997], [Juste et al. 1999], [Cheng et al. 2000], [Cheng et al. 2002], [Senjyu et al. 2002] and [Valenzuela and Smith 2002]. The results obtained by [Feltenmark 1997] were excluded from this study, as they were worse than

| Machine | SPECfp95 |
|---|---|
| HP Apollo 9000-720 | 2.02 |
| Sun Ultra 2 | 14.7 |
| (dual 200MHz UltraSPARC CPU) | |
| 500MHz Pentium PC | 14.7 |
| 800MHz Pentium III PC | 28.9 |

**Table 4.1:** SPECfp95 values

those obtained with the pioneer work by [Kazarlis et al. 1996]. On the contrary, the results published in [Juste et al. 1999], [Cheng et al. 2000], [Cheng et al. 2002] and [Senjyu et al. 2002] claimed to be much better than those published by [Kazarlis et al. 1996]. However, to reduce the search space, those authors have changed the problem formulation (the transition cost function). Thus, the results are not comparable and, therefore, only the work by [Kazarlis et al. 1996] and by [Valenzuela and Smith 2002] have been used for comparison purposes.

In the following sections we report and discuss the results obtained. The computational experience reported in [Kazarlis et al. 1996] was performed in a HP Apollo 9000-720, with a SPECfp95 of 2.02 (see Table 4.1). The Sun Ultra 2 with dual 200MHz UltraSPARC CPU used in [Valenzuela and Smith 2002], as well as the 500MHz Pentium PC used in this work, have a SPECfp95 of 14.7. As the computational tests presented in the following chapters of this thesis, were performed in a 800MHz Pentium III PC with a SPECfp95 of 28.9 all results will be reported to this machine.

### 4.3.2 Parameter tuning

To set up a search for GRASP, two parameters have to be tuned: the maximum number of iterations and $\alpha$. Therefore, it is relatively simple to find out by experimentation the "best" set of parameter values. In this work, the values for the maximum number of iterations and for $\alpha$, for each instance, were fixed by performing some systematic computational experiments. We have tested values of $\alpha$ ranging from 0 to 1, with increments of 0.1, and the maximum number of iterations studied was within the set [50, 100, 200, 500, 1000, 2000]. For the results presented in this section, this value was set to 200 iterations, for all problem instances.

| Problem size | $\alpha_{better}$ | Production cost | CPU time (sec) |
|---|---|---|---|
| 10 | - | 565 825 | 17 |
| 20 | 0.6 | 1 128 160 | 571 |
| 40 | 0.6 | 2 259 340 | 1 511 |
| 60 | 0.4 | 3 383 184 | 2 638 |
| 80 | 0.6 | 4 525 934 | 3 308 |
| 100 | 0.6 | 5 668 870 | 4 392 |

**Table 4.2:** Computational results for GRASP

### 4.3.3 Computational results

When evaluating the performance of random based methods (like metaheuristics), it is natural to test if the methods are correctly implemented by checking whether different seeds, that initialise the random number generator, do not influence the final results obtained with the method. In this work, and for that purpose, tests have been performed for the 10, 20 and 60 unit instances. For fixed values of $\alpha$, five seeds were considered and the results obtained showed that their values did not really influence the global behaviour of the method. In fact, in all cases the same results were obtained, independently of the seed being used.

For each instance, several computational tests were also done, to obtain the value of $\alpha$ leading to a better performance of the algorithm. That value was fixed at 0.6 for the 20, 40, 80 and 100 unit problems and 0.4 for the 60 unit problem. For the 10 unit problem the method performed equally well for values of $\alpha$ above 0.3. The results obtained are those presented in Table 4.2, for a maximum number of iterations equal to 200.

The introduction of the simplification procedure mentioned in Section 4.2.3, resulted in a considerable improvement in the efficiency of the algorithm for the 20, 40 and 80 unit problem. Those results, reported in Table 4.2, show that better solutions could be found within much shorter CPU times. One should also notice that the value of $\alpha$ that led to a better performance of the algorithm was, in this case, higher than that used when no simplification techniques have been considered. This adjustment of parameters can be explained, in a certain way, by the fact that the simplification technique used follows a greedy reasoning. Therefore, GRASP must introduce more randomness in the search (by increasing $\alpha$), to properly explore the search space.

| Problem size | $\alpha_{better}$ | Production cost | CPU time (sec) |
|---|---|---|---|
| 20 | 0.8 | 1 126 805 | 171 |
| 40 | 0.8 | 2 255 416 | 906 |
| 80 | 0.8 | 4 524 207 | 863 |

**Table 4.3:** Computational results for GRASP (with simplification procedure)

| | Production cost | | | | Average time (sec) | |
|---|---|---|---|---|---|---|
| Problem size | DP | LR | GA | LR–MA | GA | LR–MA |
| 10 | 565 825 | 565 825 | 565 825 | 565 827 | 221 | 87 |
| 20 | - | 1 130 660 | 1 126 243 | 1 127 254[1] | 733 | 287 |
| 40 | - | 2 258 503 | 2 251 911 | 2 249 589 | 2697 | 217 |
| 60 | - | 3 394 066 | 3 376 625 | 3 370 595 | 5840 | 576 |
| 80 | - | 4 526 022 | 4 504 933 | 4 494 214 | 10 036 | 664 |
| 100 | - | 5 657 277 | 5 627 437 | 5 616 314 | 15 733 | 1 138 |

**Table 4.4:** Computational results reported in [Kazarlis et al. 1996] and in [Valenzuela and Smith 2002]

**Comparison with other approaches**

Table 4.4 presents the best results reported by [Kazarlis et al. 1996] and by [Valenzuela and Smith 2002]. In [Kazarlis et al. 1996], the problem is solved through Dynamic Programming (DP), Lagrangian Relaxation (LR) and Genetic Algorithms (GA). In [Valenzuela and Smith 2002] the UCP is solved through Genetic Algorithms, Memetic Algorithms (MA) and Memetic Algorithms starting with an initial solution obtained through Lagrangian Relaxation (LR–MA). LR–MA performed better for all instances, except for the 10 and 20 unit ones, where the best results were achieved by the MA approach. The *Average time* columns in Table 4.4 refer to the average CPU time necessary for convergence, i.e. for obtaining a solution equal to, or better than, that of the LR approach.

When comparing the best results obtained by GRASP, with those presented in [Kazarlis et al. 1996] for Lagrangian Relaxation (LR), they seem to be comparable; GRASP presenting better results for the 20, 40, 60 and 80 unit problems, and LR for the 100 unit problem.

[1]Obtained with the MA approach

| Problem size | Relative error(%) | | |
|---|---|---|---|
| | GRASP vs LR | GRASP vs GA | GRASP vs LR–MA |
| 10 | 0.00 | 0.00 | $\approxeq 0$ |
| 20 | -0.342 | 0.049 | -0.399 |
| 40 | -0.137 | 0.157 | 0.259 |
| 60 | -0.322 | 0.194 | 0.374 |
| 80 | -0.040 | 0.428 | 0.667 |
| 100 | 0.205 | 0.736 | 0.936 |

**Table 4.5:** GRASP vs LR, GRASP vs GA and GRASP vs LR–MA: relative error

The relative error of the results (*(x - Best_Result)/Best_Result*) is presented in Table 4.5 (a minus sign is assigned to the cases where GRASP performed better).

The comparison with the Genetic Algorithms approach shows that, although being worse, the results obtained with GRASP are of the same order of magnitude. If one excludes the result obtained for the 100 unit problems, where GRASP had its worst performance (with a deviation of 0.736%, when compared with the result in [Kazarlis et al. 1996]), for the other instances the maximum deviation was of 0.428% (Table 4.5). These values are slightly increased when GRASP is compared to LR–MA. Even so, it overcomes LR–MA, for the 20-unit problem.

In Table 4.6 we report the CPU times of each approach, as if they had all been performed in a a 800MHz Pentium III PC, with a SEPCfp95 of 28.9. These values are obtained by multiplying the original values reported in Tables 4.2, 4.3 and 4.4 by their machine's SEPCfp95, divided by the 800MHz Pentium III PC SEPCfp95. The values show that GRASP is faster for the smaller problem but when the problem size increases its efficiency drastically decreases. One should also notice the remarkable reduction in CPU time introduced by the simplification procedure, in the GRASP approach (GRASP with "add-in"), for the 20, 40 and 80 unit problems.

## 4.4 Concluding remarks

In this chapter we have presented a new solution representation for the UCP and proposed a GRASP algorithm to solve the problem. Although not overcoming the best results presented in the literature, the results obtained in this work are encouraging and clearly justify further research on the topic.

| | CPU time (sec) | | | |
|---|---|---|---|---|
| Problem size | GA | LR–MA | GRASP | GRASP with “add-in” |
| 10 | 15.45 | 42.73 | 8.65 | – |
| 20 | 51.23 | 145.98 | 290.44 | 86.98 |
| 40 | 188.51 | 110.38 | 768.57 | 460.84 |
| 60 | 408.19 | 292.98 | 1341.82 | – |
| 80 | 701.47 | 337.74 | 1682.62 | 438.97 |
| 100 | 1099.6 | 680.57 | 2233.99 | – |

**Table 4.6:** CPU time – comparable values

By being able to achieve, for some instances, better results than those obtained with Lagrangian Relaxation (that is still considered to be one of the most powerful approaches for solving the UCP), and by achieving values that are similar to those obtained with Genetic Algorithms (an approach that already has some “tradition” in the area), GRASP shows that it has potential to become an additional useful technique for solving the UCP.

Nevertheless, the proposed approach still requires a careful parameter tuning, to reach good quality solutions. This is naturally a major drawback (common to any metaheuristic approach), not easily accepted by Decision Makers, as they are not confident in decision support tools that, if not correctly tuned, may lead to very bad quality results.

One should therefore consider that a main line of research is the development of techniques that reduce the influence of parameter tuning on metaheuristics’ performance. This has been one of our major research concerns, that has strongly influenced the work presented in the next chapter.

# Chapter 5

# Constraint Oriented Neighbourhoods

As mentioned before, parameter tuning is a critical requirement of most metaheuristics, because the set of parameters that is chosen may highly influence the algorithms' performance. It may be a rather longstanding process that, unless one is capable of somehow automatizing it, or of reducing the impact of the parameters on the optimisation process, must be repeated for each single problem instance under study. This rather repetitive procedure was followed in Chapter 4, where we have proposed a GRASP approach to solve the UCP. As stated, the best results reported were obtained after an accurate parameter tuning was performed.

This chapter is devoted to the presentation of a new general approach – *Constraint Oriented Neighbourhoods* (CON) – that tries to reduce the dependency of a metaheuristic on parameter tuning. The idea behind the approach is to partially control metaheuristics randomness by defining several "special" neighbourhood movements, in such a way that drastic changes in a solution in successive iterations are avoided, allowing a smoother search and, consequently, a correct intensification phase during the search process. As the intensification phase becomes less dependent on parameter tuning, we expect the heuristic to be more robust. To illustrate its main features and potential, the concept of *Constraint Oriented Neighbourhoods* will be applied to the Unit Commitment Problem in Power Systems Management. However, this concept should be seen as general as it may be applied to many other problems. The characteristics that those problems should present so that the concept can be usefully applied, are described later in this text.

The chapter is structured as follows. A discussion on metaheuristics' robustness and on the main lines of research within this area is first presented. The *Constraint Oriented Neighbourhoods* search strategy is then described and its scope of application is stated. Finally, CON is applied to the UCP and the computational results obtained by applying this strategy to a set of instances from the literature is discussed and compared with results obtained by using alternative approaches.

## 5.1 Robustness in metaheuristics

The advantages of metaheuristics are widely reported in the literature (e.g.[Glover and Kochenberger 2003]): they are usually easier to develop than optimising constructive heuristics, leading to a significant reduction in software development time and effort; they are able to easily incorporate new information/details in the model, being therefore easy to adapt to different problem variants; they can tackle complex cost functions; they are able to solve the very same problem for completely different objective functions, with minor changes in the code, etc.

However, these advantages are not a sufficient argument for their adoption by Decision Makers, who are in general reluctant to using this type of algorithms in practice. There are two main reasons for this type of attitude:

i. The performance of metaheuristics is, in general, highly dependent on parameter tuning. This tuning process is not transparent for Decision Makers and they do not usually fully understand what is the real meaning of the different parameters. Moreover, they do not want to depend on an Analyst every time that parameter tuning is required.

ii. Metaheuristics do not have a sound mathematical foundation, when compared to more traditional techniques, and in general this fact leads to a lack of confidence from the end user.

In this chapter we will focus on issue i) that is strongly related to the non satisfaction of two of the properties that, according to [Hansen and Mladenović 2003] and [Barr et al. 1995], should be found in metaheuristics to guarantee their practical and theoretical interest, namely, *user friendliness* and *robustness*.

**Property 1** *User friendliness* – "Heuristics should be well-defined, easy to understand and, most important, easy to use. This implies that they should have as few parameters as possible and ideally none." [Hansen and Mladenović 2003]

**Property 2** *Robustness* – The performance of heuristics should be consistent, performing well over a wide range of instances and parameters.

Attempts have already been made to reduce manual parameter tuning by way of adaptive metaheuristics, as in [Battiti 1996], where history-based feedback is integrated in local search for online determination of some search parameters. In terms of *robustness* improvement, several lines of research have been followed. It has been shown that parallel implementations tend to be more robust than the single thread ones [Cung et al. 2001], and it is also claimed that hybrid approaches present better results (in terms of robustness) [Ribeiro et al. 2002].

In this work, we do propose a new strategy to reduce the impact of metaheuristics' parameters on their performance, and to improve their robustness in some domains of combinatorial optimisation problems. It consists in defining several neighbourhood structures and, in each iteration, choosing one of these alternative structures, so that the new solution is not drastically different and the search process is smooth. By doing so, we aim at achieving good intensification phases, following a procedure that is less dependent on parameter tuning than usual and consequently we expect to obtain metaheuristics that are more robust. It should be noted that this search strategy is different from VNS [Hansen and Mladenović 1999] that uses the same neighbourhood for a certain number of iterations, and then changes (or not) to a different neighbourhood structure, trying to escape from local optima, not taking into account the features of the current solution.

## 5.2 Constraint Oriented Neighbourhoods

A major advantage of metaheuristics over problem specific heuristics is that they can often be easily implemented, as they are usually designed on simple and easy to understand neighbourhood structures. However, in some cases, it may be difficult to keep solution feasibility.

Let us consider a neighbourhood structure to be a two-step procedure where one first removes part of the current solution and then rebuilds it so that a feasible solution is reached. There are many cases where simple remotion procedures lead to infeasible solutions, the recovering techniques needed to reach feasibility involving complex rules that may drastically change the structure of the solutions. In such cases the algorithm may not be able of correctly exploiting the neighbourhood space, unless very restrictive metaheuristic parameters are set, and it is probably better to define more elaborate neighbourhood structures, to prevent a complex and computationally hard recovering process, and to allow a smoother search process.

Getting a smoother search process, less dependent on parameter tuning, is the major motivation for *Constraint Oriented Neighbourhoods*. This is achieved by defining neighbourhood structures that, if in a first phase lead to infeasible solutions, feasibility is easy to recover and the resulting solution is not very different in structure from the current one. In each iteration, the metaheuristic first checks what kind of constraints will be violated if part of the current solution is removed (i.e. if the state of one (or more) decision variable is changed) and then, accordingly, it applies one specific neighbourhood movement, from a set of possibilities.

Neighbourhood *structure* and neighbourhood *movement* are generally used to refer to the same concept (see Definition 10 in section 2.3.2). In this text, and within the context of CON, we will however distinguish between them, as follows. A neighbourhood *structure* is the two step procedure where: 1) part of the current solution is removed, leading (or not) to an infeasible solution; 2) a neighbourhood *movement* is applied to restore solution feasibility (if necessary), or to diversify the search process.

### 5.2.1 General description

The *Constraint Oriented Neighbourhoods* (CON) strategy assumes that a good understanding of the solution structure and of the reasoning underlying the problem decisions may be helpful for a successful search process. Therefore it implies that a pre-analysis is performed to study the structure of solutions in each particular problem. The aim of this phase is to detect those constraints that, for a general purpose neighbourhood structure (a neighbourhood that remains constant all over the search process and that may significantly change the current solution), require the use of a complex recovering algorithm – the *hard recovering constraints*.

Once the sets of *hard recovering constraints* have been identified, all possible combinations of constraint violation are enumerated and, for each scenario, a neighbourhood movement is designed. The neighbourhood movements are not necessarily different for all cases but, in any case, if $n$ types of *hard recovering constraints* are considered, one must analyse $2^n$ different scenarios. When all the constraints are violated, one can apply a general purpose movement (GPM). As shown in Table 5.1, if three types of *hard recovering constraints* are considered, one must analyse $2^3$ different scenarios. A, B and C represent three types of *hard recovering constraints* and T means that a constraint of type $i$ is violated (F, otherwise). For scenarios 3 and 4, for example, the same movement applies. If all the constraints are violated, one can apply a general purpose movement (GPM – Scenario 1).

| Scenario | Hard Recovering Constraints Violated | | | Movement |
|---|---|---|---|---|
| | A | B | C | |
| 1 | T | T | T | GPM |
| 2 | T | T | F | M1 |
| 3 | T | F | T | M2 |
| 4 | T | F | F | M2 |
| 5 | F | T | T | M3 |
| 6 | F | T | F | M1 |
| 7 | F | F | T | M3 |
| 8 | F | F | F | - |

**Table 5.1:** Example of definition of neighbourhood movements

The main steps of the basic *Constraint Oriented Neighbourhoods* strategy are presented in Figure 5.1. As the concept of CON may be applied to any metaheuristic based on Local Search, the number of generated solutions in the neighbourhood of the current one may vary, depending on the particular metaheuristic considered. The acceptance criterion will also depend on the metaheuristic considered and, as so, it is not specified in this general algorithm description.

First, an initial solution is built, by way of any simple constructive heuristic, guaranteeing a feasible solution without any particular concern on its quality. In each iteration of the process, for the current solution $X_i$, the *ChangeVariableState* step is applied ($x_a$ is the decision variable whose state is changed). This will most probably lead to an infeasible solution, where some or all types of constraints are not satisfied. So, in *CheckConstraintViolation(*$X_i \setminus x_a$) one detects the sets of *hard recovering constraints* that are not met (SetConst) and, accordingly, a specific neighbourhood movement (Neighb) is selected in *SelectNeighbMovement(SetConst)*. Finally the selected movement is applied to the current solution, in *GenerateRandomSolution*, leading to a new solution, Y, that may be accepted, or not, as the new current solution.

### 5.2.2 The reasoning behind CON

To better understand the reasoning behind the *Constraint Oriented Neighbourhood* concept consider the following example that analyses the possible impact of a General Purpose

```
Algorithm CON

i = 0
Build an initial solution $X_0$
Repeat
    $x_a$ = ChangeVariableState($X_i$)
    SetConst = CheckConstraintViolation($X_i \backslash x_a$)
    Neighb = SelectNeighbMovement(SetConst)
    Y = GenerateRandomSolution(Neighb, $X_i \backslash x_a$)
    Accept (or not) Y as the new current solution, $X_{i+1}$,
    according to the metaheuristics rules
    i++
Until Stopping criterion is met
```

**Figure 5.1:** Basic Constraint Oriented Neighbourhoods search strategy

Neighbourhood in a solution structure.

**Example 8** *Impact of a General Purpose Neihbourhood in a solution structure*

Suppose that you have to assign people to shifts, subject to some constraints: 1) Each person $r$, when assigned to a shift $t$ must remain assigned for at least $u_r$ consecutive shifts, and for no more than $l_r$ consecutive shifts. 2) After being assigned to a shift, a person will have to be off for, at least, $o_r$ consecutive shifts. 3) Each shift requires that at least $n_t$ people are assigned to it. Consider also that a General Purpose Neighbourhood for this problem would be to randomly pick a pair (r, t) in the current solution, change its state since the first shift that the person was assigned to, until shift $t$ and, if necessary, apply some recovering rules to recover solution feasibility.

Figure 5.2 exemplifies the case where such a neighbourhood is applied, for a problem with 5 people and 7 shifts. In this matrix, a 1 indicates that a person is assigned to a shift (0, otherwise). It has also been considered that $u_5 = 2$, $o_1 = 3$, $o_4 = 2$, $n_5 = 3$, $n_6 = 2$ and $n_7 = 3$.

First, the pair (5,6) is randomly chosen (Figure 5.2 - a)), and the state of person 5 is changed from 1 to 0, from t = 5, until t = 6 (Figure 5.2 - b)). The resulting solution is infeasible because $u_5 = 2$. Trying to recover feasibility, the state of pair (5,7) is changed to 0 (Figure 5.2 - c)) but the resulting solution is still infeasible because the number of persons assigned to shifts 5, 6 and 7 is not enough. So, in Figure 5.2 - d), person 4 is assigned to shifts 5 and 6, the resulting solution being still invalid because, after a shift, person 4 must be off for at least 2 consecutive periods. Therefore, the state of that person in shift 4 is changed (Figure 5.2 - e)). It is still necessary to verify the minimum number of people that are assigned to shift 7 (see Figure 5.2 - f)). By doing so, we reach a solution where person 1 stays off for 2 periods, after a shift. As it is required that the person stays off for at least 3 shifts ($o_1 = 3$), a new operation is applied to the solution. Finally, we reach a feasible solution (Figure 5.2 - g)).

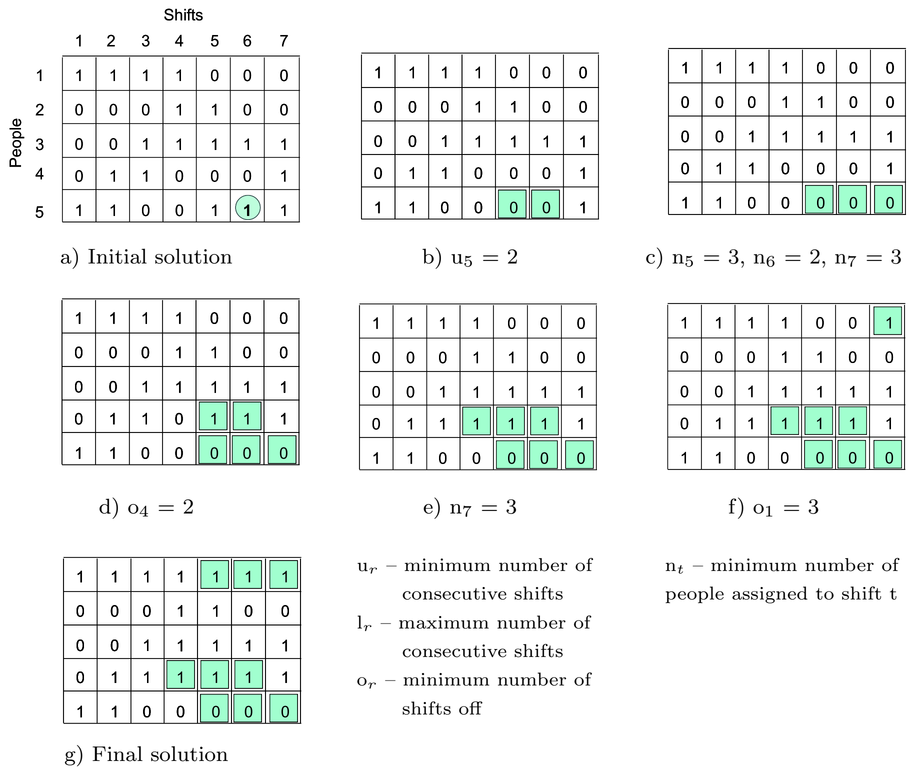

Shifts
People
a) Initial solution
b) $u_5 = 2$
c) $n_5 = 3$, $n_6 = 2$, $n_7 = 3$
d) $o_4 = 2$
e) $n_7 = 3$
f) $o_1 = 3$
g) Final solution
$u_r$ – minimum number of consecutive shifts
$l_r$ – maximum number of consecutive shifts
$o_r$ – minimum number of shifts off
$n_t$ – minimum number of people assigned to shift t

**Figure 5.2:** Effect of a general purpose neighbourhood in a solution (shifts assignment problem)

As shown, what we might expect to be a localised operation, leading to a solution similar to the current one, ended up strongly changing it. If this occurs frequently when applying a neighbourhood operation to a solution, the resulting solutions will tend to be far apart in the solution space and no intensification during the search process will be achieved. Therefore, a different approach should be developed to avoid this shortcoming. A correct intensification of the search process may be obtained by defining neighbourhood movements that lead to solutions that are similar in structure to the current one. For doing so, the solution should be analysed not as a whole, but as the result of a set of individual decisions that were made with a specific purpose. The following example illustrates this idea.

**Example 9** *Trying to understand the structure of solutions*

For the problem described in the previous example, consider the solution presented in Figure 5.3. Looking at the schedule of person 3, if he/she was off during shifts 1 and 2 and it is now decided to assign that person to shift 3, there must be a reason for making such a decision. Either it was made to comply with problem constraints (e.g. there was a person lacking in that shift) or solely to diversify the search. Following the same reasoning, why was person 1 assigned from shift 1 to shift 4, and then is off for some shifts? Probably because that person has reached the maximum number of consecutive shifts that he/she may be assigned. Therefore, if one cancels the decision of assigning person 3 to shift 4, or the one of not assigning person 1 to shift 5, different constraints will be violated. Thus, different procedures should be followed to recover solution feasibility and it might be interesting to define different neighbourhood movements, according to the type of constraints that are not met.

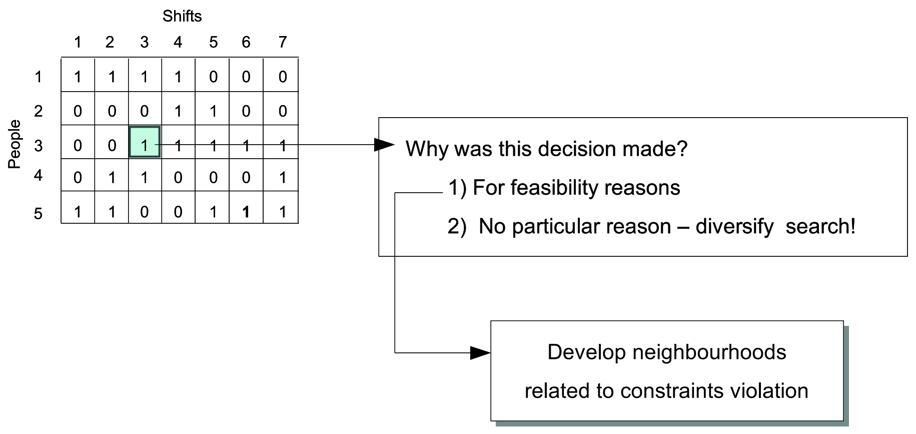

Shifts
People
1 2 3 4 5 6 7
1 1 1 1 1 0 0 0
2 0 0 0 1 1 0 0
3 0 0 1 1 1 1 1
4 0 1 1 0 0 0 1
5 1 1 0 0 1 1 1
Why was this decision made?
1) For feasibility reasons
2) No particular reason – diversify search!
Develop neighbourhoods related to constraints violation

**Figure 5.3:** Trying to understand the structure of solutions

One may ask why this strategy should lead to metaheuristics that are less dependent on parameter tuning. The underlying reason is that, when general purpose neighbourhoods lead to neighbour solutions that are very different in structure from the current solution (as in Figure 5.2), the way of preventing a random walk like search is by adopting more strict acceptance criteria, performing an accurate parameter tuning. Otherwise, in each iteration the biasing solution will be located in completely different regions of the search space, and a correct exploitation of each region will not be achieved. However, if we try to prevent that such drastic changes are allowed, as is done with the CON approach, solutions will tend to be similar in successive iterations and correctly explore a given region of the solution space, depending less on the metaheuristic parameters. One should however notice that a very smooth search process may lead to suboptimal results. So, to avoid getting stuck in local optima, CON considers a general purpose neighbourhood to be applied when e.g. all hard recovering constraints are not verified.

### 5.2.3 Scope of application

The *Constraint Oriented Neighbourhoods* search strategy should be considered for problems where some of the following conditions are met, in case general purpose neighbourhoods are applied:

i. if complex algorithms are required to recover solution feasibility;

ii. if consecutive solutions are often significantly different, and promising regions of the search space are not correctly explored;

iii. if worse solutions are often produced, leading to a significant increase in CPU time.

Classical optimisation problems such as the TSP or the Knapsack problem do not, in general, suffer from such problems and therefore do no seem interesting for using this strategy. However, other problems, such as Resource Constrained Project Scheduling [Viana and Sousa 2000], Timetabling [Burke and Petrovic 2002] or Rostering Scheduling problems [Ernst et al. 2004], as well as the UCP, might well be considered as potential cases for applying this strategy.

## 5.3 Application to the Unit Commitment Problem

In this section we present an application of *Constraint Oriented Neighbourhoods* to the Unit Commitment Problem of thermal power units. We first state the reasons to use this strategy and then present some implementation details. For comparison purposes with the work developed under Chapter 4, this strategy was embedded in a GRASP algorithm.

### 5.3.1 Adequacy of CON to tackle the UCP

The general purpose neighbourhood described in Chapter 4 for the UCP did not seem in fact very adequate to tackle the problem, both in terms of algorithm robustness and effectiveness. Two main characteristics of the problem seem to negatively affect the performance of such a neighbourhood:

i. It is a true multi-period problem. This requires complex feasibility recovering neighbourhood movement algorithms that usually lead to neighbour solutions that are significantly different from the current one.

ii. The high dependency of the load profile on the period of the day. Except for a few abnormal situations, it is easy to detect in a load profile periods corresponding to a peak in load demand, and others corresponding to low load demand. Low load periods are characterised by having most of the system units off. Therefore, assuming for example that we are using the "traditional" neighbourhood (see Chapter 4), the probability of choosing a period $t$ of the day corresponding to a low load period, and selecting a unit $i$ that is currently off, is high. If the decision is to switch on that unit, the probability of generating "interesting" solutions is low and consequently many solutions will be discarded, leading to a significant increase in computational time.

### 5.3.2 Algorithmic issues

In general, a Unit Commitment solution is represented by a binary matrix, where each row represents the operation schedule of one unit, over the planning horizon. For a given solution, a pair $(i,\ t)$ is called a "transition point" if unit $i$ changes its state from on to off or from off to on, in period $t$.

Transition points corresponding to units switching on are likely to be critical when designing neighbourhoods to tackle the UCP. If a unit was off in period $t$ - 1, and is switched on in period $t$, there is a reasoning behind this decision that should be understood – either the unit is switched on to satisfy some problem constraints (load, reserve, etc.), or simply to allow that a correct diversification is achieved. In the first case, several alternative movements, that should take into account the kind of constraints that are critical, might be applied. Accordingly different alternatives should be studied, thus defining a *Constraint Oriented Neighbourhoods* approach.

Three neighbourhood movements, corresponding to the three sets of hard recovering constraints that were detected (demand, reserve and minimum up-time constraints), were defined: Load Movement, Reserve Movement and Up-time Movement. These movements, described below, can be applied in any point of the search process, according to the evaluation of constraint violation.

**Initial operations**

For implementing the neighbourhood structure considered in this work, the following initial steps are performed. Given a solution, we start by randomly choosing a period $t$ and, for that period, a unit $i$ that is on. Then, one checks what happens to solution feasibility if the unit is switched off in the transition points immediately before and after $t$. Two things may happen: 1) solution feasibility is maintained, i.e. there was not a "rational" reason for the unit to be on and, therefore, it should be switched off with no further changes in the solution; 2) solution feasibility is not maintained and, in this case, one must analyse which constraint(s) is (are) not verified and accordingly perform a specific movement.

**Load Movement**

The *Load Movement* is applied whenever load requirements are not met, if a unit (*geroff*) is switched off in period $t_1$. To restore solution feasibility a procedure that divides the units into expensive and cheap generation units was implemented. To perform this partition, the minimum and maximum production cost, per MW, of each unit ($\lambda^i_{min}$ and $\lambda^i_{max}$, respectively) are computed, based on the derivative of the fuel cost function: $\lambda(P_i) = a_i P_i + b_i$.

**Definition 1** Unit $a$ is *cheaper* than unit $b$ if $\lambda^a_{max} \leq \lambda^b_{min}$. Then $b$ is *more expensive* than $a$.

**Definition 2** Unit $a$ is *possibly cheaper* than unit $b$ if $\lambda^a_{min} \leq \lambda^b_{min}$.

As described in Figure 5.4, for a given period $t_1$, the *Load Movement* will first select those units (SUnits) belonging to the set of *cheaper* units. If there are no such units available, or if they are not sufficient to recover load demand, the units from the set of *possibly cheaper* units are added to SUnits. After confirming that load can be recovered, *geroff* is switched off and then, while load is not recovered, units are continuously selected from SUnits to be switched on. For each such unit (*ger*), we first check for how many periods does that unit need to be switched on (*switch on period*), to comply with minimum up-time constraints, and only then is the unit switched on, in those periods. At the end, we must check if the production in $t_1$ is already enough and, if so, the process is finished. Otherwise, a new generator is selected and the algorithm is repeated.

```
Algorithm Load Movement($t_1$, geroff)

SUnits = CheaperUnits($t_1$)
if SUnits = {} or production is not enough
    SUnits = SUnits U PossiblyCheaperUnits($t_1$)
Available Prod = CheckLoad(SUnits)
if SUnits $\neq$ {} and Available Prod
    SwitchOff($t_1$, geroff)
    While SUnits $\neq$ {} and production is not enough
        ger = RandomlyChooseUnit (SUnits)
        Remove(ger, SUnits)
        switch on period  = StartUpTime(ger)
        SwitchOnGer(ger, switch on period)
        CheckProduction($t_1$)
```

**Figure 5.4:** Algorithm Load Movement

## Reserve Movement

The *Reserve Movement* is applied whenever reserve requirements are not met, even if the load requirements are, if unit $i$ is switched off in period $t_1$. A unit $i$, leading to this situation, will be called "reserve unit". A "reserve unit" is usually more expensive than the other units on, and it is not producing at its maximum production level. Based on these assumptions, the reasoning behind this movement is to replace the chosen "reserve unit" by another "reserve unit" of lower cost.

As described in Figure 5.5, for a given period $t_1$, a *Reserve Movement* starts by defining the set of *possibly cheaper* units that are off in that period. To avoid major changes in the solution, this set is reduced by considering only the units that can be switched on, only in period $t_1$ (*ReduceSet(SUnits)*). Finally, the unit(s) that will be switched on, to recover solution feasibility, is (are) randomly selected from this reduced set. The algorithm proceeds in a way very similar to that proposed for the *Load Movement*, the only difference being that, in this case, *switch on period* always equals 1.

## Up-time Movement

In a state transition point, if by switching the unit off, load and reserve requirements are still met but minimum up-time constraints are not, the unit should be set to on, only for minimum up time purposes. For such a case, a *Up-time Movement* is performed. Two situations must be considered. Supposing that the unit is on from $t$ to $t_1$:

```
Algorithm Reserve Movement($t_1$, geroff)

SUnits = PossiblyCheaperUnits($t_1$)
ReduceSet(SUnits)
Available Prod = CheckLoad(SUnits)
if SUnits ≠ {} and Available Prod
  SwitchOff($t_1$, geroff)
  While SUnits ≠ {} and production is not enough
    ger = RandomlyChooseUnit (SUnits)
    Remove(ger, SUnits)
    SwitchOnGer(ger, $t_1$)
    CheckProduction($t_1$)
```

**Figure 5.5:** Algorithm Reserve Movement

1) Move to a solution with the unit *on* from $t$ + 1 to $t_1$ + 1.

2) Move to a solution with the unit *on* from $t$ - 1 to $t_1$ - 1.

## General algorithm description

The neighbourhood movements described above have been implemented in a *Constraint Oriented Neighbourhoods* strategy, embedded in a GRASP [Feo and Resende 1995]. The resulting algorithm that was designed for the UCP is described in Figure 5.6.

For each GRASP initial solution the following procedure is considered. In line 7, a set is built, containing those units that are on in period $t$, and that are at their minimum production level ($P_{min}(t)$) at the point of transition from off to on ($t_{left}$), or at the point of transition from on to off ($t_{right}$). If there are no units verifying those conditions, a new attempt is made (line 9) to build a set of units, that are on in period $t$, and that are not at their maximum production level in $t_{left}$ or $t_{right}$.

After choosing randomly an unit $i$ from that set, and a period to apply the movement ($t_1$), the possibility of switching off unit $i$ in period $t_1$ ($x_{it_1}$) is considered (line 12). This can result in an infeasible solution, as checked in line 13, and if so, a suitable neighbourhood movement is selected and applied (lines 14 and 15). Otherwise, a GPM is applied. The new solution, Y, is then evaluated (line 16) and subject to an acceptance criterion. If accepted, it becomes the new current solution of the Local Search phase (line 18). Furthermore, if the new solution is better than the best solution found so far (X*), it becomes the new overall best solution. The algorithm proceeds until the pre-specified number of initial solutions has been reached.

**Algorithm CON-UCP**

```
number solutions = 0
Repeat
    j = 0
    Build an initial solution $X_j$
    Repeat
        Randomly choose t
        Units = ObtainUnits $P_{min}(t)$
        if Units = {}
            Units = ObtainUnits P(t)
        Randomly choose a unit i from Units
        $t_1$ = Randomly choose($t_{left}$, $t_{right}$)
        $x_{it_1}$ = ChangeVariableState($X_j$)
        SetConst = CheckConstraintViolation($X_j \backslash x_{it_1}$)
        Neighb = SelectNeighbMovement (SetConst )
        Y = Generate random solution(Neighb, $X_j \backslash x_{it_1}$)
        PreDisptach(Y)
        if f(Y) < f($X_j$)
            $X_{j+1}$ = Y
        else
            $X_{j+1}$ = $X_j$
        if number solutions = 0
            X* = $X_{j+1}$
        else
            if f($X_{j+1}$) < f(X*)
                X* = $X_{j+1}$
        j++
    Until Stopping criterion is met
    number solutions++
Until number solutions = maximum number solutions
```

**Figure 5.6:** GRASP with Constraint Oriented Neighbourhoods applied to the UCP

*An extension to the Hydrothermal Coordination problem*

Given the important role that hydro units play in the overall performance of a production system, hydrothermal coordination is an important topic of research. An extension of the algorithm described in Figure 5.6 is presented in Appendix B, for the hydrothermal coordination problem, where an integrated optimisation of the two resources (hydro and thermal) is proposed. This is achieved by discretising the production levels of the hydro units, and defining appropriate neighbourhood movements that allow changes to be made both in the thermal and in the hydro units. In spite of that, additional neighbourhoods were designed to appropriately change the state of hydro production units. This work, though preliminary, should be viewed as a starting point for further developments in the area.

## 5.4 Computational experience

The algorithm described above was used to solve the problem instances presented in [Kazarlis et al. 1996], described in Appendix A. In the following sections we report and discuss the results obtained with these tests. We also compare these results with those reported in [Kazarlis et al. 1996], [Valenzuela and Smith 2002] and with the ones obtained with the general GRASP procedure reported in Chapter 4. As pointed out in section 4.3, the CPU times of all the computational experiments will be referred to a 800MHz Pentium III PC, according to their SPECfp95 (see Table 4.1), so that the values can be compared.

### 5.4.1 Computational results

Table 5.2 replicates the results reported in Chapter 4, first presented in [Kazarlis et al. 1996] and in [Valenzuela and Smith 2002], and also the best results obtained by us, with a GRASP algorithm, for a General Purpose Neighbourhood, already discussed in the previous chapter. Again, LR stands for Lagrangian Relaxation, GA for Genetic Algorithm and LR–MA for Memetic Algorithms seeded with an initial solution obtained through Lagrangian Relaxation.

Tables 5.3 and 5.4 report the worst and best results obtained with GRASP, for several levels of refinement of the *Constraint Oriented Neighbourhoods* approach, introduced in this chapter. The *Average time* column refers to the average CPU time necessary for obtaining a solution equal to, or better than that of the LR approach.

Table 5.3 reports the results obtained, if only the GPM and the *Load Movement* are

| Problem size | LR | GA | LR–MA | GRASP |
|---|---|---|---|---|
| 10 | 565 825 | 565 825 | 565 827 | 565 825 |
| 20 | 1 130 660 | 1 126 243 | 1 127 254 | 1 126 805 |
| 40 | 2 258 503 | 2 251 911 | 2 249 589 | 2 255 416 |
| 60 | 3 394 066 | 3 376 625 | 3 370 595 | 3 383 184 |
| 80 | 4 526 022 | 4 504 933 | 4 494 214 | 4 524 207 |
| 100 | 5 657 277 | 5 627 437 | 5 616 314 | 5 668 870 |

**Table 5.2:** Previous computational results (production costs)

applied (LM). It also presents the results obtained when the three movements introduced in section 5.3.2 were implemented (CON1) and applied either in the transition point that immediately precedes $t$ ($t_{left}$) or in the transition point immediately after $t$ ($t_{right}$).

Table 5.4 reports the same results when changes are made sequentially before ($t_{left}$) and after ($t_{right}$) the selected period $t$ (CON2). Finally, CON3 differs from CON2 in the sense that, for CON3, one first makes the changes to the left (right) of $t$ and stores the solution ($X_1$). Then, the movements are applied to the right (left) and only if the resulting solution is better than the previous one, it is stored as the current solution. Otherwise, $X_1$ is considered.

| | LM | | | CON1 | | |
|---|---|---|---|---|---|---|
| Problem size | Cost of worst solution | Cost of best solution | Average time (sec) | Cost of worst solution | Cost of best solution | Average time (sec) |
| 10 | 567 570 | 566 120 | - | 566 688 | 565 825 | 101.36 |
| 20 | 1 129 930 | 1 128 220 | 84.80 | 1 127 330 | 1 126 320 | 1.09 |
| 40 | 2 255 970 | 2 250 920 | 26.37 | 2 253 140 | 2 248 710 | 38.78 |
| 60 | 3 381 020 | 3 375 910 | 49.33 | 3 377 370 | 3 371 070 | 2.90 |
| 80 | 4 509 270 | 4 504 440 | 81.31 | 4 501 910 | 4 494 880 | 9.99 |
| 100 | 5 640 040 | 5 627 300 | 152.50 | 5 624 700 | 5 615 640 | 15.01 |

**Table 5.3:** Computational results for GRASP with Load Movement and CON1

| | CON2 | | | CON3 | | |
|---|---|---|---|---|---|---|
| Problem size | Cost of worst solution | Cost of best solution | Average time (sec) | Cost of worst solution | Cost of best solution | Average time (sec) |
| 10 | 566 688 | 565 825 | 67.39 | 566 688 | 565 825 | 17 |
| 20 | 1 127 330 | 1 126 130 | 16.48 | 1 127 360 | 1 126 070 | 15.27 |
| 40 | 2 253 190 | 2 248 770 | 1.08 | 2 251 740 | 2 248 710 | 1.86 |
| 60 | 3 375 890 | 3 370 960 | 1.52 | 3 375 600 | 3 370 530 | 1.43 |
| 80 | 4 499 990 | 4 494 530 | 4.31 | 4 501 360 | 4 494 380 | 4.59 |
| 100 | 5 624 010 | 5 616 450 | 6.77 | 5 624 530 | 5 615 410 | 5.66 |

**Table 5.4:** Computational results for GRASP with CON2 and CON 3

### 5.4.2 Comparison of results

The computational results show that, even for the more elementary implementation of the CON search strategy (Table 5.3 – LM), the solutions are considerably better than those obtained with the general purpose neighbourhood, considered in Chapter 4. These results are even better, for all instances, when all the neighbourhoods described in section 5.3.2 are implemented and correctly applied, as reported in Table 5.3 – CON1.

The second *Constrained Oriented Neighbourhoods* implementation (see Table 5.4 – CON2) leads to results that, for the 40 and the 100 unit instances, are slightly worse than those presented in Table 5.3 – CON1. As mentioned, in this implementation the changes are made sequentially for two periods of time ($t_{left}$ and $t_{right}$), and the final solution ($X_2$) is the one selected as the new current solution, even if $X_1$ was better. By doing so, we may miss some interesting solutions during the search process.

Finally, for the last implementation (Table 5.4 – CON3), similar to the previous one but selecting the best solution among $X_1$ and $X_2$, there was in general a significative improvement of solution quality, both for the worst and for the best solutions obtained.

When compared with other methodologies proposed in the literature, CON does also present a better performance in the majority of the cases. As shown, the best results obtained by applying the *Constraint Oriented Neighbourhoods* search strategy to this set of instances are considerably better than those obtained with the GA approach in [Kazarlis et al. 1996]. Furthermore, for the larger problems (from 40 to 100 units), the worst results obtained, independently of the parameter set, are also better than those obtained with the GA. Besides, when compared with the results presented in [Valenzuela and Smith 2002], our approaches

| | Relative difference(%) | | | |
|---|---|---|---|---|
| Problem size | CON3 vs LR | CON3 vs GA | CON3 vs LR–MA | CON3 vs GRASP |
| 10 | 0 | 0 | $\approxeq 0$ | 0 |
| 20 | -0.408 | -0.015 | -0.105 | -0.065 |
| 40 | -0.435 | -0.142 | -0.039 | -0.298 |
| 60 | -0.698 | -0.181 | -0.002 | -0.375 |
| 80 | -0.704 | -0.235 | $\approxeq 0$ | -0.686 |
| 100 | -0.746 | -0.214 | -0.016 | -0.952 |

**Table 5.5:** CON3 vs LR, CON3 vs GA, CON3 vs LR–MA and CON3 vs GRASP: relative difference

achieve better results for all instances, except for the 80 unit one, where [Valenzuela and Smith 2002] achieve a slightly better result. Finally, a point of major importance is that, except for the 10 unit problem, the worst results obtained are also significantly better than those reached with LR.

Not surprisingly, the approach performs slightly worse for the smaller problems. In fact, when the solution space is small, search procedures closer to "random walks" have a higher probability of reaching good results (though they may take a lot of computational time). But smaller problems can be correctly tackled with many other techniques and the big challenge is on solving large problems, closer to real situations, for which effective and efficient approaches are not available.

In terms of neighbourhood movements, for the same metaheuristic, a comparison between the results in Tables 5.2 - GRASP, 5.3 and 5.4 shows how critical a correct movement definition is for a good performance of the method. As shown, the results obtained by this new approach, based on *Constraint Oriented Neighbourhoods*, have beaten the general purpose neighbourhood, for all instances, both in terms of efficiency and solution quality.

The relative difference of the results (*(x - Best_Result)/Best_Result*) is presented in Table 5.5 for LR, GA, LR–MA, GRASP and CON3 (a minus sign indicates that CON3 performed better).

In Table 5.6 we report the CPU times of each approach, as if they had all been performed in a a 800MHz Pentium III PC, with a SPECfp95 of 28.9. As shown, the approaches based on CON (LM, CON1, CON2 and CON3) are extremely fast, achieving reductions in CPU time of, at least, 100 times for the 100 unit problem, when CON3 is considered. This may be very interesting in practice, mainly for those problems with a high level of data uncertainty

– the DM will be able to use the available time to perform several runs of the algorithm, for variations in an instance's data, and thus will have the opportunity of analysing various close alternatives before making the final decision.

| | CPU time (sec) | | | | | | |
|---|---|---|---|---|---|---|---|
| Problem size | GA | LR–MA | GRASP | LM | CON 1 | CON2 | CON3 |
| 10 | 15.45 | 42.73 | 8.65 | - | 101.36 | 67.39 | 17 |
| 20 | 51.23 | 145.98 | 290.44 | 84.80 | 1.09 | 16.48 | 15.27 |
| 40 | 188.51 | 110.38 | 768.57 | 26.37 | 38.78 | 1.08 | 1.86 |
| 60 | 408.19 | 292.98 | 1341.82 | 49.33 | 2.90 | 1.52 | 1.43 |
| 80 | 701.47 | 337.74 | 1682.62 | 81.31 | 9.99 | 4.31 | 4.59 |
| 100 | 1099.6 | 680.57 | 2233.99 | 152.50 | 15.01 | 6.77 | 5.66 |

**Table 5.6:** CPU time – comparable values

A graphical representation of the same values, provided in Figure 5.7 for GA, LR–MA, GRASP and CON3, highlights the tremendous decrease in CPU times when CON3 is applied (CPU times are represented in a logarithmic scale).

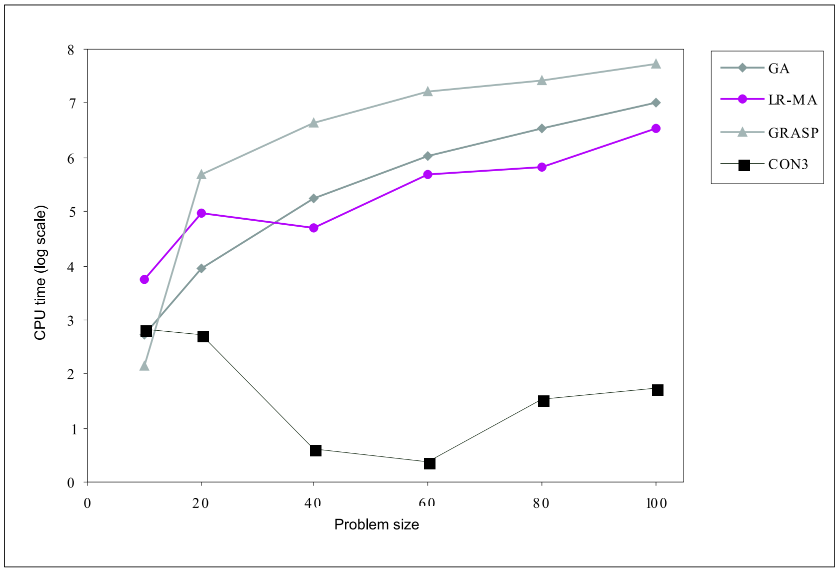


**Figure 5.7:** CPU times (log scale)

## 5.5 Concluding remarks

In this chapter we have introduced the concept of *Constraint Oriented Neighbourhoods* for Local Search based metaheuristics, defining its scope of application and strong points.

The main goal of this search strategy is to obtain robust metaheuristics, less dependent on parameter tuning and therefore more appealing to be used as alternative optimisation tools for supporting decision making. This is achieved by defining several neighbourhood movements, to be applied during the search process in a way that depends on the constraints that are violated. A correct definition of the neighbourhood movements produces neighbour solutions that are similar to the current solution, and that do not involve complex feasibility recovering algorithms. This will lead to a smoother search process and consequently to a good exploitation of the search space, that is not as dependent on parameter tuning as usual. As a result, more robust metaheuristics are obtained. For diversification purposes, we do also consider General Purpose Neighbourhood, to be applied when all hard recovering constraints are violated.

To illustrate this idea, a GRASP algorithm with *Constraint Oriented Neighbourhoods* has been designed for the Unit Commitment Problem. Computational experiments on a set of problem instances from the literature, consistently led to better results than others previously obtained, while the required CPU time has been drastically decreased.

Shorter CPU times may be very important in practice, in contexts of uncertainty, allowing the decision-maker to perform several simulations for different data values. For the Unit Commitment problem, we devise many situations where this may be extremely important, e.g. for the base UC, when load forecasts change; or, if hydro production is considered, to solve the problem for different inflow values. Finally, under market environments, having more time allows one to simulate different market conditions and even to consider recent decisions made by other market players.

It should also be noted that the computational results achieved were never worse than those obtained by more traditional approaches, like Lagrangian Relaxation, independently of the metaheuristic's parameters. This will hopefully encourage end users to consider metaheuristics as an additional interesting tool for supporting decision making processes, and ultimately promote the use of metaheuristics in commercial software packages.

# Chapter 6

# Multiobjective Constraint Oriented Neighbourhoods

In most optimisation problems, Decision Makers are confronted with multiobjective decision problems (e.g. the definition of routes when the transportation of hazardous materials is involved is not only concerned with the distances, but also with other aspects such as the minimisation of risk; in the problem of backbone design of a computer network, some of the key objectives of the designer are to minimise total costs, while maximising reliability; etc). However, these problems are not usually solved in a true multicriteria approach, and quite often one selects one of the objectives, considering it as the “most” important among all, or aggregates all the objectives into a single function.

The reluctance on solving multiobjective combinatorial problems as so has to do with their computational intractability and, until recent years, with the lack of simple and effective optimisation tools to tackle several objectives simultaneously. Fortunately, the scenario may change with emerging techniques inspired in traditional metaheuristics. Given the relative success of these approaches in solving single objective combinatorial problems, since the 90’s a considerable effort has been made at developing multiobjective metaheuristics (MOMH) that have been extended and are now able to deal with the multiobjective paradigm.

The very promising results that were obtained when solving the UCP with *Constraint Oriented Neighbourhoods*, were a major motivation to develop the new MOMH, based on the same concept, that is introduced in this chapter. As described in Chapter 5, CON considers the existence of several neighbourhood structures that are applied during the search process, according to the type of problem constraints that are violated, if part of the current solution is changed. The same reasoning is further refined in multiobjective implementations by developing neighbourhoods that are more prone at enhancing one objective than others. So, in each iteration, the neighbourhood operation to be applied to the current solution is selected according to the constraints that are violated and to the objective that one would like to improve the most.

This chapter is structured as follows. It starts with introductory sections to Multiobjective Combinatorial Optimisation (MOCO) concepts and to Multiobjective Metaheuristics (MOMH). A general description of a new metaheuristic approach, based on the CON ideas is then proposed and the method is applied to the Unit Commitment problem, considering two objectives – operating costs and emissions – both to be minimised. The results obtained by applying the approach to a set of problem instances are compared with Multiobjective Simulated Annealing (MOSA) [Ulungu et al. 1999] and with Pareto Simulated Annealing (PSA) [Czyzac and Jaszkiewicz 1998]. The comparison shows that the method is effective at solving the problem, and significantly more robust to variations in the metaheuristic parameters. Some concluding remarks close the chapter.

## 6.1 Multiobjective Combinatorial Optimisation

### 6.1.1 Basic concepts

A Multiobjective Combinatorial Optimisation (MOCO) problem can be formally stated as follows:

$$\min \quad f(X) = \{f_1(X), ..., f_m(X)\} \tag{6.1}$$

subject to:

$$X \in D$$

where $m$ is the number of objective functions, $f_k(X)$ are the objectives (all to be minimised), $X$ represents the vector of decision variables (with some binary variables reflecting the discrete or combinatorial nature of the problem) and $D$ is the space of feasible solutions. Since usually there is no solution that minimises all the objectives simultaneously, the aim is to find the set of *efficient* solutions.

**Definition 13** The image of a solution $X$ in the objective space is a point $z^X = f(X)$.

**Definition 14** Point $z^X$ *dominates* $z^Y$ *($z^X \succ z^Y$)* if $f_k(X) \leq f_k(Y), \forall_k$ and $f(X) \neq f(Y)$.

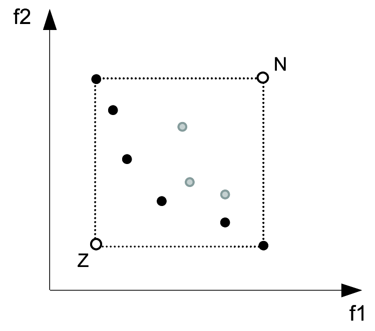


**Figure 6.1:** Solutions for a two-objective minimisation problem

**Definition 15** One solution $X$ *dominates* $Y$*,* if the image of $X$ dominates the image of $Y$*.*

**Definition 16** One solution $X$ is *efficient (*or *nondominated* or *Pareto optimal)* if there is no other feasible solution that dominates $X$*.* The image of an efficient solution is a *nondominated point.*

**Definition 17** The set of all efficient solutions of a problem is the *efficient (*or *Pareto) set.* The image of the efficient set in the objective space is the *Pareto front.*

**Definition 18** The *Zeleny point* is the point obtained by minimising each objective separately and in some sense represents the ideal, not reachable, solution.

**Definition 19** The *Nadir point* is the point defined by the worst solution that one can get for each objective, studied separately.

Figure 6.1 depicts the image of some solutions in a two objective space, f1 and f2, with both objectives to be minimised. Nondominated points are represented by black dots, dominated points are represented by grey dots and the Nadir and Zeleny points are N and Z, respectively.

When working in discrete spaces, efficient solutions may be further divided into supported and nonsupported solutions. [Steuer 1986] defines these solutions as follows:

**Definition 20** Let Z$^{\leq}$ be the convex hull of $[ND \oplus \{z \in R^k | z \leq 0\}]$, where ND is the set of nondominated solutions and $\oplus$ is the set addition operator[1]. Then, if $z^X \in$ ND is on the

[1]Let X and Y be sets in $R^k$. The set addition of X and Y is given by X$\oplus$Y $= \{z \in R^k | z = x + y, x \in X, y \in Y\}$, i.e. every point in X is added to every point in Y.

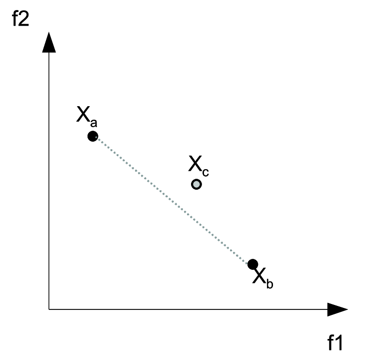


**Figure 6.2:** A nonsupported point

boundary of $Z^{\leq}$, $X$ is a *supported* nondominated solution. Otherwise, it is *nonsupported* or *convexly dominated.*

The nondominated points in Figure 6.1 do all correspond to supported solutions. An example of a nonsupported point is $X_c$ in Figure 6.2, where we assume that the nondominated set of solutions is $X_a$, $X_b$, $X_c$.

The aforementioned definitions clearly distinguish between the decision space and the objectives space. But there is naturally a direct relation between those definitions and, as soon as a solution (or a set of solutions) is (are) characterised, so is (are) its (their) corresponding image(s) on the objectives space. Therefore, if one point $z$ is nondominated one may conclude that the solution leading to that point is efficient.

In this chapter we will mostly refer to objective vectors (points). For simplification purposes, one should assume that, unless otherwise stated, whenever referring to those points, we are simultaneously referring to their corresponding solutions in the decision space.

## 6.2 Multiobjective Metaheuristics

In recent years a major effort has been made to develop MOMH to obtain a good approximation of the efficient set of solutions. As for their single objective counterparts, they may be divided in two broad sets: one that considers metaheuristics that explicitly use neighbourhood exploration/local search, and another for Evolutionary Algorithms.

The first MOMH based on Local Search was probably the algorithm designed by [Serafini 1992], inspired by Simulated Annealing. From then on several new approaches have been proposed and described in the literature, and one can now find other MOMH based on

Simulated Annealing (e.g. Pareto Simulated Annealing (PSA) [Czyzac and Jaszkiewicz 1998] and Multiobjective Simulated Annealing (MOSA) [Ulungu et al. 1999]), Tabu Search (e.g. Multiobjective Tabu Search (MOTS*) [Hansen 1997] and MOTS [Gandibleux et al. 1997]).

MOMH falling into the second type are e.g. Niched Pareto Genetic Algorithms (NPGA) by [Horn et al. 1994], Multiobjective Genetic Algorithms (MOGA) by [Fonseca and Fleming 1993] or, more recently, the Strength Pareto Evolutionary Algorithm (SPEA) by [Zitzler and Thiele 1999].

Hybrids that embed both the paradigms of Evolutionary Algorithms and of Local Search are, for example, Memetic Pareto Archived Evolutionary Strategy (M-PAES) [Knowles and Corne 2000b] and Genetic Local Search (GLS) [Jaszkiewicz 2002]).

This chapter will focus on Local Search population based MOMH. Even so, some references will be made to Multiobjective Evolutionary Algorithms.

### 6.2.1 General structure and concerns

Although following different paradigms MOMH do, generally, consider several solutions (a population) to be optimised in parallel, and use information from each solution during the process, to improve the search. Still, there are some cases where a single solution is optimised in each run of the algorithm (e.g. the multiobjective Simulated Annealing presented in [Serafini 1992]). Even so, non-population based MOMH may also be described by the same procedure of Figure 6.3, that outlines the basic structure of Local Search based MOMH, as they represent the extreme case of a population of size one.

The algorithm is initialised by letting the approximation set of solutions (PE)[2] to be empty, and by setting the parameters of the metaheuristic (e.g. initial and final temperature levels for a Simulated Annealing based MOMH), as well as the initial size of the population, $n$. According to the value of $n$, the initial population of solutions is obtained in *FindInitialPopulation()*, usually through the repetition of a constructive heuristic with a random component. In some implementations the value of $n$ may change along time. Then, and before the search process starts, PE is updated with the potentially efficient solutions in the initial population (*UpdatePE(CurrentPopulation)*).

An improvement loop does then start, involving three main steps. First, it generates

[2]Strictly speaking, one cannot guarantee that the solutions obtained through heuristic algorithms are "truly" efficient solutions. At most, one may say that they are *potentially* efficient solutions. So, in the remaining of this text, we will refer to the set of solutions obtained through metaheuristic techniques as PE, PE standing for *potentially efficient set of solutions.*

**Algorithm MOMH**

```
PE = {}
SetParameters(MH)
CurrentPopulation = FindInitialPopulation()
UpdatePE(CurrentPopulation)
Repeat
    Sol = GenerateSolution(CurrentPopulation)
    UpdatePE(Sol)
    UpdateCurrentPopulation(Sol)
Until stopping criterion is met
```

**Figure 6.3:** Local Search based Multiobjective Metaheuristics – general structure

neighbour solutions (Sol), for each solution in the population. Then, if any potentially efficient solutions are found, PE is updated. Finally, it is decided whether the new solution will replace any of the solutions in the current population (in *UpdateCurrentPopulation(Sol)*). If $n$ is not a fixed value, one may also decide to add the new solution to the population without discarding other solutions, increasing the value of $n$.

Two main questions must be solved, when using MOMH: a) how to guide the search towards the Pareto optimal front (*convergence*), and b) how to maintain a diversified population, to prevent premature convergence and achieve a well distributed approximation of the Pareto front (*diversification*).

*a) How to guide the search towards the Pareto front*

In MOMH based on Evolutionary Algorithms convergence is usually assessed by some measure defined on dominance criteria, e.g. how many individuals does an individual dominate, by how many individuals is an individual dominated, and/or through elite-preservation operators ([Fonseca and Fleming 1993], [Horn et al. 1994], [Zitzler and Thiele 1999]). Individuals achieving a better score in this measure have a higher probability of being selected for reproduction. On the other hand, MOMH based on Local Search do mostly use some multiobjective rules of acceptance [Serafini 1992] applied to weighted aggregated functions. There is not however a consensus on the best way of guiding the search. [Jaszkiewicz 2001], for example, argues that aggregating functions are advantageous as they force the solutions to explore certain areas of the solution space, while dominance assures only convergence but not dispersion over the Pareto front. But for [Knowles 2001] Local Search can be successfully combined with Pareto dominance.

*b) How to maintain a diversified approximation of the Pareto front*

To prevent the search from converging to small regions of the solution space, multiobjective Evolutionary Algorithms do generally evaluate the density of solutions in the area around a given solution, solutions located in crowded areas being less likely to be selected for updating the current population. On the other hand, metaheuristics based on Local Search usually have, for each solution of the current population, weights associated to each objective, that are controlled in such a way that solutions tend to move apart. In some methods, e.g. MOSA, the search is performed in several algorithm runs, each run using a different set of weights, while in others, e.g. [Czyzac and Jaszkiewicz 1998; Hansen 1997], the weights vary during the search process. In Figure 6.3 we assume that these decisions are handled by *GenerateSolution(CurrentPopulation)* and by *UpdateCurrentPopulation(Sol)*.

### 6.2.2 Brief survey of MOMH

This section provides an overview of some MOMH described in the literature. Although metaheuristics based on Evolutionary Algorithms are out of the scope of this work, due to the enormous attention that they have been receiving, they are also referred here. For a detailed review of the principles of MOMH and of recent developments in this field, the reader is addressed to [Ehrgott and Gandibleux 2000], [Coello 2000], [Deb 2001], [Coello et al. 2002] or [Jones et al. 2002].

#### MOMH based on Evolutionary Algorithms

Since the seminal work by [Schaffer 1985], no significant research was performed on MOMH based on Evolutionary Algorithms, until Goldberg in [Goldberg 1989] proposed a new non-dominated sorting procedure, suggesting the use of the concept of dominance to assign more copies to nondominated individuals in a population, and of a niching strategy among points of a nondominated set, for diversification purposes. These suggestions had a considerable impact, giving rise to several new MOMH based on Evolutionary Algorithms. A short description of some of these approaches is provided below.

*Vector Evaluated Genetic Algorithm* (VEGA) [Schaffer 1985]

In VEGA the population is divided into $m$ sub-populations ($m$ being the number of objectives), each sub-population $i$ being filled with individuals that are chosen from the current population, according to objective $i$. Then the $m$ sub-populations are merged and the genetic operators are applied to the whole set, to produce the new population.

*Multiobjective Genetic Algorithm* (MOGA) [Fonseca and Fleming 1993]
In MOGA each individual is assigned a rank, corresponding to the number of individuals in the current population that dominate it (nondominated points are assigned the value 1). The population is sorted according to that value, and a degree of fitness for each individual is computed. For individuals with the same rank, an average of their fitness values is computed to guarantee that all individuals are sampled at the same rate, while keeping the global population fitness constant. To prevent the search from converging to small regions of the solution space, niching methods (like *fitness sharing*) are used. In *fitness sharing*, individuals that are close get their fitness levels decreased. This implies that isolated individuals, that could initially be considered less fit for reproduction, have now more possibilities of being selected. Selection for reproduction is done according to the fitness level of each individual. An improved version of MOGA, where the Pareto dominance scheme is extended to include goal and priority information for multiobjective optimisation is presented in [Fonseca and Fleming 1998].

*Niche Pareto Genetic Algorithm* (NPGA) [Horn et al. 1994]
NPGA is based on tournament selection. Two individuals (potential candidates for reproduction) are randomly chosen among the population and compared with a subset of individuals, also randomly picked from the population. If one of the candidates is dominated by one solution in the comparison set and the other is not, the latter is selected for reproduction. Otherwise, if both of them are either dominated or nondominated, sharing is used to choose the winner. The degradation of the individual's fitness (sharing) is obtained by dividing its current fitness degree by a value that is an estimator of how crowded is the neighbourhood of that individual.

*Nondominated Sorting Genetic Algorithm* (NSGA) [Srinivas and Deb 1995]
NSGA does also classify individuals in a ranking scheme similar to the one used in MOGA and, again, fitness sharing within the same class is implemented to help maintaining a well-distributed population over the Pareto front. Once the whole population is classified, a selection scheme is used to ensure that the individuals with lower ranks still have some chances of being selected for reproduction. An improved version of this technique, NSGA-II, is described in [Deb et al. 2002]. It reduces the weaknesses of its previous versions, namely, the high computational complexity of nondominated sorting, the lack of elitism and the need of specifying a sharing parameter, by developing a fast nondominated sorting approach, a elitism-preserving approach and a parameterless niching operator, respectively.

*Strength Pareto Evolutionary Algorithm* (SPEA) [Zitzler and Thiele 1999]
SPEA is an elitist multiobjective Evolutionary Algorithm. It maintains, in all generations, an external population that stores all potentially efficient solutions found so far. In each generation step, a population that incorporates the external and the current population is built. The nondominated individuals in the external population are used to determine the fitness of individuals in the current population and do also participate in the selection process for reproduction. All nondominated points in the combined population are assigned a fitness value based on the number of solutions they dominate. To maintain diversity, a higher fitness value is assigned to a nondominated point having more dominated points in the combined population. In addition, SPEA uses a niche method to reduce the number of PE solutions stored, without destroying the characteristics of the Pareto front. An improved version of this technique, SPEA2, is described in [Zitzler et al. 2001]. The new version includes a fine-grained fitness assignment strategy, where the fitness of an individual is determined by the strength of its dominators in both archive and population (as opposed to SPEA where only archive members are considered); a nearest neighbour density estimation technique, to distinguish between individuals having the same fitness, allowing a more precise guidance of the search space; and an enhanced archive truncation method that guarantees the preservation of boundary solutions.

### MOMH based on Local Search

The other broad class of metaheuristics for multiobjective optimisation uses Local Search or neighbourhood exploration to drive the search. Some algorithms of this class are briefly described below.

*Simulated Annealing for Multiobjective Optimisation* [Serafini 1992]
Following the principles of single objective Simulated Annealing, where neighbour solutions are accepted according to a given probability function, [Serafini 1992] modified and adapted this concept, presenting several alternative transition probabilities (rules) to apply SA to multiobjective problems. Two main groups of rules are presented: the *strong criterion* rules, by which only dominating points are accepted with probability one, and the *weak criterion* rules, which state that only dominated points are accepted with probability less than one.

*Multiobjective Tabu Search* (MOTS) [Gandibleux et al. 1997]
MOTS is a population-based extension of Tabu Search. In each iteration, the neighbours of the current solution are evaluated according to a weighted scalarising function, and the

best solution in chosen. The reference point taken in the scalarising function is the “ideal” (Zeleny) point in the neighbourhood of the current solution. Diversification is achieved by modifying the weight vector in such a way that, for objectives that are significantly improved, the weights are decreased. Two tabu lists are considered, one playing the traditional role of preventing cycling, and another for the modified weight components.

*Multiobjective Tabu Search* (MOTS*) [Hansen 1997]
In a parallel, but independent work, [Hansen 1997] did also propose a MOMH based on Tabu Search. This metaheuristic, initially referred to as MOTS*, not to be confused with the one proposed by [Gandibleux et al. 1997], was later renamed as TAMOCO [Hansen 2000]. Here we will however adopt the first name, as it is still the most used one. In MOTS* a set of current solutions, each with its own tabu list, is considered. Each solution is assigned a set of weights that “force” the search to go into a certain direction in the objective space. To allow the search to go into different areas of the Pareto front, the weights assigned to each objective, for each current solution, are adjusted in such a way that the solutions are kept away from their nondominated neighbours.

*Pareto Simulated Annealing* (PSA) [Czyzac and Jaszkiewicz 1998]
PSA also considers a set of current solutions that are “optimised” in parallel, the annealing scheme being defined by the transition probabilities introduced in [Serafini 1992]. As in MOTS*, the weights assigned to each objective are dynamically changed in each iteration of the algorithm, for each of the generating solutions, in such a way that during the search process a solution will tend to move away from the other solutions for which it is nondominated, allowing the algorithm to search along the whole Pareto front.

*Multiobjective Simulated Annealing* (MOSA) [Ulungu et al. 1999]
MOSA is an extension of Simulated Annealing where a weighted aggregating function is used to evaluate the fitness of solutions. Again, the transition probabilities presented in [Serafini 1992] are used and multiple runs, for sets of different weights, are made. The weights associated to the objectives considered are kept constant in each run and are chosen in such a way that each run guides the search into a different area of the Pareto front, trying to get a well spread set of approximated nondominated points.

*Simulated Annealing for Multiobjective Optimisation* (SAMO) [Suppapitnarm et al. 2000]
SAMO is another extension of Simulated Annealing in which a different temperature parameter is associated to each objective in the problem. The algorithm considers only one solution

at a time and the annealing process adjusts each temperature independently, according to the performance of that solution in each objective during the search.

*Pareto-Archived Evolutionary Strategy* (PAES) [Knowles and Corne 2000a]
Although its name may be rather misleading, PAES should be included in the set of MOMH based on Local Search. It is initialised with a single solution, candidate solutions being generated in each iteration through neighbhourhood operations. To maintain diversity, an external archive stores potentially efficient solutions. An adaptive grid that divides the objectives space is used to evaluate how crowded is the region in which each solution lies. A candidate solution is discarded if it is dominated by the current solution, or by any other solution in the external archive. It is added to the archive, replacing the current solution, if it dominates it. If none of them dominates the other, the decisions on which solution becomes the current one and on adding or not the candidate solution to the archive are made based on the crowding mechanism. Other variants of this algorithm, with population sizes greater that one, were also proposed in [Knowles 2001].

### Hybrid MOMH

We may also find MOMH where concepts from Local Search and Evolutionary Algorithms cooperate during the search process. Hybrid MOMH were proposed by, e.g. [Knowles and Corne 2000b] and [Jaszkiewicz 2002].

*Memetic Pareto Archived Evolutionary Strategy* (M-PAES) [Knowles and Corne 2000b]
M-PAES is a hybrid MOMH proposed as a memetic variant of PAES. It incorporates a population and a crossover operator and uses the same selection mechanism of PAES. Two archives are used, a global one that stores the potentially efficient solutions and another storing a comparison set in the Local Search phase. The second archive is emptied after each local search and filled again with solutions from the global archive.

*Multiobjective Genetic Local Search* (MOGLS) [Jaszkiewicz 2002]
MOGLS is a metaheuristic that hybridises recombination operators with Local Search or, more generally, with other local improvement heuristics. A weighted aggregating function is generated at random in each iteration, and used for selecting the solutions that will be recombined to form the offspring and to guide the local optimisation of this offspring. An iteration of MOGLS consists on a single recombination of a pair of solutions and on the application of a heuristic that locally improves the value of the current scalarising function. An

approximation of the ideal point, constructed with the best known values of each objective, is used as the reference point.

### 6.2.3 Evaluation of results

Being a critical key issue in multiobjective optimisation, the evaluation of results of MOMH has been discussed by several authors (e.g. [Hansen 1998], [Zitzler et al. 2000], [Knowles 2001] and [Zitzler et al. 2003]), alternative evaluation measures having been proposed. Even so, all measures seem to present some limitations.

A current point of discussion concerns the meaning of "quality", that is definitely not clear when evaluating approximations of the Pareto set: Closeness to the Pareto front? Good dispersion of nondominated points over the Pareto front?, Zitzler et al. suggesting in [Zitzler et al. 2000] that three goals should be pursued when evaluating potentially nondominated sets of solutions:

i. A minimum distance of the resulting PE to the Pareto front.

ii. A good (in most cases uniform) distribution of the solutions found.

iii. A wide range of values, for each objective.

Still, [Knowles and Corne 2002] argue that these features are not enough to adequately compare MOMH, and that less empirical methods of evaluation should be developed.

Within this line, [Hansen and Jaszkiewicz 1998] present some outperformance relations that express the relationship between two sets of PE, and investigate whether the measures they propose comply with these relations, i.e. if an approximation is better than another, according to an outperformance relation, does the comparison method also evaluate it as being better, or at least not worse? Later, [Knowles 2001] devoted some sections of his PhD thesis to discuss and compare several quality measures, based on outperformance relations and sensitivity to scaling, among other properties. The remaining of this section makes a summary of that study, presenting some metrics proposed in the literature, studying their compatibility with the outperformace relations introduced in [Hansen and Jaszkiewicz 1998], and discussing some of their pros and cons. More details may be found in [Knowles 2001] or [Knowles and Corne 2002].

**Outperformance relations**

Three types of outperformance relations are defined in [Hansen and Jaszkiewicz 1998]: weak, strong and complete outperformance. These definitions are presented below, ND(A∪B) representing the set of nondominated points (and corresponding solutions) resulting from the union of A and B.

**Definition 21** Set A *weakly outperforms* set B (A $O_W$ B), if A $\neq$ B and if ND(A∪B) = A, i.e. if for each solution X in B there exists a solution Y in A that is equal to or dominates X, and at least one solution in A is not in B.

**Definition 22** Set A *strongly outperforms* set B (A $O_S$ B), if ND(A∪B) = A and B\ND(A∪B) $\neq \{\}$, i.e. if for each solution X in B there exists a solution Y in A that is equal to or dominates X, and at least one solution in B is dominated by one solution in A.

**Definition 23** Set A *completely outperforms* set B (A $O_C$ B), if ND(A∪B) = A and B∩ND(A∪B) = {}, i.e. if each solution in B is dominated by a solution in A.

The complete outperformance relation is the strongest and the weak outperformance relation is the weakest of the outperformance relations.

As each of the relations defines an incomplete ranking, some sets may remain incomparable. Still, one can use those relations to assess the usefulness of quantitative metrics. As referred in [Knowles 2001], "any metric which is not compatible with these relations cannot be relied upon to provide evaluations that are compatible with the notion of Pareto dominance". Compatibility and weak compatibility with an outperformance relation are defined in [Hansen and Jaszkiewicz 1998] as follows:

**Definition 24** A comparison metric M is *weakly compatible* with an outperformance relation O, if for each pair of nondominated sets A and B, such that A O B, M will evaluate approximation A as being not worse than B.

**Definition 25** A comparison metric M is *compatible* with an outperformance relation O, if for each pair of nondominated sets A and B, such that A O B, M will evaluate approximation A as being better than B.

**Metrics**

In the literature, several metrics have been proposed to evaluate the quality of the approximation set obtained through MOMH. Some of them are presented in this section. For a complete understanding of the following text, some additional concepts, introduced in [Knowles and Corne 2002], are presented.

**Definition 26** A *direct comparative metric* compares two sets A and B, using (directly) a scalar measure M(A, B) to describe how much better A is than B. If M(A, B) = c - M(B, A) for some constant c, for all pairs of nondominated sets (A, B), then M is *symmetric.*

**Definition 27** A *reference metric* uses a reference set to perform a comparison between two sets A and B. It scores both sets against the reference set and then compares the results.

**Definition 28** An *independent metric* measures some property of each set A and B, that is not dependent on the other set, nor on any reference set of points.

*The $C_1$ and $\mathcal{C}$ metrics*

A straightforward way of evaluating the closeness of a set of solutions A to a reference set R might be to calculate the number (or percentage) of solutions in A, that do also belong to R (expression (6.2)). However, if the aim is to compare the quality of two sets A and B through R, this metric may present some weaknesses. In fact, it may happen that there are no solutions neither in A nor in B that do also belong to R. In such cases, $C_1$(A) and $C_1$(B) would be null and it would not be possible to draw any conclusions about the relative quality of each set. So, comparing two approximations A and B with R, when none of them has solutions that are common to R, would not be possible.

$$C_1(A) = \frac{|A \bigcap R|}{|R|} \tag{6.2}$$

In such cases it may be more reasonable to use the quality measure proposed in [Zitzler and Thiele 1998] which, for a pair of approximation sets (A, B) calculates the fraction of solutions in B that are weakly dominated by one or more solutions in A.

Given two sets of solutions, A and B:

$$\mathcal{C}(A, B) = \frac{|\{Y \in B\} \mid \exists X \in A : X \preceq Y|}{|B|} \tag{6.3}$$

where $X \preceq Y$, means that $X$ dominates or is equal (in the objectives space) to $Y$. If $\mathcal{C}$(A, B) = 1 all points in B are dominated by, or equal to (covered by), some points in A. If $\mathcal{C}$(A, B) = 0 no point in B is covered by any point in A. This metric presents the disadvantage of not being able to determine the degree of outperformance, if one set completely outperforms the other.

This seems to be a common weakness of cardinal measures, leading [Hansen and Jaszkiewicz 1998] to state that the use of these measures seems to be reasonable only when the method being used has a high probability of finding a significant percentage of potentially nondominated solutions, otherwise two sets may be frequently incomparable. For the $\mathcal{C}$ metric, for example, when neither $\mathcal{C}$(A, B) = 1, nor $\mathcal{C}$(B, A) = 1, the two sets are incomparable, according to the weak outperformance relation and in such cases it is not advisable to draw any further conclusions on the relative quality of the two sets.

Knowles in [Knowles 2001] also shows that if three sets are compared using the $\mathcal{C}$ metric, they may not be ordered, i.e. the metric is cycle-inducing. Moreover, neither $C_1$ nor $\mathcal{C}$ provide any information concerning the shape of the approximation, i.e. on how well spread is the approximation over set R. Metric $\mathcal{C}$ does however present some advantages, e.g. it is scale and reference point independent and it requires no knowledge of the Pareto front.

In terms of compatibility, $C_1$ is weakly compatible with relations $O_C$, $O_S$ and $O_W$ and $\mathcal{C}$ is compatible with $O_S$ and $O_C$.

*The $D_{1R}$ metric*

Trying to evaluate how well spread is an approximation set A over a reference set R, [Czyzac and Jaszkiewicz 1998] proposed metric $D_{1R}$, expression (6.4), that measures the mean distance over the points in R to the nearest point in A.

$$D_{1R}(A, \Lambda) = \frac{1}{|R|} \sum_{r \in R} \min_{z \in A} \{d(z, r)\} \tag{6.4}$$

where,

$$d(z, r) = \max_{k=1 \dots m} |\lambda_k \times (f_k(z) - f_k(r))| \qquad z \in A \quad r \in R \tag{6.5}$$

$$\lambda_k = \frac{1}{\Delta_k} \tag{6.6}$$

$\Delta_k$ representing the range of objective $k$ in $R$ and $\Lambda = [\lambda_1, \dots, \lambda_m]$.

$D_{1R}$ is a non-cardinal reference metric that induces a complete ordering on the set of approximations. In terms of compatibility, it is weakly compatible with $O_W$ but it is not compatible even with $O_C$. Its major disadvantage is the strong dependence of the score on the distribution of points in R and on the choice of Λ, because all reference points have equal weights [Knowles 2001].

*The $R_1$, $R_2$ and $R_3$ metrics*

Other non-cardinal metrics, denoted by $\mathrm{R}_1$, $\mathrm{R}_2$ and $\mathrm{R}_3$, are proposed in [Hansen and Jaszkiewicz 1998]. $\mathrm{R}_1$ measures the probability of an approximation A being perceived as better than an approximation B, over an entire set of utility functions.

$$R_1(A, B, U, p) = \int_{u \in U} C(A, B, u)p(u)du \tag{6.7}$$

where

$$p(u) = \begin{cases} 1 & \text{if u*(A)} < \text{u*(B)} \\ \frac{1}{2} & \text{if u*(A)} = \text{u*(B)} \\ 0 & \text{if u*(A)} > \text{u*(B)} \end{cases} \tag{6.8}$$

U is some set of utility functions, $p(u)$ is an intensity function expressing the probability density of the utility u ∈ U and u*(A) = $\min_{z \in A}\{u(z)\}$.

**Definition 29** An *utility function* $u : \mathcal{R}^m \rightarrow \mathcal{R}$ is a model of the Decision Maker preferences, that maps each point in the objective space into a value of utility.

**Definition 30** Assuming that all objectives are to be minimised, *an utility function u is strictly compatible with the dominance relation* iff $z^1 < z^2 \Rightarrow u(z^1) < u(z^2), \forall z^1, z^2$. The set of all utility functions that are strictly compatible with the dominance relations is $\mathrm{U}_{sc}$.

$\mathrm{R}_1$ is a non-cardinal, direct comparative metric, that does not induce a total ordering. If used as a reference metric, it is referred to as $\mathrm{R}_{1\mathrm{R}}$. Both $\mathrm{R}_1$ and $\mathrm{R}_{1\mathrm{R}}$ are scaling independent and $\mathrm{R}_{1\mathrm{R}}$ can differentiate between different levels of outperformance, provided that an appropriate reference set is chosen. The weak aspects of these metrics are their dependence on the definition of a set of utility functions, and $\mathrm{R}_1$ being cycle-inducing.

In terms of compatibility, let U(A < B) = $\{u \in U | u^*(A) < u^*(B)\}$. If the probability density $p(u)$ is such that the probability of selecting a utility function u ∈ U(A < B) is positive whenever U(A < B) ≠ {} and U ⊆ $\mathrm{U}_{sc}$, $\mathrm{R}_1$ is compatible with $\mathrm{O}_W$. Under the same assumptions, $\mathrm{R}_{1\mathrm{R}}$ is only weakly compatible with $\mathrm{O}_W$ and is not compatible with $\mathrm{O}_C$.

$\mathrm{R}_2$ is a non-cardinal direct comparative metric that makes a comparison based on expected values, rather than on probabilities, calculating the expected difference between an approximation A and an approximation B. If used as a reference metric, it is referred to as $\mathrm{R}_{2\mathrm{R}}$.

$$R_2(A, B, U, p) = E(u^*(A)) - E(u^*(B)) = \int_{u \in U} u^*(A)p(u)du - \int_{u \in U} u^*(B)p(u)du \qquad (6.9)$$

$R_2$ and $\mathrm{R}_{2\mathrm{R}}$ are compatible with $\mathrm{O}_W$, under the same set of conditions outlined for $R_1$. The main disadvantage of $R_2$ is that it is based on the assumption that one is allowed to add values of different utility functions. Consequently, it depends on an appropriate scaling of the utility functions. A very positive aspect of this metric is its compatibility with all the outperformance relations and the fact of being able to differentiate between different levels of complete outperformance.

Finally, $R_3$ is similar to $R_2$ but the ratio of the best utility values is calculated, instead of the differences.

## 6.3 Constraint Oriented Neighbourhoods in MOMH

The design and implementation of a MOMH based on CON (here referred to as mCON) appeared as a natural evolution of the concepts developed in Chapter 5. If we have different neighbourhoods to tackle different problem constraints, it is also natural to use different neighbourhoods to improve the different objectives in a controlled way. This idea is the basis of the MOMH described in this chapter, and is one of the innovative achievements of this thesis.

Following the same principle of other MOMH, mCON works with a population of solutions which are improved simultaneously, aiming at reaching a good approximation of the Pareto set. For this purpose, in each iteration and for each solution in the current population, a specific neighbourhood is applied, hopefully pushing the current solution towards the Pareto front and, simultaneously, moving it to less populated areas of the current set of nondominated points. For doing so, there is a portfolio of neighbourhood movements, with different scores on two attributes: constraint recovering and direction of search. The neighbourhood movement is selected from this portfolio, according to the hard recovering constraints that are violated in a given solution and also to the objective that one would like to improve the most. First, one detects the set of hard recovering constraints that are violated, if one changes the state of one (or more) decision variable(s) in the current solution. This analysis reduces the initial portfolio to a set containing only those movements that were specifically designed to recover the satisfaction of those constraints. When working with a single objective problem, the cardinality of this set would be one and the neighbourhood movement would be automatically selected. However, in multiobjective optimisation the size of the set will generally equal the number of objectives of the problem, as one will have different movements that will differently affect the objectives. The final neighbourhood movement, selected from this reduced set, will be the one that is more prone at improving the objective with the highest priority in the current iteration.

**Example 10** *Selecting neighbourhood movements within mCON*

Suppose that you are solving a 3-objective problem, with 4 types of hard recovering constraints. Figure 6.4–a) presents the alternative neighbourhood movements that one must consider to solve this particular problem with mCON. For simplification purposes, we will only consider those scenarios where all constraints, except one, are verified. If by changing part of the current solution, constraints of type 3 are violated, the initial set of neighbourhood movements is reduced, and will only include those neighbourhoods that were designed to recover the satisfaction of type 3 constraints ($N_{13}$, $N_{23}$ and $N_{33}$ in Figure 6.4–b)). Each of these movements is more prone at improving a given objective, i.e. $N_{i3}$ will tend to produce higher improvements in objective $i$. Thus, if objective 1 is given a higher priority, the neighbourhood to apply will be $N_{13}$ (6.4–c)).

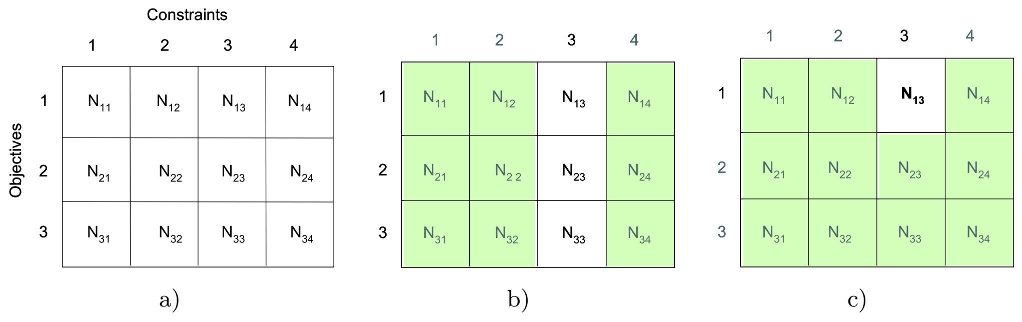


**Figure 6.4:** Selecting neighbourhood movements within mCON

mCON is described in Figure 6.5. The procedure starts by initialising the set of potentially efficient solutions, PE, by setting the parameters of the metaheuristic, and by obtaining the initial population of solutions, P. Then PE is updated with those solutions in P that are efficient (*UpdatePE(P)*) and the optimisation process starts. In each iteration, for each solu-

```
Algorithm mCON

PE = {}
SetParameters(MH)
P = FindInitialPopulation()
UpdatePE(P)
Repeat
    For each solution $X \in P$
        $x_a$ = ChangeVariableState(X)
        SetConst = CheckConstraintViolation(X\$x_a$)
        NeighbConstr = SelectNeighbMovement (SetConst)
        NeighbDir = mCONdir(X, PE)
        Neighb = SetNeighbDir(X, PE, NeighbConstr, NeighbDir)
        Y = GenerateRandomSolution(Neighb, X \$x_a$)
        UpdatePE(Y)
        UpdateCurrentPopulation(Y)
Until Stopping criterion is met
```

**Figure 6.5:** mCON – a multiobjective metaheuristic with CON

tion in the current population, and after changing the state of one or more decision variables of the solution, one detects the sets of *hard recovering constraints* that are no longer met. The result of that evaluation is saved in *NeighbConstr*. Then, a preferencial direction of search, *NeighbDir*, is selected according to the proximity of that solution to PE. Finally, based on the values of *NeighbConstr* and *NeighbDir*, a neighbourhood movement is choosen. It will hopefully prevent drastic changes in the current solution structure, and preferentially improve a specific objective. At the end, PE is updated with the new solution Y and the decision on whether Y will replace X, or not, is made in *UpdateCurrentPopulation(Y)*. In this work it has been considered that a solution should be replaced by another if the neighbour solution is not dominated by the current one. However, other criteria, e.g. the acceptance rules discussed in Serafini [1992], could also be considered as criteria for acceptability.

### 6.3.1 How to define the preferential direction of search

The definition of a search direction is a major issue in most MOMH based on Local Search, as it may strongly influence the effectiveness of the heuristic. In PSA, for example, [Czyzac and Jaszkiewicz 1998] define the preferential direction of search by associating weights to objectives, the weights being increased for the objectives that one wants to improve the most, and decreased otherwise. By increasing the weights, the probability of accepting one solution for which that objective is better, does also increase. Furthermore, the weights are changed in such a way that a solution will tend to move away from the other solutions that, until that moment, have been identified as being nondominated. In MOTS* [Hansen 1997], somehow inspired by PSA, a set of weights that "force" the search to go into a certain direction is also considered.

In mCON there are no weights associated to each objective. Even so, the definition of a preferential direction of search follows some principles that are similar to the ones considered when setting the weights in PSA and MOTS*. In spite of that, we will shortly explain these procedures, prior to the algorithm that sets the search direction in MCON.

## PSA

Figure 6.6 describes the procedure used to set the weights in PSA. In each iteration, for each solution X in the current population (S), the solution that is closer to X in S, nondominated with respect to X (X') is selected. If there is not such a solution, a set of random weights is defined in *SetRandomWeights*($\Lambda^X$). Otherwise, the weights that are currently associated to X are slightly increased for the objectives where X is better than X', and decreased for the others ($\alpha > 1$). At the end $\Lambda^X$ is normalised to guarantee that the sum of all weights equals 1. This process of setting the weights will hopefully lead to a set of solutions uniformly spread along the nondominated front.

```
Algorithm PSAweights

For each X in S
    X' = SelectClosestND(X)
    if X' = {}
        SetRandomWeights($\Lambda^X$)
    else
        For each k
            if $f_k(X) \leq f_k(X')$
                $\lambda^X_k = \alpha \lambda^X_k$
            else
                $\lambda^X_k = \lambda^X_k / \alpha$
    Normalise($\Lambda^X$)
```

**Figure 6.6:** Setting the weights in PSA

The criterion of acceptability of a movement relies on some rule of acceptance, as the ones discussed in [Serafini 1992], e.g. rule W in expression (6.10).

$$P(X,Y,T,\Lambda^X) = \min\{1; \max_{k=1,\ldots,m}\{e^{\lambda_k(f_k(X)-f_k(Y))/T}\}\} \tag{6.10}$$

**MOTS***

Figure 6.7 describes the procedure used to set the weights in MOTS*, trying to assure an optimisation towards the Pareto front. The aim is to set the weights so that each point moves away from the other points, and the whole set of points is well spread over the front. Each element $\lambda^k$ in the weight vector is set according to the proximity of other nondominated points to the current one, for objective $k$. The closer another point is, the more it should influence the weight vector, the influence being given by a decreasing positive value function, $g$. In Figure 6.7, $\Pi = [\pi_1, \pi_2, \ldots, \pi_m]$ represents the range equalisation factors (expression (6.11)) used to equalise the ranges of the objectives (see [Steuer 1986]) and $g(d)$ is a proximity function, that measures the distance between two solutions (expression (6.12)).

```
Algorithm MOTS*weights

For each X in S
    $\lambda^X_k = 0 \ \forall k$
    For each X' in S, X' nondominated by X and $f(X') \neq f(X)$
        $W = g(d(f(X'), f(X), \Pi))$
        For all objectives k where $f_k(X) < f_k(X')$
            $\lambda^X_k = \lambda^X_k + \pi^X_k w$
    if $\lambda^X_k = 0, \forall k$
        SetRandomWeights($\Lambda^X$)
    Normalise($\Lambda^X$)
```

**Figure 6.7:** Setting the weights in MOTS*

$$\pi_k = \frac{1}{Range_k}\left[\sum_{i=1}^{m}\frac{1}{Range_i}\right]^{-1} \tag{6.11}$$

$$g(d) = \frac{1}{d(f(X), f(X'), \Pi)} \tag{6.12}$$

where

$$d(f(X), f(X'), \Pi) = \sum_{k=1,\dots,m} \pi_k |f_k(X) - f_k(X')| \tag{6.13}$$

The reasoning behind the definition of the optimisation direction is explained by [Hansen 1997] with the following example, that we have adapted for objectives' minimisation. Consider the four current solutions (A, B, C and D) in Figure 6.8, and that a neighbourhood movement is going to be applied to solution A. As D is dominated by A, only solutions B and C will influence the definition of the weights. The influence of solution B in the optimisation direction, so that A moves away from it, is reflected by vector $b$. In the same way, C influences A to move away from it in direction $c$. Moreover, as B is closer to A, its influence in the definition of the weights is stronger than that of C. The final optimisation direction is given by the vector resulting from summing $b$ and $c$.

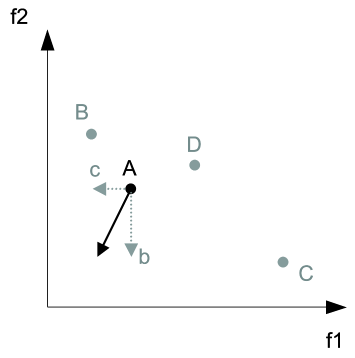


**Figure 6.8:** Setting the weights in MOTS* – an illustrative example

Having generated an optimisation direction, the neighbourhood movement is applied and evaluated through a weighted scalarising function. A set of solutions in the neighbourhood of the current solution is generated, and the solution $Y$ maximising $\Lambda \odot f(Y)$ is selected.

### mCON

In the two previous cases (PSA and MOTS*), the preferential direction of search is represented by a vector of weights associated to the objectives, each weight depending on the location of the solutions in PE. The evaluation of a specific movement is made through an aggregating function of all objectives, for MOTS*, or by a difference of weighted objectives,

```
Algorithm mCONdir (X, PE)

C = {}
For all k
    $Y_k$ = FindClosestPE(X, k)
    C = C $\cup$ $Y_k$
Sol = SelectFromCMaxEuclidian()
NeighbDir = DistMaxObj(Sol, X)
```

**Figure 6.9:** Setting the preferential direction of search

evaluated by an expression such as (6.10), or similar, for PSA.

In mCON a different methodology is followed as, rather than considering a single neighbourhood movement that optimises one objective (that differs according to the values of the weights), different neighbourhood movements are selected according to the current position of solutions. For a given solution $X$, the direction of search is defined as described in Figure 6.9. First, for each objective $f_k$, one selects, from the solutions in PE that are worse than $X$ (for $f_k$), the one that is closer to $X$ ($Y_k = FindClosestPE(X, k)$), the distance between $X$ and $Y_k$ being measured by $|f_k(X) - f_k(Y_k)|$. The maximum size of the resulting candidate set (C) is $m$, $m$ being the number of objectives of the problem. Then, the euclidian distance between $X$ and each solution in C is computed, the solution leading to the largest distance, Sol, being the one that is chosen in *SelectFromCMaxEuclidian()*. Finally, the direction of search is settled by calculating, for all the objectives for which $X$ is worse than the selected nondominated solution, the difference of values between each objective. The neighbourhood that is selected is the one that will most likely improve the objective leading to the largest difference. By doing so, one will tend to move that solution apart from the solutions that are closer to it.

The example in Figure 6.10 (minimisation of both $f_1$ and $f_2$) illustrates the procedure, for a two-objective problem. Consider that the current nondominated set is given by solutions 1, 2, 3, 4, 5 and X (Figure 6.10 a)), and that we are currently defining the search direction for X. Solution 2 is the one having a worse value for $f_2$ that is closer to X, and solution 3 is the one having a worse value for $f_1$ that is closer to X. Therefore, $C = \{2, 3\}$. As solution 3 is further away from X (in terms of euclidian distance $d_3 > d_2$) it is the one that will define the search direction for X, i.e. Sol = 3. By making this choice, one will try to move X away from its closest solution, to achieve a well spread set of nondominated points. As solution X is worse than solution 3 for $f_2$, the neighbourhood to apply should preferentially improve objective 2, for the sake of convergence. But, as the objectives are conflicting, an improvement in $f_2$ will usually result in worsening $f_1$ and, as a result, the optimisation direction ($d$ in Figure 6.10 b)) will tend to create a new solution that will be located closer to solution 3.

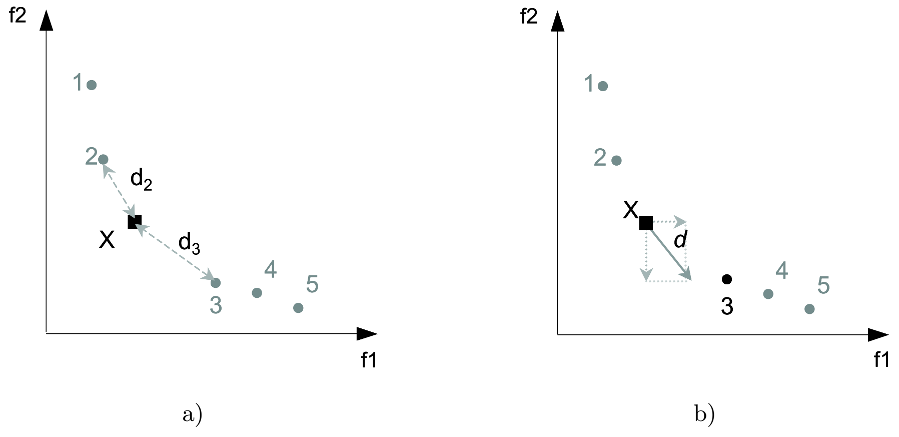


**Figure 6.10:** Defining the search direction – an example

If X falls in an extreme of PE, i.e. if it reaches the minimum value found so far for an objective $k$, the neighbourhood that is selected is the one that tries to further improve objective $k$. By doing so, we aim at exploring areas of the solution space that are near the individual optima of the objective functions.

## 6.4 mCON and the multiobjective UCP

To assess its potential mCON was used to solve a multiobjective version of the Unit Commitment Problem. This problem presents some features that make it suitable for applying the proposed approach, according to the requirements described in section 5.2.3. Moreover, though presenting several conflicting objectives, the problem has usually been solved as a single objective one (except in the works by [Srinivasan and Tettamanzi 1997] and by [Kuloor et al. 1992]), the multiobjective version deserving therefore our interest. Finally, it should be noted that this is a hard and challenging problem, in terms of computational complexity, and occupies a position of vital importance in the everyday management of a electric power production company.

### 6.4.1 Including environmental issues in the base model

Due to the emerging importance of environmental considerations, the environmental effect of thermal power generation became, in recent years, a major point of concern in many countries, leading to the development of models that do also include these issues. The emphasis has however been given to the Environmentally Constrained Economic Dispatch problem (i.e. the problem of defining the production levels of those units that are already committed) and only a few studies exist that include environmental issues in the base UC problem. Some of those studies model emissions as constraints (e.g. [Gjengedal 1996; Manzanedo et al. 2001; Wang et al. 1995]) and others as objectives to be minimised ([Srinivasan and Tettamanzi 1997; Kuloor et al. 1992]). The latter approach will be considered in this work by introducing an additional objective function that measures $SO_2$ and $CO_2$ emissions. This function is similar to that related to operating costs and can be represented as follows:

$$F_2(P_{it}) = min \sum_{t=1}^{T} \sum_{i=1}^{I} a_{1i} P_{it}^2 + b_{1i} P_{it} + c_{1i} \tag{6.14}$$

$a_{1i}$, $b_{1i}$ and $c_{1i}$ representing the emission cost parameters of unit $i$.

In general, high operation cost units tend to have lower emissions and low operation cost units tend to have high emission levels. Therefore, the two objectives are conflicting and an improvement in one of them is reflected in a deterioration of the other. So, as in general there is not a single solution that simultaneously reaches the best value for all objectives, several alternative solutions should be provided, each of them being better than the others in one objective. The aim of this multiobjective approach will therefore be to look for those alternative trade-off solutions, i.e. to produce a representative set of nondominated points. The procedures used to obtain such points/solutions are described below. These solutions will be considered in the final process of decision, when a single solution must be selected for implementation.However, this topic goes beyond the scope of this thesis and the reader ir reported to e.g. [Roy 1996] or [Gal et al. 1998] for further information on the subject.

### 6.4.2 Algorithmic details

The design of a mCON approach for the UCP, when operation and emission costs are both to be minimised, involved the design of several neighbourhood movements, that consider not only each set of hard recovering constraints, but also the objectives to be optimised.

The neighbourhood movements described in section 5.3.2 do naturally remain valid and shall be considered to improve operating costs, as they rely on swapping between cheap and expensive operating cost units. However, they do not completely fulfill the requirements of mCON, where alternative neighbourhood movements must exist to improve different objectives, and additional movements had to be designed. The final set of movements comprises five different neighbourhoods that are described below.

### Load Movements

Two movements were designed and will be considered whenever load requirements are not met, if a unit $i$ is switched off in period $t$. The *Emissions Load Movement* is used if the objective that should be improved the most is the minimisation of emissions. Otherwise, the *Operation Costs Load Movement* will be applied. Again, to restore solution feasibility a procedure that divides the units into expensive and cheap units is implemented in both cases, the concept of "cheap" and "expensive" being different, depending on the objective that is considered. In case the objective that one would like to improve the most is the one measuring operating costs, the partition of the units into sets of cheap and expensive units is computed based on the marginal derivative of the fuel cost function, $\lambda(P_i) = a_i P_i + b_i$. Otherwise, the partition is based on the marginal derivative of the emission cost function, $\lambda(P_i) = a_{1i} P_i + b_{1i}$.

For a given period $t$, both movements will first try to switch on units that belong to the set of *cheaper* units, until load requirements are recovered. If that is not possible, the units from the set of *possibly cheaper* units are selected[3]. As their selection depends on different functions, the units in each set are different, depending on the movement that is selected.

### Reserve Movements

Two movements were also designed and will be considered whenever reserve requirements are not met, even if load requirements are met, when a unit $i$ is switched off in period $t$. A unit $i$ leading to these conditions will be called "reserve unit". A "reserve unit" is usually more expensive than the other units on, and it is not producing at its maximum production level. The reasoning behind these movements is therefore to replace the chosen "reserve unit" by another "reserve unit", of lower cost.

For a given period $t$, any of the *Reserve Movements* starts by defining the set of *possibly cheaper* units that are off in that period. To avoid major changes in the solution, this set is

[3]The definition of *cheaper* and of *possibly cheaper* units is provided in section 5.3.2

simplified by just considering the units that can be switched on, only in period $t$. Finally, the unit to be switched on, so that solution feasibility is recovered, is randomly selected from this reduced set. The *Emissions Reserve Movement* is used if the objective to be improved the most is the minimisation of emissions. Otherwise, the *Operation Costs Reserve Movement* will be applied. For the same reason stated before, the set of *possibly cheaper* units depends on the objective function that is considered.

#### Up-time Movement

In a state transition point, if by switching the concerned unit off, load and reserve requirements are still met but minimum up-time constraints are not, the unit is on only for minimum up time purposes. For this case, a *Up-time Movement* is performed.

Two situations must be considered. Supposing that the unit is on from $t$ to $t_1$: 1) move to a solution with the unit on from $t$ + 1 to $t_1$ + 1; 2) move to a solution with the unit on from $t$ - 1 to $t_1$ - 1. This movement does not affect any other unit and, therefore, it is used independently of the objective selected.

## 6.5 Computational study

The method proposed in this chapter was developed with the MOMHLib++ framework [Jaszkiewicz 2003], a library of C++ classes providing the structure of several MOMH (e.g. PSA, GLS, MOSA, etc) that can be instantiated to any optimisation problem. It does also allow an easy incorporation of other multiple objective metaheuristics and provides the user with tools for evaluating the performance of a MOMH when applied to a given problem.

In the following sections we report and discuss the results obtained by applying mCON to the UCP instances proposed by [Kazarlis et al. 1996]. Some additional parameters, described in Appendix A, were required to model $SO_2$ and $CO_2$ emissions. Although attempts have also been made to compare the approach proposed in this work with those presented in [Srinivasan and Tettamanzi 1997] and in [Kuloor et al. 1992], the data required to generate the problem instances used in those works was no longer available.

The final results were compared with those obtained by applying MOSA and PSA to the same problems. These MOMH have been selected for comparison purposes because they allow different grades of refinement, concerning the use of weights to guide the search: in MOSA the weights are constant in each run of the algorithm, in PSA they are dynamically changed according to the elements currently in the set of potentially efficient solutions, and in mCON there are no weights guiding the search.

### 6.5.1 Parameter setting

The main purpose of the computational experiments presented in this section was to evaluate the quality of the set of potentially efficient solutions obtained with different approaches, when the same CPU time is provided to each approach. For that purpose, experiments have been performed as follows. First the parameters of MOSA were set and the optimisation process was performed for those parameters. PSA and mCON were then run for an amount of time equal to that spent by MOSA.

The following MOSA parameters were considered: starting temperature level = 5, final temperature level = 1, temperature decrease coefficient = 0.9, moves on the same temperature level = 1000 and stopping criterion = 1000 iterations, for populations of 5, 10 and 20 solutions. The same parameters apply to the PSA computational experiments, except the temperature decrease coefficient that was set to 0.95. Keeping that coefficient equal to 0.9 would imply that all the computational tests would be performed within a CPU time smaller than that needed for MOSA.

Concerning mCON, the stopping criterion was set to 1000 iterations. Moreover, as the initial solutions are obtained following a GRASP reasoning, $\alpha$ was set to 0.8, 0.6, 0.4, 0.2, 0.9 and 1, in this order. One should however notice that, due to the constraints on CPU time it may happen that the algorithm is stopped before all values of $\alpha$ are tried.

### 6.5.2 Computational results

In an attempt to better characterise and evaluate the results that were obtained in this work, several alternative formats of evaluation will be provided. A graphical representation of results will be given first, for the 60 unit problem, providing an idea of the form of the Pareto front. Then, information concerning the number of potentially nondominated solutions obtained by each algorithm, and their contribution to the *Reference Set* (the best set of potentially nondominated solutions found so far), will be presented. Finally, the outperformance relations and metrics $\mathcal{C}$ and $R_2$, described in section 6.2.3, will be used to assess and compare the different approaches.

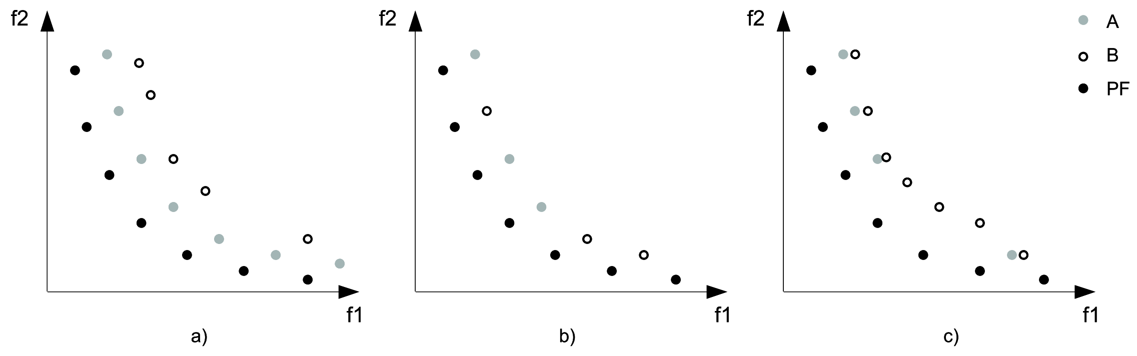


**Figure 6.11:** Graphical comparison of approximation sets

## Graphical comparison of results

A graphical representation of the set of potentially nondominated points may give some interesting information on the relative quality of two approximations of the Pareto front, when only two objectives are measured. It does however present several weaknesses and may not allow one to draw any sound conclusions, unless the sets to be compared present a strong relation of dominance between each other.

To illustrate the weaknesses of the approach, consider Figure 6.11, with three scenarios, and where A and B are two approximations of the Pareto front (PF). Which of the two approximation sets, A or B, should be considered the best?

For Figure 6.11–a), it would be generally agreed that set A is better than set B. However, for the cases presented in Figure 6.11–b) and c) it may, in general, be difficult to reach a single and consensual answer.

In b) the cardinality of A and B is the same, there is no solution in one set dominating solutions in the other set and neither A nor B, if studied separately, give more information on the possible shape of the Pareto front. So, there are no strong arguments to support that one approximation is better than the other, if there is no *a priori* information on the preferences of the DM, concerning each objective.

In c), as some points in A dominate points in B, and the opposite is not true, one may be driven to the conclusion that A is “better” than B. However, a more careful analysis of the results shows that set B explores an area of the space that A does not and which a DM may find of particular interest. Again, the lack of *a priori* information on the preferences of the DM on each objective does not allow a consensual comparison between A and B.

*Graphical assessment of mCON results*

Figures 6.12 to 6.14 present the set of potentially nondominated points obtained with mCON, for the 60 unit problem, for different values of $\alpha$, when the size of the population is kept constant (for easiness of reading, Figures 6.12 and 6.13 have been partially zoomed). For a population of 20 solutions, the maximum CPU time was reached before all $\alpha$ values were simulated. Therefore, Figure 6.14 only depicts the sets of points obtained for $\alpha = 0.6$ and $\alpha = 0.8$. Figures 6.15 and 6.16 present the results obtained, for different population sizes, when $\alpha$ is set to 0.6 and 0.8, respectively. In all figures, RS represents the set of nondominated points obtained by running all versions of mCON, PSA and MOTS, i.e. the union of the potentially efficient solutions obtained in each simulation, excluding the solutions that were found to be dominated.

These results are an example of other difficulties that may arise when a graphical representation of solutions is considered for evaluation purposes, due to the similarity of the results that were obtained when using different algorithm parameters. Consider, for example, Figure 6.13 that presents the set of potentially nondominated points obtained by mCON for the 60 unit problem, when $\alpha$ varies and the population size is kept constant and equal to 10. When comparing the solutions obtained for $\alpha = 0.2$ with those obtained for $\alpha = 1$, one would generally say that the quality of the results in the first set is better than those in the second set, because most of the points in the latter are dominated by those in the former, and the sets are similarly well spread in the objective space. However, when making the same comparison for $\alpha = 0.2$ and $\alpha = 0.4$ it becomes more difficult to draw any conclusions. In some cases the points in the set $\alpha = 0.2$ dominate those in the set $\alpha = 0.4$ but the opposite does also happen. Moreover, for $\alpha = 0.2$ mCON is capable of finding solutions for production costs around $3.43 \times 10^6$, while for $\alpha = 0.4$ it is not.

A similar difficulty arises, if we try to evaluate the relative quality of each set of solutions, for fixed values of $\alpha$, when the population size varies. Those sets are depicted in Figures 6.15 and 6.16, for $\alpha = 0.6$ and 0.8, respectively, for populations of 5, 10 and 20 solutions.

This "uncertainty" on the relative quality of each set of solutions requires the use of less intuitive evaluation measures, some of them having been introduced in section 6.2.3. They will be considered later in this study but we will first draw some conclusions on the robustness of each approach, based on the graphical representation of their results.

Figures 6.17 to 6.19 present graphically the set of potentially efficient solutions obtained for the 60 unit problem by mCON, MOSA and PSA, respectively. When compared to

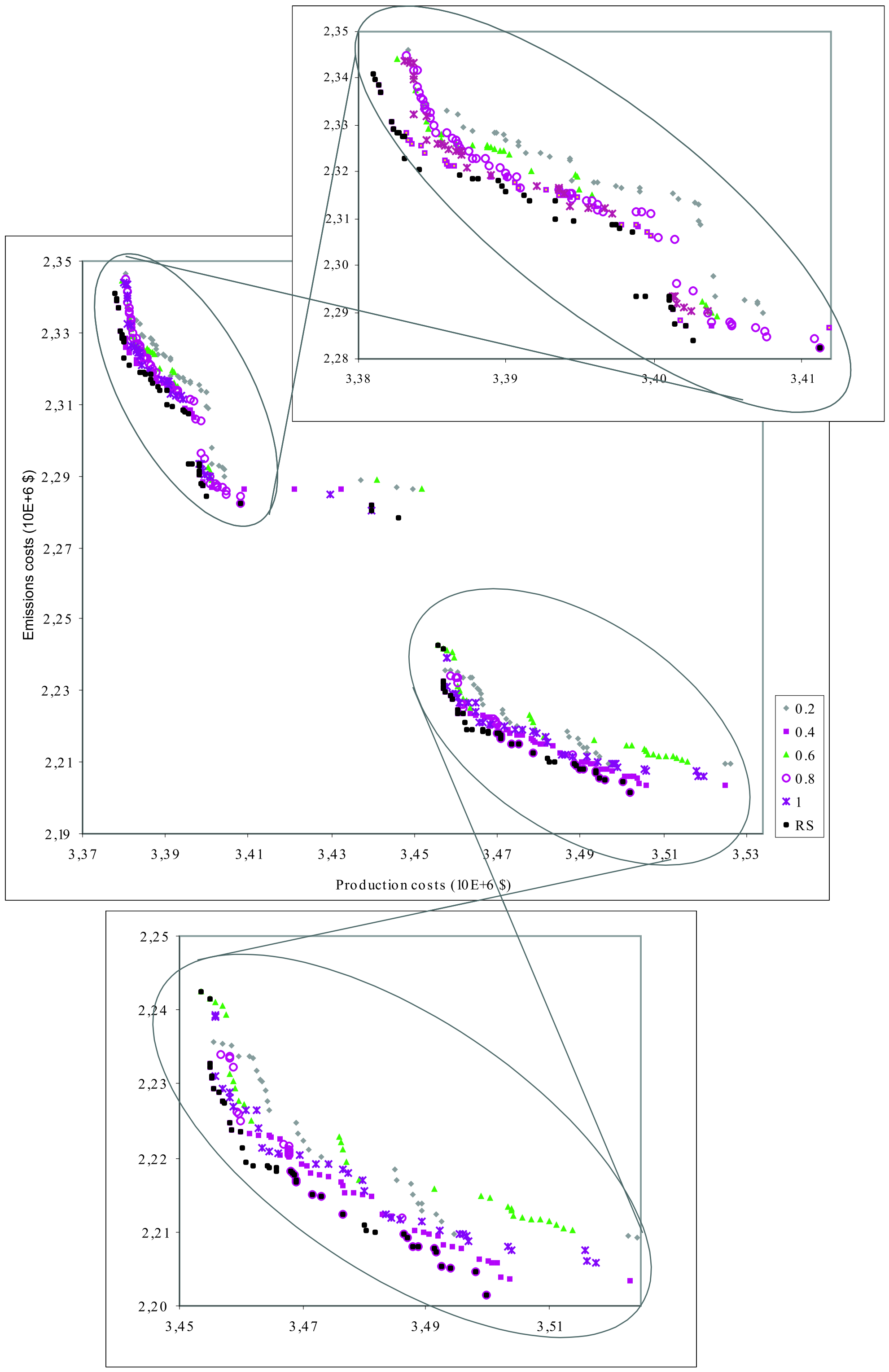


**Figure 6.12:** Graphical representation of results for the 60 unit problem, for different values of $\alpha$ (population size = 5)

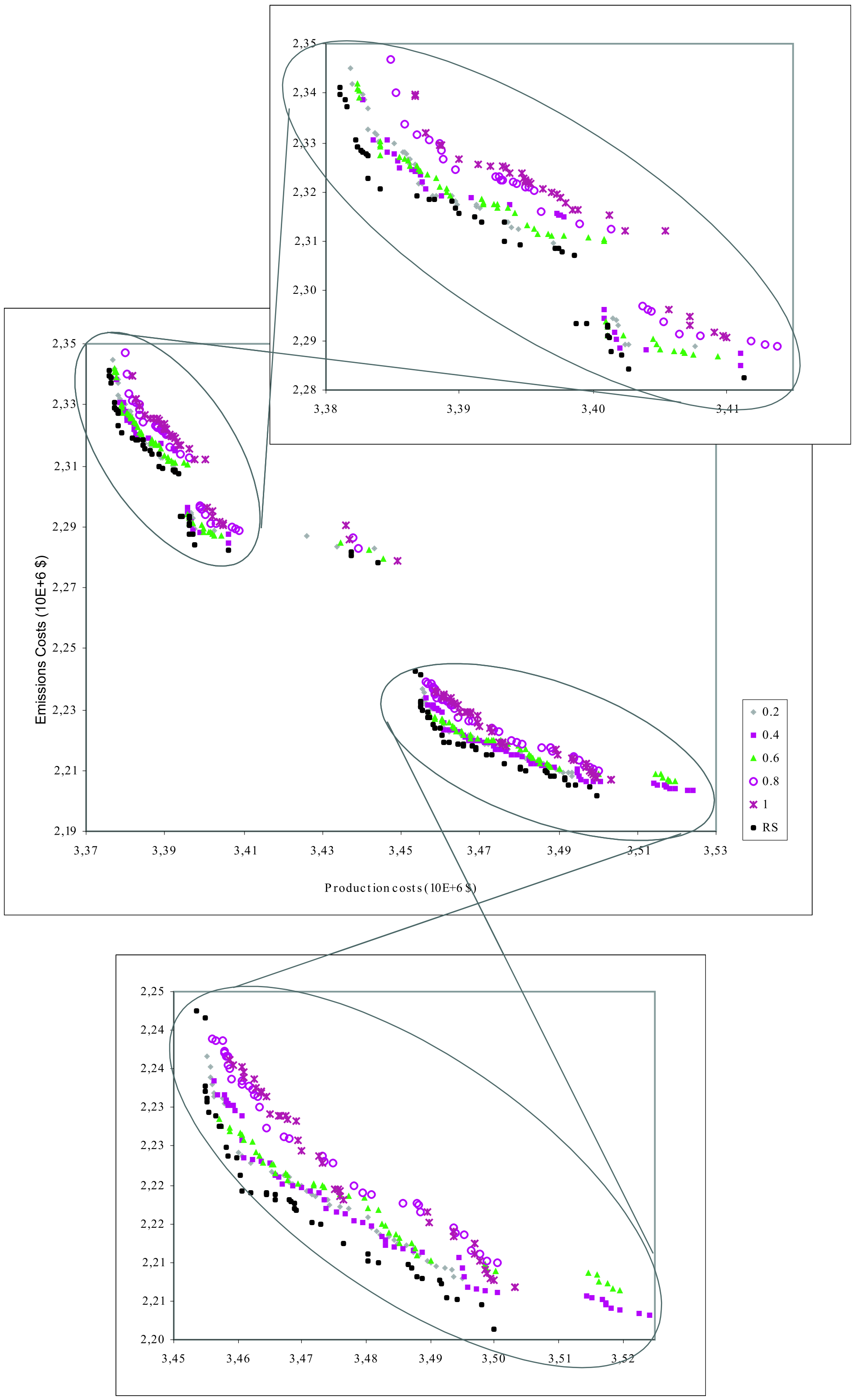


**Figure 6.13:** Graphical representation of results for the 60 unit problem, for different values of $\alpha$ (population size = 10)

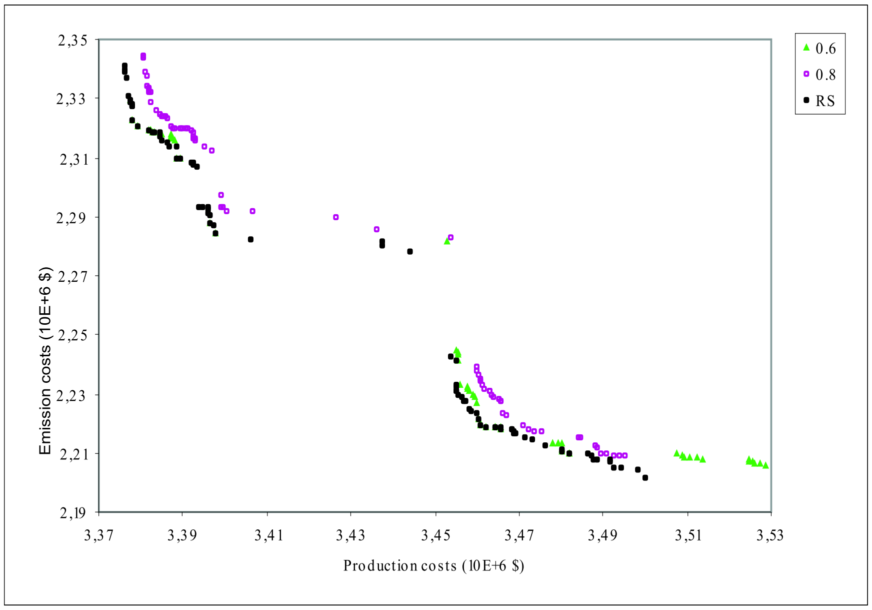


**Figure 6.14:** Graphical representation of results for the 60 unit problem, for different values of $\alpha$ (population size = 20)

MOSA, mCON seems to be much more robust for variations in the size of the population, as the dispersion of the nondominated points is smaller. When compared to PSA, the same conclusion may be reached.

### Number of potentially nondominated solutions

Table 6.1 presents the number of potentially efficient solutions obtained with mCON, considering different values for the $\alpha$ parameter of GRASP (e.g. columns under mCON 0.2 present the results obtained for $\alpha = 0.2$), and for the size of the population. #PE stands for the number of potentially nondominated solutions produced by mCON, #RS stands for the number of solutions in PE that do also belong to the Reference Set and #RefSet stands for the number of solutions in the Reference Set.

Table 6.2 presents the number of potentially efficient solutions obtained by mCON, PSA and MOSA and their contribution to the Reference Set, for all instances, for different population sizes, when the same CPU time is provided. The values under column mCON where obtained by joining the potentially efficient solutions obtained for different values of $\alpha$, for a given population size (i.e. each row in Table 6.1), and eliminating the dominated ones. As shown (see values under column #RS), the main contribution to the definition of a Reference Set is given by mCON. For the 10, 20 and 80 unit problems, PSA and MOSA do not reach

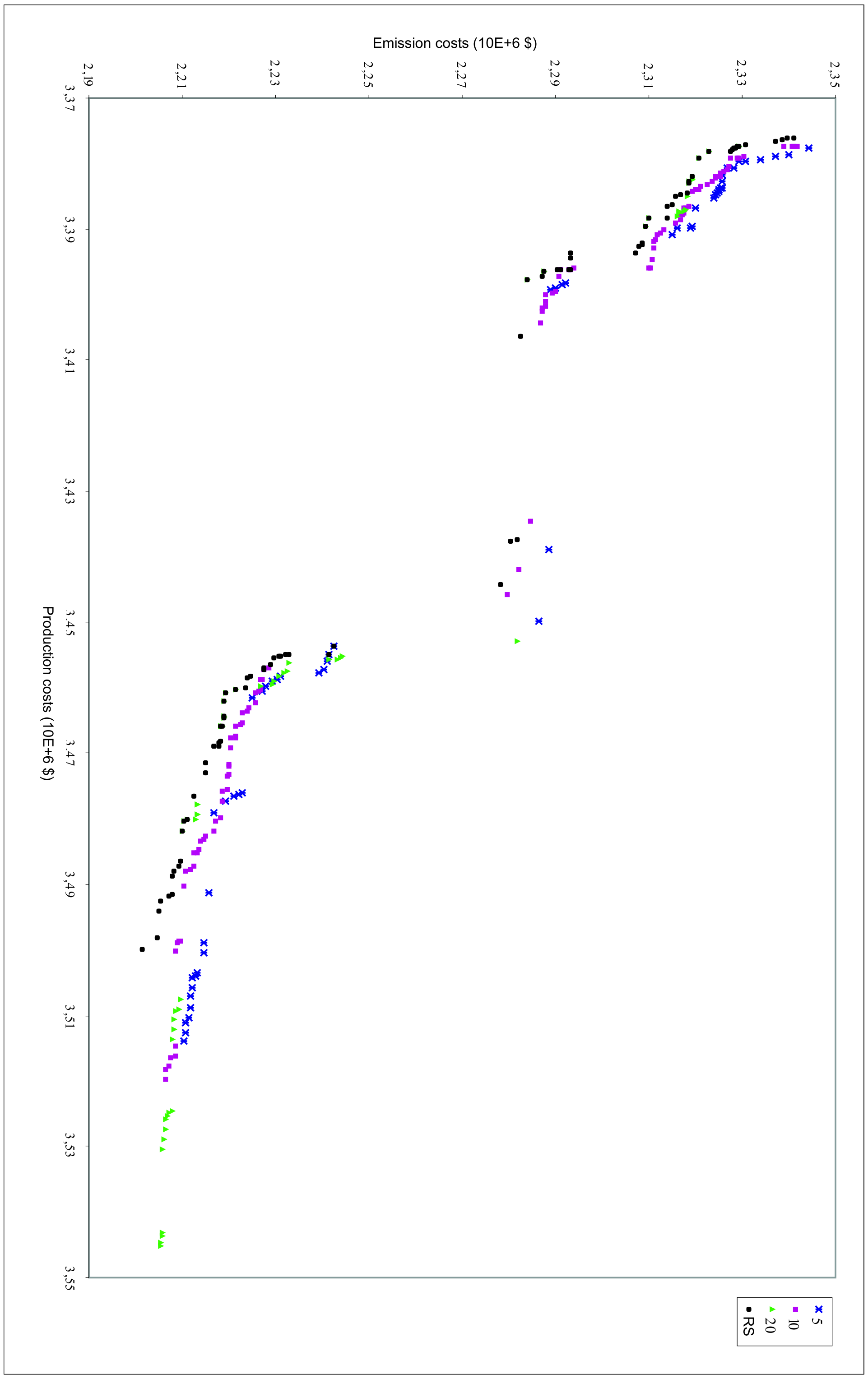


**Figure 6.15:** Graphical representation of results for the 60 unit problem, for different population sizes ($\alpha = 0.6$)

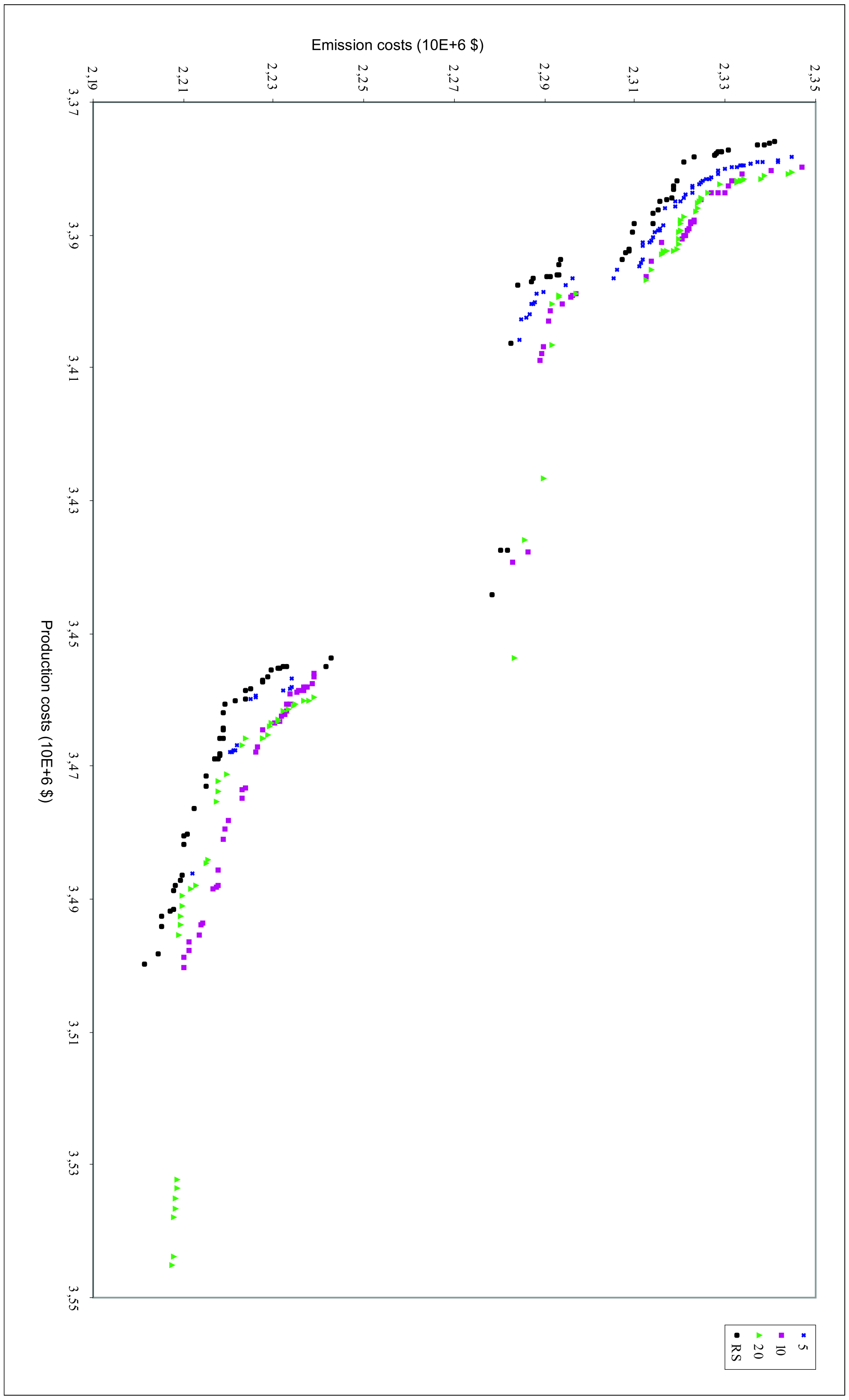


**Figure 6.16:** Graphical representation of results for the 60 unit problem, for different population sizes ($\alpha = 0.8$)

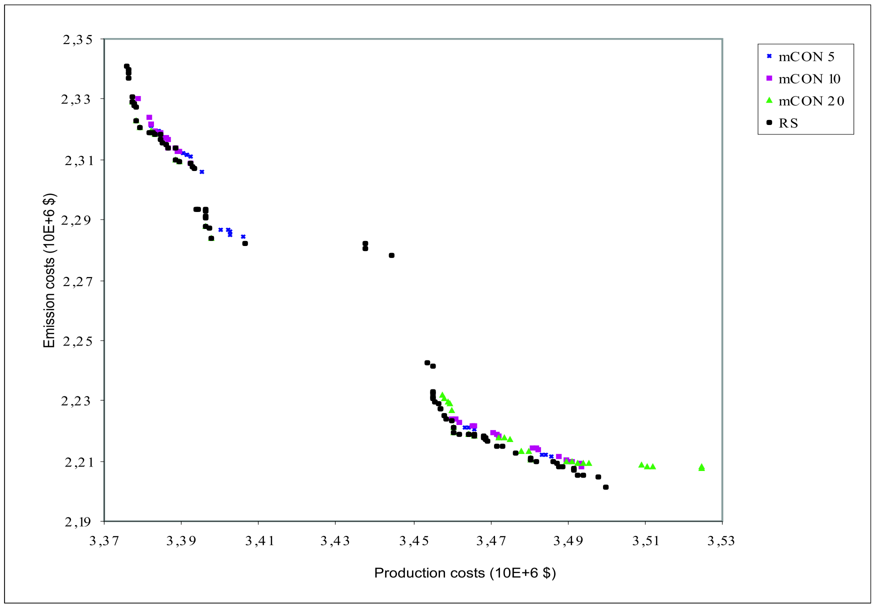


**Figure 6.17:** Graphical representation of PE for the 60 unit problem – mCON

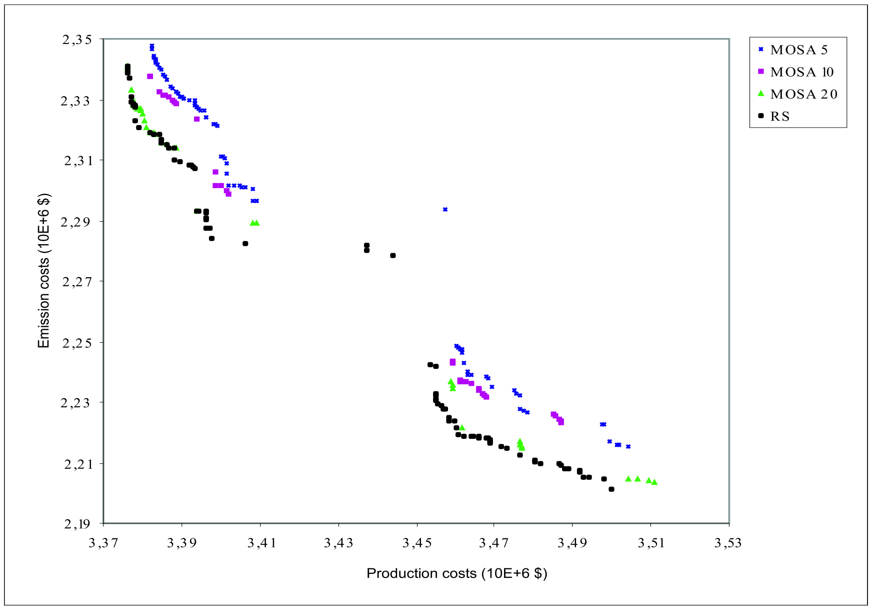


**Figure 6.18:** Graphical representation of PE for the 60 unit problem – MOSA

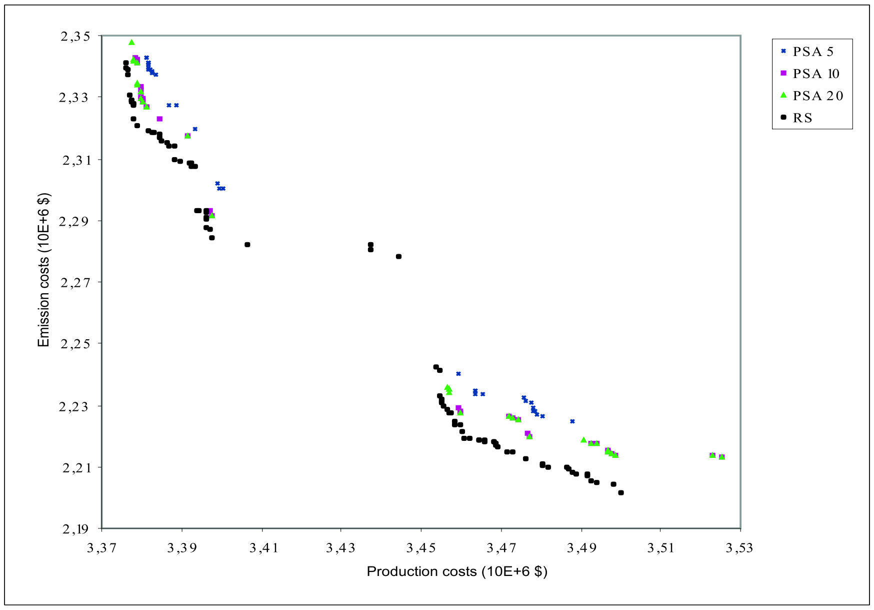


**Figure 6.19:** Graphical representation of PE for the 60 unit problem – PSA

| Prob. | Pop. | mCON 0.2 | | mCON 0.4 | | mCON 0.6 | | mCON 0.8 | | mCON 0.9 | | mCON 1 | | #RefSet |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| size | size | #PE | # RS | #PE | #RS | #PE | #RS | #PE | #RS | #PE | #RS | #PE | #RS | |
| 10 | 5 | 27 | 2 | 34 | 16 | 42 | 0 | 36 | 0 | | | | | 51 |
| | 10 | | | 41 | 0 | 50 | 0 | 52 | 1 | | | | | |
| | 20 | | | 42 | 14 | 39 | 10 | 41 | 11 | | | | | |
| 20 | 5 | 36 | 4 | 50 | 6 | 40 | 7 | 43 | 0 | 42 | 0 | 39 | 0 | 51 |
| | 10 | 48 | 0 | 40 | 16 | 40 | 1 | 92 | 1 | 59 | 0 | | | |
| | 20 | 50 | 0 | 37 | 5 | 37 | 11 | 73 | 0 | | | | | |
| 40 | 5 | | | 75 | 1 | 56 | 0 | 50 | 36 | | | | | 123 |
| | 10 | | | 73 | 4 | 59 | 6 | 78 | 0 | | | | | |
| | 20 | | | 63 | 14 | 97 | 21 | 58 | 28 | | | | | |
| 60 | 5 | 69 | 0 | 81 | 16 | 58 | 2 | 89 | 19 | 69 | 1 | 96 | 1 | 82 |
| | 10 | 96 | 7 | 79 | 0 | 103 | 4 | 68 | 0 | 73 | 0 | 80 | 0 | |
| | 20 | | | | | 63 | 21 | 78 | 0 | | | | | |
| 80 | 5 | 80 | 1 | 79 | 4 | 77 | 0 | 76 | 9 | | | | | 130 |
| | 10 | | | 87 | 22 | 99 | 27 | 115 | 14 | | | | | |
| | 20 | 83 | 11 | 83 | 14 | 68 | 2 | 85 | 28 | | | | | |
| 100 | 5 | | | | | 86 | 38 | 88 | 42 | | | | | 129 |
| | 10 | | | | | 98 | 8 | 107 | 2 | | | | | |
| | 20 | | | 101 | 19 | 93 | 1 | 129 | 8 | | | | | |

**Table 6.1:** mCON – number of potentially efficient solutions

| Prob. | Pop. | mCON | | PSA | | MOSA | | #RefSet |
|---|---|---|---|---|---|---|---|---|
| size | size | #PE | # RS | #PE | #RS | #PE | #RS | |
| 10 | 5 | 38 | 18 | 24 | 0 | 31 | 0 | 51 |
| | 10 | 51 | 1 | 19 | 0 | 34 | 0 | |
| | 20 | 47 | 35 | 25 | 0 | 47 | 0 | |
| 20 | 5 | 60 | 17 | 49 | 0 | 44 | 0 | 51 |
| | 10 | 63 | 18 | 44 | 0 | 48 | 0 | |
| | 20 | 42 | 16 | 45 | 0 | 55 | 0 | |
| 40 | 5 | 59 | 37 | 54 | 12 | 100 | 0 | 123 |
| | 10 | 96 | 10 | 67 | 12 | 57 | 9 | |
| | 20 | 83 | 58 | 67 | 12 | 64 | 9 | |
| 60 | 5 | 89 | 39 | 71 | 0 | 85 | 0 | 82 |
| | 10 | 129 | 11 | 66 | 0 | 111 | 0 | |
| | 20 | 69 | 20 | 73 | 0 | 76 | 13 | |
| 80 | 5 | 112 | 14 | 90 | 0 | 111 | 0 | 130 |
| | 10 | 114 | 63 | 112 | 0 | 117 | 0 | |
| | 20 | 99 | 54 | 116 | 0 | 130 | 0 | |
| 100 | 5 | 104 | 79 | 84 | 0 | 107 | 0 | 129 |
| | 10 | 125 | 10 | 99 | 0 | 91 | 0 | |
| | 20 | 114 | 28 | 114 | 6 | 91 | 6 | |

**Table 6.2:** Contribution of mCON, PSA and MOSA to the Reference Set

any PE solutions (also for the 60 unit problem, for PSA). For the remaining instances, some of the solutions in the Reference Set are also obtained by PSA and MOSA but, even so, when compared to mCON the contribution of these metaheuristics is smaller.

The data in these tables does not however give any information on how well spread are the sets PE over the Reference Set, nor on the distance to the Reference Set of the solutions that will not contribute for its definition. Therefore, other measures shall be used, to better characterise the set of results obtained.

### Outperformance relations and metrics

The outperformance relations reported in [Hansen and Jaszkiewicz 1998] will be considered to evaluate and compare the final results obtained with mCON, PSA and MOSA. However, as they only provide qualitative comparisons, some quantitative evaluators will also be used.

Starting with metric $\mathcal{C}$, Table 6.3 presents the results obtained by mCON with those obtained with MOSA and PSA, for populations with the same size. It also presents the outperformance relations between each set of results. In this table, W, S and C stand for

Weak, Strong and Complete outperformance, *Cover* stands for the result of $\mathcal{C}$(mCON x y, MOSA x y), and $\overline{Cover}$ for the result of $\mathcal{C}$(MOSA x y, mCON x y). mCON *x y* refers to the *x* unit problem, with a population of *y* solutions (e.g. mCON 10 5 refers to the 10 unit problem, with a population of 5 solutions). A $\surd$ sign for W, S or C means that mCON x y outperforms MOSA x y. The same convention is used for PSA. A "-" sign means that the results are not comparable.

Concerning the results reported, the coverage indicator $\mathcal{C}$ always provides better results for mCON than for MOSA or PSA, for populations of the same size ($Cover > \overline{Cover}$). This means that there are more points in the mCON sets that dominate, or are equal to, points in the MOSA and PSA sets. In the extreme case, when $Cover = 1$ and $\overline{Cover} = 0$, all points in the mCON sets dominate those in the MOSA and PSA sets.

| | | W | S | C | *Cover* | $\overline{Cover}$ |
|---|---|---|---|---|---|---|
| mCON 10 5 | MOSA 10 5 | $\surd$ | $\surd$ | $\surd$ | 1 | 0 |
| mCON 10 10 | MOSA 10 10 | $\surd$ | $\surd$ | $\surd$ | 1 | 0 |
| mCON 10 20 | MOSA 10 20 | $\surd$ | $\surd$ | $\surd$ | 1 | 0 |
| mCON 20 5 | MOSA 20 5 | $\surd$ | $\surd$ | $\surd$ | 1 | 0 |
| mCON 20 10 | MOSA 20 10 | $\surd$ | $\surd$ | $\surd$ | 1 | 0 |
| mCON 20 20 | MOSA 20 20 | $\surd$ | $\surd$ | $\surd$ | 1 | 0 |
| mCON 40 5 | MOSA 40 5 | $\surd$ | $\surd$ | $\surd$ | 1 | 0 |
| mCON 40 10 | MOSA 40 10 | - | - | - | 0.666667 | 0.1145830 |
| mCON 40 20 | MOSA 40 20 | - | - | - | 0.718750 | 0.0481928 |
| mCON 60 5 | MOSA 60 5 | $\surd$ | $\surd$ | $\surd$ | 1 | 0 |
| mCON 60 10 | MOSA 60 10 | $\surd$ | $\surd$ | $\surd$ | 1 | 0 |
| mCON 60 20 | MOSA 60 20 | - | - | - | 0.552632 | 0.362319 |
| mCON 80 5 | MOSA 80 5 | - | - | - | 0.981982 | 0 |
| mCON 80 10 | MOSA 80 10 | $\surd$ | $\surd$ | $\surd$ | 1 | 0 |
| mCON 80 20 | MOSA 80 20 | - | - | - | 0.984615 | 0 |
| mCON 100 5 | MOSA 100 5 | - | - | - | 0.981308 | 0 |
| mCON 100 10 | MOSA 100 10 | - | - | - | 0.879121 | 0.064000 |
| mCON 100 20 | MOSA 100 20 | - | - | - | 0.868132 | 0.008772 |
| mCON 10 5 | PSA 10 5 | - | - | - | 0.875000 | 0 |
| mCON 10 10 | PSA 10 10 | - | - | - | 0.789474 | 0.098039 |
| mCON 10 20 | PSA 10 20 | $\surd$ | $\surd$ | $\surd$ | 1 | 0 |
| mCON 20 5 | PSA 20 5 | $\surd$ | $\surd$ | $\surd$ | 1 | 0 |
| mCON 20 10 | PSA 20 10 | - | - | - | 0.977273 | 0.031746 |
| mCON 20 20 | PSA 20 20 | - | - | - | 0.933333 | 0.071429 |
| mCON 40 5 | PSA 40 5 | - | - | - | 0.629630 | 0.101695 |
| mCON 40 10 | PSA 40 10 | - | - | - | 0.686567 | 0.145833 |

Table 6.3: mCON vs MOSA and mCON vs PSA - *continued*

| | | W | S | C | $Cover$ | $\overline{Cover}$ |
|---|---|---|---|---|---|---|
| mCON 40 20 | PSA 40 20 | - | - | - | 0.791045 | 0.036145 |
| mCON 60 5 | PSA 60 5 | √ | √ | √ | 1 | 0 |
| mCON 60 10 | PSA 60 10 | √ | √ | √ | 1 | 0 |
| mCON 60 20 | PSA 60 20 | - | - | - | 0.945205 | 0 |
| mCON 80 5 | PSA 80 5 | √ | √ | √ | 1 | 0 |
| mCON 80 10 | PSA 80 10 | √ | √ | √ | 1 | 0 |
| mCON 80 20 | PSA 80 20 | - | - | - | 0.945205 | 0 |
| mCON 100 5 | PSA 100 5 | √ | √ | √ | 1 | 0 |
| mCON 100 10 | PSA 100 10 | √ | √ | √ | 1 | 0 |
| mCON 100 20 | PSA 100 20 | - | - | - | 0.859649 | 0.035088 |

Table 6.3: mCON vs MOSA and mCON vs PSA

Table 6.4 compares the results obtained by mCON, for different values of the population size. The results were not comparable, for all evaluations, through outperformance relations. The coverage indicators do not also allow one to draw any conclusions concerning the impact of an increase in the population size on the quality of the results. If we compare, for example, the results obtained for the 10 unit problem when populations of 5 and 10 solutions are considered, a better result is obtained for the smallest population. However, if we compare the results obtained for populations of 5 and 20 solutions, the later leads to a better coverage value.

| | | $Cover$ | $\overline{Cover}$ |
|---|---|---|---|
| mCON 10 5 | mCON 10 10 | 0.705882 | 0.105263 |
| mCON 10 5 | mCON 10 20 | 0.319149 | 0.605263 |
| mCON 10 10 | mCON 10 20 | 0 | 0.980392 |
| mCON 20 5 | mCON 20 10 | 0.492063 | 0.533333 |
| mCON 20 5 | mCON 20 20 | 0.476190 | 0.600000 |
| mCON 20 10 | mCON 20 20 | 0.500000 | 0.476190 |
| mCON 40 5 | mCON 40 10 | 0.635417 | 0.169492 |
| mCON 40 5 | mCON 40 20 | 0.228916 | 0.406780 |
| mCON 40 10 | mCON 40 20 | 0.096386 | 0.781250 |
| mCON 60 5 | mCON 60 10 | 0.720930 | 0.179775 |
| mCON 60 5 | mCON 60 20 | 0.608696 | 0.460674 |
| mCON 60 10 | mCON 60 20 | 0.536232 | 0.558140 |
| mCON 80 5 | mCON 80 10 | 0.184211 | 0.580357 |
| mCON 80 5 | mCON 80 20 | 0.090909 | 0.803571 |
| mCON 80 10 | mCON 80 20 | 0.424242 | 0.447368 |
| mCON 100 5 | mCON 100 10 | 0.888000 | 0.086539 |
| mCON 100 5 | mCON 100 20 | 0.728070 | 0.153846 |
| mCON 100 10 | mCON 100 20 | 0.149123 | 0.752000 |

**Table 6.4:** mCON – effect of population size

| | | W | S | C | $\overline{Cover}$ |
|---|---|---|---|---|---|
| Ref10 | mCON10 5 | √ | √ | - | 0.352941 |
| | MOSA10 5 | √ | √ | √ | 0 |
| | PSA10 5 | √ | √ | √ | 0 |
| | mCON10 10 | √ | √ | - | 0.0196078 |
| | MOSA10 10 | √ | √ | √ | 0 |
| | PSA10 10 | √ | √ | √ | 0 |
| | mCON10 20 | √ | √ | - | 0.686275 |
| | MOSA10 20 | √ | √ | √ | 0 |
| | PSA10 20 | √ | √ | √ | 0 |
| Ref20 | mCON20 5 | √ | √ | 0.333333 | |
| | MOSA20 5 | √ | √ | √ | 0 |
| | PSA20 5 | √ | √ | √ | 0 |
| | mCON20 10 | √ | √ | - | 0.352941 |
| | MOSA20 10 | √ | √ | √ | 0 |
| | PSA20 10 | √ | √ | √ | 0 |
| | mCON20 20 | √ | √ | - | 0.313725 |
| | MOSA20 20 | √ | √ | √ | 0 |
| | PSA20 20 | √ | √ | √ | 0 |
| Ref40 | mCON40 5 | √ | √ | - | 0.284553 |
| | MOSA40 5 | √ | √ | √ | 0 |
| | PSA40 5 | √ | √ | - | 0.097561 |
| | mCON40 10 | √ | √ | - | 0.0813008 |
| | MOSA40 10 | √ | √ | - | 0.0650407 |
| | PSA40 10 | √ | √ | - | 0.097561 |
| | mCON40 20 | √ | √ | - | 0.471545 |
| | MOSA40 20 | √ | √ | - | 0.0650407 |
| | PSA40 20 | √ | √ | - | 0.097561 |
| Ref 60 | mCON60 5 | √ | √ | - | 0.47561 |
| | MOSA60 5 | √ | √ | √ | 0 |
| | PSA60 5 | √ | √ | √ | 0 |
| | mCON60 10 | √ | √ | - | 0.134146 |
| | MOSA60 10 | √ | √ | √ | 0 |
| | PSA60 10 | √ | √ | √ | 0 |
| | mCON60 20 | √ | √ | - | 0.231707 |
| | MOSA60 20 | √ | √ | - | 0.158537 |
| | PSA60 20 | √ | √ | √ | 0 |
| Ref80 | mCON80 5 | √ | √ | - | 0.107692 |
| | MOSA80 5 | √ | √ | √ | 0 |
| | PSA80 5 | √ | √ | √ | 0 |
| | mCON80 10 | √ | √ | - | 0.476923 |
| | MOSA80 10 | √ | √ | √ | 0 |
| | PSA80 10 | √ | √ | √ | 0 |
| | mCON80 20 | √ | √ | - | 0.415385 |
| | MOSA80 20 | √ | √ | √ | 0 |
| | PSA80 20 | √ | √ | √ | 0 |
| Ref100 | mCON100 5 | √ | √ | - | 0.612403 |
| | MOSA100 5 | √ | √ | √ | 0 |
| | PSA100 5 | √ | √ | √ | 0 |
| | mCON100 10 | √ | √ | - | 0.0775194 |
| | MOSA100 10 | √ | √ | √ | 0 |
| | PSA100 10 | √ | √ | √ | 0 |
| | mCON100 20 | √ | √ | - | 0.952381 |
| | MOSA100 20 | √ | √ | - | 0.0465116 |
| | PSA100 20 | √ | √ | - | 0.0465116 |

**Table 6.5:** Comparison with the nondominated set

Finally, Table 6.5 compares all the computational tests that were performed, with the best set of potentially nondominated solutions found, the reference set – Ref x. As expected, *Cover* is, for all runs, equal to 1. Besides, one should also notice that these results do comply with those reported in Table 6.2. In fact, the simulations reported in that table that do not contribute with any solution to the Reference Set, should get a $\overline{Cover}$ of 0. Furthermore, within the same instance, those simulations that contribute with the same number of solutions, should reach equal $\overline{Cover}$ values.

To conclude the evaluation of results, the $R_2$ metric is used. As referred in section 6.2.3, this metric calculates the expected difference in the utility of two approximations A and B, or between an approximation A and a reference point $z^0$.

In this work, as suggested in [Hansen and Jaszkiewicz 1998], we will estimate the expected value of the weighted Tchebycheff scalarising function ($s_\infty$) of an approximation set A over the set of normalised weight vectors $\Lambda$ (see expression (6.15)) to evaluate and compare the results obtained by mCON, PSA and MOSA.

$$R_2(A) = E(s^*_\infty(z^0, A, \Lambda)) = \int_{\lambda \in \Lambda} s^*_\infty(z^0, A, \Lambda) p(\Lambda) d\Lambda \tag{6.15}$$

where $s^*_\infty(z^0, A, \Lambda) = \min_{z \in A}\{s_\infty(z^0, z, \Lambda)\}$ is the best value achieved by function $s_\infty(z, z^0, \Lambda)$ on approximation A and $z^0$ is a reference point. Those values are presented in Table 6.6. The reference points that were used correspond to the lowest value achieved in all simulations, for each objective, and are presented in the last two columns of the table. The value achieved by the Reference Set when evaluated through metric $R_2$ is also provided as a reference.

| Prob. | Pop. | mCON | PSA | MOSA | RefSet | Ref. Point | |
|---|---|---|---|---|---|---|---|
| size | size | | | | | Fuel | Emissions |
| 10 | 5 | 4 569.5 | 5 050.45 | 5 314.13 | 4 447.21 | 565 950 | 363 042 |
| | 10 | 4 665.25 | 5 001.7 | 5 246.16 | | | |
| | 20 | 4 321.09 | 4 848.66 | 5 153.29 | | | |
| 20 | 5 | 8 138.32 | 9 876.98 | 9 638.12 | 7 975.83 | $1.12776 \times 10^6$ | 726 215 |
| | 10 | 8 095.46 | 9 344.41 | 9 306.88 | | | |
| | 20 | 8 336.64 | 8 760.96 | 9 138.78 | | | |
| 40 | 5 | 15 914.6 | 17 479.8 | 18 376.3 | 15 124.1 | $2.25135 \times 10^6$ | $1.46147 \times 10^6$ |
| | 10 | 15 954.5 | 17 322.9 | 16 647.8 | | | |
| | 20 | 15 441 | 16 424 | 16 571.1 | | | |
| 60 | 5 | 20 696.6 | 24 163.4 | 24 715.2 | 20 531.5 | $3.376 \times 10^6$ | $2.20141 \times 10^6$ |
| | 10 | 21 068.4 | 22 684 | 23 948.9 | | | |
| | 20 | 21 139.7 | 22 414.2 | 21 649.9 | | | |
| 80 | 5 | 27 729.7 | 31 685.9 | 31 151.1 | 27 133.3 | $4.49961 \times 10^6$ | $2.93864 \times 10^6$ |
| | 10 | 27 398.6 | 30 988.7 | 31 043.9 | | | |
| | 20 | 27 556.4 | 30 017 | 30 683.4 | | | |
| 100 | 5 | 33 609 | 36 326 | 36 034.8 | 33 017.8 | $5.623830 \times 10^6$ | $3.68086 \times 10^6$ |
| | 10 | 34 047.5 | 36 034.4 | 35 160.5 | | | |
| | 20 | 33 608.4 | 35 062.3 | 34 670 | | | |

**Table 6.6:** Result of metric $R_2$

When comparing mCON with PSA and MOSA, through the $R_2$ metric, mCON achieves better results, for all cases. Moreover, PSA presents better results for most cases, when compared to MOSA, except for the 100 unit problem, for which MOSA performs better.

## 6.6 Concluding remarks

In this chapter we have described a new multiobjective metaheuristic based on the concept of *Constraint Oriented Neighbourhoods* (mCON), to tackle the Unit Commitment problem.

To assess the potential of the approach, computational tests were performed and the results were compared with those obtained with MOSA (Multiobjective Simulated Annealing) and PSA (Pareto Simulated Annealing). mCON has systematically produced better results than any of these approaches.

To avoid the need of aggregating objectives, mCON considers several neighbourhood movements to be applied during the search process in a way that depends on the current dispersion of solutions in the solution space. A correct definition of the neighbourhood movements produces neighbour solutions that are similar (due to the introduction of the concept of *Constraint Oriented Neighboourhoods*) leading to a smoother search process, and a correct selection of a neighbourhood movement during the search does hopefully lead to a well dispersed set of potentially nondominated solutions.

For the Unit Commitment problem, the computational results obtained with mCON were more robust and less dependent on variations of the metaheuristic parameters, when compared with those obtained with MOSA and PSA. The algorithm also proved to be consistently more effective, leading to better quality results in all performed runs. These experiments do support our belief that this approach can be successfully used in other classes of problems. This will probably be the case of problems where simple movements lead to unfeasible solutions, and the recovering techniques needed to reach feasibility are too complex and may drastically change the structure of the solutions. This will also be the case of those problems where worse neighbour solutions are produced too often, leading to a considerable increase in CPU time. This innovative approach can therefore be seen as a potential alternative technique to solve hard multiobjective combinatorial optimisation problems.

# Chapter 7

# Conclusions

Research work in the field of Local Search based metaheuristics has considerably grown in recent years, several new approaches based on innovative paradigms, or on merging old ideas and algorithms, having been proposed. Nevertheless, the increasing interest of the research community on these techniques has not been followed by an equivalent increase of their practical applications, due to the reluctance of Decision-Makers on supporting their decisions on methods that are not robust, highly depending on a correct tuning of some parameters.

Trying to overcome this difficulty some attempts have already been made to reduce the effect of metaheuristics' parameters on their performance through e.g. hybridisation or adaptive techniques. A different reasoning was followed in this thesis: an innovative general search strategy is proposed, *Constraint Oriented Neighbourhoods* (CON) that, if embedded in Local Search based metaheuristics, tries to make these algorithms less dependent on parameter tuning and, therefore, more robust.

A natural evolution of this work, given the practical interest of multiobjective models, was to develop another general approach – mCON – capable of correctly handling several objectives simultaneously, aiming again at reducing the influence of parameter tuning.

In a parallel line of research, the Unit Commitment of power units was studied. Due to its economical importance, this problem has for long been a point of concern for power generation companies, continuously looking for more advanced optimisation tools leading to extra reductions in production costs. The software packages currently available in this area do usually provide the end-user with optimisation heuristics based on priority lists and, mostly, based on Lagrangean Relaxation. These approaches do not however allow a correct modelling of the several discontinuities that the problem presents, as metaheuristics do. Therefore, this is an area of application for metaheuristics with high potential, in particular if we think about its natural multiobjective extensions.

## 7.1 Contributions of the thesis

The general objectives of this thesis, stated in chapter 1, were to present worthy contributions for an effective usage of metaheuristics in practice, and to propose innovative approaches to the Unit Commitment Problem. It is the author's opinion that, globally, those objectives were achieved, leading to the following more significant contributions.

### 7.1.1 Metaheuristics

Within the area of metaheuristics two main contributions should be highlighted:

i. The development of the concept of *Constraint Oriented Neighbourhoods* (CON) as a powerful way of improving the performance of Local Search based metaheuristics in certain domains of combinatorial optimisation problems. This concept aims at strongly reducing algorithms' dependency on parameter tuning. The idea behind the approach is to "control" the randomness of neighbourhood movements by applying different movements to a solution, depending on the kind of constraints that may be violated. By doing so, drastic changes in a solution are avoided, in successive iterations, thus allowing a smoother search.

ii. The development of a multiobjective metaheuristic – mCON – based on the same reasoning of the *Constraint Oriented Neighbourhoods* search strategy. The initial structure designed for single objective problems was extended and neighbourhoods that are more prone at enhancing one objective than the others are now considered. In each iteration of mCON, the neighbourhood operation applied to the current solution is selected according to the constraints that are violated and to the objective that one would like to improve the most. The approach does also avoid the need of defining and/or updating weights, that are usually required for aggregating the objectives in Local Search based multiobjective metaheuristics.

### 7.1.2 Unit Commitment

This work has also resulted in several contributions for the Unit Commitment Problem (UCP), leading to the development of a new algorithm that, when tested in a set of instances from the literature, was systematically capable of reaching results that are better than those previously published, with drastic reductions in CPU times. This algorithm is the final result of a set of intermediate contributions of this thesis that:

i. Develops an alternative representation of Unit Commitment solutions, allowing that minimum up and down time constraints are tackled more easily.

ii. Proposes an innovative GRASP methodology to solve the problem.

iii. Suggests alternative neighbourhood movements, that allow that the concept of *Constraint Oriented Neighbourhoods* is applied to the UCP, and validates the effectiveness of the approach by applying it to a set of instances from the literature.

Furthermore, the work developed in this thesis, applies the concept of *mCON* to a multiobjective UCP with two objectives to be minimised: total production costs and emissions. Again, the approach proves to be effective, leading to better results than those obtained with other MOMH proposed in the literature.

## 7.2 Guidelines for future research

The results that are presented in this thesis, concerning the application of the concept of *Constraint Oriented Neighbourhoods* to the Unit Commitment Problem, make us believe that this is a very promising technique and that one should invest on assessing the applicability of the strategy to other problems, arising either from practical situations or from the literature, both for single and for multiobjective scenarios.

Furthermore, and focusing now on on the Unit Commitment Problem, one should evolve to variants of the base problem where other types of units rather than thermal are available. In such cases a proper coordination between the different kinds of units is desirable, to obtain a rational and efficient usage of all resources.

Several lines of research may therefore be drawn, supported by the contributions of this thesis, and in depth research work may be developed. Some of those lines of research are proposed here structured as three plans for Masters dissertations, one concerned with the applicability of *Constraint Oriented Neighbourhoods* to other combinatorial optimisation problems and the other two dealing with the hydrothermal Unit Commitment Problem.

### 7.2.1 Plan A: On the applicability of Constraint Oriented Neighbourhoods to Combinatorial Optimisation problems

#### Scope and objectives

The concept of *Constraint Oriented Neighbourhoods* (CON) was first introduced in [Viana et al. 2003a], its major motivation being the achievement of a smoother search process for some types of combinatorial optimisation problems. This is the case of those problems where simple movements lead to unfeasible solutions, and the recovering techniques needed to reach feasibility involve complex rules that may drastically change the structure of the solutions; or those where worse neighbour solutions are produced too often, leading to a considerable increase in CPU time. For these purposes, neighbourhood structures are defined, in such a way that, if leading to unfeasible solutions, feasibility is easy to recover and the resulting solution is not very different in structure from the current one. This concept has already been applied to the Unit Commitment of power units with success. Still, it is expected that it is also effective at solving other combinatorial optimisation problems referred in the literature.

The aim of this dissertation is to identify and characterise a set of problems whose characteristics are suitable to apply the CON strategy, and to develop neighbourhood movements to be embedded in that strategy, so that the problems are effectively solved. Particular attention will be given to Resource Constrained Project Scheduling, Timetabling and Rostering Scheduling problems. The strategy shall be integrated in several Local Search based metaheuristics that should be assessed and compared by extensive computational experiments.

#### Plan

*Phase 1* (duration: 2 months) – literature survey on the state-of-the-art of metaheuristics; identification and characterisation of a set of problems from the literature to be solved with CON.

*Phase 2* (duration: 5 months) – design and implementation of neighbourhood structures for each of the problems considered in *Phase 1.*

*Phase 3* (duration: 2 months) – exhaustive testing, on different metaheuristics, of the methodology proposed; comparison with other methodologies from the literature.

*Phase 4* (duration:3 months) – synthesis of results and writing of dissertation.

### 7.2.2 Plan B: Hydrothermal coordination – an integrated approach

#### Scope and objectives

In a hydrothermal system a proper coordination between the hydro and the thermal systems is required to minimise the system total production costs. This is generally achieved by decomposing the base problem in two subproblems, one handling the hydro system and the other handling the thermal one. Usually, the hydro scheduling subproblem is solved first and then, for the remaining demand (the total load demand minus the hydro production), the thermal subproblem is solved. The process is repeated until the thermal marginal costs or hydro generation converge.

Metaheuristic approaches dealing with the Hydrothermal Coordination problem (HTC) have also followed this reasoning: the metaheuristic tackles the combinatorial thermal problem, while other techniques are used to solve the hydro subproblem. Hierarchical approaches of this kind may however lead to suboptimal results, due to lack of convergence of the iterative approach. Therefore, an integrated approach where both systems are tackled simultaneously is desirable.

The aim of this dissertation is to develop an effective integrated approach to solve the HTC with metaheuristics. This will be achieved by discretising the production levels of the hydro units and defining appropriate neighbourhood movements that allow changes to be made both in the thermal and in the hydro units. The approach should consider the *Constraint Oriented Neighbourhoods* paradigm and, as so, different neighbourhood movements should be designed to be applied during the search process, according to the type of constraints that are violated in each iteration. The starting point of this work will consist on analysing some previous work on the application of *Constraint Oriented Neighbourhoods* to the HTC (see Appendix B) and on proposing alternative movements.

#### Plan

*Phase 1* (duration: 2 months) – literature survey on HTC mathematical models and on the methodologies applicable to the problem and to its variants.

*Phase 2* (duration: 5 months) – design and implementation of a methodology based on *Constraint Oriented Neighbourhoods* to solve the HTC as an integrated problem.

*Phase 3* (duration: 2 months) – exhaustive testing of the methodology proposed and comparison with other methodologies from the literature.

*Phase 4* (duration: 3 months) – synthesis of results and writing of dissertation.

### 7.2.3 Plan C: Hydrothermal coordination – a multiobjective approach

#### Scope and objectives

A major concern of hydrothermal scheduling is on properly coordinating the hydro and the thermal systems, so that total production costs are minimised. But we may think of other objectives that should be also considered to achieve an effective management of the global system. For example, given the inherent uncertainty of water inflow forecasts, it may be important to measure the risk of having to commit very high production cost units (such as gas turbines), for a given hydrothermal schedule, if the actual and the predicted inflows differ.

A risk averse solution would necessarily pass through keeping the reservoir levels high, so that there would always be a slack in hydro production, in case the values of the forecasted inflows were reduced. This behaviour would naturally lead to higher production costs, because the thermal units would be used more intensively. On the other hand, a risky solution would tend to maximise the hydro production, reducing operation costs.

As these objectives are conflicting, one should consider a multiobjective approach when tackling the problem. This is the general scope of this dissertation. It will consider the mCON paradigm and, for doing so, it will be necessary to design alternative neighbourhoods, some more prone at minimising operation costs and others at minimising the risk of having to commit very high cost units. At the end some computational tests shall be performed to assess the potential of the approach.

#### Plan

*Phase 1* (duration: 2 months) – literature survey on hydrothermal coordination problems and on multiobjective metaheuristics.

*Phase 2* (duration: 5 months) – design and implementation of a methodology based on mCON to apply to the HTC.

*Phase 3* (duration: 2 months) – exhaustive testing of the methodology proposed and comparison with other methodologies from the literature.

*Phase 4* (duration: 3 months) – synthesis of results and writing of dissertation.

# Appendix A

# Case studies

## A.1 Case study A

The problem instances named *Case study A* were first presented in [Kazarlis et al. 1996]. All problems consider a 24 hour planning horizon and the number of units varies from 10 to 100 units.

In Table A.1, the base problem (for 10 units) is presented. The other problems are generated based on this one, by reproducing the 10 units as many times as necessary. The load values are also multiplied by the same increasing ratio and, in all cases, the reserve requirements are considered to be 10% of the load (Table A.2).

In Table A.1, $a$, $b$ and $c$ are the coefficients to be used in the fuel cost function, and $a_1$, $b_1$ and $c_1$ the ones to be used in the emissions objective function, in Chapter 6. In what concerns transition costs, hot start-up costs are considered when the unit has been off for a number of periods smaller or equal to the value presented in row *cold start hours*, and cold-start costs are considered otherwise. Shut-down costs are set to zero, for all instances. Finally, the *initial state* row is related to the unit initial conditions: a positive value ($+v$) means that the unit has already been on for $v$ consecutive periods and a negative value indicates that it has been off (e.g. unit 6 has been off for 3 periods).

Table A.2 presents the load and reserve requirements for a 24 hour planning horizon.

| | Unit 1 | Unit 2 | Unit 3 | Unit 4 | Unit 5 |
|---|---|---|---|---|---|
| Pmax (MW) | 455 | 455 | 130 | 130 | 162 |
| Pmin (MW) | 150 | 150 | 20 | 20 | 25 |
| a (\$/MW$^2$h) | 0.00048 | 0.00031 | 0.002 | 0.00211 | 0.00398 |
| b (\$/MWh) | 16.19 | 17.26 | 16.60 | 16.50 | 19.70 |
| c (\$/h) | 1000 | 970 | 700 | 680 | 450 |
| $a_1$ (\$/MW$^2$h) | 0.0004 | 0.0003 | 0.0022 | 0.0011 | 0.001 |
| $b_1$ (\$/MWh) | 12.19 | 10.26 | 10.60 | 15.50 | 7.70 |
| $c_1$ (\$/h) | 712 | 570 | 700 | 860 | 350 |
| min up (h) | 8 | 8 | 5 | 5 | 6 |
| min down (h) | 8 | 8 | 5 | 5 | 6 |
| hot start cost (\$) | 4500 | 5000 | 550 | 560 | 900 |
| cold start cost (\$) | 9000 | 10000 | 1100 | 1120 | 1800 |
| cold start hours (h) | 5 | 5 | 4 | 4 | 4 |
| initial state (h) | +8 | +8 | -5 | -5 | -6 |
| | **Unit 6** | **Unit 7** | **Unit 8** | **Unit 9** | **Unit 10** |
| Pmax (MW) | 80 | 85 | 55 | 55 | 55 |
| Pmin (MW) | 20 | 25 | 10 | 10 | 10 |
| a (\$/MW$^2$h) | 0.00712 | 0.00079 | 0.00413 | 0.00222 | 0.00173 |
| b (\$/MWh) | 22.26 | 27.74 | 25.92 | 27.27 | 27.79 |
| c (\$/h) | 370 | 480 | 660 | 665 | 670 |
| $a_1$ (\$/MW$^2$h) | 0.0022 | 0.003 | 0.004 | 0.0013 | 0.0023 |
| $b_1$ (\$/MWh) | 9.26 | 3.74 | 5.92 | 7.27 | 7.79 |
| $c_1$ (\$/h) | 370 | 480 | 660 | 665 | 670 |
| min up (h) | 3 | 3 | 1 | 1 | 1 |
| min down (h) | 3 | 3 | 1 | 1 | 1 |
| hot start cost (\$) | 170 | 260 | 30 | 30 | 30 |
| cold start cost (\$) | 340 | 520 | 60 | 60 | 60 |
| cold start hours (h) | 2 | 2 | 0 | 0 | 0 |
| initial state (h) | -3 | -3 | -1 | -1 | -1 |

**Table A.1:** Data for the 10 unit problem

| Hour | Load (MW) | Reserve (MW) | Hour | Load (MW) | Reserve (MW) |
|---|---|---|---|---|---|
| 1 | 700 | 70 | 13 | 1400 | 140 |
| 2 | 750 | 75 | 14 | 1300 | 130 |
| 3 | 850 | 85 | 15 | 1200 | 120 |
| 4 | 950 | 95 | 16 | 1050 | 105 |
| 5 | 1000 | 100 | 17 | 1000 | 100 |
| 6 | 1100 | 110 | 18 | 1100 | 110 |
| 7 | 1150 | 115 | 19 | 1200 | 120 |
| 8 | 1200 | 120 | 20 | 1400 | 140 |
| 9 | 1300 | 130 | 21 | 1300 | 130 |
| 10 | 1400 | 140 | 22 | 1100 | 110 |
| 11 | 1450 | 145 | 23 | 900 | 90 |
| 12 | 1500 | 150 | 24 | 800 | 80 |

**Table A.2:** Load and reserve requirements (MW)

## A.2 Case study B

The problem instance named *Case study B* was presented in [Redondo 1999] and will be used in Appendix B. It considers a 24 hour planning horizon and includes 70 thermal and 30 hydro units. This problem partially represents the Spanish electrical generation system and, except for the minimum production level of thermal units, the data presented represents the real parameters of the system.

Tables A.3 and A.4 present the characteristics of the thermal and of the hydro units, respectively. In Table A.3, C and D stand for start up and shut down costs, and E is a multiplicative factor to apply to all costs. $R_s$ and $R_b$ stand for minimum up and down rates. In Table A.4, DownUnit stands for the reservoirs that are downstream from unit $i$, $V_{ini}$ for the reservoirs initial volume, and $V_{max}$ and $V_{min}$, for its maximum and minimum volume, respectively. Finally, $V_{\text{END}_{max}}$ and $V_{\text{END}_{min}}$, stand for a reservoir maximum and minimum final volumes.

Load and reserve requirements are presented in Table A.5.

| Units | initial state (h) | Pmax (MW) | Pmin (MW) | a (Te/\$/MW$^2$h) | b (Te/\$/MWh) | c (Te/\$/h) | C (Te) | D (Te) | E (\$/Te) | $R_s$ (MW) | $R_b$ (MW) | min up (h) | min down (h) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 9 | 325.8 | 89.05 | 1.300 | 1745 | 121591 | 5446153 | 20 | 0.7350 | 144 | 144 | 1 | 1 |
| 2 | 9 | 513.14 | 117.55 | 0.670 | 1724 | 142733 | 8626821 | 20 | 0.7350 | 180 | 180 | 1 | 1 |
| 3 | -5 | 297.92 | 46.35 | 0.560 | 2049 | 40363 | 1159344 | 20 | 2.0560 | 242 | 242 | 1 | 1 |
| 4 | -2 | 297.92 | 46.35 | 0.560 | 2049 | 40363 | 1159344 | 20 | 2.0560 | 242 | 242 | 1 | 1 |
| 5 | -4 | 200.00 | 31.35 | 0.820 | 1969 | 57437 | 1455752 | 20 | 1.2200 | 138 | 138 | 1 | 1 |
| 6 | -2 | 509.01 | 31.35 | 0.710 | 1701 | 150070 | 3200808 | 20 | 1.2200 | 300 | 300 | 1 | 1 |

Table A.3: Data for the thermal generators - *continued*

| Units | initial state (h) | Pmax (MW) | Pmin (MW) | a (Te/\$/MW$^2$h) | b (Te/\$/MWh) | c (Te/\$/h) | C (Te) | D (Te) | E (\$/Te) | R$_s$ (MW) | R$_b$ (MW) | min up (h) | min down (h) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 7 | 1 | 888.72 | 443.60 | 0 | 1000 | 0 | 2914194 | 99 | 1 | 120 | 120 | 170 | 1 |
| 8 | 1 | 888.72 | 443.60 | 0 | 1000 | 0 | 2914194 | 99 | 1 | 120 | 120 | 170 | 1 |
| 9 | 2 | 330.75 | 79.25 | 0.880 | 1864 | 93450 | 7043750 | 20 | 0.62220 | 168 | 168 | 1 | 1 |
| 10 | 1 | 888.15 | 442.16 | 0 | 1000 | 0 | 2914194 | 99 | 1 | 120 | 120 | 170 | 1 |
| 11 | 1 | 888.15 | 442.16 | 0 | 1000 | 0 | 2914194 | 99 | 1 | 120 | 120 | 170 | 1 |
| 12 | 2 | 522.50 | 85.50 | 0.100 | 2029 | 115837 | 5522987 | 20 | 0.9510 | 356 | 356 | 1 | 1 |
| 13 | -2 | 285.00 | 34.20 | 0.910 | 1913 | 61259 | 1803597 | 20 | 1.3030 | 180 | 180 | 1 | 1 |
| 14 | -2 | 517.32 | 66.85 | 0.300 | 2042 | 100894 | 2451637 | 20 | 1.6770 | 392 | 392 | 1 | 1 |
| 15 | 2 | 517.32 | 66.85 | 0.300 | 2042 | 100894 | 2451637 | 20 | 1.6770 | 392 | 392 | 1 | 1 |
| 16 | 1 | 945.45 | 455.53 | 0.000 | 1000 | 0 | 3100206 | 99 | 1 | 120 | 120 | 170 | 1 |
| 17 | 2 | 131.13 | 30.22 | 13.160 | -213 | 16073 | 3267956 | 20 | 0.6200 | 70 | 70 | 1 | 1 |
| 18 | 2 | 131.13 | 32.08 | 2.030 | 190 | 64534 | 3225090 | 20 | 0.6200 | 65 | 65 | 1 | 1 |
| 19 | 2 | 311.85 | 74.65 | 2.140 | 1106 | 202574 | 6734233 | 20 | 0.6200 | 160 | 160 | 1 | 1 |
| 20 | 2 | 330.75 | 81.75 | 2.490 | 1030 | 185463 | 6907227 | 20 | 0.6200 | 165 | 165 | 1 | 1 |
| 21 | 2 | 330.75 | 81.75 | 1.690 | 1332 | 164434 | 6907227 | 20 | 0.6200 | 165 | 165 | 1 | 1 |
| 22 | -9 | 274.55 | 47.5 | 0 | 2380 | 32000 | 628593 | 20 | 2.0780 | 96 | 96 | 1 | 1 |
| 23 | -2 | 274.55 | 47.5 | 0.640 | 2019 | 69474 | 1074505 | 20 | 2.0780 | 178 | 178 | 1 | 1 |
| 24 | 2 | 141.60 | 35.40 | 2.650 | 1942 | 68064 | 3771370 | 20 | 0.6070 | 74 | 74 | 1 | 1 |
| 25 | -3 | 496.60 | 38.20 | 0.510 | 1872 | 103819 | 3131582 | 20 | 1.3280 | 235 | 235 | 1 | 1 |
| 26 | -7 | 133.94 | 33.94 | 0.690 | 2520 | 22714 | 2870456 | 20 | 0.7080 | 74 | 74 | 1 | 1 |
| 27 | 2 | 330.75 | 70.90 | 0.960 | 1858 | 88924 | 6322696 | 20 | 0.7080 | 191 | 191 | 1 | 1 |
| 28 | 1 | 152.80 | 75.92 | 0 | 1000 | 0 | 501043 | 99 | 1 | 40 | 40 | 170 | 1 |
| 29 | -4 | 144.15 | 34.41 | 8.150 | 528 | 135203 | 3009552 | 20 | 0.7390 | 60 | 60 | 1 | 1 |
| 30 | 2 | 330.75 | 85.05 | 1.650 | 1476 | 179304 | 5941907 | 20 | 0.7390 | 161 | 161 | 1 | 1 |
| 31 | 2 | 522.50 | 85.50 | 0.480 | 1864 | 100532 | 5440091 | 20 | 0.9770 | 276 | 276 | 1 | 1 |
| 32 | 2 | 519.75 | 123.80 | 1.300 | 1709 | 301548 | 9999999 | 20 | 0.7100 | 276 | 276 | 1 | 1 |
| 33 | -2 | 60.45 | 16.27 | -9.580 | 2401 | 27629 | 1364325 | 20 | 0.6820 | 24 | 24 | 1 | 1 |
| 34 | -3 | 143.22 | 37.66 | 13.500 | -663 | 178234 | 3050325 | 20 | 0.682 | 64 | 64 | 1 | 1 |
| 35 | 2 | 330.75 | 100.65 | 1.180 | 1710 | 115704 | 6006173 | 20 | 0.6820 | 121 | 121 | 1 | 1 |
| 36 | -2 | 202.23 | 49.61 | 1.990 | 1875 | 56816 | 2096997 | 20 | 1.0160 | 105 | 105 | 1 | 1 |
| 37 | 2 | 330.75 | 108.70 | 3.090 | 1405 | 213279 | 6398451 | 20 | 0.6160 | 111 | 111 | 1 | 1 |
| 38 | 2 | 330.75 | 108.70 | 3.090 | 1405 | 213279 | 6398451 | 20 | 0.6160 | 111 | 111 | 1 | 1 |
| 39 | 2 | 330.75 | 108.70 | 3.090 | 1405 | 213279 | 6398451 | 20 | 0.6160 | 111 | 111 | 1 | 1 |
| 40 | 2 | 330.75 | 108.70 | 3.090 | 1405 | 213279 | 6398451 | 20 | 0.6160 | 111 | 111 | 1 | 1 |
| 41 | 2 | 199.10 | 36.2 | 2.370 | 1511 | 121064 | 4180708 | 20 | 0.7300 | 131 | 131 | 1 | 1 |
| 42 | 2 | 283.08 | 67.9 | 0.400 | 2224 | 47797 | 5083092 | 20 | 0.7550 | 150 | 150 | 1 | 1 |
| 43 | 2 | 251.10 | 63.25 | 1.890 | 1536 | 101589 | 4524810 | 20 | 0.7430 | 60 | 60 | 1 | 1 |
| 44 | 2 | 330.75 | 100.65 | 1.840 | 1403 | 166427 | 5524829 | 20 | 0.7430 | 121 | 121 | 1 | 1 |
| 45 | -4 | 332.50 | 43.70 | 1.470 | 1618 | 120095 | 2532752 | 20 | 1.8430 | 235 | 235 | 1 | 1 |
| 46 | -3 | 358.40 | 66.50 | 0.240 | 1989 | 109896 | 1443773 | 20 | 1.8350 | 222 | 222 | 1 | 1 |
| 47 | -2 | 517.29 | 66.85 | 0.180 | 2124 | 99450 | 2266055 | 20 | 1.8350 | 300 | 300 | 1 | 1 |
| 48 | 2 | 141.60 | 35.40 | 5.080 | 1118 | 96248 | 3534072 | 20 | 0.6440 | 75 | 75 | 1 | 1 |
| 49 | -5 | 62.85 | 20.92 | 16.330 | 1374 | 47399 | 1224070 | 20 | 0.7200 | 15 | 15 | 1 | 1 |
| 50 | 2 | 236.22 | 71.60 | 0 | 2425 | 14807 | 4226747 | 20 | 0.7200 | 69 | 69 | 1 | 1 |
| 51 | 2 | 330.75 | 81.25 | 1.400 | 1409 | 136476 | 6024431 | 20 | 0.7200 | 165 | 165 | 1 | 1 |
| 52 | -4 | 325.50 | 46.50 | 0.550 | 2094 | 73451 | 1239037 | 20 | 2.0050 | 235 | 235 | 1 | 1 |
| 53 | 2 | 330.75 | 99.25 | 1.980 | 1553 | 135658 | 6541614 | 20 | 0.7080 | 132 | 132 | 1 | 1 |
| 54 | 2 | 330.75 | 99.25 | 1.980 | 1553 | 135658 | 6541614 | 20 | 0.7080 | 132 | 132 | 1 | 1 |

Table A.3: Data for the thermal generators - *continued*

| Units | initial state (h) | Pmax (MW) | Pmin (MW) | a (Te/\$/MW$^2$h) | b (Te/\$/MWh) | c (Te/\$/h) | C (Te) | D (Te) | E (\$/Te) | R$_s$ (MW) | R$_b$ (MW) | min up (h) | min down (h) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 55 | 2 | 330.75 | 99.25 | 1.980 | 1553 | 135658 | 6541614 | 20 | 0.7080 | 132 | 132 | 1 | 1 |
| 56 | 1 | 1018.03 | 491.35 | 0 | 1000 | 0 | 3275283 | 99 | 1 | 120 | 120 | 170 | 1 |
| 57 | 1 | 958.82 | 462.22 | 0 | 1000 | 0 | 3144047 | 99 | 1 | 120 | 120 | 170 | 1 |
| 58 | 9 | 430.00 | 215.00 | 0 | 1000 | 0 | 1440500 | 99 | 1 | 430 | 430 | 1 | 1 |
| 59 | -2 | 80.00 | 20.00 | 0 | 2299 | 20057 | 8193762 | 20 | 0.1980 | 40 | 40 | 1 | 1 |
| 60 | -2 | 272.00 | 47.00 | 0.630 | 2020 | 68566 | 1074505 | 20 | 2.0780 | 178 | 178 | 1 | 1 |
| 61 | -2 | 329.00 | 47.00 | 0.550 | 2094 | 73451 | 1239037 | 20 | 2.0050 | 235 | 235 | 1 | 1 |
| 62 | -2 | 329.00 | 47.00 | 0.550 | 2204 | 73451 | 1211839 | 20 | 2.0050 | 235 | 235 | 1 | 1 |
| 63 | -2 | 310.00 | 75.00 | 0.450 | 2006 | 93919 | 7504813 | 20 | 0.3430 | 150 | 150 | 1 | 1 |
| 64 | -2 | 65.00 | 9.50 | 0 | 2780 | 15000 | 311543 | 20 | 2.0780 | 46 | 46 | 1 | 1 |
| 65 | -2 | 65.00 | 9.50 | 0 | 2780 | 15000 | 310343 | 20 | 2.0780 | 46 | 46 | 1 | 1 |
| 66 | -2 | 162.00 | 28.50 | 0 | 2480 | 30000 | 650124 | 20 | 2.0540 | 105 | 105 | 1 | 1 |
| 67 | -2 | 162.00 | 28.50 | 0 | 2480 | 30000 | 650124 | 20 | 2.0540 | 105 | 105 | 1 | 1 |
| 68 | -2 | 66.00 | 15.00 | 0 | 2780 | 15000 | 388349 | 20 | 1.5750 | 36 | 36 | 1 | 1 |
| 69 | -2 | 140.00 | 15.00 | 0 | 2480 | 30000 | 809718 | 20 | 1.5750 | 100 | 100 | 1 | 1 |
| 70 | -2 | 150.00 | 20.00 | 0 | 2480 | 30000 | 835390 | 20 | 1.5750 | 110 | 110 | 1 | 1 |

Table A.3: Data for the thermal generators

| Units | Down Unit | V$_{ini}$ (Hm$^3$) | V$_{max}$ (Hm$^3$) | V$_{min}$ (Hm$^3$) | V$_{\text{END}_{max}}$ (Hm$^3$) | V$_{\text{END}_{min}}$ (Hm$^3$) |
|---|---|---|---|---|---|---|
| 1 | 3 | 40.740 | 99.90 | 0 | 41.8800 | 35.4000 |
| 2 | 3 | 40.460 | 99.10 | 0 | 41.52 | 35.368 |
| 3 | 4 | 40.230 | 99.38 | 0 | 41.2700 | 35.3000 |
| 4 | - | 0 | 0 | 0 | 0 | 0 |
| 5 | 6 | 169.000 | 281.60 | 0 | 170.8000 | 164.2000 |
| 6 | 7 | 4.632 | 7.72 | 0 | 5.5584 | 1.0056 |
| 7 | 8 | 4.512 | 7.52 | 0 | 5.4144 | 1.0096 |
| 8 | 9 | 0.426 | 0.71 | 0 | 0.6112 | 0.2408 |
| 9 | 19 | 0.510 | 0.85 | 0 | 0.6120 | 0.3080 |
| 10 | 11 | 73.200 | 122.00 | 0 | 74.8400 | 68.5600 |
| 11 | 19 | 3.792 | 6.32 | 0 | 5.5504 | 1.0336 |
| 12 | 13 | 1.200 | 2.00 | 0 | 1.8400 | 0.5600 |
| 13 | 14 | 22.680 | 37.80 | 0 | 23.2160 | 15.1440 |
| 14 | 16 | 2.610 | 4.35 | 0 | 4.1320 | 1.1880 |
| 15 | 16 | 283.800 | 473.00 | 0 | 285.5600 | 274.0400 |
| 16 | 20 | 126.000 | 210.00 | 0 | 128.2000 | 117.8000 |
| 17 | 18 | 31.500 | 52.50 | 0 | 33.8000 | 23.2000 |
| 18 | 20 | 1.662 | 2.77 | 0 | 1.7944 | 0.8296 |
| 19 | 21 | 2.526 | 4.21 | 0 | 3.0312 | 1.0208 |
| 20 | 21 | 4.758 | 7.93 | 0 | 5.7096 | 1.0064 |
| 21 | 22 | 3.660 | 6.10 | 0 | 5.3920 | 2.0280 |
| 22 | 23 | 116.800 | 194.80 | 0 | 118.1600 | 108.1600 |
| 23 | 26 | 1.500 | 2.50 | 0 | 2.0000 | 0.9000 |
| 24 | 25 | 363.600 | 606.00 | 0 | 364.32 | 354.8800 |
| 25 | 26 | 89.700 | 149.50 | 0 | 90.64 | 81.7600 |

Table A.4: Data for the hydro generators - *continued*

| Units | Down Unit | $V_{ini}$ ($Hm^3$) | $V_{max}$ ($Hm^3$) | $V_{min}$ ($Hm^3$) | $V_{END_{max}}$ ($Hm^3$) | $V_{END_{min}}$ ($Hm^3$) |
|---|---|---|---|---|---|---|
| 26 | 27 | 4.002 | 6.67 | 0 | 4.8024 | 1.0016 |
| 27 | 29 | 24.500 | 40.84 | 0 | 25.4000 | 15.6000 |
| 28 | 29 | 1.200 | 2.00 | 0 | 1.8400 | 0.5600 |
| 29 | 0 | 10.080 | 16.80 | 0 | 11.0960 | 5.0640 |
| 30 | 0 | 2000.000 | 3000.00 | 0 | 3000.0000 | 1500.8000 |

Table A.4: Data for the hydro generators

| Hour | Load | Reserve | Hour | Load | Reserve | Hour | Load | Reserve | Hour | Load | Reserve |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 16742 | 1674.2 | 13 | 20241 | 2024.1 | 25 | 16330 | 1633.0 | 37 | 20221 | 2022.1 |
| 2 | 15212 | 1521.2 | 14 | 19931 | 1993.1 | 26 | 15059 | 1505.9 | 38 | 20057 | 2005.7 |
| 3 | 14220 | 1422.0 | 15 | 19326 | 1932.6 | 27 | 14073 | 1407.3 | 39 | 19342 | 1934.2 |
| 4 | 13723 | 1372.3 | 16 | 19015 | 1901.5 | 28 | 13569 | 1356.9 | 40 | 18955 | 1895.5 |
| 5 | 13394 | 1339.4 | 17 | 19294 | 1929.4 | 29 | 13384 | 1338.4 | 41 | 19114 | 1911.4 |
| 6 | 13422 | 1342.2 | 18 | 19619 | 1961.9 | 30 | 13388 | 1338.8 | 42 | 19484 | 1948.4 |
| 7 | 14183 | 1418.3 | 19 | 19586 | 1958.6 | 31 | 14288 | 1428.8 | 43 | 19451 | 1945.1 |
| 8 | 15671 | 1567.1 | 20 | 18918 | 1891.8 | 32 | 15160 | 1516.0 | 44 | 19046 | 1904.6 |
| 9 | 16407 | 1640.7 | 21 | 18391 | 1839.1 | 33 | 16389 | 1638.9 | 45 | 18581 | 1858.1 |
| 10 | 17706 | 1770.6 | 22 | 18690 | 1869.0 | 34 | 17939 | 1793.9 | 46 | 19391 | 1939.1 |
| 11 | 19120 | 1912.0 | 23 | 18844 | 1884.4 | 35 | 19107 | 1910.7 | 47 | 19193 | 1919.3 |
| 12 | 20021 | 2002.1 | 24 | 17632 | 1763.2 | 36 | 19760 | 1976.0 | 48 | 17763 | 1776.3 |

**Table A.5:** Load and reserve requirements (MW)

# Appendix B

# Integrated hydro and thermal scheduling with Constraint Oriented Neighbourhoods

In this Appendix we present some preliminary work on the application of *Constraint Oriented Neighbourhoods* to hydrothermal scheduling. Although not conclusive, it is, in our opinion, a good starting point for further research.

In a hydrothermal system a proper coordination between the hydro and the thermal systems is required to minimise the system total production costs. This is generally achieved by decomposing the base problem in two subproblems, one handling the hydro system and the other handling the thermal one. Usually, the hydro scheduling subproblem is solved first and then, for the remaining demand (the total load demand minus the hydro production) the thermal subproblem is solved. The process is repeated until the thermal marginal costs or hydro generation converge.

Metaheuristics approaches dealing with the Hydrothermal Coordination problem (HTC) do also follow this reasoning: the metaheuristic tackles the combinatorial thermal problem, while other techniques are used to solve the hydro subproblem. Hierarchical approaches may however lead to suboptimal results, due to lack of convergence of the iterative approach and, in spite of that, an integrated approach where both systems are tackled simultaneously is desirable.

This section proposes an integrated approach for solving the HTC with metaheuristics.

This is achieved by discretising the production levels of the hydro units and defining appropriate neighbourhood movements that allow changes to be made both in the thermal and in the hydro units. As the approach is based on the *Constraint Oriented Neighbourhoods* paradigm introduced in Chapter 5, different neighbourhood movements are designed to be applied during the search process, according to the type of constraints that are violated in each iteration.

The section is structured as follows. The structure of the solution representation of the HTC problem is first described. Then an algorithm to obtain initial feasible solutions is presented. The section proceeds with a description of the first neighbourhood movements designed to solve the problem and an assessment of these movements is performed, showing their unsatisfactory performance and thus justifying further research on the topic.

## B.1 HTC solution representation

The solution structure for the HTC will be divided in three levels. Two levels will represent the thermal problem, using the binary and integer solution representations proposed in Chapter 4, and a third level will represent the hydro units. As the length of the string representing the integer coding of thermal units varies with the number of state transitions, the integer codification is represented in a separate data structure. Still, they are managed in parallel.

The discretisation of hydro units production is achieved by setting a maximum number of possible levels of production and, accordingly, selecting a level for each hydro unit. If the maximum number of levels is set to n, the output of a unit that is at production level l ($l \leq n$) will be $\frac{l}{n}$ of its maximum output.

Figure B.1 presents a solution for a system with I thermal units and P hydro units, for a planning horizon of T periods, when the maximum number of hydro production levels is set to 4, a 0 meaning that the unit is not discharging.

## B.2 Building feasible initial solutions

The construction phase algorithm for the HTC problem keeps most of the particularities of the algorithm described in Chapter 4 for the thermal problem, and includes some further procedures to handle the hydro units.

Again, the sum of the maximum production capacity of the thermal units whose state

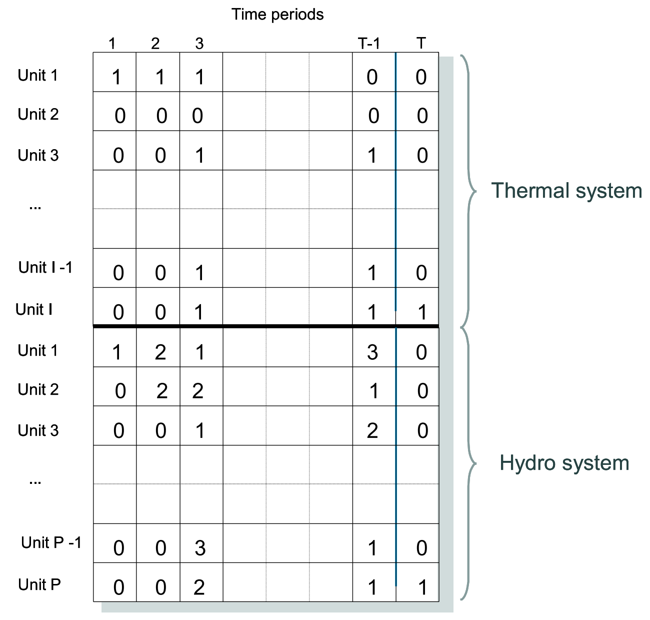


**Figure B.1:** Hydro-thermal solution representation

had to be fixed to on, due to specific problem constraints, is first computed (prodT in Figure B.2). The maximum production capacity of must-run hydro units, as well as that of the units whose corresponding reservoir level is currently above its maximum final level is also computed (prodH). If the units that are set to produce are capable of satisfying the load and reserve requirements on their own, no further calculations are needed for that period. Otherwise, a *Restricted Candidate List* containing a set of candidate hydro and thermal units is generated and units are selected from that list until all requirements are reached for that period. By then, the thermal operating levels that lead to minimum operating costs are set and the solution current cost is computed. The MakeRCLHTC procedure guarantees that the solution will remain feasible.

## B.2.1 The MakeRCLHTC procedure

The MakeRCLHTC procedure (Figure B.3) works as follows. First, the current thermal and hydro production (prodT and prodH, respectively) are computed. Two intermediate *Restricted Candidate Lists* are then built, RCLThermal storing the candidate thermal units and built according to the procedure described in section 4.2.2, and RCLHydro storing all the hydro units that have not been committed yet (the must-run ones and those whose

**Algorithm ConstructionHTC**

```
For t = 1 to T
    SwitchOnUnitsThermal(mustRun, minUpTime)
    SwitchOnUnitsHydro(mustRun, aboveEndLevel)
    prodT = CheckCurrentProductionThermal($P_{max}$)
    prodH = CheckCurrentProductionHydro($P_h$)
    if prodT + prodH < load + reserve
        MakeRCLHTC(Solution)
    CalculateDispatchValues(Solution)
    CalculateCurrentCost(Solution)
```

**Figure B.2:** Building an initial solution for the HTC

current reservoir level was above its maximum final level are therefore excluded from that set) and that are allowed to discharge in the current period of study. The next operation is to compute the sum of the maximum production of the thermal and of the hydro units within those lists (prodTrcl and prodHrcl, respectively). If the current production and the maximum production of the units belonging to the intermediate lists is below demand and reserve requirements, all units must be switched on and the thermal units that initially were not candidates to be switched on are studied (let us recall the definition given in Chapter 4 for candidate and no-candidate thermal units: candidate thermal units are those that if switched on in period $t$, will satisfy the minimum down-time constraints, while no-candidate thermal units are those that, if chosen, must be switched on since the last period they were on until period $t$ is reached, to satisfy minimum down-time constraints). Otherwise, if the sum is above the required value, an iterative procedure is started, running until that value is reached. In this process, the units in RCLThermal and RCLHydro are ranked, according to their associated *greedy functions*, those reaching a certain threshold being stored in the final RCL (RCL = Achieve(RCLThermal, RCLHydro, $\alpha$)) and excluded from RCLThermal and RCLHydro (UpDate(RCLThermal, RCLHydro)). Units are then randomly chosen from RCL until there are no further units in that list or load and reserve requirements have been reached. If these requirements have not been reached, the value of $\alpha$ is incremented and the process is repeated.

Expression (B.1) defines the *greedy function* $gfh(i)$ for a hydro unit $i$, in a generic period of time. $resLevel(i)$ represents the current level of the reservoir connected to hydro unit $i$ and $minEndLevel(i)$ its minimum allowed level at the end of the planning horizon. For reservoirs closer to their minimum end level, the corresponding hydro unit will get a higher value and,

**Algorithm MakeRCLHTC**

```
prodT = CheckCurrentProductionThermal($P_{max}$)
prodH = CheckCurrentProductionHydro($P_h$)
RCLThermal  = MakeRCLThermal()
RCLHydro = MakeRCLHydro()
prodTrcl = CalcProdT(RCLThermal)
prodHrcl = CalcProdH(RCLHydro)
if prodT + prodH + prodTrcl  + prodHrcl < load + reserve
    SwitchOnAllUnitsRCL()
    SwitchOnUnits (ThermalNoCandidates)
else
    Initialise($\alpha$)
    prodTrcl = prodHrcl = 0
    While prodT + prodH + prodTrcl  + prodHrcl < load + reserve and RCL $\neq$ {}
        RCL = Achieve(RCLThermal, RCLHydro, $\alpha$)
        Update(RCLThermal, RCLHydro)
        While prodT + prodH + prodTrcl  + prodHrcl < load + reserve and RCL $\neq$ {}
            RandomChooseUnit(RCL)
        Update($\alpha$)
```

**Figure B.3:** Algorithm MakeRCL in HTC

consequently, will be less prone of being selected if the initial value of $\alpha$ is sufficiently low (see expression B.2). The *greedy function* for thermal units remains the one defined in section 4.6.

$$gfh(i) = \frac{1}{resLevel(i) - minEndLevel(i)} \quad \forall t \tag{B.1}$$

$$\underline{gfh_t} \ \le \ gfh(i) \ \le \ \underline{gfh_t} \ + \ \alpha \, ( \, \overline{gfh_t} - \underline{gfh_t}) \tag{B.2}$$

## B.3 CON applied to the HTC problem

The base structure of the CON algorithm for the HTC problem is presented in Figure B.4. As for the thermal problem, presented in Chapter 5, we build a set containing those units that are on in period t (UnitsT), and that are at their minimum production level at the point of transition from off to on ($t_{left}$), or at the point of transition from on to off ($t_{right}$). If there are no units satisfying those conditions, a new attempt is made to build a set of units that are on in period t and that are not at their maximum production level in $t_{left}$ or $t_{right}$. A set (UnitsH) containing the hydro units that are off in period t, or that are not at their maximum turbination level, is also built. It is then decided whether changes will be applied to hydro or thermal units and, if the thermal ones are selected, the procedure presented in Figure 5.6, in Chapter 5, is used. Otherwise, if the hydro option is chosen, a different procedure that considers neighbourhood movements specifically designed to tackle hydro constraints is applied. For peak periods, when the demand is above a given percentage of the maximum demand (MaxLoad), the procedure is consecutively applied to the units in UnitsH, until there are no further units left. For the other periods a single unit is selected and the operations are performed only for that unit.

### B.3.1 Neighbourhood movements

Hydro units are handled here in a similar way to the case of thermal units. For a given solution, why are some units off or not producing at their maximum production levels? Several reasons may exist: 1) their reservoirs may have reached the minimum allowed level; 2) the downstream reservoirs may have reached their maximum allowed level, or 3) there would be a system overload if hydro production was increased (i.e. even if the thermal units were producing at their minimum, the total hydro and thermal production would go over the demand level). Therefore, depending on the reason that restricts a unit's hydro production, it might be advisable to apply different changes in a solution, this reasoning leading, again, to the development of different neighbourhood structures.

In this section we propose an approach based on *Constraint Oriented Neighbourhoods* to solve the HTC. It considers five distinct neighbourhood movements: three directly related to thermal units, and two related to hydro units. Concerning the thermal units movements, they remain the same developed in Chapter 5, namely Load Movement, Reserve Movement and Up-time Movement. The movements related to hydro units aim at increasing a generator's hydro production in a given period of time – IncreaseHydro Movement – and at transferring water resources from one period to another – TransferHydroProd Movement. They are described in detail below.

*IncreaseHydro Movement*

The *IncreaseHydro Movement* is applied whenever an increase in a generator hydro production is not forbidden neither due to its current reservoir level, nor due to downstream reservoir levels or system overload, although the maximum increase that is allowed is constrained. So, this movement computes the maximum increase in the generators hydro pro-

**Algorithm CON-HTC**

```
number solutions = 0
Repeat
    i = 0
    Build an initial solution $X_0$
    Repeat
        Randomly choose t
        UnitsT = Obtain UnitsT $P_{min}(t)$
        if  UnitsT = {}
            UnitsT = Obtain UnitsT P(t)
        UnitsH = Obtain UnitsH(t)
        RandomChoose(Hydro, Thermal)
        if Thermal
            RandomChooseUnit(UnitsT)
            $t_1$ = Randomly choose($t_{left}$, $t_{right}$)
            $x_{ab}$ = Remove element($X_i$, $t_1$, $i$)
            Check constraint violation Thermal($t_1$)
            Select neighbourhood movement Thermal($t_1$)
            $X_{i+1}$ = Apply movement Thermal($X_i \setminus x_{ab}$, $t_1$ )
            Accept (or not) $X_{i+1}$ as the current solution,
            according to the metaheuristics rules
        else
            if load(t) $\geq$ % $\times$ MaxLoad
                While UnitsH $\neq$ {}
                    ger = RandomChooseUnit(UnitsH)
                    Check constraint violation Hydro(ger, t)
                    Select neighbourhood movement Hydro(ger, t)
                    $X_{i+1}$ = Apply movement Hydro(t, MovementDone)
                    Accept (or not) $X_{i+1}$ as the current solution,
                    according to the metaheuristics rules
            else
                    ger = RandomChooseUnit(UnitsH)
                    Check constraint violation Hydro(ger, t)
                    Select neighbourhood movement Hydro(ger, t)
                    $X_{i+1}$ = Apply movement Hydro($X_i \setminus x_{ab}$, t)
                    Accept (or not) $X_{i+1}$ as the current solution,
                    according to the metaheuristics rules
        i++
    Until Stopping criterion is met
    number solutions++
Until number solutions = maximum number solutions
```

**Figure B.4:** Algorithm CON for the HTC problem

duction and increases it of, at most, that value.

For a given generator *ger* and for a specific period *t*, the algorithm (Figure B.5) first calculates the maximum increase that is allowed in the reservoir volume – maxVarVolume – so that the water discharges that have been scheduled from *t* until the end of the planning horizon can still be made. As the volume of discharge is also constrained by the level of the reservoirs that are downstream, CheckDownstreamReservoirs(ger, t) computes the maximum volume of water that *ger* may discharge, so that the levels of the downstream reservoirs do not go over their maximum. The minimum of these two volumes is kept as the current maxVarVolume value. Another constraint to the increase of water discharge is a possible system overload, checked in CheckSystemOverload(ger, t). Finally, maxVarVolume is constrained by the technical characteristics of the generator.

As soon as maxVarVolume is definitely set, the maximum increase in the turbination level of the generator and, consequently, the maximum increase in its production level – turbMaxGer and PhMaxGer, respectively – are computed. However, to avoid premature convergence the production level of the generator is not necessarily set to its maximum, and one rather chooses randomly an integer value – levelProdh – between 0 and the maximum number of possible levels of production (hLevels), introduced in section B.1. Instead of considering that hydro production may vary from 0 to its maximum technical level, it is now considered that it may vary from its current level ($\mathrm{P}_h$) to $\mathrm{P}_h$ + PhMaxGer. Thus, a 0 levelProdh reflects no changes in the current production of the hydro unit, while if levelProdh is set to l, it will represent an increase of $\frac{l}{n} \times PhMaxGer$ in the hydro power output.

After setting the current level of production, PhGer, the system reservoirs level are updated and, to conclude, the algorithm checks if it is possible to switch off any thermal units in period t.

*TransferHydroProd Movement*

The *TransferHydroProd Movement* is applied whenever an increase in hydro production is not constrained by the levels of the downstream reservoirs or system overload, but is constrained by its current reservoir level. By transferring water resources from one period to another, the movement allows an increase in the production of a hydro generator in period t, by reducing it in another period.

For a given generator *ger* and for a specific period t, the algorithm (Figure B.6) first finds a period t1 where *ger* is producing. It then computes the maximum volume of water that can be transferred from t1 to t – maxVarVolume – constrained by the reservoir's maximum

```
Algorithm IncreaseHydro

maxVarVolume = CheckFutureDischarges(ger, t)
maxVarVolume = min(maxVarVolume, CheckDownstreamReservoirs(ger, t))
maxVarVolume = min(maxVarVolume, CheckSystemOverload(ger, t))
maxVarVolume = min(maxVarVolume, CheckMaxTurb(ger))
turbMaxGer = maxVarVolume $\times 10^6$ / 3600
PhMaxGer = turbMaxGer $\times \rho$
levelProdh = RandomChoose(hLevels)
PhGer = PhGer + turbMaxGer $\times \rho \times$ levelProdh / hLevels
UpdateReservoirLevels()
CheckSwitchOffThermal(t)
```

**Figure B.5:** Algorithm IncreaseHydro Movement

```
Algorithm TransferHydroProd

$t_1$ = FindProductionPoint(ger, t)
maxVarVolume = CalcMaxWaterTransfer($t_1$, t)
CheckLevelDown()
maxHydroDecrease = CalcMaxDecrease($t_1$)
maxVarVolume = min(maxVarVolume, maxHydroDecrease)
If maxVarVolume > 0
    maxHydroDecrease = min(maxHydroDecrease, $t_1$, Ph)
    maxHydroIncrease = NoOverload(t)
    IncreaseHydroProd(t)
    ReduceHydroProd($t_1$)
    UpdateReservoirLevels(ger)
    CheckSwitchOffThermal(t)
```

**Figure B.6:** Algorithm TransferHydroProd Movement

level, if $t_1 < t$, and by its minimum level if $t_1 > t$. After checking the downstream reservoirs maximum levels, CheckLevelDown(), the maximum hydro decrease that will not force any thermal units to be switched off is computed – maxHydroDecrease. If it is possible to transfer hydro production from $t_1$ to t (maxVarVolume $> 0$), maxHydroDecrease is updated so that it will never go over the current production of *ger* in $t_1$ (Ph) and it does not lead to a system overload in t. After increasing the production in t and decreasing it in $t_1$, the reservoir levels are updated and, if possible, some thermal units are switched off in period t.

## B.4 Computational experience

To evaluate its adequacy, this approach has been used to solve a HTC problem from the literature [Redondo 1999], with 70 thermal and 29 hydro units, described in Appendix A. The results of some computational experiments show that, although robust, i.e. the quality of the results did not strongly vary with changes in the metaheuritiscs parameters, the algorithm was not capable of reaching good quality solutions, when compared to the best solution presented in [Redondo 1999]. Therefore, they will not be discussed in this section.

An analysis of some typical results, and a comparison between the production levels of the hydro units before and after the optimisation, will however be done. It clearly shows that there was not a strong difference between the two solutions. In fact, they were extremely similar, leading us to believe that the movements that were designed to apply to the hydro units were not capable of correctly exploring the solution space.

Figure B.7 presents the production levels of the 29 hydro units, for a 24 hour horizon, before being optimised, while Figure B.8 represents the solution obtained after applying the *Constraint Oriented Neighbourhoods* to the initial solution. Finally, Figure B.9 highlights the differences between the two solutions. A 0 means that the production level of hydro unit i in period t has not changed during the search process, positive values indicate an increase in hydro production from the initial to the final solution and a negative value indicates a decrease. As clearly shown, the impact of the hydro movements was low, in most cases the production remaining unchanged. Further research is therefore necessary to design neighbourhood movements that are capable of correctly analyse and evaluate alternative discharge values of the hydro units.

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 72 | 72 | 72 | 72 | 72 | 72 | 72 | 72 | 72 | 13 | 56 | 72 | 69 | 43 | 72 | 72 | 0 | 13 | 72 | 43 | 43 | 72 | 13 | 72 |
| 2 | 26 | 26 | 26 | 26 | 26 | 26 | 26 | 26 | 26 | 26 | 26 | 26 | 26 | 26 | 26 | 26 | 26 | 26 | 26 | 26 | 26 | 26 | 26 | 26 |
| 3 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 |
| 4 | 72 | 72 | 72 | 20 | 72 | 8 | 72 | 72 | 72 | 72 | 22 | 5 | 72 | 72 | 36 | 69 | 72 | 0 | 10 | 72 | 72 | 72 | 0 | 72 |
| 5 | 130 | 130 | 0 | 82 | 25 | 63 | 88 | 130 | 61 | 39 | 119 | 0 | 0 | 0 | 0 | 38 | 95 | 129 | 130 | 130 | 95 | 48 | 44 | 95 |
| 6 | 0 | 13 | 7 | 6 | 7 | 0 | 7 | 7 | 0 | 0 | 0 | 16 | 0 | 13 | 10 | 31 | 9 | 36 | 22 | 22 | 11 | 36 | 30 | 36 |
| 7 | 7 | 7 | 0 | 7 | 0 | 5 | 0 | 0 | 7 | 9 | 1 | 0 | 24 | 6 | 0 | 20 | 9 | 38 | 16 | 30 | 18 | 10 | 35 | 33 |
| 8 | 0 | 0 | 2 | 0 | 2 | 0 | 2 | 2 | 0 | 0 | 3 | 8 | 0 | 6 | 3 | 10 | 7 | 0 | 13 | 9 | 6 | 6 | 2 | 13 |
| 9 | 64 | 64 | 64 | 64 | 64 | 64 | 64 | 64 | 64 | 64 | 64 | 64 | 64 | 64 | 64 | 64 | 64 | 64 | 64 | 64 | 64 | 64 | 64 | 64 |
| 10 | 50 | 50 | 50 | 50 | 50 | 50 | 50 | 50 | 50 | 50 | 50 | 50 | 50 | 50 | 50 | 50 | 50 | 50 | 50 | 50 | 50 | 50 | 50 | 50 |
| 11 | 18 | 18 | 18 | 0 | 0 | 0 | 0 | 18 | 18 | 18 | 18 | 18 | 18 | 18 | 18 | 18 | 18 | 18 | 18 | 18 | 18 | 18 | 18 | 18 |
| 12 | 18 | 18 | 18 | 18 | 18 | 18 | 18 | 18 | 18 | 18 | 18 | 18 | 18 | 18 | 18 | 18 | 18 | 18 | 18 | 18 | 18 | 18 | 18 | 18 |
| 13 | 58 | 0 | 58 | 58 | 58 | 58 | 58 | 0 | 58 | 58 | 58 | 58 | 58 | 24 | 0 | 58 | 0 | 58 | 58 | 24 | 0 | 58 | 58 | 58 |
| 14 | 258 | 258 | 258 | 258 | 258 | 258 | 118 | 156 | 156 | 251 | 118 | 258 | 258 | 0 | 0 | 258 | 156 | 31 | 258 | 0 | 156 | 118 | 118 | 258 |
| 15 | 234 | 234 | 234 | 234 | 234 | 234 | 234 | 234 | 234 | 234 | 234 | 234 | 234 | 234 | 234 | 234 | 234 | 234 | 234 | 234 | 234 | 234 | 234 | 234 |
| 16 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 |
| 17 | 36 | 36 | 36 | 36 | 36 | 36 | 36 | 36 | 36 | 36 | 36 | 36 | 36 | 36 | 36 | 36 | 36 | 36 | 36 | 36 | 36 | 36 | 36 | 36 |
| 18 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 |
| 19 | 38 | 38 | 38 | 38 | 38 | 38 | 38 | 38 | 38 | 38 | 38 | 38 | 38 | 38 | 38 | 38 | 38 | 38 | 38 | 38 | 38 | 38 | 38 | 38 |
| 20 | 34 | 34 | 34 | 8 | 34 | 34 | 20 | 20 | 29 | 34 | 34 | 34 | 34 | 34 | 34 | 34 | 34 | 34 | 32 | 34 | 34 | 34 | 34 | 34 |
| 21 | 235 | 97 | 254 | 81 | 254 | 254 | 254 | 0 | 254 | 173 | 254 | 254 | 254 | 254 | 0 | 89 | 254 | 254 | 254 | 195 | 190 | 254 | 73 | 254 |
| 22 | 32 | 32 | 32 | 32 | 32 | 32 | 32 | 32 | 32 | 32 | 32 | 32 | 32 | 32 | 32 | 32 | 32 | 32 | 32 | 32 | 32 | 32 | 32 | 32 |
| 23 | 0 | 225 | 225 | 0 | 0 | 0 | 225 | 3 | 0 | 0 | 203 | 203 | 0 | 225 | 225 | 0 | 225 | 137 | 0 | 225 | 225 | 176 | 0 | 0 |
| 24 | 145 | 145 | 145 | 145 | 133 | 145 | 118 | 145 | 145 | 145 | 145 | 145 | 145 | 145 | 145 | 145 | 145 | 145 | 145 | 145 | 145 | 145 | 145 | 145 |
| 25 | 50 | 50 | 50 | 28 | 50 | 50 | 50 | 50 | 50 | 50 | 23 | 50 | 50 | 23 | 50 | 50 | 50 | 17 | 50 | 50 | 50 | 50 | 46 | 50 |
| 26 | 112 | 112 | 112 | 112 | 112 | 112 | 0 | 37 | 112 | 112 | 112 | 112 | 112 | 112 | 112 | 112 | 112 | 112 | 112 | 112 | 112 | 112 | 112 | 112 |
| 27 | 0 | 0 | 0 | 0 | 59 | 0 | 0 | 59 | 0 | 0 | 0 | 0 | 37 | 0 | 0 | 59 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 28 | 120 | 120 | 120 | 120 | 120 | 120 | 120 | 120 | 120 | 120 | 120 | 120 | 120 | 120 | 120 | 120 | 120 | 120 | 120 | 120 | 120 | 120 | 120 | 120 |
| 29 | 4200 | 4200 | 4200 | 4200 | 4200 | 4200 | 4200 | 4200 | 4200 | 4200 | 4200 | 4200 | 4200 | 4200 | 4200 | 4200 | 4200 | 4200 | 4200 | 4200 | 4200 | 4200 | 4200 | 4200 |

**Figure B.7:** Hydro units initial schedule

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 72 | 72 | 72 | 72 | 72 | 72 | 72 | 72 | 72 | 13 | 56 | 72 | 69 | 43 | 72 | 72 | 0 | 13 | 72 | 43 | 43 | 72 | 13 | 72 |
| 2 | 26 | 26 | 26 | 26 | 26 | 26 | 26 | 26 | 26 | 26 | 26 | 26 | 26 | 26 | 26 | 26 | 26 | 26 | 26 | 26 | 26 | 26 | 26 | 26 |
| 3 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 |
| 4 | 72 | 72 | 72 | 20 | 72 | 8 | 72 | 72 | 72 | 72 | 22 | 5 | 72 | 72 | 36 | 69 | 72 | 0 | 10 | 72 | 72 | 72 | 0 | 72 |
| 5 | 130 | 130 | 0 | 82 | 25 | 63 | 88 | 130 | 61 | 39 | 119 | 0 | 0 | 0 | 0 | 38 | 95 | 129 | 130 | 130 | 95 | 48 | 44 | 95 |
| 6 | 0 | 13 | 7 | 6 | 7 | 0 | 7 | 7 | 0 | 0 | 0 | 16 | 0 | 13 | 10 | 31 | 9 | 36 | 22 | 22 | 11 | 36 | 30 | 36 |
| 7 | 7 | 7 | 0 | 7 | 0 | 5 | 0 | 0 | 7 | 9 | 1 | 0 | 24 | 6 | 0 | 20 | 9 | 38 | 16 | 30 | 18 | 10 | 35 | 33 |
| 8 | 0 | 0 | 2 | 0 | 2 | 0 | 2 | 2 | 0 | 0 | 3 | 8 | 0 | 6 | 3 | 10 | 7 | 0 | 13 | 9 | 6 | 6 | 2 | 13 |
| 9 | 64 | 64 | 64 | 64 | 64 | 64 | 64 | 64 | 64 | 64 | 64 | 64 | 64 | 64 | 64 | 64 | 64 | 64 | 64 | 64 | 64 | 64 | 64 | 64 |
| 10 | 50 | 50 | 50 | 50 | 50 | 50 | 50 | 50 | 50 | 50 | 50 | 50 | 50 | 50 | 50 | 50 | 50 | 50 | 50 | 50 | 50 | 50 | 50 | 50 |
| 11 | 18 | 18 | 0 | 0 | 0 | 0 | 18 | 18 | 18 | 18 | 18 | 18 | 18 | 18 | 18 | 18 | 18 | 18 | 18 | 18 | 18 | 18 | 18 | 18 |
| 12 | 18 | 18 | 18 | 18 | 18 | 18 | 18 | 18 | 18 | 18 | 18 | 18 | 18 | 18 | 18 | 18 | 18 | 18 | 18 | 18 | 18 | 18 | 18 | 18 |
| 13 | 58 | 0 | 58 | 58 | 58 | 58 | 58 | 0 | 58 | 58 | 58 | 58 | 58 | 24 | 0 | 58 | 0 | 58 | 58 | 24 | 0 | 58 | 58 | 58 |
| 14 | 258 | 258 | 258 | 258 | 258 | 258 | 118 | 156 | 156 | 251 | 118 | 258 | 258 | 0 | 0 | 258 | 156 | 31 | 258 | 0 | 156 | 118 | 118 | 258 |
| 15 | 234 | 234 | 234 | 234 | 234 | 234 | 234 | 234 | 234 | 234 | 234 | 234 | 234 | 234 | 234 | 234 | 234 | 234 | 234 | 234 | 234 | 234 | 234 | 234 |
| 16 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 |
| 17 | 36 | 36 | 36 | 36 | 36 | 36 | 36 | 36 | 36 | 36 | 36 | 36 | 36 | 36 | 36 | 36 | 36 | 36 | 36 | 36 | 36 | 36 | 36 | 36 |
| 18 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 |
| 19 | 38 | 38 | 38 | 38 | 38 | 38 | 38 | 38 | 38 | 38 | 38 | 38 | 38 | 38 | 38 | 38 | 38 | 38 | 38 | 38 | 38 | 38 | 38 | 38 |
| 20 | 34 | 34 | 34 | 8 | 34 | 20 | 16 | 34 | 34 | 34 | 34 | 34 | 34 | 34 | 34 | 33 | 34 | 34 | 34 | 34 | 34 | 33 | 34 | 34 |
| 21 | 235 | 97 | 254 | 81 | 254 | 254 | 254 | 0 | 254 | 173 | 254 | 254 | 254 | 254 | 0 | 89 | 254 | 254 | 254 | 195 | 190 | 254 | 73 | 254 |
| 22 | 32 | 32 | 32 | 32 | 32 | 32 | 32 | 32 | 32 | 32 | 32 | 32 | 32 | 32 | 32 | 32 | 32 | 32 | 32 | 32 | 32 | 32 | 32 | 32 |
| 23 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 184 | 225 | 225 | 225 | 225 | 225 | 225 | 0 | 225 | 225 | 225 | 89 | 0 | 0 | 225 | 0 |
| 24 | 145 | 145 | 145 | 145 | 133 | 118 | 145 | 145 | 145 | 145 | 145 | 145 | 145 | 145 | 145 | 145 | 145 | 145 | 145 | 145 | 145 | 145 | 145 | 145 |
| 25 | 50 | 50 | 50 | 28 | 50 | 50 | 23 | 50 | 50 | 50 | 50 | 50 | 50 | 50 | 36 | 36 | 50 | 50 | 50 | 50 | 17 | 46 | 50 | 50 |
| 26 | 112 | 112 | 112 | 112 | 112 | 112 | 0 | 37 | 112 | 112 | 112 | 112 | 112 | 112 | 112 | 112 | 112 | 112 | 112 | 112 | 112 | 112 | 112 | 112 |
| 27 | 0 | 0 | 0 | 0 | 59 | 0 | 0 | 59 | 0 | 0 | 0 | 0 | 37 | 0 | 0 | 59 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 28 | 120 | 120 | 120 | 120 | 120 | 120 | 120 | 120 | 120 | 120 | 120 | 120 | 120 | 120 | 120 | 120 | 120 | 120 | 120 | 120 | 120 | 120 | 120 | 120 |
| 29 | 4200 | 4200 | 4200 | 4200 | 4200 | 4200 | 4200 | 4200 | 4200 | 4200 | 4200 | 4200 | 4200 | 4200 | 4200 | 4200 | 4200 | 4200 | 4200 | 4200 | 4200 | 4200 | 4200 | 4200 |

**Figure B.8:** Hydro units final schedule

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 5 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 6 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 7 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 8 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 9 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 10 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 11 | 0 | 0 | 1 | 0 | 0 | 0 | 18 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 12 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 13 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 14 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 15 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 16 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 17 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 18 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 19 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 20 | 0 | 0 | 0 | 0 | 0 | -14 | -4 | 13 | 5 | 0 | 0 | 0 | 0 | 0 | 0 | -1 | 0 | 0 | 2 | 0 | 0 | -1 | 0 | 0 |
| 21 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 22 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 23 | 0 | -225 | -225 | 0 | 0 | 0 | -225 | -3 | 184 | 225 | 22 | 22 | 225 | 0 | 0 | 0 | 0 | 87 | 225 | -136 | -225 | -176 | 225 | 0 |
| 24 | 0 | 0 | 0 | 0 | 0 | -27 | 27 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 25 | 0 | 0 | 0 | 0 | 0 | 0 | -28 | 0 | 0 | 0 | 28 | 0 | 0 | 28 | -14 | -14 | 0 | 33 | 0 | 0 | -33 | -4 | 4 | 0 |
| 26 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 27 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 28 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 29 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure B.9:** Difference between initial and final hydro units schedule

# References

E. Aarts and J.K. Lenstra. *Local Search in Combinatorial Optimization.* John Wiley & Sons, Chichester, UK, 1997.

E. Aarts and H.M.M. Ten Eikelder. Simulated Annealing. In P.M. Pardalos and M.G.C. Resende, editors, *Handbook of Applied Optimization.* Oxford University Press, 2002.

S. Al-Agtash and R. Su. Augmented lagrangian approach to hydrothermal scheduling. *IEEE Transactions on Power Systems*, 13(4):1392–1400, 1998.

K. Aoki, M. Itoh, T. Satoh, K. Nara, and M. Kanezashi. Unit commitment in a large-scale power system including fuel constrained thermal and pumped-storage hydro. *IEEE Transactions on Power Systems*, PWRS – 2:1077–1084, 1987.

K. Aoki, T. Satoh, and M. Itoh. Optimal long-term unit commitment in large scale systems including fuel constrained thermal and pumped-storage hydro. *IEEE Transactions on Power Systems*, 4:1065–1073, 1989.

M.F. Argüello, J.F. Bard, and G. Yu. A GRASP for aircraft routing in response to groundings and delays. *Journal of Combinatorial Optimization*, 1:211–228, 1997.

R. Azencott. *Simulated Annealing: Parallelization Techniques.* John Wiley & Sons, 1992.

L. Bahiense, G.C. Oliveira, and M. Pereira. A mixed integer disjunctive model for transmission network expansion. *IEEE Transactions on Power Systems*, 16:560–565, 2001.

X. Bai and S.M. Shahidehpour. Hydrothermal scheduling by Tabu Search and decomposition method. *IEEE Transactions on Power Systems*, 11:968–974, 1996.

X. Bai and S.M. Shahidehpour. Extended neighbourhood search algorithm for constrained unit commitment. *Electrical Power and Energy Systems*, 19(5):349–356, 1997.

R. Baldick. The generalized unit commitment problem. *IEEE Transactions on Power Systems*, 10(1):465–475, 1995.

A. Baíllo, M. Ventosa, A. Ramos, and M. Rivier. Strategic unit commitment for generation in deregulated electricity markets. In B.F. Hobbs, M.H. Rothkopf, R.P. O'Neill, and H. Chao, editors, *The Next Generation of Electric Power Unit Commitment Models*, pages 227–248. Kluwer Academic Publishers, 2001.

J.F. Bard. Short-term scheduling of thermal-electric generators using Lagrangian Relaxation. *Operations Research*, 36(5):756–766, 1988.

R.S. Barr, B.L. Golden, J. Kelly, W.R. Stewart, and M.G.C. Resende. Designing and reporting on computational experiments with heuristic methods. *Journal of Heuristics*, 1: 9–32, 1995.

R. Battiti. Reactive search: Toward self-tuning heuristics. In V.J. Rayward-Smith, editor, *Modern Heuristic Search Methods*, pages 61–83. John Wiley & Sons, 1996.

J. Batut and A. Renaud. Daily generation scheduling optimization with transmission constraints: a new class of algorithms. *IEEE Transactions on Power Systems*, 7(3):982–989, 1992.

C. Beltran and F.J. Heredia. Unit commitment by Augmented Lagrangian Relaxation: testing two decomposition approaches. *Journal of Optimisation Theory and Applications*, 112(2):295–314, 2002.

D.P. Bertsekas, G.S. Lauer, N.R. Sandell, and T.A. Posbergh. Optimal short-term scheduling of large-scale power systems. *IEEE Transactions on Automatic Control*, 28(1):1–11, 1983.

F. Bock. An algorithm for solving traveling-salesman and related network optimization problems. In *Proceedings of the 14$^{th}$ National Meeting of the Operations Research Society of America*, St. Louis, USA, 1958.

A. Borghetti, A. Frangioni, F. Lacalandra, A. Lodi, S. Martello, C.A. Nucci, and A. Trebbi. Lagrangian Relaxation and Tabu Search approaches for the unit commitment problem. In J. Tomé Saraiva and M.A. Matos, editors, *Proceedings of the IEEE 2001 Porto PowerTech Conference*, Porto, Portugal, 2001a.

A. Borghetti, A. Frangioni, F. Lacalandra, C.A. Nucci, and P. Pelacchi. Using of a cost-based unit commitment algorithm to assist bidding strategy decisions. In A. Borghetti, C.A. Nucci, and M. Paolone, editors, *Proceedings of the IEEE 2003 PowerTech Bologna Conference*, Bologna, Italy, 2003.

A. Borghetti, G. Gross, and C.A. Nucci. Auctions with explicit demand-side bidding in competitive electricity markets. In B.F. Hobbs, M.H. Rothkopf, R.P. O'Neill, and H. Chao, editors, *The Next Generation of Electric Power Unit Commitment Models*, pages 53–74. Kluwer Academic Publishers, 2001b.

E. Burke, E. Hart, G. Kendall, J. Newall, P. Ross, and S. Schulenburg. Hyper-heuristics: an emerging direction in modern search technology. In F. Glover and G. Kochenberger, editors, *Handbook of Metaheuristics*. Kluwer Academic Publishers, 2003.

E.K. Burke and S. Petrovic. Recent research directions in automated timetabling. *European Journal of Operational Research*, 140(2):266–280, 2002.

G.W. Chang, M. Aganagic, J.G. Waight, J. Medina, T. Burton, S. Reeves, and M. Christoforidis. Experiences with mixed integer linear programming based approaches on short-term hydro scheduling. *IEEE Transactions on Power Systems*, 16(4):743–749, 2001.

H. Chen and X. Wang. Cooperative coevolutionary algorithm for unit commitment. *IEEE Transactions on Power Systems*, 17(1):128–133, 2002.

P-H. Chen and H-C. Chang. Genetic aided scheduling of hydraulically coupled plants in hydrothermal coordination. *IEEE Transactions on Power Systems*, 11(2):975–981, 1996.

C-P. Cheng, C-W. Liu, and C-C. Liu. Unit commitment by Lagrangian Relaxation and Genetic Algorithms. *IEEE Transactions on Power Systems*, 15(5):707–714, 2000.

C-P. Cheng, C-W. Liu, and C-C. Liu. Unit commitment by annealing-Genetic Algorithm. *Electrical Power and Energy Systems*, 24:149–158, 2002.

C.A. Coello. An updated survey of GA-based multiobjective optimization techniques. *ACM Computing Surveys*, 32(2):109–143, 2000.

C.A. Coello, D.A. Van Veldhuizen, and G.B. Lamont. *Evolutionary Algorithms for Solving Multi-Objective Problems*. Kluwer Academic Publishers, New York, 2002.

A.I. Cohen and S.H. Wan. A method for solving the fuel constrained unit commitment problem. *IEEE Transactions on Power Systems*, PWRS – 2(3):608–614, 1987.

A.I. Cohen and M. Yoshimura. A branch-and-bound algorithm for unit commitment. *IEEE Transactions on Power Apparatus Systems*, PAS – 102(2):444–451, 1983.

E. Costamagna, A. Fanni, and G. Giacinto. A Tabu Search algorithm for the optimisation of telecommunication networks. *European Journal of Operational Research*, 106:357–372, 1998.

T.G. Crainic, M. Toulouse, and M. Gendreau. Towards a taxonomy of parallel Tabu Search heuristics. *INFORMS Journal on Computing*, 9(1):61–72, 1997.

G.A. Croes. A method for solving traveling salesman problems. *Operations Research*, 6: 791–812, 1958.

V-D. Cung, S. Martins, C. Ribeiro, and C. Roucairol. Strategies for the parallel implementation of metaheuristics. In C.C. Ribeiro and P. Hansen, editors, *Essays and Surveys on Metaheuristics*, pages 263–308. Kluwer Academic Publishers, 2001.

P. Czyzac and A. Jaszkiewicz. Pareto Simulated Annealing – a metaheuristic technique for multiple objective combinatorial optimization. *Journal of Multi-Criteria Decision Analysis*, 7:34–37, 1998.

D. Dasgupta and D.R. McGrevor. Thermal unit commitment using Genetic Algorithms. *IEE Proceedings – Generation Transmission and Distribution*, 141:459–465, 1994.

K. Deb. *Multi-Objective Optimization using Evolutionary Algorithms.* John Wiley & Sons, Chichester, UK, 2001.

K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective Genetic Algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, 2002.

M. Dorigo and G. di Caro. The ant colony optimization metaheuristic. In D. Corne, M. Dorigo, and F. Glover, editors, *New Ideas in Optimization*. McGraw-Hill, 1999.

H. Duo, H. Sasaki, T. Nagata, and H. Fujita. A solution for unit commitment using Lagrangian Relaxation combined with Evolutionary Programming. *Electric Power Systems Research*, 51:71–77, 1999.

M. Ehrgott and X. Gandibleux. A survey and annotated bibliography of multiobjective combinatorial optimization. *OR Spektrum*, 22:425–460, 2000.

L. Eliasson. Start-stop costs in hydro systems. In *Hydro scheduling in competitive electricity markets conference*, Trondheim, Norway, May 2002. SINTEF.

A.T. Ernst, H. Jiang, M. Krishnamoorthy, and D. Sier. Staff scheduling and rostering: A review of applications, methods and models. *European Journal of Operational Research*, 153(1):3–27, 2004.

S.R. Erwin, J.S. Griffith, J.T. Wood, K.D. Le, J.T. Day, and C.K. Yin. Using an optimization software to lower overall company electric production costs for southern company. *Interfaces*, 21(1):27–41, 1991.

U. Faigle and W. Kern. Note on the convergence of Simulated Annealing algorithms. *SIAM Journal of Control and Optimisation*, 29:153–159, 1991.

U. Faigle and W. Kern. Some convergence results for probabilistic Tabu Search. *ORSA Journal on Computing*, 4:32–37, 1992.

S. Feltenmark. *On Optimization of Power Production*. PhD thesis, Royal Institute of Technology, Stockholm, Sweden, 1997.

T.A. Feo and M.G.C. Resende. A probabilistic heuristic for a computationally difficult set covering problem. *Operations Research Letters*, 8:67–71, 1989.

T.A. Feo and M.G.C. Resende. Greedy randomized adaptive search procedures. *Journal of Global Optimization*, 6:109–133, 1995.

L.A.F.M. Ferreira. On the convergence of the classic hydrothermal coordination algorithm. *IEEE Transactions on Power Apparatus and Systems*, 9:1002–1008, 1994.

P. Festa and M.G.C. Resende. GRASP: An annotated bibliography. In C.C. Ribeiro and P. Hansen, editors, *Essays and Surveys on Metaheuristics*. Kluwer Academic Publishers, 2001.

A. Fink, S. Voss, and D.L. Woodruff. Metaheuristic class libraries. In F. Glover and G. Kochenberger, editors, *Handbook of Metaheuristics*. Kluwer Academic Publishers, 2003.

M. Fischetti and A. Lodi. Local branching. *Mathematical Programming*, 98:23–47, 2003.

C.M. Fonseca and P.J. Fleming. Genetic Algorithms for multiobjective optimization: Formulation, discussion and generalization. In *Proceedings of the ICGA'93*, pages 416–423, 1993.

C.M. Fonseca and P.J. Fleming. Multiobjective optimization and multiple constraint handling with evolutionary algorithms – Part I: A unified formulation. *IEEE Transactions on System, Man, and Cybernetics – Part A: System and Humans*, 28:26–37, 1998.

R. Fuentes and V. Quintana. Semidefinite programming: a practical application to hydrothermal coordination. In *Proceedings of the 14th Power Systems Computation Conference*, Seville, Spain, June 2002.

T. Gal, T. Hanne, and T. Stewart, editors. *Advances in Multiple Criteria Decision Making.* Kluwer Academic, Dordrecht, 1998.

F.D. Galiana, A.L. Motto, A.J. Conejo, and M. Huneault. Decentralized nodal-price self-dispatch and unit commitment. In B.F. Hobbs, M.H. Rothkopf, R.P. O'Neill, and H. Chao, editors, *The Next Generation of Electric Power Unit Commitment Models*, pages 271–292. Kluwer Academic Publishers, 2001.

X. Gandibleux, N. Mezdaoui, and A. Fréville. A Tabu Search procedure to solve multiobjective combinatorial optimization problems. In R. Caballero, F. Ruiz, and R. Steuer, editors, *Advances in multiple objective and goal programming*, volume 455 of *Lecture Notes in Economics and Mathematical Systems*, pages 291–300. Springer, 1997.

J. García, J. Román, J. Barquín, and A. González. Strategic bidding in deregulated power systems. In *Proceedings of the 13th Power Systems Computation Conference*, Trondheim, Norway, 1999.

M.R. Garey and D.S. Johnson. *Computers and Intractability – A Guide to the Theory of NP-Completeness.* Freeman, San Francisco, 1979.

M. Gendreau. Recent advances in Tabu Search. In C.C. Ribeiro and P. Hansen, editors, *Essays and Surveys in Metaheuristics.* Kluwer, 2002.

M. Gendreau. An introduction to Tabu Search. In F. Glover and G. Kochenberger, editors, *Handbook of Metaheuristics.* Kluwer Academic Publishers, 2003.

M. Gendreau, A. Hertz, and G. Laporte. A Tabu Search heuristic for the vehicle routing problem. *Management Science*, 40:1276–1290, 1994.

T. Gjengedal. Emission constrained unit commitment (ECUC). *IEEE Transactions on Energy Conversion*, 11:132–138, 1996.

F. Glover. Future paths for integer programming and links to artificial intelligence. *Computers and Operations Research*, 13:533–549, 1986.

F. Glover. Tabu Search – Part II. *ORSA Journal on Computing*, 2:4–32, 1990.

F. Glover and S. Hanafi. Tabu Search and finite convergence. *Discrete Applied Mathematics*, 119:3–36, 2002.

F. Glover and G. Kochenberger. *Handbook of Metaheuristics.* Kluwer Academic Publishers, Boston, 2003.

F. Glover and M. Laguna. *Tabu Search.* Kluwer Academic Publishers, Boston, 1997.

F. Glover and M. Laguna. Tabu Search. In P.M. Pardalos and M.G.C. Resende, editors, *Handbook of Applied Optimization.* Oxford University Press, 2002.

F. Glover, E.D. Taillard, and D. de Werra. A user's guide to Tabu Search. *Annals of Operations Research*, 41:3–28, 1993.

A.M. Gofferion. Generalized Benders decomposition method. *Journal of Optimization Theory and Application*, 10(4), 1972.

D. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning.* Addison-Wesley, Cambridge, MA, 1989.

R. Gollmer, A. Möller, Nowak M.P., W. Römisch, and R. Schultz. Primal and dual methods for unit commitmnet in a hydrothermal power system. In *Proceedings of the 13th Power Systems Computation Conference*, Trondheim, Norway, 1999.

V. Granville, M. Krivanek, and J.-P. Rasson. Simulated Annealing: a proof of convergence. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16:652–656, 1994.

G. Gross and D. Finlay. Generation supply bidding in perfectly competitive electricity markets. *Computational and Mathematical Organization Theory*, 6:83–98, 2000.

X. Guan, P.B. Luh, H. Yan, and P. Rogan. Optimization based scheduling of hydrothermal power systems with pumped-storage units. *IEEE Transactions on Power Systems*, 9(2): 1023–1031, 1994.

X. Guan, E. Ni, R. Li, and P.B. Luh. An optimization based algorithm for scheduling hydrothermal power systems with cascade reservoirs and discrete hydro constraints. *IEEE Transactions on Power Systems*, 12:1775–1780, 1997.

X. Guan, E. Ni, P.B. Luh, and Y-C. Ho. Optimization-based bidding strategies for deregulated electric power markets. In B.F. Hobbs, M.H. Rothkopf, R.P. O'Neill, and H. Chao, editors, *The Next Generation of Electric Power Unit Commitment Models*. Kluwer Academic Publishers, 2001.

H. Guo. *Algorithm selection for sorting and probabilistic inference: a machine learning based approach*. PhD thesis, Kansas State University, Manhattan, Kansas, 1996.

M.P. Hansen. Tabu Search for multiobjective optimization: MOTS. In *Proceedings of the 13$^{th}$ International Conference on Multiple Criteria Decision Making*, pages 574–586, University of Cape Town, Republic of South Africa, 1997.

M.P. Hansen. *Metaheuristics for multiple objective combinatorial optimization*. PhD thesis, Technical University of Denmark, Lyngby, Denmark, 1998.

M.P. Hansen. Tabu Search for multiobjective combinatorial optimization: Tamoco. *Control and Cybernetics*, 294:799–818, 2000.

M.P. Hansen and A. Jaszkiewicz. Evaluating the quality of approximations to the nondominated set. Technical Report IMM–REP–1998–7, 1998.

P. Hansen. The steepest ascent mildest descent heuristic for combinatorial programming. In *Proceedings of the Congress on Numerical Methods in Combinatorial Optimization*, Capri, Italy, 1986.

P. Hansen and N. Mladenović. Variable Neighborhood Search. *Computers and Operations Research*, 24:1097–1100, 1997.

P. Hansen and N. Mladenović. An introduction to variable Neighborhood Search. In S. Voss, S. Martello, I.H. Osman, and C. Roucairol, editors, *Metaheuristics: Advances and Trends in Local Search Paradigms for Optimization*, pages 433–458. Kluwer Academic Publishers, 1999.

P. Hansen and N. Mladenović. Developments of variable Neighborhood Search. In C.C. Ribeiro and P. Hansen, editors, *Essays and Surveys on Metaheuristics*, pages 415–440. Kluwer Academic Publishers, 2001a.

P. Hansen and N. Mladenović. Variable Neighborhood Search: Principles and applications. *European Journal of Operational Research*, 130:449–467, 2001b.

P. Hansen and N. Mladenović. Variable Neighborhood Search. In P.M. Pardalos and M.G.C. Resende, editors, *Handbook of Applied Optimization*, pages 221–234. Oxford University Press, 2002.

P. Hansen and N. Mladenović. Variable Neighborhood Search. In F. Glover and G. Kochenberger, editors, *Handbook of Metaheuristics.* Kluwer Academic Publishers, 2003.

P. Hansen, N. Mladenović, and D. Perez-Brito. Variable neighborhood decomposition search. *Journal of Heuristics*, 7:335–350, 2001.

S. Hao. A study of basic bidding strategy in clearing pricing auctions. *IEEE Transactions on Power Systems*, 15:975–980, 2000.

S. Hao, G.A. Angelidis, H. Singh, and A.D. Papalexopoulus. Consumer payment minimization in power pool auctions. *IEEE Transactions on Power Systems*, 13:986–991, 1998.

H.H. Happ, R.C. Johnson, and W.J. Wright. Large scale hydrothermal unit commitment method and results. *IEEE Transactions on Power Apparatus and Systems*, 90:1373–1383, 1971.

J.P. Hart and A.W. Shogan. Semi-greedy heuristics – an empirical study. *Operations Research Letters*, 6:107–114, 1987.

D. Henderson, S.H. Jacobson, and A.W. Johnson. The theory and practice of Simulated Annealing. In F. Glover and G. Kochenberger, editors, *Handbook of Metaheuristics.* Kluwer Academic Publishers, 2003.

W.J. Hobbs, G. Hermon, S. Warner, and G.B. Sheblé. An enhanced dynamic programming approach for unit commitment. *IEEE Transactions on Power Systems*, 3(3):1201–1205, 1988.

J.H. Holland. *Adaptation in Natural and Artificial Systems.* University of Michigan Press, Ann Arbor, MI, 1975.

J. Horn, N. Nafpliotis, and D.E. Goldberg. A Niched Pareto Genetic Algorithm for multi-objective optimization. In *Proceedings of the* $1^{st}$ *IEEE Conference on Evolutionary Computation, IEEE World Congress on Computational Intelligence*, pages 82–87, Piscataway, New Jersey, June 1994.

K-Y. Huang, H-T. Yang, and C-L. Huang. A new thermal unit commitment approach using Constraint Logic Programming. *IEEE Transactions on Power Systems*, 13(3):936–945, 1998.

S-J. Huang. Enhancement of thermal unit commitment using immune algorithms based optimization approaches. *Electrical Power and Energy Systems*, 21:245–252, 1999.

S-J. Huang and C-L. Huang. Application of genetic-based Neural Networks to thermal unit commitment. *IEEE Transactions on Power Systems*, 12:654–660, 1997.

Y. Ikura, G. Gross, and G.S. Hall. PG&E's state-of-the-art scheduling tool for hydro systems. *Interfaces*, 16(1):65–82, 1986.

J.M. Jacobs. Artificial power markets and unintended consequences. *IEEE Transactions on Power Systems*, 12(2):968–972, 1997.

A. Jaszkiewicz. Comparison of Local Search-based metaheuristics on the multiple objective knapsack problem. *Foundations of Computing and Decision Sciences*, 26(1):99–120, 2001.

A. Jaszkiewicz. Genetic Local Search for multiobjective combinatorial optimization. *European Journal of Operational Research*, 137(1):50–71, 2002.

A. Jaszkiewicz. *MOMHLib++: Multiple Objective MetaHeuristics Library in C++.* http://www-idss.cs.put.poznan.pl/ jaszkiewicz/MOMHLib, 2003.

D.S. Johnson, C.H. Papadimitriou, and M. Yannakakis. How easy is Local Search? *Journal of Computer and System Sciences*, 37:79–100, 1988.

R.B. Johnson, A.J. Svoboda, C. Greif, A. Vojdani, and F. Zhuang. Positioning for a competitive electric industry with PG&E's hydrothermal optimization. *Interfaces*, 28(1):53–74, 1998.

D. Jones, S.K. Mirrazavi, and M. Tamiz. Multiobjective metaheuristics: An overview of the current state-of-the-art. *European Journal of Operational Research*, 137(1):1–9, 2002.

J. Jozefowska, G. Waligora, and J. Weglarz. Tabu list management methods for a discrete-continuous scheduling problem. *European Journal of Operational Research*, 137:288–302, 2002.

K.A. Juste, H. Kita, E. Tanaka, and J. Hasegawa. An Evolutionary Programming solution to the unit commitment problem. *IEEE Transactions on Power Systems*, 14:1452–1459, 1999.

S.A. Kazarlis, A.G. Bakirtzis, and V. Petridis. A Genetic Algorithm solution to the unit commitment problem. *IEEE Transactions on Power Systems*, 11:83–92, 1996.

S. Kirkpatrick, C.D. Gelatt, and M.P. Vecchi. Optimization by Simulated Annealing. *Science*, 220:671–680, 1983.

J. Knowles. *Local-Search and Hybrid Evolutionary Algorithms for Pareto Optimization*. PhD thesis, University of Reading, Reading, UK, 2001.

J.D. Knowles and D. Corne. Approximating the nondominated front using the Pareto Archived Evolution Strategy. *Evolutionary Computation*, 8(2):149–172, 2000a.

J.D. Knowles and D.W. Corne. M-PAES: A memetic algorithm for multiobjective optimization. In *Proceedings of the 2000 Congress on Evolutionary Computation CEC2000*, pages 325–332, 2000b.

J.D. Knowles and D.W. Corne. On metrics for comparing nondominated sets. In *Proceedings of the 2002 Congress on Evolutionary Computation Conference CEC02*, pages 711–716. IEEE Press, 2002.

D.P. Kothari and A. Ahmad. An expert system approach to the unit commitment problem. In *Proceedings of the IEEE TENCON'93*, Beijing, China, 1993.

H.W. Kuhn and A.W. Tucker. Nonlinear programming. In *Proceedings of the* $2^{nd}$ *Berkeley Symposium on Mathematical Programming Statistics and Probability*, 1951.

S. Kuloor, G.S. Hope, and O.P. Malik. Environmentally constrained unit commitment. *IEE Proceedings – C*, 139:122–128, 1992.

G. Laporte and I.H. Osman. Metaheuristics in combinatorial optimization: A bibliography. *Annals of Operations Research*, 63:513–628, 1996.

F.N. Lee. Short-term unit commitment – a new method. *IEEE Transactions on Power Systems*, 3(2):421–428, 1988.

F.N. Lee. A fuel constrained unit commitment method. *IEEE Transactions on Power Systems*, 4:1208–1218, 1989.

F.N. Lee. The coordination of multiple constrained fuels. *IEEE Transactions on Power Systems*, 6:699–707, 1991.

F.N. Lee and Q. Feng. Multi-area unit commitment. *Transactions on Power Systems*, 7: 591–599, 1992.

F.N. Lee, J. Huang, and R. Adapa. Multi-area unit commitment via sequential method and a DC power flow network model. *IEEE Transactions on Power Systems*, 9:279–287, 1994.

C. Li, A. Svoboda, X. Guan, and H. Singh. Revenue adequate bidding strategies in competitive electricity markets. *IEEE Transactions on Power Systems*, 14:492–497, 1999.

C-a. Li, E. Hsu, A.J. Svoboda, C.L. Tseng, and R.B. Johnson. Hydro unit commitment in hydrothermal optimisation. *IEEE Transactions on Power Systems*, 12:764–769, 1997a.

C-a. Li, R.B. Johnson, and A.J. Svoboda. A new unit commitment method. *IEEE Transactions on Power Systems*, 12(1):113–119, 1997b.

C-a. Li, R.B. Johnson, A.J. Svoboda, C.L. Tseng, and E. Hsu. A robust unit commitment algorithm for hydrothermal optimization. *IEEE Transactions on Power Systems*, 13:1051–1056, 1998.

S. Li, S.M. Shahidehpour, and C. Wang. Promoting the application of expert systems in short-term unit commitment. *IEEE Transactions on Applied Supercondutivity*, 3(1):286–292, 1993.

H.R. Lourenço, O.C. Martin, and T. Stutzle. Iterated Local Search. In F. Glover and G. Kochenberger, editors, *Handbook of Metaheuristics*. Kluwer Academic Publishers, 2003.

P.G. Lowery. Generating unit commitment by Dynamic Programming. *IEEE Transactions on Power Apparatus Systems*, PAS – 85(5):422–426, 1966.

P.B. Luh, D. Zhang, and R.N. Tomastik. An algorithm for solving the dual problem of hydrothermal scheduling. *IEEE Transactions on Power Systems*, 13(2):593–600, 1998.

X. Ma, A. A. El-Keib, R.E. Smith, and H. Ma. A Genetic Algorithm based approach to thermal unit commitment of electrical power systems. *Electrical Power Systems Research*, 34:29–36, 1995.

M. Madrigal and V.H. Quintana. Existence and determination of competitive equilibrium in unit commitment power pool auctions: Price setting and scheduling alternatives. *IEEE Transactions on Power Systems*, 16(3):380–388, 2001.

T.T. Maifeld and G.B. Sheblé. Genetic-based unit commitment algorithm. *IEEE Transactions on Power Systems*, 11:1359–1370, 1996.

V. Maniezzo and A. Carbonaro. Ant colony optimization: an overview. In C.C. Ribeiro and P. Hansen, editors, *Essays and Surveys on Metaheuristics*, pages 469–492. Kluwer Academic Publishers, 2001.

A.H. Mantawy and Y.L. Abdel-Magid. A new fuzzy unit commitment model and solution. In *Proceedings of the 14$^{th}$ Power Systems Computation Conference*, Seville, Spain, June 2002.

A.H. Mantawy, Y.L. Abdel-Magid, and S.Z. Selim. A Simulated Annealing algorithm for unit commitment. *IEEE Transactions on Power Systems*, 13:197–204, 1998a.

A.H. Mantawy, Y.L. Abdel-Magid, and S.Z. Selim. Unit commitment by Tabu Search. *IEE Proceedings – Generation Transmission and Distribution*, 145:56–65, 1998b.

A.H. Mantawy, Y.L. Abdel-Magid, and S.Z. Selim. Integrating Genetic Algorithms, Tabu Search and Simulated Annealing for the unit commitment problem. *IEEE Transactions on Power Systems*, 14(3):829–836, 1999.

F. Manzanedo, M.P. Donsion, and J.L. Castro. An evolution based algorithm for environmentally constrained thermal scheduling problems. In *Proceedings of the IEEE 2001 Porto PowerTech Conference*, Porto, Portugal, Sept 2001.

R. Martí. Multi-start methods. In F. Glover and G. Kochenberger, editors, *Handbook of Metaheuristics*. Kluwer Academic Publishers, 2003.

H.R. Mashhadi, H.M. Shanechi, and C. Lucas. A new Genetic Algorithm with lamarckian individual learning for generation scheduling. *IEEE Transactions on Power Systems*, 18 (3):1181–1186, 2003.

M.A. Matos. Introdução ao problema de escalonamento e pré-despacho. *Apontamentos para a disciplina de DOSE – FEUP*, 1999.

A. Merlin and P. Sandrin. A new method for unit commitment at Electricité de France. *IEEE Transactions on Power Apparatus and Systems*, PAS–102:1218–1225, 1983.

Z. Michalewicz. *Genetic Algorithms + Data Structures = Evolution Programs.* Springer-Verlag New York, Inc., 1994.

S. Mokhtari, J. Sing, and B. Wollenberg. A unit commitment expert system. *IEEE Transactions on Power Systems*, 3(1):272–277, 1988.

J.A. Muckstadt and S.A. Koenig. An application of Lagrangian Relaxation to scheduling in power generation systems. *Operations Research*, 25:387–403, 1977.

R. Nayak and J.D. Sharma. A hybrid Neural Network and Simulated Annealing approach to the unit commitment problem. *Computers and Electrical Engineering*, 26:461–477, 2000.

E. Ni, X. Guan, and R. Li. Scheduling hydrothermal power systems with cascade and head-dependent reservoirs. *IEEE Transactions on Power Systems*, 14:1127–1132, 1999.

P. Oliveira, S. McKee, and C. Coles. Optimal scheduling of a hydro thermal power generation system. Research report 17, Department of Mathematics, University of Strathclyde, Glasgow, Great Britain, 1990.

P. Oliveira, S. McKee, and C. Coles. Optimal scheduling of a hydrothermal power generation system. *European Journal of Operational Research*, 71:334–340, 1993.

S.O. Orero and M.R. Irving. A Genetic Algorithm for generator scheduling in power systems. *Electrical Power and Energy Systems*, 18(1):19–26, 1996.

S.O. Orero and M.R. Irving. A combination of Genetic Algorithm and Lagrangian Relaxation decomposition techniques for the generation unit commitment problem. *Electric Power Systems Research*, 43:149–156, 1997a.

S.O. Orero and M.R. Irving. Large scale unit commitment using a hybrid Genetic Algorithm. *Electrical Power and Energy Systems*, 19:45–55, 1997b.

S.O. Orero and M.R. Irving. A Genetic Algorithm modelling framework and solution technique for short-term optimal hydrothermal scheduling. *IEEE Transactions on Power Apparatus and Systems*, PAS–13:501–517, 1998.

S.O. Orero and M.R. Irving. Large scale unit commitment using a hybrid Genetic Algorithm. *Electrical Power and Energy Systems*, 19:45–55, 1999.

Z. Ouyang and S.M. Shahidehpour. Short-term unit commitment expert system. *Electrical Power Systems Research*, 20:1–13, 1990.

Z. Ouyang and S.M. Shahidehpour. Heuristic multi-area unit commitment with economic dispatch. *IEE Proceedings – C*, 138:242–252, 1991.

Z. Ouyang and S.M. Shahidehpour. A multi-stage intelligent system for unit commitment. *IEEE Transactions on Power Systems*, 7:639–646, 1992.

N.P. Padhy. Unit commitment using hybrid models: a comparative study for Dynamic Programming, Expert System, Fuzzy System and Genetic Algorithms. *Electric Power and Energy Systems*, 23:827–836, 2000.

C.K. Pang and H.C. Chen. Optimal short-term thermal unit commitment. *IEEE Transactions on Power Apparatus and Systems*, PAS − 95:1336–1346, 1976.

C.H. Papadimitriou. *Computational Complexity.* Addison-Wesley, Reading, MA, 1994.

E. Piñana, I. Plana, V. Campos, and R. Martí. GRASP and Path Relinking for the matrix bandwidth minimization. *European Journal of Operational Research*, 153(1):200–210, 2004.

L.S. Pitsoulis and M.G.C. Resende. Greedy randomized adaptive search procedures. In P.M. Pardalos and M.G.C. Resende, editors, *Handbook of Applied Optimization.* Oxford University Press, 2002.

G.K. Purushothama and L. Jenkins. Simulated Annealing with Local Search – a hybrid algorithm for unit commitment. *IEEE Transactions on Power Systems*, 18(1):273–278, 2003.

N. Rajaković and Z.M. Ružić. Sensitivity analysis of an optimal short-term hydrothermal schedule. *IEEE Transactions on Power Systems*, 8(3):1235–1241, 1993.

R. Rajaraman, L. Kirsch, F.L. Alvarado, and C. Clark. Optimal self-commitment under uncertain energy and reserve prices. In B.F. Hobbs, M.H. Rothkopf, R.P. O'Neill, and H. Chao, editors, *The Next Generation of Electric Power Unit Commitment Models*, pages 93–116. Kluwer Academic Publishers, 2001.

M.V. Rakić and Z.M. Marković. Short-term operation and power exchange planning of hydrothermal power systems. *IEEE Transactions on Power Systems*, 9(1):359–365, 1994.

J.L.M. Ramos, A.T. Lora, J.R. Santos, and A.G. Exposito. Short-term hydrothermal coodination based on interior point nonlinear programming and Genetic Algorithms. In *Proceedings of the IEEE 2001 Porto PowerTech Conference*, Porto, Portugal, Sept 2001.

V.J. Rayward-Smith, I.H. Osman, C.R. Reeves, and G.D. Smith. *Modern Heuristic Search Methods.* John Wiley & Sons, Chichester, 1996.

N.J. Redondo. *Coordinación hidrotérmica en el corto plazo mediante técnicas de relajación Lagrangiana*. PhD thesis, Universidad de Castilla-La Mancha, Spain, 1999. (in Spanish).

N.J. Redondo and A.J. Conejo. Short-term hydrothermal coordination by Lagrangian Relaxation: solution of the dual problem. *IEEE Transactions on Power Systems*, 14:89–95, 1999.

C.R. Reeves. *Modern heuristiques techniques for combinatorial problems*. McGraw-Hill, 1995.

Y. Ren and F.D. Galiana. Minimum consumer payment scheduling and pricing in electricity markets. In *Proceedings of the 14$^{th}$ Power Systems Computation Conference*, Seville, Spain, June 2002.

M.G.C. Resende and C.C. Ribeiro. Greedy randomized adaptive search procedures. In F. Glover and G. Kochenberger, editors, *Handbook of Metaheuristics*. Kluwer Academic Publishers, 2003.

M.G.C. Resende and J.P. Sousa. *Metaheuristics: Computer Decision-Making*. Kluwer Academic Publishers, Dordrecht, The Netherlands, 2004.

C. Ribeiro, E. Uchoa, and R. Werneck. A hybrid GRASP with perturbations for the Steiner problem in graphs. *INFORMS Journal on Computing*, 14:228–246, 2002.

C.W. Richter Jr and G.B. Sheblé. Generic algorithm evolution of utility bidding strategies for the competitive marketplace. *IEEE Transactions on Power Systems*, 13:256–261, 1998.

C.W. Richter Jr, G.B. Sheblé, and D. Ashlock. Comprehensive bidding strategies with genetic programming/finite state automata. *IEEE Transactions on Power Systems*, 14(4): 1207–1212, 1999.

R.Z. Ríos-Mercado and J.F. Bard. Heuristics for the flow line problem with setup costs. *European Journal of Operational Research*, pages 76–98, 1998.

B. Roy. *Multicriteria Methodology for Decision Aiding*. Kluwer Academic, Dordrecht, 1996.

A. Rudolf and R. Bayrleithner. A Genetic Algorithm for solving the unit commitment problem of a hydrothermal power system. *IEEE Transactions on Power Systems*, 14: 1460–1468, 1999.

S. Ružić and N. Rajaković. Optimal distance method for lagrangian multipliers updating in short-term hydrothermal coordination. *IEEE Transactions on Power Systems*, 13:1439–1444, 1998.

S. Ružić, N. Rajaković, and A. Vućković. A flexible approach to short-term hydrothermal coordination – Part I: problem formulation and general solution procedure. *IEEE Transactions on Power Systems*, 11:1564–1571, 1996.

S. Saneifard, N.R. Prasad, and H.A. Smolleck. A fuzzy logic approach to unit commitment. *IEEE Transactions on Power Systems*, 12(2):988–995, 1997.

H. Sasaki, M. Watanabe, and R. Yokoyama. A solution method of unit commitment by artificial Neural Networks. *IEEE Transactions on Power Systems*, 7:974–981, 1992.

J.D. Schaffer. Multiple objective optimization with vector evaluated Genetic Algorithms. In *Proceedings of the* 1$^{st}$ *International Conference on Genetic Algorithms*, pages 93–100, 1985.

S. Sen and D.P. Kothari. Optimal thermal generating unit commitment: a review. *Electrical Power and Energy Systems*, 20:443–451, 1998a.

S. Sen and D.P. Kothari. Optimal thermal generating unit commitment of large power system: a novel approach. In *Proceedings of the TENCON '98 - IEEE International Conference on Global Connectivity in Energy, Computer, Communication and Control*, pages 474 – 478, 1998b.

T. Senjyu, H. Yamashiro, K. Uezato, and T. Funabashi. A unit commitment problem by using Genetic Algorithm based on unit characteristic classification. In *Proceedings of the 2002 IEEE Winter Power Meeting*, New York, USA, 2002.

P. Serafini. Simulated Annealing for multiobjective optimization problems. In *Proceedings of the 10$^{th}$ International Conference on MCDM*, pages 221–248, Taipei, July 1992.

G.B. Sheblé. Solution of the unit commitment problem by the method of unit periods. *IEEE Transactions on Power Systems*, 5(1):257–260, 1990.

G.B. Sheblé. *Computational Auction Mechanisms for Restructured Power Industry Operation*. Kluwer Academic Publishers, 1999.

G.B. Sheblé and G.N. Fahd. Unit commitment literature synopsis. *IEEE Transactions on Power Systems*, 9:128–135, 1994.

G.B. Sheblé and L. Grigsby. Decision analysis solution of the unit commitment problem. *Electric Power Systems Research*, 11:85–93, 1986.

G.B. Sheblé, T.T. Maifield, K. Brittig, and G. Fahd. Unit commitment by Genetic Algorithm with penalty methods and a comparison of lagrangian search and Genetic Algorithm – economic dispatch example. *Electrical Power and Energy Systems*, 18(6):339–346, 1996.

N. Srinivas and K. Deb. Multiobjective optimization using nondominated sorting in Genetic Algorithms. *IEEE Transactions on Evolutionary Computation*, 2(3):221–248, 1995.

D. Srinivasan and A.G.B. Tettamanzi. An evolutionary algorithm for evaluation of emission compliance options in view of the clean air act amendments. *IEEE Transactions on Power Systems*, 12(1):336–341, 1997.

R.E. Steuer. *Multiple criteria optimization: Theory, computation, and optimization.* John Wiley & Sons, 1986.

A. Suppapitnarm, K.A. Seffen, G.T. Parks, and P.J. Clarkson. A Simulated Annealing algorithm for multiobjective optimisation. *Engineering Optimisation*, 33:55–85, 2000.

K.S. Swarup and S. Yamashiro. Unit commitment solution methodology using Genetic Algorithm. *IEEE Transactions on Power Systems*, 17(1):87–91, 2002.

K.S. Swarup and S. Yamashiro. A Genetic Algorithm approach to generator unit commitment. *Electric Power and Energy Systems*, 12(25):679–687, 2003.

S. Takriti and J.H. Birge. Using integer programming to refine lagrangian-based unit commitment solutions. *IEEE Transactions on Power Systems*, 15(1):151–156, 2000.

J.H. Talaq, F. el Hawary, and M.E. el Hawary. A summary of environmental/ecomic dispatch algorithms. *IEEE Transactions on Power Systems*, 9(3):1508–1516, 1994.

E.G. Talbi. A taxonomy of hybrid metaheuristics. *Journal of Heuristics*, 8(5):541–564, 2002.

J.M. Thompson and K.A. Dowsland. A robust Simulated Annealing based examination timetabling system. *Computers & Operations Research*, 25:637–648, 1998.

S.K. Tong and S.M. Shahidehpour. An innovative approach to generation scheduling in large scale hydro thermal power systems with fuel constrained units. *IEEE Transactions on Power Systems*, 5:665–673, 1990.

S.K. Tong, S.M. Shahidehpour, and Z. Ouyang. A heuristic short-term unit commitment. *Transactions on Power Systems*, 6(3):1210–1216, 1991.

C.A. Tovey. Tutorial on computational complexity. *Interfaces*, 32(3):30–61, 2002.

C-l. Tseng, C-a. Li, and S.S. Oren. Solving unit commitment by a unit decommitment method. *Journal of Optimization Theory and Applications*, 105(3):707–730, 2000.

C-l. Tseng and S.S. Oren. A unit decommitment method in power system scheduling. *Electrical Power and Energy Systems*, 19(6):357–365, 1997.

E.L. Ulungu, J. Teghem, P. Fortemps, and D. Tuyttens. MOSA method: a tool for solving multiobjective combinatorial optimization problems. *Journal of Multicriteria Decision Analysis*, 8:221–236, 1999.

R.J.M. Vaessens, E. Aarts, and J.K. Lenstra. A Local Search template. In Reinhard Männer and Bernard Manderick, editors, *Proceedings of the 2$^{nd}$ Conference on Parallel Problem Solving from Nature*, pages 65–74, Amsterdam, 1992. North-Holland.

J. Valenzuela and M. Mazumdar. Probabilistic unit commitment under a deregulated market. In B.F. Hobbs, M.H. Rothkopf, R.P. O'Neill, and H. Chao, editors, *The Next Generation of Electric Power Unit Commitment Models*, pages 139–152. Kluwer Academic Publishers, 2001.

J. Valenzuela and A.E. Smith. A seeded memetic algorithm for large unit commitment problems. *Journal of Heuristics*, 8(2):173–195, 2002.

P.J.M. van Laarhoven and E. Aarts. *Simulated Annealing: theory and applications.* Reidel Publishing Company, Dordrecht, Holland, 1987.

S. Vemuri and L. Lemonidis. Fuel constrained unit commitment. *Transactions on Power Systems*, 7(1):410–415, 1992.

A. Viana and J.P. Sousa. Using metaheuristics in multiobjective resource constrained project scheduling. *European Journal of Operational Research*, 120:359–374, 2000.

A. Viana, J.P. Sousa, and M.A. Matos. GRASP with Constraint Oriented Neighbourhoods: an application to the Unit Commitment Problem. In *Proceedings of the 5$^{th}$ Metaheuristics International Conference – MIC 2003*, Tokyo, Japan, August 2003a.

A. Viana, J.P. Sousa, and M.A. Matos. Using GRASP to solve the unit commitment problem. *Annals of Operations Research*, 120(1):117–132, 2003b.

V.V. Vidal. *Applied Simulated Annealing.* Springer Verlag, 1993.

C. Wang and S.M. Shahidehpour. A decomposition approach to nonlinear multi-area generation scheduling with tie-line constraints using expert systems. *Transactions on Power Systems*, 7:1409–1418, 1992.

C. Wang and S.M. Shahidehpour. Power generation scheduling for multi-area hydrothermal systems with tie line constraints, cascade reservoirs and uncertain data. *IEEE Transactions on Power Systems*, 8(3):1333–1340, 1993.

S.J. Wang, S.M. Shahidehpour, D.S. Kirschen, S. Mokhtari, and G.D. Irisarri. Short-term generation scheduling with transmission and environmental constraints using an augmented Lagrangian Relaxation. *IEEE Transactions on Power Systems*, 10:1294–1301, 1995.

T.G. Werner and J.F. Verstege. An evolution strategy for short-term operation planning of hydrothermal power systems. *IEEE Transactions on Power Systems*, 14(4):1362–1368, 1999.

L.A. Wolsey. *Integer Programming*. Wiley Interscience, 1998.

K.P. Wong and Y.W. Wong. Short-term hydrothermal scheduling. Part I – Simulated Annealing approach. *IEE Proceedings – Generation Transmission and Distribution*, 141(5): 497–501, 1994.

K.P. Wong and Y.W. Wong. Thermal generator scheduling using hybrid genetic/Simulated Annealing approach. *IEE Proceedings – Generation Transmission and Distribution*, 142 (4):372–380, 1995.

K.P. Wong and Y.W. Wong. Combined Genetic Algorithm/Simulated Annealing/fuzzy set approach to short-term generation scheduling with take-or-pay fuel contract. *IEEE Transactions on Power Systems*, 11:128–136, 1996.

A.J. Wood and B.F. Wollenberg. *Power Generation Operation and Control*. John Wiley & Sons, 1996.

R.N. Wu, T.H. Lee, and E.F. Hill. Effect of interchange on short-term hydrothermal scheduling. *IEEE Transactions on Power Systems*, 6(3):1217–1223, 1991.

Y-G. Wu, C-Y. Ho, and D-Y. Wang. A diploid genetic approach to short-term scheduling of hydrothermal system. *IEEE Transactions on Power Systems*, 15(4):1268–1274, 2000.

J. Xu and R.D. Christie. Decentralised unit commitment in competitive energy markets. In B.F. Hobbs, M.H. Rothkopf, R.P. O'Neill, and H. Chao, editors, *The Next Generation of Electric Power Unit Commitment Models*, pages 293–315. Kluwer Academic Publishers, 2001.

H.Y. Yamin. Review on methods of generation scheduling in electric power systems. *Electric Power Systems Research*, 69:227–248, 2004.

H. Yan, P.B. Luh, X. Guan, and P.M. Rogan. Scheduling of hydrothermal power systems. *IEEE Transactions on Power Apparatus and Systems*, 8:1358–1365, 1993.

H-T. Yang, P-C. Yang, and C-L. Huang. Parallel Genetic Algorithm approach to solving the unit commitment problem: implementation on the transputer networks. *IEEE Transactions on Power Systems*, 12:661–668, 1997.

P-C. Yang, H-T. Yang, and C-L. Huang. Solving the unit commitment problem with a Genetic Algorithm through a constraint satisfaction technique. *Electrical Power Systems Research*, 37:55–65, 1996.

S. Yin Wa Wong. An enhanced Simulated Annealing approach to unit commitment. *Electrical Power and Energy Systems*, 20:359–368, 1998.

D. Zhang, P.B. Luh, and Y. Zhang. A bundle method for hydrothermal scheduling. *IEEE Transactions on Power Systems*, 14(4):1355–1361, 1999.

F. Zhuang and F.D. Galiana. Towards a more rigorous and practical unit commitment by Lagrangian Relaxation. *IEEE Transactions on Power Apparatus and Systems*, 3(2): 763–773, 1988.

F. Zhuang and F.D. Galiana. Unit commitment by Simulated Annealing. *IEEE Transactions on Power Systems*, 5:311–318, 1990.

E. Zitzler, M. Laumanns, and L. Thiele. SPEA2: Improving the Strength Pareto Evolutionary Algorithm for multiobjective optimization. In *Proceedings of the EUROGEN 2001 – Evolutionary Methods for Design, Optimisation and Control with Applications to Industrial Problems*, Barcelona, Spain, 2001.

E. Zitzler and L. Thiele. Multiobjective optimization using Evolutionary Algorithms – a comparative case study. In A.E. Eiben, T. Bäck, M. Schoenauer, and H-P. Schwefel, editors, *Proceedings of the 5$^{th}$ International Conference on Parallel Problem Solving from Nature (PPSN-V)*, pages 292–301, Berlin, Germany, 1998. Springer.

E. Zitzler and L. Thiele. Multiobjective Evolutionary Algorithms: A Comparative Case Study and the Strength Pareto Approach. *IEEE Transactions on Evolutionary Computation*, 3(4):257–271, 1999.

E. Zitzler, L. Thiele, and K. Deb. Comparison of multiobjective evolutionary algorithms: empirical results. *Evolutionary Computation*, 8(2):173–195, 2000.

E. Zitzler, L. Thiele, M. Laumanns, C.M. Fonseca, and V.G. Fonseca. Performance assessment of multiobjective optimizers: An analysis and review. *IEEE Transactions on Evolutionary Computation*, 7(2):117–132, 2003.

PROFISSÃO DE FEBRE

quando chove,
eu chovo,
faz sol,
eu faço,
de noite,
anoiteço,
tem deus,
eu rezo,
não tem,
esqueço,
chove de novo,
de novo, chovo,
assobio no vento,
daqui me vejo,
lá vou eu,
gesto no movimento

Paulo Leminski