On decomposition and piecewise linearization in petroleum production optimization

Vidar Gunnerud

Doctoral Thesis

Submitted for the Partial Fulfilment of the Requirements for the Degree of

philosophiae doctor

May 15, 2011

Department of Engineering Cybernetics
Faculty of Information Technology,
Mathematics and Electrical Engineering
Norwegian University of Science and Technology

ISBN 978-82-471-2874-9 (Printed version)
ISBN 978-82-471-2875-6 (Electronic version)
ISSN 1503-8181

ITK Report 2011-7-W
Doctoral thesis at NTNU, 2011:164

Summary

Maintaining good or optimal operations of large and complex petroleum assets is not a trivial task. There are numerous decisions to be made where many of the decisions affects each other. Relying entirely on human interpretation of data is futile; this means that decision support tools are important for efficient and safe operations. Further, as complexity of the assets has increased over the years, so have the requirements for the tools developed to support operational decisions.

Decision support tools come in many forms. The simplest ones would only display measurements in a suitable way, and the operators and engineers would then, based on their knowledge and experience make and implement decisions. Often when measurements are noisy or unreliable, a low pass filter or some uncertainty indication may help. Complex decision support tools may embed model-based estimation and optimization. This work targets methods for optimization-based decision support.

In petroleum assets with rate dependent gas to oil, or water to oil ratios, and with limited gas and/or water handling capacity, it is often a nontrivial task to maximize value throughput. This challenge has been addressed by several commercial and academic actors, as the potential additional values of increased production is large. The motivation for this PhD research has been to attack this real time production optimization problem from a new angle which has many advantages with respect to finding the optimal operational strategy.

The contribution of this work may be divided into four parts. The first is related to modeling. A full field production system consisting of many wells, manifolds and pipelines is challenging to describe in a suitable optimization formulation. The approach in this research has been to transform all nonlinearities into piecewise linear approximations. It is then possible to represent the problem as a mixed integer linear problem, which comes with many advantages with respect to solvability.

As the wells usually are clustered in groups, the second contribution is related to exploration of this structure to be able to solve large full field production systems; in our case more than sixty wells. This is done by decomposing the full field problem into sub-problems for each cluster of wells. The coordination between these sub-problems is handled by using Dantzig-Wolfe decomposition theory.

The decomposed problem naturally lets itself parallelize. Therefore, to further decrease the solution time, parallelization of the solution algorithm is explored to take advantage of the latter years development in computational architectures.

Dantzig-Wolfe decomposition theory has certain limitations when the optimization problem contains integer and binary decisions, as is needed when modeling on/off valves and routing of wells. More precisely, it is not an exact method, and cannot guarantee convergence to the optimal solution, even though, for this class of problems it gets fairly close. However, to overcome this flaw, a branch & price algorithm which handles the integer properties, is proposed and implemented. For this problem it provides an optimal solution.

Preface

This thesis is submitted to the Norwegian University of Science and Technology (NTNU) for partial fulfillment of the requirements for the degree of philosophiae doctor. The doctoral work has been performed at the Engineering Cybernetics department, under the supervision of professor Bjarne A. Foss, and with co-supervision from Bjørn Nygreen, Marta Dueñas Díez and Erik Ydstie. The research was sponsored by the Center for Integrated Operations in the petroleum industry, via the project for real-time production optimization.

Through these years I have been fortunate to interact with numerous people who have offered their advice and inspiration. It will not be possible to mention all by names, however I feel that the following persons deserve special recognition.

To start, I am sincerely grateful for the close and fruitful research collaboration I have shared with my supervisor professor Bjarne A. Foss. For all the good discussions we have had, and the ideas and directions he has given, he certainly deserves a special recognition. A special thanks also goes to professor Bjørn Nygreen, for the many many hours he has spent helping me understand the nature of optimization. Further, I would like to thank Dr. Marta Dueñas Díez and professor Erik Ydsti for introducing me to their industrial and academic networks, and for fruitful discussions.

I have also enjoyed working together with Erlend Torgnes. Our collaboration started by me supervising him during the work on his master thesis. However, in the continuation, we co-authored two journal papers that are an essential part of my PhD thesis. Professor Ken McKinnon should also be mentioned; his insight and eye for details have significantly lifted the quality of my work.

Sincere thanks goes to professor Andrew Conn, Espen Langvik, professor Mikael Rönnquist, professor Asgeir Tomasgard, professor Stein-Erik Fleten, Nina Cecilie Walberg, Randi Vestbø, Eirik Hagem, Pia Glæserud, Jannicke Aschjem Syrdalen, Øystein Dale Hem, Alexander Svensen, Epen Erlingsen, Torgeir Strat, Vilde Hushoved Godø, Brage Rugstad Knudsen and Michael Wartmann, for contributing in papers, sharing ideas, important feedback and organizing student projects.

Finally, I will use this opportunity to thank my family, sister, brother and parents, for their support and for their encouragement when pursuing my interest. Special thanks goes to my father, Jan Gunnerud for giving me valuable insight in the art of making information interesting. A just as special thanks goes to my mum, Reidun Gunnerud for spending countless hours during my childhood helping me learn to read and write.

Contents

Abbreviations

B&B	Branch & bound
B&C	Branch & cut
BD	Benders decomposition
B&P	Branch & price
DWD	Dantzig-Wolf decomposition
ESP	Electric-submersible pump
GOR	Gas oil ratio
GORM	Gas oil ratio model
IO	Integrated operations
IP	Integer programming
IPR	Inflow performance relationship
LD	Lagrangian decomposition
LP	Linear programming
LR	Lagrangian relaxation
MILP	Mixed integer linear programming
MINLP	Mixed integer non-linear programming
NLP	Non-linear programming
NBB	Non-linear branch & bound
OA	Outer approximation
QP	Quadratic programming
RMP	Restricted master problem
RTPI	Real time production improvement
RTPO	Real time production optimization
SOS2	Special ordered sets of type 2
SQP	Sequential quadratic programming
VLP	Vertical lift performance curve
WPC	Well performance curve

Chapter 1

Introduction

1.1 Petroleum production optimization

This opening section will present important aspects of petroleum production optimization. Production optimization will be put into context and further focus on how production engineers solve this problem in their everyday work and which software tools they use for analyses. The concept of mathematical optimization for decision support in production optmization is central to this work. Hence, an overview of this field, in particular the methods which are applied in this work is provided. Finally, the motivation for and contribution of this PhD thesis are reviewed.

1.2 Background on petroleum production

Hydrocarbons for oil and gas production are deposited in subsurface porous rocks. There will always be a sealing rock above a reservoir of hydrocarbons and free gas is usually present above a layer of oil and water. Hydrocarbon assets are produced through wells which are drilled and perforated in hydrocarbon rich zones. This allows reservoir fluids to flow to the surface provided the reservoir pressure is high enough. Wells streams are typically connected to pipelines and transported to a downstream process plant. Its main purpose is to separate oil, water and gas. Oil and gas are exported through export pipelines or ships, and some of the gas may be re-injected into a reservoir for pressure support to increase the recovery of oil. Water is purified and deposited for instance in the sea.

To elaborate further on the exploitation of hydrocarbon assets we will briefly discuss this using a timeline. Reserves are typically discovered by analyses of seismic surveys. Subsequently, promising prospects are investigated closer by drilling exploration wells. Some exploratory wells may produce for a limited time to gain further insight into reservoir behavior. Some key parameters are the bounds of a reservoir or the oil and gas in place, faults and permeability distributions. Given a positive outcome of the above analysis the owners need to decide on how to develop the asset. This includes choice of technology and a longterm production plan. It is quite common to define three production phases; a ramp-up phase, a plateau phase and a decline phase. Due to the large financial investments it is important to start production as early as possible. Hence, production starts from a small number of wells and increases as more wells are drilled and come online. During the plateau rate production is kept fairly constant even though the distribution of oil, water and gas may change. Typically the gas-oil-ratio or the water-oil-ratio increases with time due to depletion effects. There is often an active drilling program, in-field drilling, during the plateau phase. In-field drilling is done to compensate for lowered production from existing wells, and to take advantage of new reservoir knowledge from historical production data and possibly new seismic surveys. This knowledge may provide new opportunities for well placement in order to increase recovery. The plateau rate is characterized by small changes in the reservoir pressure. Pressure compensation is accomplished either by nature itself through a large gas cap or a strong water drive, or by injecting gas and water. The depletion phase is characterized by an exponential drop in the pressure. This provides challenges to production. It is common to enhance production from wells through gas-lift or submersible pumps. Gas-lift reduces the density of the liquid column in a well whereas a pump increases downhole pressure. There are many other options not mentioned here like chemical injection, microbial injection and thermal recovery.

Norway is a large oil and gas producer, and all its production comes from offshore installations. Hence, this theses will have a bias towards these types of systems.

1.3 Petroleum production optimization set into context

As alluded to above the development of a field asset requires planning on multiple horizons. On a long-term horizon, strategic decisions are made on field development (e.g., choice of technology and export options, investment strategies, and recovery strategies despite uncertainties) Saputelli et al. (2007). For offshore assets, the choice of technology may include subsea solutions and the issue of processing the reservoir fluid either offshore or onshore. Export choices include pipelines or liquefied natural gas tankers for gas, and pipelines or oil tankers for heavier components. Furthermore, option management may point to a flexible, and thereby more costly development to allow for future development, such as tie-ins from possible neighboring assets. The analyses and subsequent development plan seek to maximize the net present value of the asset or maximizing oil recovery.

On a medium-time horizon, typically 3 months to 2 years, production rates and, if applicable, injection rates are decided. Depending on the life cycle of an asset, decisions may also involve a drilling program. During the ramp-up stage it is important to plan, drill, and commission new wells to reach some predefined plateau rate as soon as possible. During plateau production, there may be an in-field drilling program for production and injection wells. This program involves decisions about the location and completion of wells. During the decline phase of a field, lift technology may be an issue involving decisions (e.g., the distribution of limited amounts of lift gas between wells). A detailed reservoir simulator is usually an important planning tool on the medium time horizon. A reservoir simulator model may be very complex if the geology is complex for reasons like heterogeneities, faults and shale layers, to represent flow patterns accurately. Reservoir models may contain millions of grid cells for large fields. Furthermore, complexity increases if phase behavior is complicated. Gas-condensate reservoirs often necessitate the use of a compositional simulator instead of a black-oil simulator to get accurate prediction of phase behavior, which increases computational load significantly because of thermodynamics. Such models are updated, quite infrequently, based on historical data and other knowledge.

On a short-time horizon, typically days to weeks, production optimization in which both the subsurface part (i.e., reservoir and wells) and the surface part (i.e., collection and downstream production equipment) of the system is important. We denote this real-time production optimization (RTPO). RTPO is a commonly used abbreviation in the industry, but there seems to be some confusion on the actual meaning of the term: therefore, we propose a definition starting with the term real-time.

- *Real time is the frequency at which meaningful decisions can be made.*

RTPO is vaguely used for a multitude of production improvement programs covering improved visualization, new surface or subsurface instrumentation, an improved workflow supported by a decision support tool, and as a proxy model to predict coning effects and retuned low-level controllers. This broad definition may introduce unnecessary and resource-consuming confusion within the industry, because optimization is actually a well-defined term. We therefore distinguish between real-time production improvement (RTPI) and RTPO as follows:

- *Real-time production improvement (RTPI) includes all improvement programs* *that use real-time data.*

- *Real-time production optimization (RTPO) includes all decision-support tools, with a typical decision horizon of days to weeks, in which a mathematical optimization problem is an essential component.*

Hence, RTPI is a much wider term than RTPO. Similar definitions of RTPO are found several places in the literature Saputelli et al. (2003).

Production may be constrained by reservoir conditions, such as coning effects or the production equipment like pipeline capacity or downstream water-handling capacity. Constraints may hence move from one part of the system to another over time. Water production may be low early and increase dramatically during the decline phase of a reservoir, thereby making water-handling capacity an issue. Decision variables in RTPO include production and possibly injection rates, artificial lift inputs like lift-gas rates and electric-submersible pump (ESP) rates, and routing of well streams.

There are also faster time scales, including supervisory and regulatory control, such as those presented in several references (e.g., the operational hierarchy in Figure 1 Saputelli et al. (2007)). This thesis however focuses on the RTPO problem.

1.3.1 How production engineers optimize production

Modern offshore organizations include at least one onshore team in addition to an offshore platform-based team. These teams are well connected through frequent videoconference meetings and other collaboration tools and common workflows Hepsø (2006). RTPO as discussed in this thesis mainly resides with the onshore team who performs analyses and makes recommendations to the offshore team where the final decisions are made and implemented. The reason for ignoring a recommendation may hinge on uncertainties related to the production data and models. Therefore recommendations which involve a change from a current operating point will only be implemented if there is a substantial gain to be made.

The daily workflow for a production engineer typically starts with a review of production and individual well performance. Sometimes it is also necessary to act on an abnormal situation, for example an underperforming well. Later during the morning, videoconferencing is used to discuss and agree on the current situation, and make or adjust plans and production goals for the next 24 hours. In terms of content such meetings extend far beyond RTPO and include light maintenance, logistics, reservoir issues and drilling.

The production engineers need to analyze the current situation and recommend a production strategy for the next day. This may require analyses on an individual well level, cluster level, i.e. a set of neigboring wells, as well as the complete system. An optimization-based tool for allocation and routing needs to fit into the daily workflow of the onshore production engineers and be flexible in the sense that it can be integrated into similar analytical tools. This may for instance be done through a portal solution in which the allocation and routing optimizer is one of several analysis tools.

1.3.2 Production optimization in an IO perspective

Integrated Operations (IO) is a hot topic in the petroleum industry today. There exists several definitions, however, the IO-Center at NTNU describes it like this;

- *IO is the integration of people, work processes and technology to make smarter decisions and better execution. It is enabled by the use of ubiquitous real-time data, collaborative techniques and multiple expertise across disciplines, organizations and geographical locations.*

Alternative extensively used terms to IO are i-field used by Chevron, Field of the Future by BP, Smarter Oilfield by IBM, Smart Fields by Shell and Digital Oil Field by Baker Hughes. They all have more or less the same meaning. However, some definitions focus more on the work process and soft issues rather than technology and vice versa.

RTPO plays an important role in IO. The flow of information from the different sensors, valves, pumps etc. in a petroleum production asset is enormous. Far from all this data is used in a sensible way today and several disciplines are involved. Models and methods that use this information to improve decisions are therefore very welcome in the industry. RTPO is an excellent example since it provides a means to integrate multiple information sources for decision support in multi-skilled teams. Figure 1.1 is meant to illustrate the fact that mathematics-based production optimization will rely on suitable work processes and appropriate visualization tools to fully exploit its potential.

1.3.3 Software tools

Production engineers in different petroleum companies, and even various assets within the same company, use different tools and software packages. This section adressess commonly used software packages. Since the research performed in this

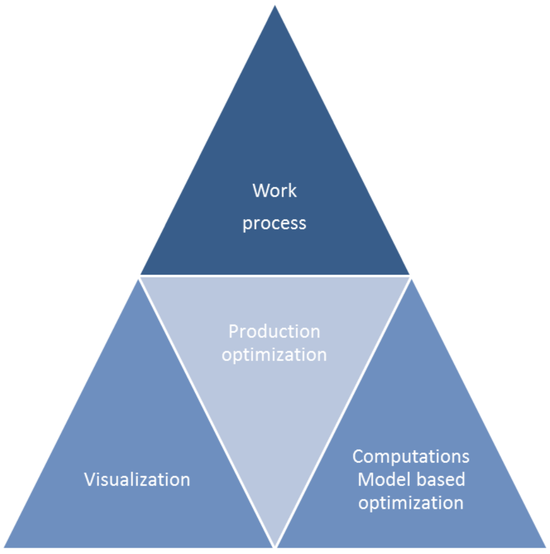


Figure 1.1: This figure illustrates the link between mathematical optimization and other key issues in production optimization; work processes and visualization.

thesis is inspired by petroleum production at the Troll oil and gas field operated by Statoil it is biased towards the tools that Statoil is using. Thus, it provides a concrete example of the mix of tools which are applied in an operational setting.

The main two groups of tools are collaboration tools and analysis tools. Collaboration tools are important in production engineer's daily workflow and they use several of them. The most important at Troll is the Process Data Portal (PDP). This tool is used to present and trend all data relevant for production. This includes pressures, temperatures and valve positions at different locations in the subsea network, and downhole in the wells. Production rate estimates are also presented. Further, there is a tool which aggregates the production information into daily reports. This report is also used by managers and other staff in addition to the production engineers. A specially designed Excel spread sheet is used to create production plans for the coming week.

A less frequently used software tool than the ones mentioned above is used for loss reports. This tool presents an estimate of how much higher the production could have been if everything had worked perfectly. Low loss values indicate efficient production. Production Surveillance, Analysis & Forecasting (OFM) from Schlumberger is used to organize data from well tests, and to predict future behavior of the wells.

At Troll the production engineers mainly use three analysis tools for production optimization. The first is FlowManager from FMC Technologies. FlowManager is used to estimate production rates from the different wells based on pressure and temperature measurements. It includes simple models of the inflow from the reservoir into the well, the vertical part of the well and the choke valve. These models are updated regularly, typically three times a year, from well tests. Within FlowManager, there is a package for production optimization, MaxPro. This package may be used to optimize production from parts of the production subsea system. This means allocating production between wells and routing a well stream to one of several pipelines. In addition the engineers may use an optimization package from Petroleum Experts called the General Assignment Package (GAP). In both the latter cases the Gas Oil Ratio Model (GORM), which is an in-house Statoil model, is used to estimate the relation between gas and oil inflow to the wells.

Both production optimization software packages mentioned above use Sequential Quadratic Programming (SQP) algorithms as their core algorithm since the models, e.g. the pressure drop in flow lines, are nonlinear. Further, the problems contain binary decisions related to turning wells on and off, and routing them to different production lines. MaxPro handles the binary variables heuristically, while GAP needs to fix them before running the optimization. Hence, to evaluate different routing alternatives with GAP one may need to apply an exhaustive search. It should be noted that available software tools are not able to perform an optimization analysis of the complete Troll oil field. This observation was an important starting point for this PhD research project.

1.4 Overview of optimization techniques

This section aims at giving the reader a brief overview of the problem classes and algorithms that should be considered when attacking the RTPO problem. A reader familiar with optimization may skip parts of this section.

This section starts with linear programming (LP) problem as this is the steppingstone for other classes of problems. Further, the special case of LP where some of the variables are constrained to be integer variables (MILP) are discussed, before nonlinear problems (NLP) and nonlinear problems with integer variables (MINLP) are introduced. Piecewise linearization is of great importance to this work, the same is true for decomposition methods, and a short introduction is therefore given. Decomposition methods potential in combination with the recent advances in parallel computer architecture is also mentioned. The section ends with some comments on heuristics and derivative free optimization. The concepts are deepened in the core sections of this thesis where the treatment is more formal than in this introductory chapter.

1.4.1 Linear programming

The work on linear optimization problems, as we know it today, started with Dantzig in the late 1940s and his development of the simplex method in 1947. The method enabled engineers and economists to develop large models and analyze them in a systematic and efficient way. His discovery coincided with the development of the first computers, and the simplex method is one of the first applications that explore the power of this new revolutionary technology.

In the early 1980s it was discovered that by using techniques from nonlinear programming it was possible to solve large linear programs efficiently. An important characteristic of these methods is that the inequality constraints of the problem should be strictly satisfied, and the group of methods is known as interior-point methods. By the early 1990s, a subclass of these methods, the primal-dual methods, distinguish themself as more efficient than the others, and proved to be strong competitors to simplex algorithms on large LP problems (Nocedal and Wright, 2006).

There are two groups of algorithms which have proven to work well for LP problems, the simplex and interior-point algoritms. The simplex algorithm solves problems in exponential time. However, in practice solution time is proportional to the number of linear equations m. It can be shown that interior-point algorithms solve LP-problems in polynomial time. One advantage of simplex algorithms is the fact that they can benefit from a hot-start situation, i.e. when there exists a good estimate of the active set. This is often the case in real-time applications like RTPO.

Interior-point algorithms share several common features that distinguish them from the simplex method. Each interior-point iteration can make significant progress towards the solution, but is expensive to compute. The simplex method on the other hand, usually requires a large number of inexpensive iterations. Geometrically speaking, the simplex method works its way around the boundary of the feasible polytope, testing a sequence of vertices in turn until it finds the optimal one. Interior-point methods approach the boundary of the feasible set only when converging to the optimal solution.

From the beginning of the 1940s and until today, the implementation of the simplex and interior-point methods has improved continuously and tremendously. Currently LP-problems with millions of variables are readily solved on state-of-the-art computers (Dantzig and Thapa, 2003a) (Nocedal and Wright, 2006).

A general LP implies the minimization or maximization of a linear function subject to linear constraints. The feasible set is a polytope, i.e. a convex, connected set with flat polygonal faces. The contours of the objective function are planar, and hence, the solution, if unique, is in the cross section of a subset of the linear constraints. A general LP maximization problem is shown in (1.1) where it may be noted that the inequalities can be transformed to equalities by adding slack/artificial variables.

$$\max_{x \in \Re^n} \quad c^T x \tag{1.1a}$$

subject to

$$Ax = b \tag{1.1b}$$

$$x \geq 0 \tag{1.1c}$$

Where c and x are vectors in $\Re^n$, $b \in \Re^m$ and $A \in \Re^{m \times n}$. The solution of (1.1) is characterized by the first order Karush-Kuhn-Tucker (KKT) conditions. Under mild conditions, convexity of the problem ensures that these are sufficient conditions for a global maximum. The Lagrangian function for (1.1) is given by

$$\mathcal{L}(x, \lambda, s) = c^T x - \lambda(Ax - b) - sx \tag{1.2}$$

and the KKT conditions for the optimal solution of (1.1)

$$A^T \lambda + s = c \tag{1.3a}$$

$$Ax = b \tag{1.3b}$$

$$x^T s = 0 \tag{1.3c}$$

$$x, s \geq 0 \tag{1.3d}$$

where $\lambda \in \Re^m$ and $s \in \Re^n$ are the dual variables (also known as Lagrangian multipliers) of the constraints (1.1b) and (1.1c).

Simplex algorithms

There is a number of variants of the simplex method; the one described in a few words here is known as the primal revised simplex method. Another commonly used version is the dual simplex method which works particulary well together with branch & bound solving MILP problems due to efficient hot-start.

The principle of the simplex method is to move from one vertex to an adjacent one for which the basis differs in exactly one dimension, i.e. which variables are allowed to be nonzero. On most steps (but not all), the value of the objective function $c^T x$ increases. Another type of step occurs when the problem is unbounded, where one can move infinitely far without ever reaching a vertex. A major issue at each simplex iteration is to decide which variable to remove from the basis and which to replaced it with from outside the basis, and different strategies exist. For a comprehensive discussion on simplex, see Dantzig and Thapa (2003a).

Interior-point algoritms

As for the simplex method consider the linear problem (1.1). The primal-dual method finds the solution by applying variants of Newton's method to the three equalities in (1.3) and modify the search directions and step length so that the inequalities $x, s \geq 0$ are satisfied strictly at every iteration. The equations (1.3) are linear except for (1.3c), which is only mildly nonlinear, and hence not difficult to solve by themselves. However, the problem becomes much more tricky due to the non-negativity requirement (1.3d), which gives rise to all the complications in the design and analysis of interior-point methods.

The procedure for determining the search direction originates from Newton's method for nonlinear equations. A linear model for (1.3) around the current point is formed and a search direction $(\Delta x, \Delta\lambda, \Delta s)$ is obtained. A full step along the Newton's search direction may violate the bound $x, s \geq 0$, so a line search is performed

$$(x, \lambda, s) + \alpha(\Delta x, \Delta\lambda, \Delta s)$$

to determine the line search parameter $\alpha \in (0, 1]$. Often one can take only small steps along this direction before violating the conditions $x, s \geq 0$. Hence, the pure Newton direction, sometimes known as the affine scaling direction, does not allow much progress toward a solution.

Most primal-dual methods use a less aggressive Newton direction, one that does not aim directly for a solution of (1.3a) - (1.3c), but rather for a point whose pairwise products of $x_i s_i = \sigma^k \mu^k$, and where $\sigma^k \mu^k$ is reduced at each iteration k, where μ^k is the current duality measure and $\sigma^k \in (0, 1]$ is the reduction factor. The optimal solution is found when (1.3) is satisfied, i.e. $\mu^k = 0$ and $x, s \geq 0$

Advances in linear programming

As mentioned, advanced LP solution methods started with Dantzig and the simplex algorithm in the late 1940s. The next quantum leap came with the interior-point methods in the 1980s. However, one has never seen as much progress in the efficiency of the simplex method as after the interior-point method made its entry. And even though interior-point works very well on large LP problems, it is not as efficient in combination with integer variables. This is due to the dual simplex method's capability of handling hot-starts after introducing new constraints. In the last decade, the largest gain in algorithmic development are related to the handling of sparsity in large matrices. To illustrate the progress, the improvement on a standard problem with 50000 equality constraints was close to 2.0×10^6 times from 1987 to 2002 applying state-of-the-art software and hardware. About 2.5×10^3 times was due to algorithmic improvements and the rest (800 times) was related to hardware advances (Grossmann, 2011).

Two very efficient simplex implementations are used in the CPLEX and XPRESS solvers which also handle integer variables. The open source code IPOPT is a very efficient interior-point implementation.

1.4.2 Mixed integer linear programming

The first serious attempts to describe mixed integer linear programming (MILP) models started in the 1960s and 1970s with single-stage and multi-stage planning problems of the type that arise regularly in practice. A general MILP, which extends the LP problems outlined above, is the minimization or maximization of a linear function subject to linear constraints where one or more of the variables are integer, e.g. (1.4).

$$\max_{x \in \Re^n, y \in \mathbb{Z}^l} \quad c^T x + h^T y \tag{1.4a}$$

subject to

$$Ax + Hy = b \tag{1.4b}$$

$$x, y \geq 0 \tag{1.4c}$$

Where c and x are vectors in $\Re^n$, $h \in \Re^l$, $y \in \mathbb{Z}^l$, $b \in \Re^m$, $A \in \Re^{m \times n}$ and $H \in \Re^{m \times l}$. Simplex and interior-point are general methods which have proven to work very well for general linear programs. In contrast to LPs, which in most cases can be solved efficiently, and in polynomial time as a worst case, integer programming problems are NP hard and in many practical situations no efficient algorithm exists. Given that a general polynomial time algorithm to solve MILP was found, it would enable us to solve a large class of problems efficiently, including the famous traveling salesman problem. However, most scientists believe that no such algorithm exists (Williams, 2005).

Most exact methods for solving MILPs today fall into one of the following two categories:

- Cutting plane methods
- Enumerative methods (where branch & bound methods is a subclass)

It is also quite common to combine the two approaches in branch & cut methods. Below cutting planes, branch & bound (B&B) and branch & cut (B&C) methods are presented.

Cutting plane methods

Since first introduced by Gomory (1958) the cutting plane methods have been implemented in many different variants, however the general idea persists. The integrality requirements are relaxed and the problem is solved as an LP. Given that the solution of the relaxed problem has integer values for all variables originally required to be integer, the problem is solved. If not, additional constraints, i.e. cutting planes, are added that make prior non-integer solutions infeasible. This procedure continues iteratively until an optimal integer solution is found. For more on cutting planes see the book by Wolsey (1998).

Branch & bound

Branch & bound enumerates through a search tree while it introduces new constraints that divide the LP solution space into subspaces which removes prior feasible LP solutions. B&B methods have shown great results and are currently by far the most used methods for solving problems where integer variables play an important role (Williams, 2005). Through solving changing versions of the original problem, the goal is to gradually obtain better lower and upper bounds, and terminate when the gap is less than some predefined value.

The algorithm starts by solving a LP relaxation of (1.4), known as the root node, which establishes the first upper bound, i.e. $y \in \mathbb{Z}^l$ is replaced by $y \in \Re^l$. The next step is to identify y's in this solution with non-integer values. One of the y's is selected to branch on, based on some branching rule (Nemhauser and Wolsey, 1999), and thereby two new child subproblems is added to a list of unsolved problems known as open nodes. This variable may be named $\tilde{y}$, and the value of it may be represented by $\tilde{y}^{LP}$. The first child node will have a upper bound on $\tilde{y}$ equal to $\lfloor \tilde{y}^{LP} \rfloor$, while the second will have a lower bound equal to $\lceil \tilde{y}^{LP} \rceil$.

The solution space is thus divided into two disjunct parts, while the part in between, not containing any integer feasible solutions, is cut off. For this reason, the LP relaxations of a child node will always give a lower upper bound solution compared to its parent node. This process continuous through the tree search where the child nodes inherit its parents branching rules, in addition to its own rule.

In addition to choosing $\tilde{y}$, a B&B algorithm needs to decide which node to handle next, i.e. the search strategy. There are three basic strategies and numerous combinations of them. The depth first strategy will choose child nodes of the current node, fast going deep into the tree, generating many and tight branching rules and hopefully create a feasible solutions fast. The best first strategy will choose the unsolved node with the highest upper bound, which is the node that potentially can create the best feasible solution. The breadth first will basically choose the unsolved nodes closest to the root node. For more on how to choose or combine these strategies, see T'kindt et al. (2004) and Williams (1993).

If the LP relaxed solution of a node is feasible with respect to all integer variables i.e. that all y's take non-negative integer values, this solution is a feasible solution of (1.4), and a lower bound. All nodes with an upper bound lower than the best feasible solution, may hence be removed. Nodes may also be removed if the LP relaxation is infeasible. The algorithm terminates when there are no more unsolved nodes or if the duality gap is satisfied.

Branch & cut

B&C is a hybrid solution method that introduces cutting constraints in addition to the dividing constraints generated by B&B. When B&C is used to solve a MILP, an initial solution, i.e. the root node, is established by solving the problem relaxed as an LP. If the solution obtained has fractional values for one or more of the integer variables, a cutting plane algorithm is initialized. It is used to generate constraints that are violated by the current fractional solution, but which are satisfied by all feasible integer points. At some point, dependent on the chosen strategy, the algorithm starts branching and divides the LP solution space into subspaces using B&B. Basically the algorithm alternates between introducing constraints that do not cut away any integer feasible points, i.e. cutting planes, and constraints that divide the LP solution space into subspaces, i.e. B&B. For a more thorough introduction to B&C the reader is referred to Mitchell (2001) and Wolsey (1998).

Advances in mixed integer linear programming

When MILP modeling started in the 1960s and 1970s, solvers were only able to solve toy problems, and so efforts were concentrated on simple and rapid heuristics. Motivated by the successes in tackling the traveling salesman problem and pure binary problems by strong cutting planes in the early 1980s, a systematic study of the polyhedral structure of production planning problems was initiated. As a results, today there is considerable knowledge on how to formulate many simple production planning submodels as MILPs. Such knowledge, combined with the remarkable progress of general MILP solvers, enables the solution of many practical production planning problems that were considered far out of reach fifteen or so years ago.

Since MILP solvers solve LP problems in each node in the B&B tree, MILP solvers always rely on efficient simplex implementations. Hence, MILP solvers benefit from the remarkable efficiency increase in LP solvers. All leading MILP solvers today are using some version of B&C.

1.4.3 Non-linear programming

Non-linear programming (NLP) is the term used to describe an optimization problem when the objective or constraints functions are nonlinear. As LP and MILP problems are well studied over several decades only some of the subclasses of NLPs are equally well treated. Hence, there are no efficient methods for solving general NLPs. Even simple looking ones with as few as ten variables can be extremely challenging, while problems with a few hundred variables can be intractable. Methods for general NLPs (1.5) can therefore take several different approaches, each of which involves some compromise (Boyd and Vandenberghe, 2004). A common formulation is given below.

$$\max_{x\in\Re^n} \; f(x) \tag{1.5a}$$

$$s.t.$$

$$g(x) \geq 0, \qquad g : \Re^n \rightarrow \Re^m \tag{1.5b}$$

$$x \geq 0 \tag{1.5c}$$

$f(x)$ is a function mapping the vector x to a scalar value, while $g(x)$ is a vector of functions.

NLP problems can be divided into two main classes, convex and non-convex problems. A convex optimization problem is one in which the objective function (1.5a) is concave, or convex for a minimization problem, and the feasible set is convex. The reason for dividing the NLP problems into these groups is that for the convex ones, all local optimal solutions are also global optimal solutions. For the non-convex problems however, this is not true, since there is no way to characterize the global solution using only local information. It may be hard to verify convexity for NLPs. It such cases it is necessary to assume that they are non-convex.

The same techniques are applied for both groups of problems. However, for non-convex problems, starting points, convexification techniques and metaheuristics are important to increase the chance of finding the global optimal solution.

The most popular solution methods for NLPs are interior-point methods and Sequential quadratic programming (SQP) algorithms (Nocedal and Wright, 2006). These methods are generally considered the most powerful algorithms for large-scale nonlinear programming. The principal ideas of interior-point methods and SQP are described briefly below.

Interior-point algorithms

The success of interior-point methods for LPs stimulated renewed interest in them for NLPs. A new generation of methods and software for NLPs emerged in the late 1990s. Numerical experience indicated that interior-point on average was faster than SQP on large problems, particularly when the number of free variables were large. They were not yet necessarily robust, but significant advances has emerged on their design and implementation.

Some of the key ideas, such as primal-dual steps, carry directly over from LP. However, the treatment of non-convexity, the strategy for updating the barrier parameter in the presence of nonlinearities, and the need to ensure progress toward the solution, are several of the important new challenges that arise.

The Lagrangian function and KKT conditions for the nonlinear program (1.5) can be written as:

$$\mathcal{L}(x, \lambda, s) = f(x) - \lambda g(x) \tag{1.6}$$

$$\nabla f(x) - \nabla g(x)\lambda = 0 \tag{1.7a}$$
$$g(x) \geq 0 \tag{1.7b}$$
$$x^T s = 0 \tag{1.7c}$$
$$s \geq 0 \tag{1.7d}$$

As for the linear case, (1.7c) is replaced with $x_i s_i = \sigma^k \mu^k$, and the KKT conditions are solved for a sequence of positive parameters $\sigma^k \mu^k$ that converges to zero, while maintaining $x, s > 0$. The goal is to converge to a point which satisfies the KKT conditions (1.7).

Compared to the KKT conditions of the LP problem (1.3) the KKT system is no longer linear, and hence, solving it may provide several challenges. When applying Newton's method to (1.7), several modifications and extensions are enforced to obtain a robust solution algorithm (Nocedal and Wright, 2006) (Boyd and Vandenberghe, 2004).

Sequentiel quadratic programming

SQP is an efficient method for solving nonlinear constrained optimization. As for the interior-point method we consider the inequality constrained problem (1.5). The SQP approach can be used both in line search and trust-region frameworks, and is appropriate for both small and large problems.

The idea behind the SQP approach is to approximate the NLP problem (1.5) by a convex quadratic programming (QP) subproblem at the current iterate x_k, see (1.8) stated below. The solution of this QP problem, using for instance an active-set method, provides a search direction p.

$$\max_{p} \quad f(x_k) + \nabla f(x_k)^T p + 1/2 p^T \nabla^2_{xx} \mathcal{L}(x_k, \lambda_k) p \tag{1.8a}$$
subject to
$$\nabla g(x_k) p + g(x_k) \geq 0 \tag{1.8b}$$

The active-set method has several similarities to the simplex method, which actually is an active set method for LP problems. In essence, the simplex method starts by making a guess of the optimal active set, then repeatedly use dual information to drop one constraint from the current estimate of active constraints, and add a new one, until optimality is detected. Active-set methods for convex QPs differ from the simplex method in that the iterates are not necessarily vertices of the feasible polytop. It finds a step from one iterate to the next by solving a quadratic subproblem in which some of the inequality constraints, and all equality constraints are imposed as equalities i.e. the active-set. The algorithm terminates when the solution does not change from one iteration to the next, and the Lagrangian multiplier for all the active inequalities are zero or negative.

When using a line search approach, the new iterate is found by $x_{k+1} = x_k + \alpha \cdot p$ where α is a positive scalar. The method typically applies a merit function (1.9), and does a back tracking line search on this to decide the line search step α.

$$\phi(x;\mu) = f(x) + \mu \sum_{i=1}^{m} \max\{0, -g_i(x)\} \tag{1.9}$$

As an alternative to the line search approach, it is possible to use a trust region method to find the new iterate x_{k+1}. The usual way to handle this is to include the following constraint into the above QP problem.

$$||p|| \leq \Delta_k \tag{1.10}$$

Δ_k defines the trust region for the quadratic approximation at x_k. It will be expanded or reduced from one iteration-point to the next according to the quality of the approximated QP problem (Nocedal and Wright, 2006).

Advances in nonlinear programming

The SQP algorithm is considered the most robust of the two, however, the largest problems are solved with interior-point methods. By the late 1990s, a new generation of methods and software for nonlinear programming had emerged. Numerical experiments indicated that the interior-point method outperforms SQP on large problems, particularly when the degrees of freedom is large (Nocedal and Wright, 2006). Significant advances are still being made, and there are examples of mildly nonlinear problems with more than a million variables being solved with IPOPT (Wachter and Biegler, 2005).

Important solvers on NLP problems are SNOPT and MINOS which originates from Stanford University, and IPOPT from Carnegie Mellon University and IBM. Information about all this solvers are e.g. found on the web page of the General Algebraic Modeling System GAMS (2011).

1.4.4 Mixed integer nonlinear programming

MINLP problems is a very challenging class of problems, and it is mostly during the last two decades interesting results from the algorithmic development have emerged. Increased computational power is of course one important factor in additional to algorithmic improvements. In many ways MINLP solution methods are based on elements from both NLP and MILP techniques, put together in a framework to handle both the nonlinear relations and integer variables.

The MINLP problem is stated below:

$$\max_{x \in \Re^n, y \in \mathbb{Z}^l} f(x,y) \tag{1.11a}$$

subject to

$$g(x,y) \geq 0, \qquad g : \Re^{n+l} \rightarrow \Re^m \tag{1.11b}$$

$$x, y \geq 0 \tag{1.11c}$$

There are basically two approaches for solving (1.11), either a nonlinear branch & bound (NBB) or an outer approximation (OA) method; or a combination of the two.

The NBB method uses a B&B framework. In each node of the search three, it will relax the integer variables, and solve the NLP's with an interior-point or an SQP algorithm. In the same way as for MILP problems, B&B introduces new constraints on the integer variables in the NLPs to gain integrality.

If the problem is convex, except for the integer variables, the solution of the sub-NLPs are globally optimal. This will enable the algorithm to keep track of the upper bound through the search tree, and hence guarantee a globally optimal solution to the overall problem, in the end. On the other hand, if the NLP's are not convex, the solution of the sub-NLPs could be sub optimal, and hence, the solution is no longer necessarily an upper bound. Therefore, the algorithm must take care when cutting off nodes due to the lower upper bound. Heuristic cut-off rules may be used, so that sub-NLPs that might have a sub optimal solution not necessarily are cut off. In some cases it is possible to convexify the sub-NLPs by relaxing the constraints and objective function into a convex problem. The solution of this problem can then be used as an upper bound on the original sub-NLP.

The OA method uses a somewhat different approach to solve (1.11). It creates an outer, linear approximation of the problem and solves this as an MILP. Each iteration starts by solving this problem, creating an upper bound. Further, the integer variables y are fixed and the resulting NLP problem is solved. If the solution is feasible, it provides a lower bound for (1.11). To strengthen the MILP relaxation, a new cut is added based on this feasible solution. On the other hand, if the solution is infeasible, a related feasibility problem is solved, and the solution of this gives rise to cuts that strengthen the MILP relaxation. The algorithm finds an optimal solution in a finite number of iterations provided that the problem is convex. In the case of non-convexity, the NBB method or a hybrid is more appropriate. For more on solutions of MINLP problems see Bonami et al. (2008).

Advances in MINLP problems

Solving this class of problems is still considered a science, rather than a technology. Even really small problems can be very hard to solve. However, there is interesting research with a potentially large impact going on in this area.

Some of the important software packages and solvers for these classes of problems are the BONMIN solver within the COIN-OR environment, and the BARON solver developed at Carnegie Mellon University. The first one is exact for convex problems, but should only be considered as a heuristic in the non-convex case. The latter on the other hand, includes convexification techniques and is thus able to create an upper bound also for non-convex problems (Tawarmalani and Sahinidis, 2002).

The size of the problems solved today is highly dependent on the problem characteristics, but numbers of variables could be in the order of thousands if the integer properties are not too challenging.

1.4.5 Piecewise linearization of nonlinear functions

Nonlinear constraints, compared to linear relations between variables, complicates an optimization problem. A quite common technique is to piecewise linearize these nonlinearities and transform a NLP into a linear problem with some binary requirements given that the nonlinearities make the problem non-convex. The binary requirements could either be explicit included in the optimization problems as variables or implicitly handled by the algorithm through a specialized branch & bound search. Below a brief description of two alternative approaches are given.

A nonlinear function, e.g. $f(x) = x^2$, could be piecewise linearized as shown in Figure 1.2. This could be done by adding weighting variables to the breakpoints; origin, A, B and C, here named $\lambda_1, ..., \lambda_4$, and replacing $f(x) = x^2$ with (1.12). However, these equations do not require that only two adjacent λ's can be non-zero, which has to be the case if the nonlinearity should be represented by the piecewise linear line.

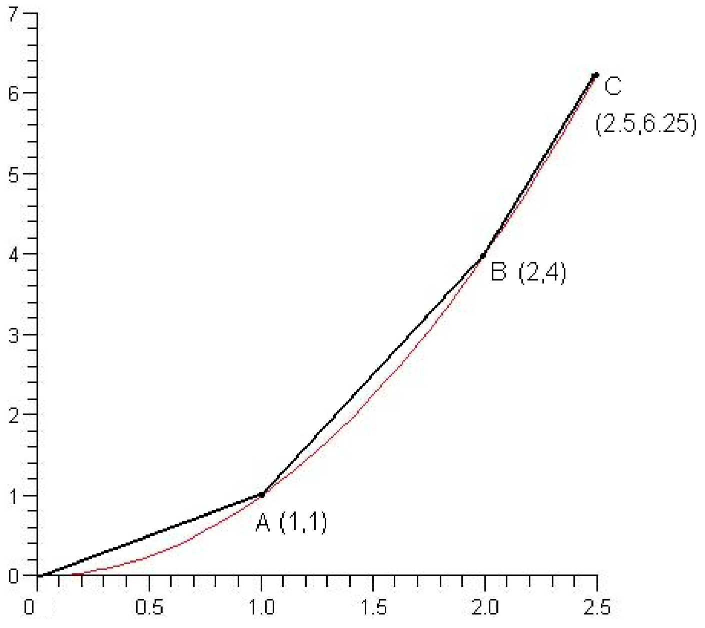


Figure 1.2: Piecewise linear function $f(x) = x^2$

$$\begin{aligned} x &= 0\lambda_1 + 1\lambda_2 + 2\lambda_3 + 2.5\lambda_4 && (1.12a) \\ f(x) &= 0\lambda_1 + 1\lambda_2 + 4\lambda_3 + 6.25\lambda_4 && (1.12b) \\ 1 &= \lambda_1 + \lambda_2 + \lambda_3 + \lambda_4 && (1.12c) \end{aligned}$$

To cope with this unintended effect it is possible to assign binary variables to each interval, $\delta_1, \delta_2, \delta_3$, and let the sum of the binary variables equal one, see (1.13).

$$\begin{aligned} \lambda_1 &\leq \delta_1 && (1.13a) \\ \lambda_2 &\leq \delta_1 + \delta_2 && (1.13b) \\ \lambda_3 &\leq \delta_2 + \delta_3 && (1.13c) \\ \lambda_4 &\leq \delta_3 && (1.13d) \\ 1 &= \delta_1 + \delta_2 + \delta_3 && (1.13e) \end{aligned}$$

Alternatively, some solvers let the user defined $\lambda_1, ..., \lambda_4$ as a Special ordered set of type 2 (SOS2). Given circumstances like when the LP relaxation of the problem results in an interpolation between the origin and C, the solver would introduce a branching rule that removes this solution, e.g. create two subproblems where the first allows interpolation between origin, A and B, and the second allows interpolation between B and C.

The paragraphs above addresses single input - single output functions. This reformulation is also possible to generalize to multi input - multi output functions, though with an increased computational cost.

Suppose $z = g(x, y)$ is a nonlinear function of x and y. A grid of values of (x, y) and associated weighting variables λ_{ij} is defined. If the values of (x, y) at the grid points are denoted by (X_i, Y_j) the function can be approximated by the following equations:

$$x = \sum_i \sum_j X_i \lambda_{ij} \tag{1.14a}$$

$$y = \sum_i \sum_j Y_j \lambda_{ij} \tag{1.14b}$$

$$z = \sum_i \sum_j g(X_i, Y_j) \lambda_{ij} \tag{1.14c}$$

$$1 = \sum_i \sum_j \lambda \tag{1.14d}$$

In addition, only neighboring λ's should be non-zero, which can be imposed by:

$$\xi_i = \sum_j \lambda_{ij} \tag{1.15a}$$

$$\eta_j = \sum_i \lambda_{ij} \tag{1.15b}$$

for all i, j. ξ and η are two separate SOS2 sets which enforces the solution to lie in a rectangle between four neighboring λ's. However, only a triangle of neighboring weighting variables is needed to represent interpolation in two dimensions. Hens using four corner points will result in non-uniqueness where different combinations of values of the weighting variables represent the same coordinate (x, y), but with a different value on z. It is possible to get around this by introducing a third SOS2 set limiting the neighborhood to only a triangle. For a more comprehensive explanation, please see Williams (2005), and for alternative formulations DAmbrosio et al. (2010).

1.4.6 Decomposition techniques

When a problem becomes too large or too challenging to solve within reasonable time, decomposition approaches may be applied if the right problem structure is present. The principle is to decompose the original problem into smaller subproblems and coordinated them with a master problem. There are two different approaches, price or resource directive methods (Molina, 1979).

There are several reasons for applying decomposition methods to a problem. A very large model might not fit into the available memory, whereas decomposition may allow smaller pieces to fit. Large feasible regions which could result in an extensive B&B tree and solution time might be reduced by using decomposition. Decomposition may also make a model more manageable, and if it is terminated before reaching its optimum, lower and upper bound on the optimal objective value may be found for convex problems. There exists multiple decomposition techniques. The three most common methods, Lagrangian relaxation (LR), Dantzig–Wolfe decomposition (DWD) and Benders decomposition (BD), will be presented below. In addition the relationship between them is commented upon, and also how to handle integer variables through branch & price (B&P).

The discussion of LR, DWD and BD is limited to convex problems, while the B&P section considers MILP problems.

Lagrangian relaxation

Lagrangian relaxation (LR) is described in Beasley (1993). The idea behind LR is to attach Lagrangian multipliers to some of the constraints, relax these into the objective function, and then solve the resulting problem. When applying LR, the problem can in some cases be divided into smaller isolated subproblems, either naturally, i.e. by relaxing "common constraints" or by splitting variables and relaxing their equality binding, provided the objective and the relaxed constraints are additive in these variables. Different approaches can be used to find the values of the Lagrangian multipliers, for instance a sub-gradient method. LR is categorized as a price–directive decomposition method (Patriksson, 2008). A simple example of the procedure can be found in Beasley (1993).

LR does not necessarily provide the optimal solution, but an upper bound (for a maximization problem). By including a heuristic which creates feasible solutions together with the relaxation, both the solution and the size of the duality gap can be established. It may be noted that the bound found by Lagrangian relaxation will be equal to or tighter than the bound found by linear relaxation.

Dantzig-Wolfe decomposition

Decomposition methods are designed to make use of structures in the constraint matrix. Dantzig–Wolfe decomposition (DWD) is well suited for problems with a block angular structure, see Figure 1.3 and it applies a price directive way to coordinate the subproblems (Tebboth, 2001). This structure is exploited by splitting the original problem into smaller subproblems, also known as pricing problems, while the common constraints, ref. A_0 in 1.3 remain in a master problem. The constraints of each subproblem are accounted for by a convexity constraint in the master problem. This master problem contains far fewer constraints than the original problem, but possibly a large number of variables are introduced instead (Williams, 2005).

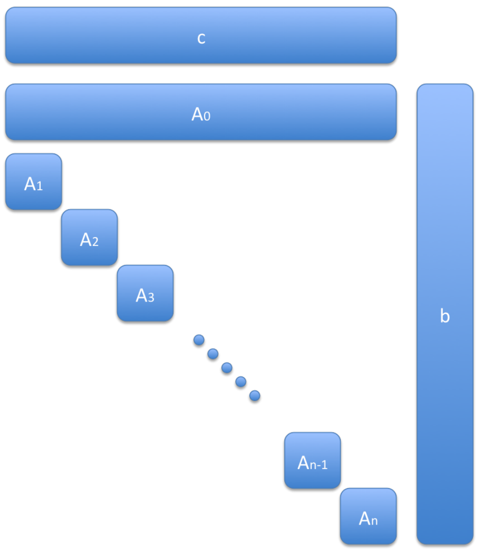


Figure 1.3: Illustration of block angular structure (Figure 2.1 in Conejo et al. (2006))

The subproblems' consumption of the relaxed common resources/constraints is accounted for by cost terms in their objective function. By changing the cost parameter a change in the optimal solution, and the use of the common resources, is expected. Each solution generated from the subproblems is included as column with an associated weighting variable in the master problem. These columns represent corner points of the convex subproblems. The master problem only contains information about subproblem solutions which have been computed, and is therefore known as the restricted master problem (RMP). By solving the RMP and retrieving the Lagrangian multipliers for the common constraints, the procedure uses these values as the cost parameters in the subproblem objective functions. This iterative procedure concludes when the subproblems cease to suggest new and better solutions given the changes in the cost on the consumption of the common resources.

In order to start the algorithm, initial columns must be provided, so that a feasible solution of the RMP can be established to ensure that proper Lagrangian multiplier information is passed to the subproblems. One example of a DWD application can be found in Williams (2005). Other sources on the topic are Lübbecke and Desrosiers (2005), Desrosiers (2005) and Savelsbergh (1997).

Benders decomposition

Benders decomposition (BD) is suitable for problems with a set of complicating variables, like integer variables, and a set of non-complicating variables, like continuous variables. BD was originally developed for MILP, and is categorized as resource directive decomposition. The non-complicating variables of the original problem is removed and placed in a subproblem. The new problem, only containing the complicated variables, is named the master problem. This problem would have a large number of constraints to account for the removed variables, and may be difficult to solve. Therefore, it is usually impractical to express all these constraints explicitly. The non-complicating variables are instead accounted for by feasibility or optimality constraints in the master problem, but only when needed.

The algorithm works with a small subset of these constraints, and as for DWD, the resulting problem is known as a RMP. As mentioned, the non-complicating variables are placed in a subproblem with their appropriate constraints. This subproblem generates new constraints to the RMP, also known as cuts. For a given set of values for the complicated variables, the subproblem finds values for the non-complicating variables. The Lagrangian multipliers from this solution are used to make a new optimality constraint or feasibility constraint which is added to the RMP. The resolved RMP with the new constraints included, computes information that is used in the subproblem to generate new cuts. Hence, this is an iterative process between the RMP and the subproblem, and the RMP will eventually converge toward the optimal solution as long as the subproblem is bounded and feasible. See also Dantzig and Thapa (2003b) and McCarl and Spreen (2004).

Relationship between decomposition schemes

There is a strong relationship between DWD and LR when considering LPs. If the linking or complicating constraints in a convex problem are put in the master problem of a DWD formulation, and if they are dualized in the objective function of a LR formulation, the optimal value of the LR and the DWD reformulation will be the same. The resulting subproblems will also be identical. As mentioned LR may apply a subgradient method to compute the multipliers whereas the DWD RMP solves an LP problem (Geoffrion, 1974). The equivalence between DWD and LR is a consequence of the equivalence of dualization and convexification (Pereira Pinto da Cunha e Alvelos, 2005).

Lemma (10.14) in Dantzig and Thapa (2003b) states that "solving the original problem by applying the DWD procedure to the dual of the original LP results in a procedure that is identical to the BD procedure." We recognize this relationship by the fact that DWD generates columns while BD generates the corresponding dual equivalents, i.e. rows. The DWD master problem sends dual solutions to the subproblems and receives a primal solutions in return, the reverse is true for BD.

Branch & price

B&P combines B&B and column generation, and it was specifically designed to handle IP models containing a large number of variables (Barnhart et al., 1998). Similarly to DWD, sets of columns are left out of the RMP problem since there are too many columns to handle efficiently, and because the necessary reformulation of the problem leads to models where most of the associated variables are zero in the optimal solution. To check if a LP solution is optimal, a subproblem, i.e. the pricing problem, is solved to locate columns with positive reduced cost (maximization problems). If such columns are found they are added to the RMP and the RMP is reoptimized. If no columns with a positive reduced cost can be found and the solution does not satisfy the integrality requirements, branching is done. After branching, there may exist columns with a positive reduced cost, but not currently present in the RMP. B&P will handle this by applying column generation in every node of the B&B tree to produce new columns for the RMP.

B&P is an exact method and will therefore, as a pure B&B search, in the end find the optimal integer solution. Within B&P there is however a variety of alternative strategies available for searching for the optimal solution. To start the B&P procedure an initial feasible RMP solution is needed as the subproblems require information about the Lagrangian multipliers of a RMP solution. Once the initial RMP solution has been obtained the remaining decisions evolve around how to branch, how to manage columns and what to branch on. For more on the topic, see Danna and Le Pape (2005).

Advances in decomposition techniques

The theoretical framework for decomposition of optimization problems has been around since the 1960s. However, as parallel computer architecture has boosted computational power the last decade, these methods have definitely increased their potential as they now easily can be solved in parallel instead of sequentially. Recent important work on this topic includes e.g. Brach & Price, and specially designed cuts (Desaulniers et al., 2005). It should finally be mentioned that filtering techniques to smooth changes in the dual variables have proven efficient for faster convergence.

1.4.7 Parallel computation

Parallel computing operates on the principle that large problems are divided into smaller ones that can be solved concurrently. There are many forms of parallel computing, bit-level, instruction level, data level and task level. Parallelization has been in use for many years, mainly in high-performance computing, but interest in it has grown the last years due to the physical constraints preventing a further increase in CPU clock frequency. Parallel computing has hence become the dominant paradigm in computer architecture, mainly in the form of multicore processors (Asanovíc et al., 2006).

The speed-up from parallelization would ideally be linear, i.e. doubling the number of processing elements should halve the runtime, and doubling it a second time should again halve the runtime once more. However, very few parallel algorithms achieve this ideal speed-up. For small numbers of processing elements, a near-linear speed-up is fairly easy to obtain while for large numbers of processing elements it tends to converge to a constant value. How well an optimization algorithm exploits a parallel computer architecture is hence important for success in solving large and challenging optimization problems.

The main steps of the simplex algorithm, i.e. moving from one basic feasible point to the next, is a sequential procedure, and does not naturally lend itself to parallel computing. However, given a basic feasible point, there are many neighboring points that can be evaluated before choosing one of them as the next iterate. This part of simplex is computationally expensive in problems with many variables and it may be parallelized. In addition, it is also possible to solve the same problem on many different CPUs with different initial points and with different strategies for choosing the next basic feasible point. For interior-point and SQP methods, many of the same things as for the simplex method hold. This discussion alludes to the fact that many algorithms may be candidates for parallel solutions even though they initially appear to be sequential.

B&B algorithms may be parallelized. For an MILP, at each node in the B&B tree, there is a stand-alone LP problem, and there is no reason for not solving all open nodes, i.e. identified, but unsolved nodes in the B&B tree, in parallel. Therefore, there will be a significant ramp up delay, as the root node has to be solved first. In a regular B&B algorithm one usually branches so as to create two child nodes. Hence, in the next step there will only be need for two CPUs. However, one may branch on several variables at the same time, creating more than two child nodes. There are potentially ramp down effects as well, when there are only a few nodes left to solve to establish the upper bound. The effect of parallelization is that the upper and lower bounds are not updated in the same way as in a sequential implementation. Nodes that are cut off in a sequential implementation due to a too low upper bound, i.e. an earlier found feasible solution, may be calculated in the parallel version as the feasible solution might not yet be found. For this reason it should be expected that more nodes need to be calculated in a parallel implementation of B&B than in a standard implementation.

There are inherent structures in optimization problems which lend themselves to parallel computer architectures. This is the case for optimization problems that apply B&P, DWD and LR. The subproblems can easily be solved in parallel instead of sequentially, without making any change to the rest of the algorithm. However, if the subproblems have a large span in solution time, the algorithm still has to wait for the slowest, which results in increased idle times for the CPUs and thereby reduced efficiency.

1.4.8 Heuristics

Heuristics are solution methods that try to obtain a good solution fast, but offer no, or in some case a bad, formal proof of the solution values distance to the global optimum. Knowledge of the application is essential for developing efficient heuristics.

Heuristics are often used in combination with an exact solution method or when no good exact method exists. There are generally two classes of heuristics, construction heuristics and local search heuristics (Winker and Gilli, 2004). Construction heuristics create, as the name implies, solutions from an empty start. The solutions produced by a construction heuristic are often the starting point for a local search heuristics. The local search will then try to perturb this solution to improve it further. Two concepts that are central to local search heuristics are intensification and diversification (Blum and Roli, 2003). A search that focuses on intensification will seek the best solution possible without any long term search considerations, thereby risking getting stuck in a local solution. Diversifying elements in a heuristic search, may in many ways try to do the opposite, which is to search in areas of the solution space which are unexplored.

Further, metaheuristic refers to strategies that modify other heuristics to strike a balance between intensification and diversification, and improve on solutions that are normally generated in local search algorithms i.e. avoid getting stuck in a local optimum. Metaheuristics usually add a random element to the procedure or do not allow one to step to solutions that have been visited recently. For more on heuristics and metaheuristics see Winker (2001).

1.4.9 Derivative free optimization

Many engineering businesses, and others as well, have simulators which are used for analyses. However, in simulator-based analyses, there are often more or less clarified goals that should be reached and constraints which must be honored. Simulation analyses are often done in a trial and error setting, to find a good enough solution.

The above problem can be cast into an optimization problem. By recognizing this, it might be possible to find better solutions in a faster way than with a trial and error approach. Simulators may contain complex models which embed complex phenomena like phase transitions, reaction kinetics and flows, and advanced numerical solvers. For this reason black box or derivative free optimization becomes interesting.

One way of looking at derivative free optimization is as a systematic trial and error approach, of which the simple Nelder and Mead method from 1965 (Nelder and Mead, 1965), is a good example. Despite of, or maybe because of its simplicity, the method is used in numerous applications; many of them with success. However, more sophisticated methods with much better properties exist.

By running simulators for different values of the input variables, also known as sampling, it is possible to build local approximate models of the underlying simulator, e.g. by using quadratic polynomials. The accuracy of such models decreases away from the sample points, and hence the concept of a trust region could be applied in the following way: A solution is improved based on optimizing the approximate model within a trust region. The solution of this problem is a candidate for an improved solution. The candidate is accepted if its error compared to the outcome of the simulator is less than some threshold value. Further, if the error is small, the trust region is enlarged; if not, it is kept constant or even reduced before proceeding to the next iterate.

For further reading on the subject, the introduction to derivative free optimization in Conn et al. (2009) is very useful. It is currently the only comprehensive book on this topic.

Advances in derivative free optimization

Derivative free optimization should be considered an interesting option when gradients are difficult or impossible to compute, and the advances during the last decade have been quite significant. It is hard to provide numbers on how many variables it is possible to handle since this is extremely dependent on the underlying simulators. However, there are examples of problems with up to a few hundred variables which have been successfully handled (Conn et al., 2009).

1.5 Problem instances and datasets

Through this thesis, the Statoil operated Troll west region, a large oil and gas field with one of the largest subsea developments in the world, is used as a test case to study and validate the optimization techniques which are developed. At Troll the production technology team mainly uses three software packages to model the well inflow, the flow through the vertical part of the well, and the subsea production network. These software packages are to some extent integrated.

The GORM software suite is used to predict inflow from the reservoir to the bottom hole wellbore. This model is integrated with FlowManager from FMC, through inflow performance ratio (IPR) tables, which state the flow rates of gas and oil for given bottom hole pressures. The flow of water is kept as a constant fraction of the oil flow, identified through well tests. With these IPR tables, together with pressure and temperature measurements, FlowManager calculates flow rates from each well. The production optimization packages in FlowManager, i.e. MaxPro, is implemented for some of the production clusters at Troll. In addition to MaxPro, the petroleum technology team is also using GAP to perform production optimization. GAP is also integrated with GORM through IPR tables. GAP and MaxPro however, are not integrated and acts as substitutes for each other.

As mentioned earlier, even though there exists software packages that address the production optimization problem at Troll, this is a very challenging problem which could benefit from improved optimization capability. Current technology cannot solve the full production allocation and routing problem.

The five papers that this thesis is based on, as listed Section 1.8.1, apply different data sets. The complexity and the realism of the datasets have increased as the methods which have been developed for solving them have improved. Comments on these data sets, and the relation between them are given below.

- The data set in Foss et al. (2009) is generated indirectly from the software packages described above. Simplified approximate functions were developed by the author to represent the real problem in a reasonable way. These semi realistic functions, representing the well and pipeline models, were used to generated tables i.e. breakpoints for the piecewise representations used in the optimization model. In this paper the problem instances is solved with one common constraint i.e. total gas capacity.

- The data set in Gunnerud and Foss (2009) have also one common constraint i.e. total gas capacity. It is also based on semi realistic functions, however slightly more advanced to better capture physics that in Foss et al. (2009).

- In Gunnerud et al. (2010) a realistic data set were developed directly from the GORM and GAP simulation software packages. An interface to GAP was created and simulations were run for different wellhead pressures and flow rates in pipelines, to create tables, i.e. breakpoints. However, we only have access to GORM and GAP models for three out of eight production clusters. The wells and pipelines in these clusters were therefore duplicated and slightly modified to create a full field dataset. The data set is realistic in the sense that real data is not expected to be more challenging or change the solution time or solution quality of the optimization problem significantly. As for the previous papers, only total gas capacity is restricted.

- In Torgnes et al. (2010), five variants of the realistic data set were generated. The solution time is quite sensitive to changes in data sets, we therefore reports average results to increase the reliability of our conclusions. In this paper the problem instances have two, as opposed to one common constraint as in the earlier publications, i.e. both total gas and water production is restricted.

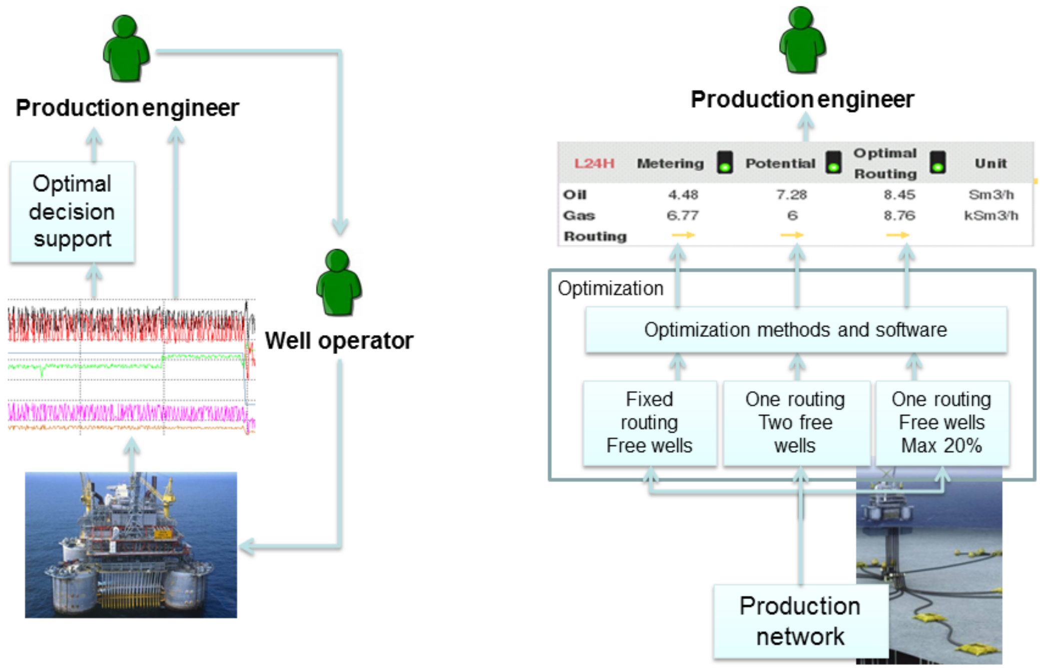
- Gunnerud et al. (2011) also uses a realistic data set, however this data set also accounts for changes in absolute pressure, which were generated from additional simulation in GAP. As for Torgnes et al. (2010) the problem instances have two common constraints, total gas and water production.

The difference in the data sets do not allow for a direct comparison between the results presented in Chapter 2, 3 and 4.

1.6 Motivation

Petroleum production engineers face several challenges in their daily work on optimizing the production from onshore and offshore facilities. Some of these challenges are so complex that human interpretation alone will not be enough to secure the best possible decisions. Hence, it makes sense to embed optimization-based decision support tools into work processes to improve the decision process to exploit the installed facilities in the best possible way. Best possible may imply maximizing oil production, keeping production on target rates, minimize down-time or combinations thereof. The situation is illustrated in Figure 1.4. The figure illustrates one of the work processes of a petroleum production engineer. This person analyses well data and recommends changes to the well operator who makes the final decision and implements actions. On the Norwegian Continental Shelf the production engineer is typically located onshore and the well operator offshore. A decision support tool may provide the production engineer with a tool to improve the analyses by adding information to the conventional analysis of raw data. The right hand figure illustrates a concrete analysis in which the engineer studies the effect of three alternative production changes.

This thesis will present optimization-based methods which are motivated by the need for such decision support tools for challenging problems. This includes nonlinear problems which includes routing options. The methods will be applied to real data and hence validated in a realistic setting.



L24H	Metering	🟢	Potential	🟢	Optimal Routing	🟢	Unit
Oil	4.48		7.28		8.45		Sm3/h
Gas	6.77		6		8.76		kSm3/h
Routing	→		→		→		

Figure 1.4: Production engineer workflow

1.7 Main contributions

In light of the challenges mentioned above the main contributions of this thesis include the following four elements:

1. A piecewise linear formulation of the RTPO problem, transforming the MINLP into MILP which allows the use of well established optimization techniques and also keep track of the global optimal solution (Chapter 2).
2. Decomposition strategies which exploit the structure of a problem and divide it into smaller tractable subproblems. For the RTPO problem this results in a linear increase in solution time with respect to then number of subproblems defined by the number of production clusters (Chapter 2 and 3).
3. Algorithmic adaption to exploit a parallel computer architecture. For the RTPO problem this gives significant improvements (Chapter 3).
4. A branch & bound framework which finds the global optimal solution, also of the decomposed problem, i.e. branch & price, has been developed for the RTPO problem (Chapter 4).

1.8 Thesis outline and list of papers

The continuation of the thesis is presented in five chapters:

- **Chapter 2** creates an optimization model of the RTPO problem. It starts by presenting a MINLP formulation, and transform this into a MILP formulation. Further, it explores how it is possible to use Lagrangian relaxation and Dantzig-Wolfe decomposition to divide the problem into smaller subproblems to be able to solve much large problems. The chapter is based on Foss et al. (2009), Gunnerud and Foss (2009) and Gunnerud et al. (2010).
- **Chapter 3** describes the same problem as presented in the previous chapter. In addition it looks into the possibility of solving the subproblems in parallel to exploit the gain in computational power in new computers. The chapter is based on Gunnerud et al. (2010) and Torgnes et al. (2010).

- **Chapter 4** addresses the challenges related to decomposing a problem containing several binary decisions, such as on/off valves and routing valves for each well. This is handled by branching on continuoues gas and water flow rate variables from the subproblems in a branch & price framework. This chapter is based on Gunnerud et al. (2011).

- **Chapter 5** concludes the thesis and gives suggestions for further work.

The material presented in this thesis is either published, recently submitted for publication or accepted for publication. Below, all papers that I have contributed to are listed. The papers are divided into two groups depending on the impact they have had on this thesis. The selected papers contain research that is essential for what is presented here, while the list of secondary papers are more peripheral. I have also included a list of presentations and outplacements conducted during the PhD work.

1.8.1 List of selected papers

- B. Foss, V. Gunnerud and M. Dueñas Díez, *"Introduction to decomposition of petroleum production optimization problems"*, published in SPE Journal 2009 (Foss et al. (2009)).

- V. Gunnerud and B. Foss, *"Lagrangian Decomposition of Oil-Production Optimization Applied to the Troll West Oil Rim"*, published in Computers & Chemical Engineering 2009 (Gunnerud and Foss (2009)).

- V. Gunnerud, E. Torgnes and B. Foss, *"Parallel Dantzig-Wolfe decomposition for real-time optimization - Applied to a complex oil field"*, published in Journal of Process Control 2010 (Gunnerud et al. (2010)).

- E. Torgnes, V. Gunnerud, E. Hagem, M. Rönnquist and B. Foss, *"Parallel Dantzig-Wolfe decomposition of Petroleum production allocation problems"*, accepted for publication in Journal of the Operational Research Society 2010 (Torgnes et al. (2010)).

- V. Gunnerud, B. Nygreen, K. McKinnon and B. Foss, *"Oil production optimization Solved by Piecewise linearization in a Branch & Price framework"*, submitted to Computers and Operations Research Journal 2011 (Gunnerud et al. (2011)).

1.8.2 List of secondary papers

- M. Wartmann, V. Gunnerud, B. Foss and E. Ydstie, *"Distributed optimization and control of offshore oil production: the intelligent platform"*, proceedings FOCAPO conference June 29 - July 2, 2008 Boston (Wartmann et al. (2008)).
- V. Gunnerud, B. Foss, B. Nygreen, R. Vestbø and N. Walberg, *"Dantzig-Wolfe decomposition for real-time optimization - applied to the Troll west oil rim"*, key-note paper, proceedings ADCHEM conference July 12-15, 2009 Istanbul (Gunnerud et al. (2009)).
- E. Erlingsen, T. Strat, V. Gunnerud, B. Nygreen and M. Dueñas Díez, *"Decision Support in Long Term Planning of Petroleum Production Fields"*, submitted to SPE Intelligent Energy conference 2012 (Erlingsen et al. (2012)).
- S-E. Fleten, V. Gunnerud, Ø . Dahl Hem, and A. Svendsen, *"Real option valuation of offshore petroleum field tie-ins"*, submitted to Resource and Energy Economics Journal 2010 (Fleten et al. (2010)).
- Ø. Dahl Hem, A. Svendsen, S-E. Fleten and V. Gunnerud, *"Real options analysis of strategic decisions in offshore petroleum production"*, accepted for the Real Options conference, Turku, Finland, June 15-18, 2011 (Dahl Hem et al. (2011)).

1.8.3 List of presentations

- *Nordic Process Control Workshop*, NPCW-14, Helsinki, Finland, August 2007
- *Guest lecture at IBM T.J. Watson research center*, New York, USA, June 2008
- *Guest lecture at University of Edinburgh*, Edinburgh, Scotland, August 2008
- *International Conference on Integrated Operations in the Petroleum Industry*, IO08, Trondheim, Norway, October 2008
- *Smart Field Annual meeting at Stanford University*, San Francisco, USA, April 2009
- *European Conference on Operational Research*, EURO XXIII, Bonn, Germany, July 2009
- *Guest lecture at Repsol research*, Madrid, Spain, November 2009
- *Workshop - set of lectures at Petrobras*, Rio de Janeiro, Brazil, August 2010

- *Guest lecture at Universidade Federal de Santa Catarina*, Florianopolis, Brazil, August 2010
- *Guest lecture at Carnegie Mellon University*, Pittsburgh, USA, February 2011
- *Guest lecture at IBM T.J. Watson research center*, New York, USA, February 2011

1.8.4 List of outplacements

- Statoil - Troll oil and gas field, production engineer, Bergen, Norway, May - November 2009
- FMC Technologies, system development, Asker, Norway, February - July 2010

Chapter 2

Piecewise linearization and comparison of Lagrangian relaxation and Dantzig-Wolfe decomposition

The results presented in this chapter is based on three papers. The first is Foss et.al 2008, "Lagrangian Decomposition of Oil-Production Optimization Applied to the Troll West Oil Rim", published in the SPE Journal. The second paper is Gunnerud and Foss 2009, "Oil production optimization - A piecewise linear model, solved with two decomposition strategies", published in Computers and Chemical Engineering. The text in this chapter is mainly taken from this particular paper. The third paper is Gunnerud et al. (2010), "Parallel Dantzig-Wolfe decomposition for real-time optimization - Applied to a complex oil field", published in Journal of Process Control.

Abstract

This chapter presents a new method for real-time optimization of process systems with a decentralized structure where the idea is to improve computational efficiency and transparency of a solution. The contribution lies in the application and assessment of Lagrangian relaxation and Dantzig-Wolfe methods, which allow us to efficiently decompose a real-time optimization problem. Furthermore, all nonlinearities are modeled by piecewise linear models, resulting in a mixed integer linear program, with the added benefit that error bounds on the solution can be computed.

The merits of the method are studied by applying it to a semi-realistic model of the Troll west oil rim, a petroleum asset with severe production optimization challenges due to rate dependent gas-coning wells. This study indicates that both the Lagrangian relaxation and in particular the Dantzig-Wolfe approach offers an interesting option for complex production systems. Moreover, the method compares favorably with the non-decomposed method.

Keywords: Oil production planning, Piecewise linearization, Mixed Integer Linear Programming, Lagrangian relaxation, Dantzig-Wolfe decomposition.

2.1 Introduction

Development of a petroleum field asset requires planning on several horizons. On a long-term horizon, typically from one year and up to the field's lifetime, strategic reservoir planning is based on market conditions, field properties and strategic considerations of the developing company. Decisions related to technology for an offshore field will include; how to develop the subsea solution, whether to process the fluid onshore or offshore, and how to export the different products produced. It is also possible to include extra flexibility. By for instance accepting a higher investment cost, it may be possible to allow future development such as tie-ins from possible neighboring assets. The analyses and subsequent development plan seek to maximize the net present value of the asset by maximizing oil and gas recovery while honoring safety and environmental constraints. Nygreen et al. (1998) discuss some of these issues.

On a medium time horizon, often referred to as tactical reservoir management, the planner will seek to extract as much oil and gas from the reservoir as possible, within the bounds of the earlier strategic decisions. In the Troll field which will be discussed later, the extraction of gas was severely limited to ensure higher pressure in the reservoir for an easier extraction of oil. Usually during the green field stage it is important to plan, drill and commission new wells to reach some pre-defined plateau rate as soon as possible. During the plateau production there may be an in-field drilling program for production and/or injection wells. This involves decisions on the location and completion of wells. Later, during the decline phase of a field, artificial lift technology may be applied to boost production.

Operational production planning considers shorter time horizons, typically days and weeks, and is usually denoted real-time production optimization (RTPO) problem. Production may be constrained by reservoir conditions such as coning effects and/or the production equipment like pipeline capacity or downstream water handling capacity. Hence, this requires modeling of both the sub-surface part (reservoir and wells) and the surface part (pipelines and downstream production equipment) of the value chain. Decision variables in RTPO include production and possibly injection rates, artificial lift inputs like lift gas rates and electric submersible pump (ESP) rates, and routing of well streams. The goal will be to maximize daily production rates. Overviews on RTPO can be found in Wang (2003), Saputelli et al. (2005) and Bieker et al. (2006a). RTPO is a widely used expression in the industry. In this chapter, however, by RTPO we mean planning which involves the solution of a mathematical optimization problem.

This chapter centers on a RTPO problem. The main contribution is a mixed integer linear program (MILP) formulation of the production network combined with a decomposition strategy. Two decomposition methods, Lagrangian relaxation (LR) and Dantzig-Wolfe decomposition (DWD), are explored and tested on a realistic example. A key feature of our approach is the use of a divide-and-conquer strategy to decompose a RTPO problem into tractable subproblems. Related references are Foss et al. (2009) where LR was introduced on a rather conceptual level and Gunnerud et al. (2009) where DWD was initially proposed as a decomposition method for RTPO.

There has been some interest in applying decomposition techniques within the process systems literature. Alabi and Castro (2009) apply DWD for refinery planning while Cheng et al. (2008) use DWD to coordinate decentralized model predictive controllers for plant-wide control. In particular decentralized target calculations are coordinated by applying DWD.

There are many similarities between developing and operating a petroleum asset and a downstream process system, like a refinery. Hence, it makes sense to frame RTPO within the well established process control hierarchy which includes regulatory control, supervisory control, local dynamic or static optimization, site-wide optimization, and scheduling and planning; see e.g. Backx et al.. As mentioned earlier RTPO means short-term production planning. This is comparable to real-time optimization (RTO) in the process industries. One obvious difference, however, is the fact that a petroleum field in a life-cycle perspective is a depletable asset which can and should be viewed as a batch process as opposed to a plant like a refinery. Hence, conditions will vary significantly during the lifetime of a petroleum asset.

The remainder of this chapter is organized as follows. To begin with, the complete nonlinear RTPO model is presented. Then the techniques used for handling the nonlinearities, i.e. how to create piecewise linear representations and formulate the MILP model, is discussed. Further, we look into why the problem is suitable for decomposition and present two decomposition schemes. Subsequently, results from a field case are presented and a discussion on challenges related to the alternative solution methods is included before some conclusions end the chapter.

2.2 The real-time production optimization problem

2.2.1 Methods and technology

RTPO applications exist in limited numbers. Two commercial products are GAP from Petroleum Experts and MaxPro from FMC Technologies. Both model the wells and pipeline systems, and solve the optimization problem by combining linear and nonlinear techniques. Wang (2003) provides a comprehensive overview of models and solution algorithms for different problems in the industry, again including both linear and nonlinear formulations with appropriate techniques for solving them. Bieker et al. (2006a) present an overview of the oil production problem which includes a description of production planning, processing facilities and well model updating. Another recommendable reference is Saputelli et al. (2003) since it ties RTPO to application challenges such as the availability of appropriate technologies. The value chain may be divided into an upstream part, which includes reservoirs, wells, pipelines and a downstream part which includes a process system for separating oil, water and gas as well as some export facility. The literature mentioned above takes a silo approach in the sense that the upstream part is optimized without including a model of the downstream system. Usually the downstream boundary is a fixed pressure on the inlet separator of the downstream process. Optimizing across this boundary by including the upstream and downstream system is rarely seen. One exception is Foss and Halvorsen (2009) which shows that a significant gain can be made by bridging the gap.

Some work discuss the consistency between production optimization and the medium term horizon decisions involving a full field reservoir simulator. In Awasthi et al. (2007) model consistency is emphasized while Awasthi et al. (2008) focus on decomposition between time scales and a moving-horizon approach is used for operational planning. As in all model-based applications model maintenance is important. It is particularly important to update well models since these models may change significantly over time. Cramer et al. (2009) present a data driven monitoring approach towards this end.

The literature is fairly limited on optimization models and solution algorithms for upstream petroleum production systems. Bieker (2007) solves the problem by piecewise linearization of nonlinearities and end up with a MILP formulation. Network topology is kept quite simple and in particular no routing issues are included.

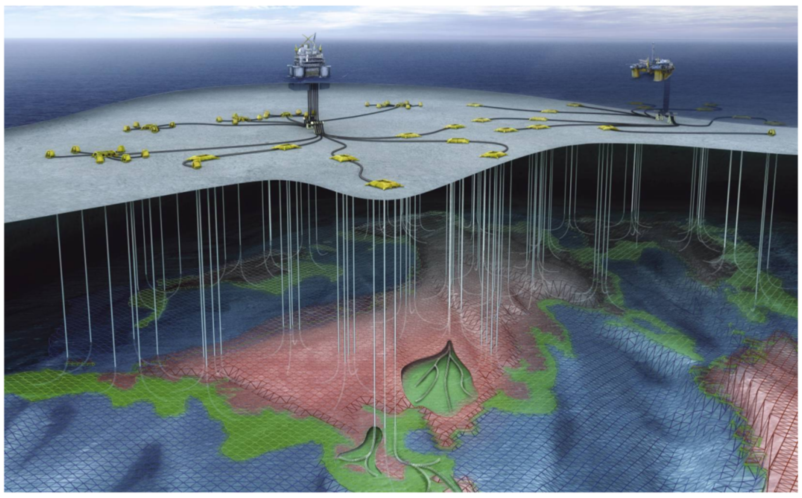


Figure 2.1: Troll West structure (StatoilHydro 2008)

Kosmidis et al. (2005) use a similar approach on a richer network topology. They allow for routing of the fluids from wells between different pipelines and to different separators. Kosmidis et al. (2005) piecewise linearize the well models. They do, however, end up with a MINLP model, which results in a completely different solution algorithm compared to MILP solvers.

2.2.2 Problem structure

Figure 2.1 illustrates the Troll B and C platforms including the subsea production network. The structure is typical for a large scale offshore oil and gas production system. Such systems can be divided into clusters where one cluster contains a collection of wells which are connected to a platform through common pipelines. An illustration of a cluster is shown in Figure 2.2. Each cluster consists of a number of wells, manifolds and importantly several production lines. Troll C, which we revisit later, contains eight clusters, each with two parallel production lines, two manifolds, and up to eight wells per cluster. The fluids from these eight clusters feed into a common platform-based process section.

Inflow from the reservoir into the wellbore, i.e. in the bottom part of the well, is known as the inflow performance relationship (IPR). It depends on reservoir pressure, pressure in the wellbore as well as the condition of the well itself. Further, a vertical lift performance curve (VLP) is commonly used to relate downhole conditions to wellhead (surface) conditions. This relationship depends on well geometry, and fluid rates and composition. Reservoir conditions change over time due to the

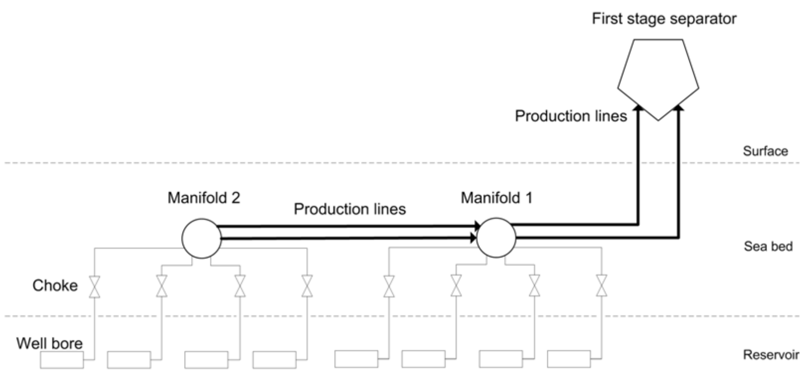


Figure 2.2: Typical topology structure of a cluster

drainage effect. Since we are interested in short-term optimization it is fair to assume constant conditions on the optimization horizon of interest. The well stream entering a manifold is routed to one of the production lines, see Figure 2.2. The fluid is then transported through these pipelines to the platform, and the pressure drop along the lines is modeled by multiphase flow models.

There are constraints related to each cluster. They arise from reservoir analyses and capacity limits in the production equipment itself. As an example short-term production boosting may harm long-term drainage efficiency. The reason is that increased pressure gradients may damage the formation close to wells and hence reduce long term productivity. Further, there will as always be capacity limits on production equipment like wells, valves and pipelines, as well as constraints which originate from the downstream part of the value chain, for instance gas and water handling capacities. All appropriate constraints are detailed in the next section.

A complete formulation is complex, thus the RTPO problem is both challenging and hard to solve. It may be noted, however, that most of the constraints are local to each cluster. This observation is essential for the decomposition approaches applied later.

2.3 Model formulation

In the following we present a system model which encompasses a substantial class of upstream production systems. It is based on a relatively general network topology including fields like Troll. We start by stating the complete model and from there, derive the MILP model which is used as a foundation for the two decomposition strategies.

2.3.1 Nonlinear model

In the interest of clarity, we formulate the problem for clusters containing only one manifold. An extension to several manifolds per cluster, as in Figure 2.2, is quite straightforward and is actually implemented in the case example discussed later.

At Troll C, there are two parallel pipelines transporting the produced fluid from a cluster. The number of pipelines can vary from one asset to another, from simple applications with only one line to network structures with even more than two lines, and with other possible structures than the tree structure in our model.

The following indexing conventions are used throughout the chapter. Each cluster is identified by a single index i. Each well is identified by two indices ij, the index of the cluster in which it lies and its own index within the cluster. Each pipeline is also identified by two indices il, the index of the cluster where it lies and its own pipeline index.

The most common variables are illustrated in Figure 2.3. The choke valve is found between the reservoir and the manifold, and is used to control the flow from the well. Some definitions are needed. In Table 2.1 all the indices are given, while the sets are given in Table 2.2. Further, Table 2.3 contains the data, while Table 2.4 contains the variables we use.

Table 2.1: Indices

i	-	Cluster
j	-	Well
l	-	Pipeline
p	-	Phase

Table 2.2: Sets

$\mathcal{I}$	-	Set of clusters
$\mathcal{J}_i$	-	Set of wells in cluster i
$\mathcal{L}_i$	-	Set of pipelines in cluster i
$\mathcal{P}$	-	Set of phases (g for gas, o for oil, and w for water)

In the following we present the objective function and constraints. The constraints are divided into two groups; common constraints and local constraints for each cluster.

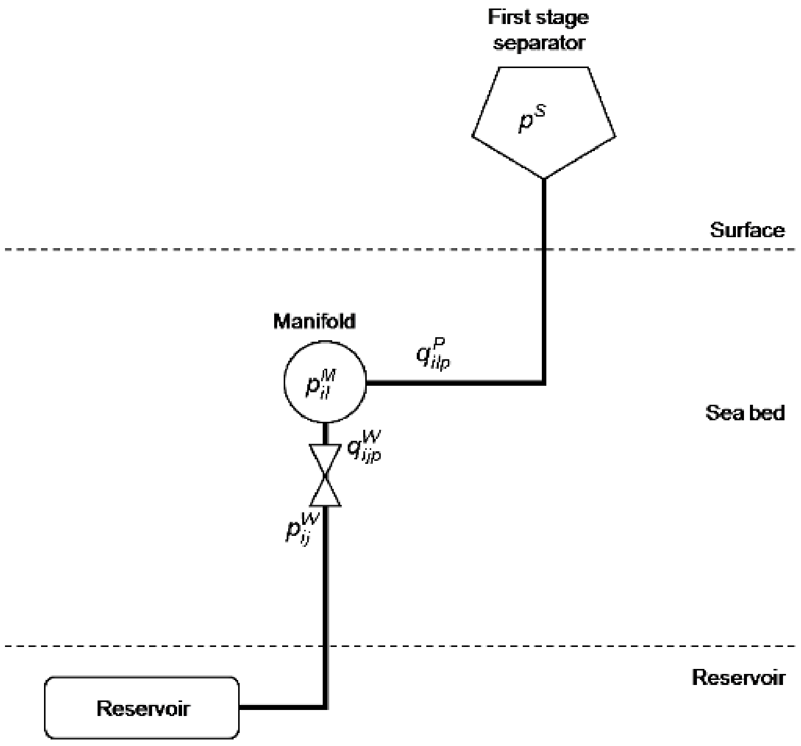


Figure 2.3: Cluster topology with key variables

Table 2.3: Data

C^T_p	-	Capacity of phase p in the first stage separator at platform
$f^P_{il}()$	-	Function for pressure drop in pipeline l
$f^W_{ij}()$	-	Function for pressure drop in pipeline from wellbore to manifold
$f^{IN}_{ijp}()$	-	Well inflow model for each phase p
$f^{WPC}_{ijp}()$	-	Well performanc curve model for each phase p
p^R_{ij}	-	Reservoir block pressure near well j
p^S	-	Pressure at first-stage separator on platform
q^M_{ij}	-	Maximum liquid production from well ij

Table 2.4: Variables

p_{il}^{M}	-	Pressure at the manifold level in pipeline l
p_{il}^{D}	-	Pressure drop across pipeline l
p_{ij}^{W}	-	Pressure at manifold level upstream choke
p_{ij}^{WF}	-	Wellbore flowing pressure, well j
q_{ip}^{C}	-	Flowrate of phase p from cluster i
q_{ilp}^{P}	-	Flowrate of phase p in line l
q_{ijp}^{W}	-	Flowrate of phase p from well j
x_{ij}	-	Binary variable. 1 if well j is closed 0 otherwise
y_{ijl}	-	Binary variable. 1 if well j is routed to line l 0 oterwise

Objective function

The objective function sums the oil flowrates from all clusters.

$$\max \ \mathrm{Z} = \sum_{i \in \mathcal{I}} q_{io}^{C} \tag{2.1}$$

It should be noted that the objective function is additive on a cluster level. Further, the use of oil flowrate is a common production measure since it reflects short-term revenue in most cases.

Common capacity constraints

The inequality below defines the common constraints. It states that the sum of gas and water rates from all clusters must be less than the gas handling and water handling capacities of the downstream processing equipment. Hence, these are the only constraints which connect the clusters, all other constraints apply for each cluster separately.

$$\sum_{i \in \mathcal{I}} q_{ip}^{C} \leq C_{p}^{T}, \ \ p \in \{g, w\} \tag{2.2}$$

Cluster constraints

All constraints, except for the common capacity constraints, are defined for all clusters $i \in \mathcal{I}$. In this section this statement will be omitted for the sake of simplicity.

The well model consists of two parts. First, the multiphase flow from the reservoir into the wellbore of each well is defined. It depends on reservoir pressure, geological properties of the formation close to the well and the wellbore itself, and it is usually represented by the well's IPR curve.

$$q_{ijp}^{W} = f_{ijp}^{IN}(p_{ij,}^{R} p_{ij}^{WF}), \quad \forall\ j \in \mathcal{J}_i,\ \ p \in \mathcal{P} \tag{2.3}$$

Further, the pressure upstream the choke has to be equal to the bottomhole flowing pressure p_{ij}^{WF} subtracted the pressure loss between the wellbore and the wellhead:

$$p_{ij}^{W} = p_{ij}^{WF} - f_{ij}^{W}(q_{ijg}^{W}, q_{ijo}^{W}, q_{ijw}^{W}), \quad \forall\, j \in \mathcal{J}_i \tag{2.4}$$

This represents the VLP curve of a well.

The two nonlinear functions representing IPR and VLP respectively can be combined to create a nonlinear well performance curve (WPC). This equation then links wellhead pressure to the flowrate of each phase from one particular well.

$$q_{ijp}^{W} = f_{ijp}^{WPC}(p_{ij,}^{R}\, p_{ij}^{W}), \quad \forall\, j \in \mathcal{J}_i,\ \ p \in \mathcal{P} \tag{2.5}$$

Long-term recovery may impose limits on liquid production from a well.

$$q_{ijo}^{W} + q_{ijw}^{W} \le q_{ij}^{M}, \quad \forall\, j \in \mathcal{J}_i \tag{2.6}$$

The well routing constraint (2.7) assures that flow from a well j either is closed or routed to one of the pipelines leaving the manifold.

$$x_{ij} + \sum_{l \in \mathcal{L}_i} y_{ijl} = 1, \quad \forall\, j \in \mathcal{J}_i \tag{2.7}$$

The mass balance constraints (2.8) sum the flow from all wells that are routed to a particular pipeline.

$$q_{ilp}^{P} = \sum_{j \in \mathcal{J}_i} q_{ijp}^{W} y_{ijl}, \quad \forall\, l \in \mathcal{L}_i,\ \ p \in \mathcal{P} \tag{2.8}$$

Simliarly (2.9) aggregates the flow from all pipelines in cluster i to one variable q^C_{ip}, i.e. the total production of phase p from cluster i.

$$\sum_{l \in \mathcal{L}_i} q^P_{ilp} = q^C_{ip}, \quad \forall\, p \in \mathcal{P} \tag{2.9}$$

Pressure relations in manifolds and separator: Constraint (2.10) assures that the pressure for line l at the manifold in cluster i must be lower than the pressure upstream the choke in all wells j connected to the manifold and routed to line l. The reason for the inequality, instead of an equality, is to allow a pressure drop across the choke, i.e. when the choke is partly closed.

$$p^M_{il} y_{ijl} \leq p^W_{ij}, \quad \forall\, j \in \mathcal{J}_i,\ l \in \mathcal{L}_i \tag{2.10}$$

The pressure drop in pipelines between the manifold and the separator is formulated in (2.11). As the separator pressure is fixed the absolute pressure will not be included in the pressure drop model. For a problem containing more than one manifold in a cluster, this will not be the case and absolute pressure should be considered.

$$p^S = p^M_{il} - f^P_{il}(q^P_{ilg}, q^P_{ilo}, q^P_{ilw}), \quad \forall\, l \in \mathcal{L}_i \tag{2.11}$$

2.3.2 MILP model

There are especially two factors that make the above optimization problem challenging. First, the nonlinearities related to well models (2.5) and pressure losses in pipelines (2.11), and second, well routing, which forces integer properties to be taken into consideration, cf. (2.7). These aspects have been key issues when choosing the solution approach for this optimization problem.

We piecewise linearize all nonlinearities which transform the MINLP into a MILP, to take advantage of the features that come with this formulation. This includes the possibility to use algorithms like simplex, and branch & bound, see e.g. Rardin (2000), and the ability to compute performance bounds on a global solution.

Compared to Bieker (2007) we treat several clusters, which makes decomposition interesting. Further, we include the ability to route well flows to different pipelines. Kosmidis et al. (2005) end up with a MINLP model since they keep the nonlinear models for the pipeline pressure drops as opposed to our MILP formulation.

Table (2.5) - (2.8) define the new indices, sets, data and variables needed for linearizing the problem.

Table 2.5: Indices

k	-	Interpolation coordinate for the piecewise linearization of the WPC
n_p	-	Interpolation coordinate for the piecewise linearization of the pressure drop in pipelines, dependent on phase p (n_g for gas, n_o for oil, n_w water)

Table 2.6: Sets

$\mathcal{K}_{ij}$	-	Set of interpolation coordinates for the piecewise linearization of the well performance curve (WPC)
$\mathcal{N}_{ilp}$	-	Set of interpolation coordinates for the piecewise linearization of the pressure drop in the pipelines

Table 2.7: Data

$p^D_{iln_gn_on_w}$	-	Breakpoints for pressure drop in pipeline il
p^{MAX}_{ij}	-	Maximum well head pressure
p^{WPC}_{ijk}	-	Pressure at manifold level before choke, breakpoint k, well ij
$q^D_{ilpn_p}$	-	Flow rate in pipeline il of phase p corresponding to breakpoint n_p
q^{MAX}_{ijp}	-	Maximum flow of phase p from well ij
q^{WPC}_{ijpk}	-	Flow rate of phase p from well ij corresponding breakpoint k

Gas and water capacity constraints

The capacity constraints (2.2) are already linear, however we would like to restate them here to have a complete MILP model in this section.

$$\sum_{i\in\mathcal{I}} q^C_{ip} \leq C^T_p, \ \ \forall\, p \in \{g, w\} \tag{2.12}$$

Linearization of well performance curves

We apply a modal formulation to piecewise linearize the WPC (2.5). By this we replace the nonlinear constraints with SOS2 sets (Williams, 2005). The link between wellhead pressure p^W_{ij} and flow q^W_{ijp} is given by (2.13) - (2.17). γ_{ijk} are the

Table 2.8: Variables

q^S_{ijlp}	-	flow from well ij to line il of phase p into line l
γ_{ijk}	-	Weighting variable associated with each breakpoint k, well ij
$\lambda_{iln_gn_on_w}$	-	Weighting variable associated with breakpoint n_g, n_o, n_w in well ij
η_{ilpn_p}	-	Weighing variable associated with the SOS2 sets for the piecewise linearization pipeline models

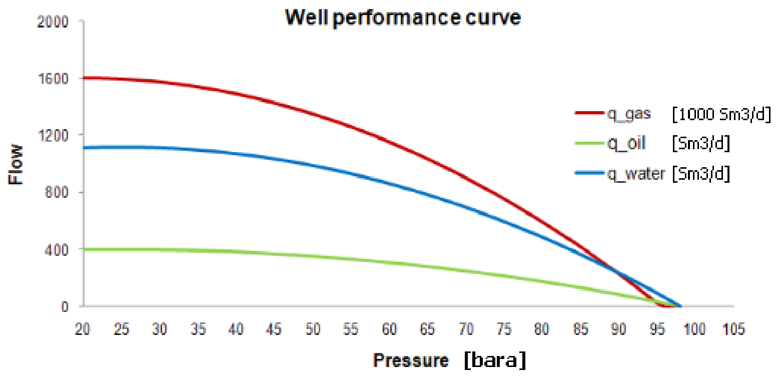


Figure 2.4: WPC based on typical data from a gas-coning well

weighting variables introduced by the SOS2 set which decides on the weighting of the breakpoints $(p^{WPC}_{jk}, q^{WPC}_{jpk})$.

$$p^W_{ij} = \sum_{k \in \mathcal{K}_{ij}} p^{WPC}_{ijk} \gamma_{ijk}, \; J \in \mathcal{J}_i \tag{2.13}$$

$$q^W_{ijp} = \sum_{k \in \mathcal{K}_{ij}} q^{WPC}_{ijpk} \gamma_{ijk}, \; j \in \mathcal{J}_i, \; p \in \mathcal{P} \tag{2.14}$$

$$\sum_{k \in \mathcal{K}_{ij}} \gamma_{ijk} = 1, \;\; j \in \mathcal{J}_i \tag{2.15}$$

$$\gamma_{ijk} \geq 0, \;\; j \in \mathcal{J}_i, \;\; k \in \mathcal{K}_{ij} \tag{2.16}$$

$$\gamma_{ijk} \quad \text{is SOS2 for } k, \quad j \in \mathcal{J}_i \tag{2.17}$$

Note that p^W_{ij} is mapped to three flow variables q^W_{ijp}, cf. Figure 2.4.

Linearization of well routing constraints

To handle well routing, several steps are made. The linear routing constraints (2.7) are restated.

$$x_{ij} + \sum_{l \in \mathcal{L}_i} y_{ijl} = 1, \;\; \forall\, j \in \mathcal{J}_i \tag{2.18}$$

Further, we have changed the nonlinear constraints (2.8) and (2.10) into a set of linear constraints. A new set of variables, q^{S}_{ijlp}, which are the flowrates of phase p from well ij to pipelines il leaving the manifold, is defined. Further, two new sets of constants, q^{MAX}_{ijp} and p^{MAX}_{ij} are introduced. q^{MAX}_{ijp} defines the maximum possible flowrate for each phase from a given well, and p^{MAX}_{ij} defines the maximum possible pressure drop over the manifold. (2.19) - (2.22) then describe new mass and pressure balances for wells and manifolds.

$$\begin{aligned}
\sum_{l \in \mathcal{L}_i} q^{S}_{ijlp} &= q^{W}_{ijp}, \;\; \forall\, j \in \mathcal{J}_i, \;\; p \in \mathcal{P} && (2.19)\\
q^{S}_{ijlp} &\leq q^{MAX}_{ijp} y_{ijl}, \;\; \forall\, j \in \mathcal{J}_i, \;\; l \in \mathcal{L}_i, \;\; p \in \mathcal{P} && (2.20)\\
q^{P}_{ilp} &= \sum_{l \in \mathcal{L}_i} q^{S}_{ijlp}, \;\; \forall\, l \in \mathcal{L}_i, \;\; p \in \mathcal{P} && (2.21)\\
p^{M}_{il} &\leq p^{W}_{ij} + p^{MAX}_{ij}(1 - y_{ijl}), \;\; \forall\, j \in \mathcal{J}_i, \;\; l \in \mathcal{L}_i && (2.22)
\end{aligned}$$

To elaborate (2.22) limits the pressure in the manifold when the connection between well ij and pipeline il is open. When the connection is closed, however, then $y_{ijl} = 0$ and (2.20) ensures that there is no flow from the well into that pipeline, and (2.22) is relaxed so that there is no link between the wellhead and pipeline pressures.

Linearization of pressure drop in pipelines

The pressure drop in the pipelines depends on gas, oil and water flowrates as shown in (2.11). Hence, it is necessary to select breakpoints in three dimensions, i.e. the gas, oil and water flowrates, this results in a more complicated remodeling procedure than for the WPC curves. Bieker (2007) used a similar procedure for a pipeline model with four inputs since he also includes the absolute pressure at the inlet of the pipeline.

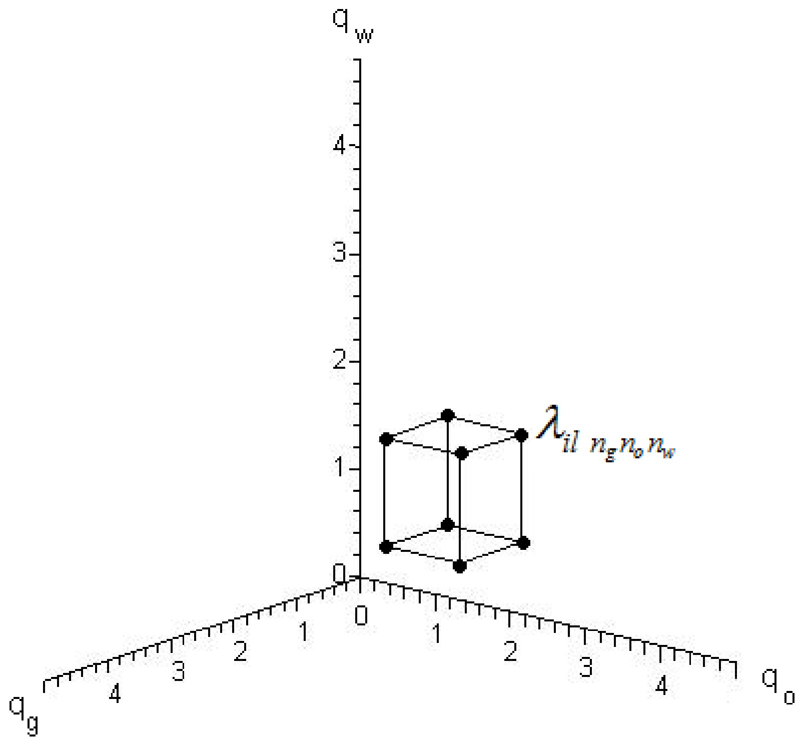


Figure 2.5: Illustration of breakpoint coordinates

To explain this procedure, we start by defining a grid for each pipeline il $(q^P_{ilg}, q^P_{ilo}, q^P_{ilw})$, not necessarily equidistant, in three dimensions. q^P_{ilg} will be the flowrate of gas in pipeline l, and similar for oil and water. Associated non-negative weighting variables $\lambda_{iln_gn_on_w}$ with each point in the grid are also defined, where $n_g \in \mathcal{N}_{ig}$, $n_o \in \mathcal{N}_{io}$, $n_w \in \mathcal{N}_{iw}$. Further, $\mathcal{N}_{ig}$, $\mathcal{N}_{io}$, and $\mathcal{N}_{iw}$ are the sets of flowrate break points along the gas, oil and water axes, respectively, see Figure 2.5.

If the values of $(q^P_{ilg}, q^P_{ilo}, q^P_{ilw})$ at the breakpoints are denoted $(q^D_{ilgn_g}, q^D_{ilon_o}, q^D_{ilwn_w})$ and the associated pressure drop $(p^D_{ilpn_gn_on_w})$, it is possible to approximate function (2.11) by means of the following relations:

$$p^D_{il} = \sum_{n_g \in \mathcal{N}_{ig}} \sum_{n_o \in \mathcal{N}_{io}} \sum_{n_w \in \mathcal{N}_{iw}} p^D_{ipn_g n_o n_w} \lambda_{iln_g n_o n_w}, \ \forall\, l \in \mathcal{L}_i \tag{2.23}$$

$$q^P_{ilp} = \sum_{n_g \in \mathcal{N}_{ig}} \sum_{n_o \in \mathcal{N}_{io}} \sum_{n_w \in \mathcal{N}_{iw}} q^D_{ilpn_p} \lambda_{iln_g n_o n_w}, \ \forall\, l \in \mathcal{L}_i,\, p \in \mathcal{P} \tag{2.24}$$

$$1 = \sum_{n_g \in \mathcal{N}_{ig}} \sum_{n_o \in \mathcal{N}_{io}} \sum_{n_w \in \mathcal{N}_{iw}} \lambda_{iln_g n_o n_w}, \ \forall\, l \in \mathcal{L}_i \tag{2.25}$$

$$\lambda_{iln_g n_o n_w} \geq 0, \ \forall\, l \in \mathcal{L}_i, \ n_g \in \mathcal{N}_{ig}, \ n_o \in \mathcal{N}_{io}, \ n_w \in \mathcal{N}_{iw} \tag{2.26}$$

In addition at most eight neighboring $\lambda_{iln_g n_o n_w}$ can be non-zero as illustrated by Figure 2.5. This condition is a generalization of a SOS2 set, and can be imposed as below, where η_{ilpn_g}, η_{ilpn_o} and η_{ilpn_w} are auxiliary weighting variables of $\mathcal{N}_{ig}$, $\mathcal{N}_{io}$ and $\mathcal{N}_{iw}$ elements defined as SOS2 sets.

$$\eta_{ilgn_g} = \sum_{n_o \in \mathcal{N}_{io}} \sum_{n_w \in \mathcal{N}_{iw}} \lambda_{iln_g n_o n_w}, \ \forall\, l \in \mathcal{L}_i, \ n_g \in \mathcal{N}_{ig} \tag{2.27}$$

$$\eta_{ilon_o} = \sum_{n_g \in \mathcal{N}_{ig}} \sum_{n_w \in \mathcal{N}_{iw}} \lambda_{iln_g n_o n_w}, \ \forall\, l \in \mathcal{L}_i, \ n_o \in \mathcal{N}_{io} \tag{2.28}$$

$$\eta_{ilwn_w} = \sum_{n_g \in \mathcal{N}_{ig}} \sum_{n_o \in \mathcal{N}_{io}} \lambda_{iln_g n_o n_w}, \ \forall\, l \in \mathcal{L}_i, \ n_w \in \mathcal{N}_{iw} \tag{2.29}$$

$$\eta_{ilpn_p} \geq 0, \ \forall\, l \in \mathcal{L}_i, \ p \in \mathcal{P}, \ n_p \in \mathcal{N}_{ip} \tag{2.30}$$

$$\eta_{ilpn_p} \quad \text{is SOS2 for } n_p, \ \forall\, l \in \mathcal{L}_i, \ p \in \mathcal{P} \tag{2.31}$$

Finally, the relationship between the pressure in the manifold and the flowrates q^P_{ilp} in the pipelines (2.11) can be formulated.

$$p^S = p^M_{il} - p^D_{il}, \ \forall\, l \in \mathcal{L}_i \tag{2.32}$$

Aggregated flow variable

To obtain a complete model formulation the linear constraints (2.9) are restated.

$$\sum_{l \in \mathcal{L}_i} q^P_{ilp} = q^C_{ip}, \ \forall\, p \in \mathcal{P} \tag{2.33}$$

Constraints (2.13) - (2.33), together with the common capacity constraints (2.12) and the objective function (2.1), defines the complete MILP problem. This problem will be the basis for the two decomposition strategies investigated in the next section.

2.4 Decomposition strategies

Decomposition approaches for optimization problems is a mature field in operations research. In this section we argue that decomposition is a suitable strategy for the RTPO problem. Alternative decomposition approaches exist and their applicability depends on the structure of the underlying problem. We present two alternative and related strategies, Lagrangian relaxation (LR) and Dantzig-Wolfe decomposition (DWD), as a means to decompose the RTPO problem. A third strategy, Bender decomposition (Dantzig and Thapa, 2003a), is excluded since it cannot exploit the structure in our problem efficiently.

The general idea is to relax global constraints, i.e. constraints which span across large parts of a problem. There are alternative ways of doing this. The basic mechanism in all decomposition principles, however, is to decompose the original problem into smaller subproblems which are coordinated by a master problem. An iterative procedure is then used to achieve convergence towards a global optimal solution.

There are two aspects to consider when choosing which constraints to place in the subproblem. It should be easy to solve since it is re-optimized several times, and it should provide good quality bounds on the intermediate solutions. As mentioned earlier, the presented RTPO problem has a decentralized topology, which is a suitable structure for some decomposition strategies. An oil and gas producing cluster has many internal couplings such as mass and pressure balances, and routing of wells. Further, there are only a few common constraints, in this case total gas and water production. By relaxing these constraints a decomposition strategy may result in one subproblem for each cluster i. Each subproblem can be assigned its own part of the objective function if this is additive on a cluster basis as in (2.1), and if a second term takes common capacity constraints into account.

2.4.1 Lagrangian relaxation

Lagrangian relaxation is a well known technique for finding upper bounds on maximization problems. This involves a technique that attaches Lagrangian multipliers to some of the constraints of the original problem, and relaxes them into the objective function. The resulting problem is then solved, and the value provides an upper bound on the solution of the original maximization problem. If the solution also is feasible with respect to the original problem, it provides a lower bound as well. Further, if common constraints are subject to relaxation, the resulting problem will fall apart into smaller optimization problems, one for each subproblem. A general description can be found in Beasley (1993).

When building a solution method using LR, two issues have to be decided. First, which constraints to relax to define the subproblems, and second, the strategy for how to update the Lagrangian multipliers for the relaxed constraints. Both are presented below.

Relaxed constraints/ Sub-problems

The Lagrangian objective function for the RTPO is presented below. In this case the gas and water handling capacity constraints (2.12) are relaxed. When including all other constraints in the MILP model (2.13) - (2.33) this will represent the Lagrangian problem for the RTPO problem. π_g^{GAP} and π_w^{GAP} are the Lagrangian multipliers associated with the gas and water handling capacity constraints.

$$\max Z = \sum_{i \in \mathcal{I}} q_{io}^C + \pi_g^{CAP}(C_g^T - \sum_{i \in \mathcal{I}} q_{ig}^C) + \pi_w^{CAP}(C_w^T - \sum_{i \in \mathcal{I}} q_{iw}^C) \tag{2.34}$$

This objective function, together with the MILP constraints (2.13) - (2.33), define the Lagrangian problem. By relaxing (2.12) the problem can be separated into one subproblem for each cluster. Hence, we obtain the following objective function (2.35) for each cluster i.

$$\max Z_{(i)} = q_{io}^C - \pi_g^{CAP} q_{ig}^C - \pi_w^{CAP} q_{iw}^C \tag{2.35}$$

This objective function will be solved subject to local constraints only (2.13) - (2.33).

Lagrangian multiplier update

When the subproblems are defined the next step involves updating the Lagrangian multipliers. There are several alternatives in the literature, herein, the subgradient methods described in Beasley (1993) has been implemented.

The procedure is iterative, and generates multipliers in a systematic fashion from an initial set of multipliers. Upper and lower bounds on the solution have to be calculated after each iteration. These bounds can also be used to quantify the quality of the solutions obtained. The upper bound Z_{UB} will equal (2.34) while the lower bound is usually found by some heuristics. In this case we only update this bound when the sum of production of gas and water from all the subproblems is less than the handling capacities. It can then be defined to be equal to the original objective function.

$$Z_{LB} = \sum_{i \in \mathcal{I}} q_{io}^{C} \tag{2.36}$$

Algorithm structure

1. Select ν, satisfying $0 < \nu \leq 2$. Then initialize a lower bound, Z_{LB}, from some heuristics of the problem. We choose $Z_{LB} = 0$. Then select an initial set of Lagrangian multipliers, i.e. π_g^{CAP}, π_w^{CAP}.

2. Solve $\mathcal{I}$ local optimization problems by using the Lagrangian multipliers and get an upper bound on the solution, i.e. Z_{UB}. If the solution also is feasible with respect to the original problem, update the Z_{LB} as well.

3. Compute the subgradients G_g^{CAP}, G_w^{CAP} for the relaxed constraints

$$G_g^{CAP} = \sum_{i \in \mathcal{I}} q_{ig}^{C} - C_g^{T} \tag{2.37}$$

$$G_w^{CAP} = \sum_{i \in \mathcal{I}} q_{iw}^{C} - C_w^{T} \tag{2.38}$$

4. Define a step size T

$$T = \frac{\nu(Z_{UB} - Z_{LB})}{\sum_{p \in \{g,w\}} (G_p^{CAP})^2} \tag{2.39}$$

   The step size depends on the gap between the lower and upper bounds and the user defined parameter ν. The denominator acts as a scaling factor.

5. Update π_p^{CAP} using the rule below and then go to step 2 to resolve the Lagrangian subproblem with this new set of multipliers.

$$\pi_g^{CAP} = \max(0, \pi_g^{CAP} + TG_g^{CAP}) \tag{2.40}$$

$$\pi_w^{CAP} = \max(0, \pi_w^{CAP} + TG_w^{CAP}) \tag{2.41}$$

Convergence and integer handling

The iterative procedure needs a termination criterion. The procedure may terminate after a certain number of iterations or when $Z_{UB} - Z_{LB}$ is below some value, e.g. 1% of Z_{UB}.

LR and the subgradient method is developed for LP problems (Beasley, 1993). Since the subproblems are MILP, the method will not necessarily converge to a 0% gap due to the integer variables, as would be the case for LP problems.

2.4.2 Dantzig-Wolfe decomposition

When applying DWD to the RTPO problem the subproblems will be identical to LR if the same common constraints are subject to relaxation. However, while the Lagrangian multipliers are updated by a simple heuristic in the LR case, the update is now done by solving LP-master problem.

DWD principle

We start by assuming linear constraints and continuous variables, i.e. a LP-problem instead of a MILP problem. The master problem is a reformulation of the original problem. By taking advantage of the fact that a convex combination of basic feasible points, i.e. corner points of the feasible set defined by the linear constraints of a problem, also is a feasible solution, an alternative formulation can be achieved. Each basic feasible point in each subproblem is then represented as a variable in the master problem. Note that each basic feasible point in a subproblem represents a specific production setup for this subproblem. The number of basic feasible points for any practical problem can clearly be prohibitively high, and in reality only a small number of these basic feasible points will ever enter the basis in the master problem. The idea is then to restrict the master problem by reducing the number of basic feasible points. This is called a restricted master problem (RMP). Hence, we start with a few basic feasible points and check if the solution of the original problem is within a convex combination of these points. If this is not the case, new basic feasible points are included in a structured way until the optimal solution has been found, Dantzig and Thapa (2003a) and Williams (2005). Details of the algorithm are given below with some related comments specific to the RTPO problem above.

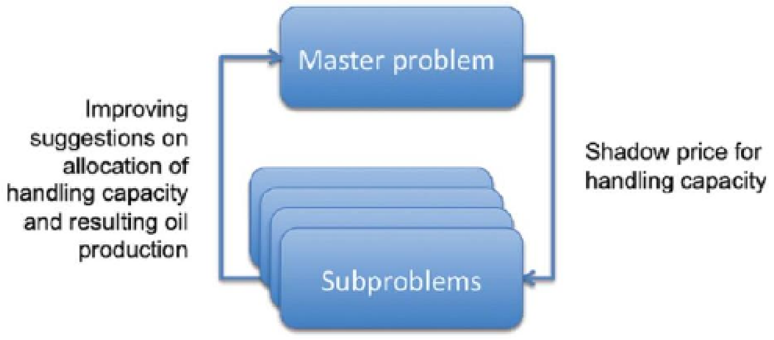

Master problem
Improving suggestions on allocation of handling capacity and resulting oil production
Shadow price for handling capacity
Subproblems

Figure 2.6: Iteration structure for Dantzig-Wolfe decomposition (DWD) and Lagrangian relaxation (LR)

Algorithm structure

1. Choose two initial basic feasible points for each local optimization problem, i.e. two different modes of operating a cluster.

2. Specify the RMP for the given set of basic feasible points. Then solve it and compute values for the Lagrangian multipliers for the global constraints, i.e. π_g^{CAP}, π_w^{CAP}.

3. Solve $\mathcal{I}$ local optimization problems for the Lagrangian multipliers computed in 2.

4a. For $i \in \mathcal{I}$: If the solution of a local optimization problem i lies outside the convex set defined by the basic feasible points used in 2; add these basic feasible points to the RMP, and go to 2. Hence, the RMP is then resolved, also including these new basic feasible points generated in 3.

4b. If the solutions of all the local optimization problems are unchanged, the optimal solution has been found; and the algorithm terminates.

To illustrate, the main iteration loop is shown in Figure 2.6. This figure is also applicable for LR if "Master problem" is replaced by "Subgradient updating".

Sub-problem

The procedure is to update the Lagrangian multipliers such that the consumption of the relaxed common constraints converge to their optimal values. Let q_{ig}^C, q_{io}^C and q_{iw}^C denote the output flow of the phases in a feasible solution to cluster i. The corresponding reduced cost $\overline{c}_{(i)}$, if these values are used to construct a column in the master problem:

$$\bar{c}_{(i)} = q^C_{io} - \pi^{CAP}_g q^C_{ig} - \pi^{CAP}_w q^C_{iw} - \pi^C_i, \ \ \forall\, i \in \mathcal{I} \tag{2.42}$$

Where π^{CAP}_g and π^{CAP}_w are the Lagrangian multipliers in the master problem for the common gas and water handling capacity constraints (2.44) and (2.45) while π^C_i is the multiplier for the convexity constraint for cluster i (2.46). The objective in the subproblem for this cluster is to find the maximum reduced cost among feasible solutions for the cluster. By maximizing $\bar{c}_{(i)}$, given the local constraints (2.13) - (2.31), this column will be found. If this value is positive then a new column, say s, is added to the master problem with Z_{isg}, Z_{iso} and Z_{isw} equal to the optimal values of q^C_{ig}, q^C_{io} and q^C_{iw}, respectively, from the subproblem. s is added to set $\mathcal{S}_i$, which is used in (2.43).

As mentioned π^C_i are the Lagrangian multipliers for the convexity constraints in the RMP defined below. Since no subproblem variables are associated with it, it will only act as a constant in the subproblem.

Restricted master problem

The RMP can now be formulated. The production rate corresponding to each Z_{isp} represents one basic feasible point s from subproblem i, i.e. a possible production allocation for cluster i. Z_{isp} could in principle include the optimal value of all decision variables for subproblem i after solving it given π^{CAP}_g and π^{CAP}_w. However, only the variables also present in the objective function and the common constraints will be relevant for the RMP. Hence, Z_{isp} will for this RTPO problem contain the flow variables ($q^C_{ig}, q^C_{io}, q^C_{iw}$), but no pressure and routing variables. μ_{is}, the optimization variables for the RMP, is the weight the RMP will give this basic feasible point after it is solved to optimality. The objective function of the restricted master problem is given in (2.43). Further, (2.44) and (2.45) represent the constrained common resources, while (2.46) are the convexity constraints.

$$\max \sum_{i \in \mathcal{I}} \sum_{s \in \mathcal{S}_i} Z_{iso}\mu_{is} \tag{2.43}$$

$$\sum_i \sum_{s \in \mathcal{S}_i} Z_{isg}\mu_{is} \leq C_g^T \tag{2.44}$$

$$\sum_i \sum_{s \in \mathcal{S}_i} Z_{isw}\mu_{is} \leq C_w^T \tag{2.45}$$

$$\sum_{s \in \mathcal{S}_i} \mu_{is} = 1, \quad \forall\, i \in \mathcal{I} \tag{2.46}$$

$$\mu_{is} \geq 0 \tag{2.47}$$

Integer handling

DWD guarantees to find optimal solutions for LP problems. If it is extended to a MILP problem, branch & price (Desrosiers and Lübbecke, 2006) or some heuristics have to be applied to handle the integer variables. When solving the master problem, we have not imposed integer restrictions on μ_{is}, i.e. the RMP is solved as a LP to achieve Lagrangian multipliers for (2.44) - (2.46). The resulting solution may then be infeasible with respect to the original MILP problem since a convex combination of two basic feasible points may not be feasible. As an example a basic feasible point represents a specific production allocation for a particular cluster, while another basic feasible point represent another allocation for the same cluster. A convex combination of these is impossible if the two allocations have different routings, i.e. different integer solutions.

As mentioned, integer variables could be handled in several ways. In this work we apply the following heuristic approach: If a sufficient number of basic feasible points is generated up front, a feasible solution can simply be found by demanding integer values for μ_{is} by solving the RMP as an integer programming (IP) problem.

Convergence

For both LP and MILP problems, upper and lower bounds on the objective function can be computed. The LP solution of the RMP plus the sum of the objective values of the subproblems, i.e. the reduced cost, will act as an upper bound Z_{UB}, Karlof (2006). In the LP case, the solution of the RMP alone will give a feasible lower bound, while for the MILP problems, the master problem has to be solved as an IP or some heuristics have to be applied to create the feasible lower bound Z_{LB}. By using these bounds actively during the optimization process, it is possible to terminate the algorithm when an acceptable gap is achieved.

2.5 Implementation

An RTPO has been investigated by two decomposition strategies, LR and DWD. These strategies are compared with the solution of the integrated problem before relaxation. In this way, we compare the effect of the two decomposition strategies against each other, and against a regular global method.

Piecewise linear well and pipeline models were generated as follows: Initially analytic functions approximating data from Troll C were created. These functions were then used to generate breakpoints for the piecewise representation of the models. This information was subsequently stored as tables. For the well models; this means that for every wellhead pressure breakpoint, there is a related gas, oil and water flowrate. For the pipeline models on the other hand; every combination of gas, oil and water flowrates give a pressure drop across it. If the solution algorithm requests a pressure between the breakpoints, it will simply linearly interpolate between the neighboring points. The reason for introducing the analytic functions was to obtain flexibility when choosing the interpolation points for the well models used in the optimization. The underlying data points vary in resolution since some operating regions are more heavily sampled than others.

Both optimization algorithms are implemented in Xpress-IVE which is a state-of-the-art software for mixed integer linear problems. Breakpoint tables together with other topology information are supplied through data files. The user is able to choose between solving the problem by a global solution algorithm, or with LR or DWD. In the case of the global method, all data will be loaded into the solver and solved directly.

2.6 Results

The Troll field is a huge oil and gas field on the continental shelf west of Norway. There are severe production optimization challenges due the size of the asset and because of rate dependent gas-coning wells (Hauge and Horn, 2005). We study a model of the Troll C production system shown in Figure 2.1 where primarily oil is produced from an oil rim through more than 50 wells.

This system changes with time since new wells are drilled and commissioned continuously. Further, the reservoir conditions change significantly due to medium and long-term depletion effects. Hence, well models are usually updated twice a year by running well tests to collect data for parameter estimation. Within a day or a week, however, changes are usually quite limited. Therefore RTPO as presented in this chapter has a potential for increasing value creation.

For the moment, only one common constraint is active; the total handling capacity for gas production (2.12). Water handling will, however, become an issue in the future as the reservoir drains and therefore produces more water. The production system includes 8 clusters and each cluster contains $6-8$ wells and two manifolds.

The purpose of the numerical study is to investigate three solution strategies

1. A global strategy where all clusters are solved in one large MILP problem.
2. The LR strategy we propose in this chapter.
3. The DWD strategy we propose in this chapter.

The results shown in Table 2.9 are an extension of the results in Gunnerud et al. (2009). The values in the table are representative numbers for the different cases and the three alternative methodologies. The computations are performed on an IBM Thinkpad T60P with a $2.33GHz$ processor.

The solution strategies were tested on instances with for two, four, six and eight clusters as shown in Table 2.9, and the average gas capacity per cluster was chosen at about 3000 Sm^3/day per cluster. The large number of continuous and discrete variables per cluster comes from the piecewise linearization approach. A typical well model is linearized by $20-100$ breakpoints. The total number of breakpoints in the pressure drop models is much higher since it is necessary to interpolate in three dimension; the gas, oil and water flowrates.

2.7 Discussion

There are two main contributions in this chapter. The first is the formulation of a full field RTPO model only containing linear and integer variables while the second is a detailed description of two methods for decomposing the RTPO problem into smaller and hence easier problems.

As mentioned earlier, by piecewise linearizing all nonlinearities and thereby creating a MILP formulation it is possible to apply known and well developed algorithms such as the simplex algorithm and the branch & bound algorithm to solve the RTPO problem. An additional effect of this formulation is the fact that the underlying wells and pipeline models are completely disconnected from the MILP formulation. Hence, they could be replaced on a later occasion without affecting the structure and

Table 2.9: Results

No of clusters	2	4	6	8
Gas cap [Sm^3/day]	3000	12000	18000	24000
Continuous variables	13898	27805	41766	55688
Discrete variables	1029	2134	3725	4819
Constraints	491	981	1639	2185
Strategy 1 - global				
Solution time [min]	0.26	7.38	237.0	720.0
Oil prod [Sm^3/day]	1777	6487	11641	14365
Optimality gap [%]	0.00	0.00	0.00	7.50
Strategy 2 - LR				
Solution time [min]	1.42	8.54	18.2	19.2
Oil prod [Sm^3/day]	1774	6467	11640	14440
Optimality gap [%]	0.17	0.30	0.21	0.38
LR iterations	12	32	16	13
Strategy 3 - DWD				
Solution time [min]	1.86	1.43	6.16	11.3
Oil prod [Sm^3/day]	1777	6458	11629	14473
Optimality gap [%]	0.02	0.46	0.36	0.15
DWD iterations	10	4	3	5

solution procedure of the MILP formulation. Hence data points can be created from any reservoir, well and pipeline simulator as an alternative to real production data.

Flexible accuracy is an attribute of the formulation since the break points can be placed arbitrarily. The price to pay for higher accuracy, however, is a longer calculation time.

There are still challenges using the piecewise linearization approach for pipeline models. Since the pressure drop depends on the gas, oil and water flowrates the number of weighting variables increases with the power of three. 20 breakpoints for each phase will e.g. result in 8000 interpolation elements ($\lambda_{iln_gn_on_w}$). In addition there will be $20 \times 3 = 60$ integer variables (η_{ilpn_p}) in this case. Since the number of weighting variables increases so rapidly it is essential to minimize the number of breakpoints. This has however not been studied in any detail in the present chapter.

The second contribution in this chapter, the decomposition methods, is tested in a realistic numerical study. The decomposition strategies, DWD and LR, perform better than the global method for the combined rate allocation and routing problem on all instances. Moreover, it may be observed that the global method does not converge for the eight cluster instance and it has a hard time solving instances consisting of more than 6 clusters.

To elaborate on the above the global method finds the optimal solution for all except the full field instance with 8 clusters. In this case it terminates after 12 hours with more than a 7.5% duality gap. DWD and LR terminate with less than 0.5% duality gap for all instances.

The global method works fine for the 2 and 4 cluster instances. This is not a surprising result since the benefit of a decomposition strategy is limited in these cases. For larger instances, however, we observe that the two decomposition methods are much faster than the global method. Further, DWD performs better than LR. The reason is the mechanism for updating of Lagrangian multipliers. The DWD master problem finds good multipliers with fewer iterations than for the LR case, and usually converges after fewer iterations. This is observed in the table where DWD converges in 3 and 5 iterations for the 6 and 8 cluster instances while LR requires more than 10 iterations. The run-time for solving the DWD master problem, i.e. the LP-problem, is negligible compared to solving the local MILP problems even though it is more time-consuming than solving the LR master problem. Hence, this is no issue when comparing DWD and LR.

DWD is more stable with respect to solution time than LR. DWD has fewer tuning parameters and works well for different data sets, LR is actually quite sensitive to perturbations of the data set. A small change may for instance double the solution time.

Decomposition algorithms like DWD or LR have some interesting properties. First, the subproblems may be solved by different algorithms or even different software packages. This feature has a potential for value chain optimization applications which may encompass reservoir, wells, pipelines and downstream processing facilities. The duality gap, however, can only be computed if upper and lower bounds on the solution can be found. This is in general not possible if the subproblems are nonlinear programs as opposed to MILPs. Second, it is quite obvious that decomposition methods are suitable for parallel computing since each subproblem is self-contained and has no direct dependence on the other subproblems.

The routing and well allocation problem is usually solved by re-optimizing the stationary optimization problem, typically once a day. Hence, it is treated in a quasi-dynamic way. A couple of wells are usually selected for more frequent production changes to compensate for variations in for instance gas processing capacity. The use of dynamic models is an issue, in particular during start-up of wells. Start-up occurs quite often since wells are shut-in from time to time due to maintenance or operational problems. Further, applications with long pipelines may benefit from dynamic pipeline models provided the dynamics are important for optimal performance.

There are at least two trends which can aid the implementation of the methodology proposed in this chapter. First, the possibility to include third party applications into commonly used software systems for monitoring and control purposes has increased significantly in the latter years. Second, there is an initiative to develop a new standard for production optimization, PRODML, which is supported by many leading vendors and end users.

2.8 Conclusions

This chapter presents a complete MILP model for the RTPO problem in the petroleum industries. Further, the chapter argues that decomposition is well suited for this problem. There are many reasons for this. Decomposition methods clearly outperform a global method for full field problems in terms of computational efficiency. Further, DWD gives better performance than LR on all the relevant instances tested in this chapter and is therefore the preferred decomposition method. This is because of a more efficient updating algorithm of the Lagrangian variables. The MILP formulation allows the computation of a duality gap on the solution of the production optimization problem. This is clearly information of interest to any user.

Chapter 3

Decomposition and parallel computation

The results presented in this chapter is based on Gunnerud et al. (2010), "Parallel Dantzig-Wolfe decomposition for real-time optimization - Applied to a complex oil field", published in Journal of Process Control, and on Torgnes et al. (2011) "Parallel Dantzig-Wolfe decomposition of petroleum production allocation problems", accepted for publication in Journal of the Operational Research Society. The text of the chapter is mainly taken from the latter paper. My contribution to this paper has been on the development of the methodology, while the writing and computations have mainly been done by Erlend Torgnes.

Abstract

This article discusses the optimization of a petroleum production allocation problem through a parallel Dantzig-Wolfe algorithm. Petroleum production allocation problems are problems in which the determination of optimal production rates, lift gas rates and well connections are the central decisions. The motivation for modelling and solving such optimization problems stems from the value that lies in an increased production rate and the current lack of integrated software that considers petroleum production systems as a whole. Through our computational study, which is based on realistic production data from the Troll West field, we show the increase in computational efficiency that a parallel Dantzig-Wolfe algorithm offers. In addition, we show that previously implemented standard parallel algorithms lead to an inefficient use of parallel resources. A more advanced parallel algorithm is therefore developed to improve efficiency, making it possible to scale the algorithm by adding more CPUs and thus approach a reasonable solution time for realistic sized problems.

Keywords: Optimization, Dantzig-Wolfe, Decomposition, Oil and Gas, Parallel Computing

3.1 Introduction

The large value of a small improvement in the production rate combined with the often complex nature of the production systems, make it interesting to investigate the potential for using optimization methods to support planning of the production for petroleum fields. Previous work has shown that decomposition methods can be powerful tools for solving large problems with a certain intrinsic structure often met in the process industry, see Tebboth (2001), Cheng et al. (2008) and Alabi and Castro (2009). Moreover, Gunnerud and Foss (2009), Foss et al. (2009) and Gunnerud et al. (2010) have shown that decomposition approaches, such as Dantzig-Wolfe decomposition, work very well for petroleum production allocation problems. However, they also show that a further reduction in solution time is necessary for such algorithms to be of practical value. This because such problems require that a solution is provided within hours as production planners should be able to evaluate and re-optimize iteratively. In this article we study petroleum production allocation problems with an intrinsic decentralized structure, and further develop existing solution methods for such problems by exploring advanced parallel algorithms in combination with Dantzig-Wolfe decomposition. A naive parallel algorithm was employed with some success in Gunnerud et al. (2010), but the need for a more scalable algorithm, that enables one to reduce the solution time by adding more CPUs, and further testing was raised. In this article, by testing on realistic data we specifically study how the hardware platform employed can be utilized so that solutions can be obtained within a time-frame that ensures their practical value.

The greatest advances in computing power on easily accessible hardware have the last few years come through an increase in the number of CPUs in a computer, and not as a result of an increase in CPU clock speed. It is therefore important for optimization practitioners to design algorithms that can utilize parallelization if they are to continue to benefit from the advances in computing power predicted by Moore's law (Moore, 1965), (Moore, 1975). While the performance of high end CPUs has increased by about 40% a year the last decades, memory access speed has only improved by roughly 10% in the same period. This constitutes a growing gap between CPUs' speed and memory, and is a bottleneck for utilizing advances in computing power. However, parallel platforms generally also yield better memory system performance because they provide larger aggregate caches and higher aggregate bandwidth to the memory system, thus furthering the importance of designing algorithms that allow utilization of parallelization (Grama et al., 2003).

Tebboth (2001) demonstrated that for certain problems a significant reduction in the solution time can be obtained by designing a parallel Dantzig-Wolfe algorithm. We will show how parallel Dantzig-Wolfe algorithms also can be used to reduce the solution time for petroleum production allocation problems. In this paper we are able to reduce the solution time by up to 69%, compared to a sequential algorithm, by developing a simple parallel Dantzig-Wolfe algorithm for a realistic production allocation problem resembling the Troll West field. By implementing a more advanced strategy we are able to reduce the solution time by up to 84%, and illustrate how the algorithm could be scaled if a larger parallel platform with more CPUs was available.

The layout of the rest of this chapter is as follows: The next section present the properties of the production system combined with the mathematical model for the problem. Section 3.4 present the solution methods used while Section 3.5 present the parallel algorithms that have been developed and employed. Section 3.6 present the Troll field test case and the results are presented in Section 3.7. Their implications are further discussed in Section 3.8 and the main conclusions presented in the final section.

3.2 Problem Formulation

The production allocation problem in this article represents a general offshore platform production system that consists of a platform and a set of wells grouped in clusters, each connected to the platform through one or several pipelines. Each cluster is again comprised of a number of manifolds at the sea floor which the wells are connected to. A typical field will have a number of such clusters, depending on how old the field is, as fields normally are under continuous development. An example of the layout of a single cluster is illustrated in Figure 3.1. In this example there are two manifolds, four wells connected to each manifold and two pipelines between the manifolds and from the manifolds up to the separator.

An essential part of modelling optimization problems for petroleum fields is defining the time horizon for the problems. Different frameworks that divide planning in petroleum production have been introduced, see Ulstein et al. (2005), Schlumberger (2005) and Foss et al. (2009). Foss et al. (2009) define decisions with a time horizon

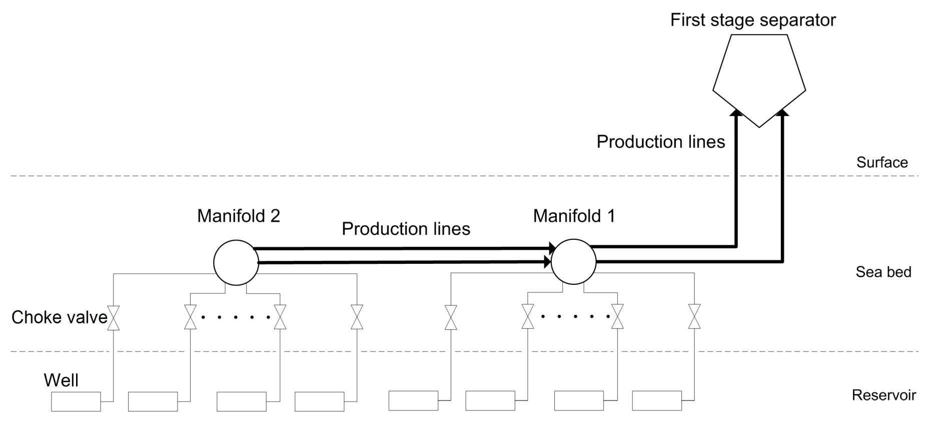


Figure 3.1: Layout of a well cluster for a typical field

of a day to a few weeks real-time production optimization (RTPO). In this chapter we consider production planning with a time horizon of a day to a week and assume that the fields we plan for are in steady-state over this time period. With such a time horizon central decisions include the rate of production from each well and how to route the well flow if there are multiple production lines. Ideal allocation of available gas and water capacity is often a bottleneck since the total oil production may be constrained by the gas and/or water handling capacity at the platform. Other constraints may also come into play, for instance available gas for gas-lift or pressure bounds in the inlet separator. As mentioned, production allocation problems require that a solution is provided within hours, preferably within minutes, as the production planners should be able to solve, evaluate and re-optimize iteratively. An allocation suggested by a solution of the production allocation problem serves as a recommendation to the operating engineers who may or may not adhere to the suggested allocation. The operating engineers might choose to ignore a recommendation if the estimated gain is small or considered too uncertain.

When well interaction can be ignored, a table representing the performance of each well can be calculated and the wells prioritized according to this table until the production capacity is reached. Such problems can thus be solved rather easily. However, production systems in which well flow interaction is significant require a different formulation and solution approach (Wang et al., 2002). In the type of problems we discuss in this chapter several wells in one cluster are connected to the same pipelines in at least one of the clusters. For these the pressure/production from one well influences the pressure/production in other wells in the same cluster. Such production allocation problems are therefore much harder to solve as nonlinear pressure-flow relations have to be modelled to take into account the well interaction. The flow from the reservoir into the wellbore and up to the manifold consists of a gas, oil and water phase. Modelling a multiphase flow has been and continues to be a challenge. The reason for this is that the pipe flow has a flow regime that is very dependent on several other factors like the flow rate of the different phases, the pipe characteristics and the angle of the pipe at different points. For a thorough description of multiphase flow and different flow regimes, see Chapter 7 in Brennen (2005)

The production allocation problem that results from the above described production system is a mixed integer nonlinear program (MINLP). This is because of the nonlinear pressure-flow relations and the fact that there may be more than one production line, either of which a well could be routed to, thus introducing integer variables and making the problem nonconvex. We approximate the production system by a mixed integer linear program (MILP) by using piecewise linearization and special ordered sets of type 2 (SOS2) to represent the multidimensional nonlinear functions that characterize the pressure and flow behavior in the wells and the pipelines. The SOS2 requirement is necessary since the LP relaxation of the MILP will not give neighboring solutions for weighting variables on the piecewise linear curve. This is a result of us wanting to maximize oil flow rates and minimizing water and gas rates while all three pressure-flow relations are convex. The formal problem formulation is presented in Section 3.3. Piecewise linearization and SOS2 has on a number of occasions been used to linearize nonlinear pressure-flow relations in different petroleum optimization problems, see Handley-Schachler et al. (2000), Kosmidis et al. (2005) and Bieker et al. (2006b). More recently it has also been used for production systems and reservoirs with the same characteristics as in this article, see Gunnerud and Foss (2009) and Gunnerud et al. (2010). An obvious advantage of resorting to a MILP formulation is the simplicity with which an upper bound, and hence a duality gap, can be found.

3.2.1 Decentralized optimization and parallel computation

The production allocation problem has a block angular structure as defined by Williams (2005). The limited handling capacity of gas and water leads to a set of common constraints, while each cluster of wells correspond to blocks of constraints that can be solved independently if the common capacity constraints on gas and water are ignored. After a piecewise linearization was introduced this special structure was utilized by testing different decomposition methods in Gunnerud and Foss (2009). Gunnerud and Foss (2009) tested both Lagrangian relaxation (LR) and Dantzig-Wolfe decomposition (DWD). The authors showed that utilizing decomposition methods for the problem resulted in large reductions in solution time and that DWD performed better than LR as a result of a more efficient updating algorithm for the dual variables.

Combined with a decentralized optimization approach such as DWD and LR the block angular structure of the problem lends itself to parallel algorithms with the potential of a further reduction in solution time. A conceptual explanation of how this can be done, using DWD as an example, can be found in Chapter 6 in Kontoghiorghes (2005), and is explained briefly below.

In the parallel algorithm's most basic form the subproblems are solved concurrently in different solution processes, each normally equating to one CPU, using what is termed barrier synchronization (Foster, 1995). Each CPU performs its task until it reaches the barrier. It then stops and waits until all tasks have reached the barrier before they are synchronized. The basic barrier criterion in a parallel DWD algorithm is that a subproblem has been solved to its optimality. Using this criterion all subproblems are solved to optimality before the serial processing of the master problem and the, in most cases, subsequent parallel resolving of the subproblems. In the most straightforward parallel implementation, the mapping of tasks to processes is static, that is, before the algorithm starts it is predetermined which CPU handles which subproblem and in which sequence (Grama et al., 2003). The objective of a good mapping is to obtain load balancing, i.e. distributing work among processes so that all processes are constantly busy. Load balancing can also be considered as a minimization of the total process idle time. Good load balancing is central for an optimal utilization of available parallel resources when the solution processes are subject to a barrier synchronization point since the slowest *bottleneck* problem will determine the performance of the algorithm. How difficult it is to obtain good load balancing is determined by the granularity of the algorithm, which is the ratio of parallel computation to communication and serial computation. A DWD algorithm for a production allocation problem is normally coarse-grained as the time to solve the master problem is small compared to the subproblems. This makes load balancing harder as it will be more likely that some processes are idle for a larger part of the time.

Ho et al. (1988) were the first to implement a parallel DWD strategy. The results they obtained clearly indicate that a significant reduction in solution time can be obtained by utilizing parallel processes if a problem has the suitable structure. Their initial analysis revealed that a static mapping of tasks to processes and barrier synchronization is unattractive as much time is lost by processes being idle when the subproblems have unequal solution times. A strategy they tried proved successful on several problems and is termed the Accelerated feedback strategy. It is in general terms described as a strategy where processes are kept active if possible, with dual prices from the master problem and candidate solutions from the subproblems communicated as soon as they become available. In this strategy the master problem is resolved every time a subproblem has finished solving and any subproblem not currently being solved is restarted when the master problem finishes.

In Gnanendran and Ho (1993) the authors implemented a more advanced parallel DWD code to improve load balancing. In their article they conclude that measures such as modifying the way in which columns are sent to the master problem does increase speedup (ratio of best serial time to parallel time) and efficiency (speedup per processor used). However, they remark that it is hard to obtain a high efficiency when the serial solving of the master problem is computationally difficult compared to the subproblems.

In Tebboth (2001) the parallel DWD algorithms introduced in Ho et al. (1988) and in Gnanendran and Ho (1993) were further developed. On certain large scale primal block angular problems Tebboth was able to obtain a reduction in the solution time of 63% using a sequential DWD algorithm compared to a commercial implementation of the simplex method. A further speedup of 2.42 and 5.22 was obtained using multi-processor PCs with respectively four and seven CPUs, with the theoretical maximum speedup being 4 and 7. His main conclusion was that for problems with a block angular structure it is worth examining the potential of a parallel DWD algorithm, in particular when the context demands rapid results, as he demonstrated that it has the potential of beating modern commercial implementations of the simplex method for certain practical problems.

One of the most recent works on parallelization of DWD algorithms is, as mentioned earlier, Gunnerud et al. (2010). The algorithm developed in this article makes use of a static mapping of tasks to processes and barrier synchronization and is a very basic implementation of a parallel algorithm. Though the authors obtained a speedup of about 2.7 with eight CPUs, this is well below the theoretical maximum of eight and there are clear scaling issues with this algorithm. If the subproblems have unequal solution times, time will be lost as all other CPUs have to wait for the slowest subproblem to be solved. When more CPUs are used to solve the problem scaling will increasingly become an issue as more and more CPUs have to wait for the slowest subproblem to finish.

Ideally the speedup would be linear when introducing multiple CPUs and a parallel algorithm. That is, a doubling of the number of CPUs would equate to a halving of the solution time. However, virtually all optimization algorithms have some parts that have to be processed sequentially. The minimum solution time is therefore equal to the time it takes to execute the sequential steps in the algorithm (Amdahl, 1967). The proportion of the algorithm that has to be processed sequentially thereby limits the maximum performance increase. However, as was shown recently in Rios and Ross (2009) a large speedup can be obtained if the necessary structure is present in the problem. An easily accessible and good introduction to parallel computing is Foster (1995). For an overview of parallel computing in linear programming, which is central for solution methods applied in this article, the reader is referred to Hall (2008).

3.2.2 Modelling the field

The formulation introduced in Foss et al. (2009) and Gunnerud and Foss (2009) is the basis for the mathematical model we will use. We represent the pipeline flow using the common Black Oil model, which entails simplifying the representation of reservoir flow to one flow variable for each of the phases gas, oil and water (Fevang et al., 1997). We model the problem with these three flow variables and solve an optimization problem that aims to maximize oil production while not exceeding the handling capacity of gas and water in the first stage separator at the platform. Maximization of the oil production rate is a reasonable objective as it ensures the best utilization of the resources invested in the production process. Over the life of a field the total oil production can also often be increased by increasing the production rate as a higher production rate will likely extend the period in which oil production at the field is profitable.

To ease readability the model presented in Section 3.3 is formulated for clusters with only one manifold. However, expanding the model to also allow for clusters with multiple manifolds is rather straightforward and was implemented to produce the results we discuss later.

Modelling reservoir performance

The pressure drop in the pipelines from the wells and up to the manifold is, as a function of gas, oil and water rates, highly nonlinear. To model the interaction between the reservoir and a well, inflow performance relationship (IPR) curves and vertical lift performance (VLP) curves are typically used (Dueñas Díez et al., 2005). The IPR curves model the inflow from the reservoir into the wellbore, i.e. into the bottom part of the well, and are typically a function of the pressure gradient between the wellbore and the reservoir. The VLP curves account for the hydrodynamic and friction effects in the well. The VLP and IPR curves can be combined into a single curve, a well performance curve (WPC), which is used to represent the production rates for a well at different wellhead pressures. The WPCs, illustrated in Figure 3.2, are hence a function of one variable and nonlinear as they describe the multiphase flow mentioned above. Similarly to Gunnerud and Foss (2009), we will approximate these curves using piecewise linearization. The wellhead pressure, p_{ij}^{WELL}, is linked to the flow of the different phases, q_{ijp}^{WELL}, using (3.3) and (3.4), while (3.5) - (3.7) ensure that a solution is an interpolation between two neighboring breakpoints in the piecewise linearization. There is only one weighting variable, γ_{ijk}, associated with each breakpoint in the linearization of the WPCs. Note therefore that the pressure drop is mapped to three flow variables.

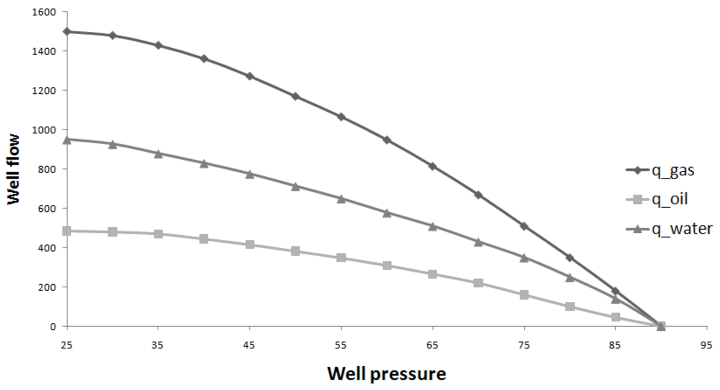


Figure 3.2: Illustration of a well performance curve (WPC)

Modelling pressure-flow relations in the pipelines

In the production systems several wells are often connected to the same production lines. The pressure and production in different wells therefore affect each other and as a result it is necessary to model the pressure drop in the pipelines from the manifold to the separator. Modelling the pressure drop in these pipelines is complex and, in reality, it is a function of a number of variables including temperature, absolute pressure, pipe angle and the flow rates of the different fluids. Models exist that produce a pressure drop given a certain flow regime. However, the flow regime is typically not constant over time and creating a correct pressure drop function is therefore very challenging. More information regarding the challenges of modelling pressure-flow relations can be found in e.g. Hauge and Horn (2005).

The pressure-flow relations in the pipelines will, as in Gunnerud and Foss (2009) and Foss et al. (2009), be modeled as a function of the three phases in the pipelines: gas, oil and water rates. By extending the concept of piecewise linearization to three dimensions these nonlinear pressure-flow relations are also linearized. This is done by introducing a grid of values $(q_{ilg}^{PIPE}, q_{ilo}^{PIPE}, q_{ilw}^{PIPE})$ for each pipeline l in cluster i representing the flow of the different phases in the pipeline. Associated with each point in the grid there is a non-negative weighting variable, $\lambda_{iln_gn_on_w}$, where $n_g \in \mathcal{N}_{ig}$, $n_o \in \mathcal{N}_{io}$ and $n_w \in \mathcal{N}_{iw}$ are the set of break points along the axis representing respectively gas, oil and water flow in a pipeline, see Figure 3.3.

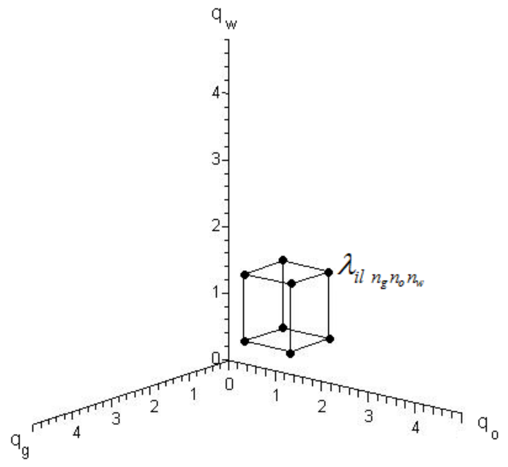


Figure 3.3: The eight neighbors of an interpolation point in 3 dimensions

If we denote the values of $(q_{ilg}^{PIPE}, q_{ilo}^{PIPE}, q_{ilw}^{PIPE})$ at the breakpoints as $(Q_{ign_g}^{DROP}$, $Q_{ion_o}^{DROP}, Q_{iwn_w}^{DROP})$ and the associated pressure drop $(P_{in_g n_o n_w}^{DROP})$, the tridimensional pressure drop function can be approximated by (3.13) - (3.16).

To ensure that an interpolated solution is only between neighboring points we have to make sure that at most eight neighboring weighting variables can be strictly positive. These eight neighbors correspond to each of the corner points in a cube surrounding the solution, as illustrated in Figure 3.3. This is a generalization of the SOS2 conditions and can be imposed by introducing (3.17) - (3.21) to our formulation.

A more thorough description of the linearization of the pressure-flow relations in the pipelines can be found in Gunnerud and Foss (2009), who further developed methods for piecewise linearization introduced in Beale and Tomlin (1969) and expanded on in Williams (2005) and Bieker (2007).

3.3 Optimization model

Declarations

Indices

i	Cluster
j	Well
l	Pipeline
p	Phase
k	Interpolation coordinate for the piecewise linearization of the WPC
n_p	Interpolation coordinate for the piecewise linearization of the pressure drop in the pipelines, dependent on phase p (n_g for gas, n_o for oil and n_w for water)

Sets

$\mathcal{I}$	Set of clusters
$\mathcal{J}(i)$	Set of wells in cluster i
$\mathcal{L}(i)$	Set of pipelines in cluster i
$\mathcal{P}$	Set of phases (g for gas, o for oil and w for water)
$\mathcal{K}(i,j)$	Set of interpolation coordinates for the piecewise linearization of the WPC for well j connected to cluster i
$\mathcal{N}(i,p)$	Set of interpolation coordinates for the piecewise linearization of the pressure drop contributed by phase p in the pipelines from the manifold to the separator in cluster i

Data

C_p^{TOT}	Maximum capacity of phase p in the first–stage separator at the platform
$P_{in_gn_on_w}^{DROP}$	Pressure drop in a pipeline from the manifold to the separator in cluster i corresponding to interpolation coordinate n_p for all phases p, in the piecewise linearization of the pressure drop in pipelines
P_i^{MAX}	Maximum pressure before choke for all wells in cluster i
P_{ij}^{MIN}	Minimum pressure before choke for well j at the manifold in cluster i
P^{SEP}	Pressure at the first stage separator on the platform
P_{ijk}^{WPC}	Pressure at the manifold level before the choke in the pipeline from well j connected to the manifold in cluster i in the piecewise linearization of the WPC with interpolation coordinate k
$Q_{ipn_p}^{DROP}$	Flow rate from the manifold to the separator in cluster i corresponding to interpolation coordinate n_p for that particular phase p in the piecewise linearization of the pressure drop in the pipelines
Q_{ijp}^{MAX}	Maximum flow of phase p from well j in cluster i up to the manifold
Q_{ijpk}^{WPC}	Flow rate of phase p from well j connected to the manifold in cluster i corresponding to interpolation coordinate k in the piecewise linearization of the WPC

Variables

p_{il}^{M}	Pressure at manifold level (upstream side) in pipeline l in cluster i
p_{il}^{DROP}	Pressure drop across a section of pipeline l with flow from the manifold and up to the separator in cluster i
p_{ij}^{WELL}	Pressure at manifold level upstream from choke in pipeline from well j connected to the manifold in cluster i
q_{ilp}^{PIPE}	Flow rate of phase p in pipeline l from the manifold and up to the separator in cluster i
q_{ijlp}^{WRP}	Flow rate of phase p routed to pipeline l from well j connected to the manifold in cluster i
q_{ijp}^{WELL}	Flow rate of phase p from well j to the manifold in cluster i
x_{ij}	Binary variable. Equals 1 if well j in cluster i is closed, 0 otherwise
y_{ijl}	Binary variable. Equals 1 if well j in cluster i is routed to pipeline l, 0 otherwise
γ_{ijk}	Weighting variable associated with each interpolation coordinate k in the piecewise linearization of the WPC for well j in cluster i
$\lambda_{iln_gn_on_w}$	Weighting variable associated with the interpolation coordinates for n_g, n_o and n_w in the piecewise linearization of the pressure drop in pipeline l from the manifold to the separator in cluster i
$\eta_{ilpn_p}^{SUB}$	Weighting variable used in the modelling of the SOS2 variables in the piecewise linearization of the pressure drop in pipelines for the interpolation coordinate for pipeline l in cluster i in interpolation set n_p dependent on phase p

3.3.1 Objective function

As discussed earlier the objective is to maximize the total quantity of oil produced. This is expressed in the objective function as the sum of oil flow in all pipelines from all clusters.

$$\max\ Z = \sum_{i\in\mathcal{I}} \sum_{l\in\mathcal{L}(i)} q_{ilo}^{PIPE} \tag{3.1}$$

3.3.2 Common constraints

The constraints can be divided into common constraints and well cluster constraints. Handling capacity is the only common constraint over all clusters.

Handling capacity The sum of gas rates and the sum of water rates from all clusters must, respectively, be less than the gas handling and water handling capacity of the first stage separator.

$$\sum_{i \in \mathcal{I}} \sum_{l \in \mathcal{L}(i)} q_{ilp}^{PIPE} \leq C_p^{TOT} \ , \ p = \{g, w\} \tag{3.2}$$

3.3.3 Cluster constraints

The constraints for linearization of well performance curves, mass balance, pressure requirements, linearization of pressure drop, closing of wells, removal of symmetry and requirements on variables all apply within one well cluster. As all well cluster constraints are duplicated for each cluster, i, we have omitted writing $\forall\, i \in \mathcal{I}$ in the following constraints for simplicity.

Linearization of well performance curves The linearization of the WPCs is expressed in the constraints below. As mentioned earlier the wellhead pressure determines the flow of gas, oil and water from a well to the manifold it is connected to. The γ variables are modeled as SOS2.

$$p_{ij}^{WELL} = \sum_{k \in \mathcal{K}(i,j)} P_{ijk}^{WPC} \gamma_{ijk} \ , \ \forall\, j \in \mathcal{J}(i) \tag{3.3}$$

$$q_{ijp}^{WELL} = \sum_{k \in \mathcal{K}(i,j)} Q_{ijpk}^{WPC} \gamma_{ijk} \ , \ \forall j \ \in \mathcal{J}(i), \ p \in \mathcal{P} \tag{3.4}$$

$$\sum_{k \in \mathcal{K}(i,j)} \gamma_{ijk} = 1 \ , \ \ \forall j \ \in \mathcal{J}(i) \tag{3.5}$$

$$\gamma_{ijk} \geq 0 \ , \ \ \forall j \ \in \mathcal{J}(i), \ k \in \mathcal{K}(i,j) \tag{3.6}$$

$$\gamma_{ijk} \text{ is SOS2} \ , \ \ \forall j \ \in \mathcal{J}(i), \ k \in \mathcal{K}(i,j) \tag{3.7}$$

Note that this linearization is one-dimensional. A certain pressure, p_{ij}^{WELL}, determines the flow rate of all three phases, q_{ijp}^{WELL}, and it is therefore only one SOS2 weighting variable, γ_{ijk}, per breakpoint for each WPC.

Routing of wells These constraints ensure that the wells are closed or routed to one of the two pipelines.

$$x_{ij} + \sum_{l \in \mathcal{L}(i)} y_{ijl} = 1 \ , \ \forall \, j \in \mathcal{J}(i) \tag{3.8}$$

Mass balance The mass balance constraint, (3.9), forces the flow from a well to be equal to the sum of all flows routed to the different pipelines from that well (q_{ijlp}^{WRP}). Inequality (3.10) ensures that the flow from a well to a particular pipeline is zero if the corresponding routing variable is equal to zero. Further, (3.11) ensures that the flow of gas, oil or water in a pipeline has to be equal to the sum of flow from all wells routed to that pipeline.

$$q_{ijp}^{WELL} = \sum_{l \in \mathcal{L}(i)} q_{ijlp}^{WRP} \ , \ \forall \, j \in \mathcal{J}(i), \ p \in \mathcal{P} \tag{3.9}$$

$$q_{ijlp}^{WRP} \le Q_{ijp}^{MAX} \, y_{ijl} \ , \quad \forall \, j \in \mathcal{J}(i), \ l \in \mathcal{L}(i), \ p \in \mathcal{P} \tag{3.10}$$

$$q_{ilp}^{PIPE} = \sum_{j \in \mathcal{J}(i)} q_{ijlp}^{WRP} \ , \ \ \forall \, l \in \mathcal{L}(i), \ p \in \mathcal{P} \tag{3.11}$$

Manifold pressure requirements The following constraints enforce that the pressure in pipeline l at the manifold in cluster i has to be lower than the maximum pressure upstream from the choke in all wells connected to the manifold in cluster i routed to pipeline l. $P_i^{MAX} - P_{ij}^{MIN}$ is used as the smallest possible Big M.

$$p_{il}^{M} \le p_{ij}^{WELL} + (P_i^{MAX} - P_{ij}^{MIN}) \, (1 - y_{ijl}) \ , \ \forall \, j \in \mathcal{J}(i), \ l \in \mathcal{L}(i) \tag{3.12}$$

Linearization of pressure drop As described in Section 3.2.2 the pressure drop in the pipelines is modeled in three dimensions as a function of gas, oil and water with the help of (3.13) - (3.16).

$$q_{ilp}^{PIPE} = \sum_{n_g \in \mathcal{N}_{ig}} \sum_{n_o \in \mathcal{N}_{io}} \sum_{n_w \in \mathcal{N}_{iw}} Q_{ipn_p}^{DROP} \lambda_{iln_gn_on_w} \ , \ \forall \, l \in \mathcal{L}(i), \ p \in \mathcal{P} \tag{3.13}$$

$$p_{il}^{DROP} = \sum_{n_g \in \mathcal{N}_{ig}} \sum_{n_o \in \mathcal{N}_{io}} \sum_{n_w \in \mathcal{N}_{iw}} P_{in_gn_on_w}^{DROP} \lambda_{iln_gn_on_w} \ , \ \forall \, l \in \mathcal{L}(i) \tag{3.14}$$

$$\sum_{n_g \in \mathcal{N}_{ig}} \sum_{n_o \in \mathcal{N}_{io}} \sum_{n_w \in \mathcal{N}_{iw}} \lambda_{iln_gn_on_w} = 1 \ , \ \forall \, l \in \mathcal{L}(i) \tag{3.15}$$

$$\lambda_{iln_gn_on_w} \geq 0, \ \forall \, l \in \mathcal{L}(i), \ n_g \in \mathcal{N}_{ig}, \ n_o \in \mathcal{N}_{io} \ , \ n_w \in \mathcal{N}_{iw} \tag{3.16}$$

Equations (3.17) - (3.21) ensure the tridimensional SOS2 neighbor conditions explained earlier.

$$\eta_{ilgn_g} = \sum_{n_o \in \mathcal{N}_{io}} \sum_{n_w \in \mathcal{N}_{iw}} \lambda_{iln_gn_on_w} \ , \ \forall \, l \in \mathcal{L}(i), \ n_g \in \mathcal{N}_{ig} \tag{3.17}$$

$$\eta_{ilon_o} = \sum_{n_g \in \mathcal{N}_{ig}} \sum_{n_w \in \mathcal{N}_{iw}} \lambda_{iln_gn_on_w} \ , \ \forall \, l \in \mathcal{L}(i), \ n_o \in \mathcal{N}_{io} \tag{3.18}$$

$$\eta_{ilwn_w} = \sum_{n_g \in \mathcal{N}_{ig}} \sum_{n_o \in \mathcal{N}_{io}} \lambda_{iln_gn_on_w} \ , \ \forall \, l \in \mathcal{L}(i), \ n_w \in \mathcal{N}_{iw} \tag{3.19}$$

$$\eta_{ilpn_p} \geq 0, \ \forall \, l \in \mathcal{L}(i), \ p \in \mathcal{P} \ , \ n_p \in \mathcal{N}_{ip} \tag{3.20}$$

$$\eta_{ilpn_p} \text{ is SOS2}, \ \forall \, l \in \mathcal{L}(i) \ , \ p \in \mathcal{P}, \ n_p \in \mathcal{N}_{ip} \tag{3.21}$$

Pipeline pressure requirements The pressure drop in a pipeline from the manifold and up to the separator needs to equal the pressure difference. The pressure at the first stage separator is fixed at P^{SEP}.

$$p_{il}^{DROP} = p_{il}^{M} - P^{SEP} \ , \ \forall \, l \in \mathcal{L}(i) \tag{3.22}$$

Requirements on variables Binary requirements must be put on the variable x_{ij}, which defines whether a well is open or closed. The same is true for y_{ijl} which contains information on whether a well has been routed to a certain pipeline or not. Nonnegativity constraints apply to all other variables.

Removal of symmetry There is symmetry in the solution space as the parallel pipelines between the manifolds and up to the separator in each cluster are modelled as having exactly the same physical properties. If we know that a certain single well always will produce in a solution, this symmetry can be removed by fixing this well's routing variable so that it is always routed to the same pipeline.

3.4 Solution method

Introducing the piecewise linearization of the pipeline pressure-flow relations increases the problem size substantially with respect to variables and constraints, and a large proportion of the new variables will be complicating SOS2 weighting variables. However, the reformulation comes with the benefit that once the WPCs and the pipeline pressure drop has been linearized it is possible to combine branch & bound with linear solvers to solve the problem. A solution method that can handle integer variables, such as branch & bound, is necessary as a result of the binary routing variables and to ensure that the SOS2 conditions are satisfied.

Though DWD was developed in the early 1960's and optimization methods have been widely used in the process and petroleum industry since then, DWD has received little attention for solving optimization problems in this industry. As we mentioned earlier, notable recent exceptions are the works of Cheng et al. (2008), Alabi and Castro (2009), Gunnerud and Foss (2009) and Gunnerud et al. (2010).

The general idea behind utilizing DWD is to break down a large problem into smaller parts that together can be handled more easily than the original problem - basically a divide and conquer strategy. The first step towards implementing this strategy is selecting how to partition the original problem into one coordinating master problem and a number of subproblems. Choosing the downstream capacity constraints as the common constraints to relax and placing them in the master problem is a natural choice when implementing a DWD for the production allocation problem presented in the previous section. Colombani and Heipcke (2006) state that one should ideally have as few common constraints as possible. By placing the handling capacity constraints in the master problem we ensure that there are at most two common constraints, one representing gas capacity and a second representing water capacity.

Decomposing the problem in this way also makes it easier for planners to understand the model, since it becomes conceptually similar to the way production planners typically try to maximize the production at a field. Production planners try to find good solutions for the whole system by combining solutions for each well cluster. Similarly, we let each cluster equate to one subproblem and then combine the solutions for the purpose of creating one good solution for the entire system.

3.4.1 Redefinition of variables

To allow the separation of the problem into a master problem and a number of subproblems, a redefinition of the variables is necessary. If the solution space given by the constraints is convex and bounded, a solution can be described by a convex combination of the extreme points - termed the convexification approach (Vanderbeck, 2006). However, as the model presented in Section 3.3 is a MILP it does not have a convex solution space. An alternative to the convexification approach is to represent a solution using the integer points in the solution space - termed the discretization approach (Nemhauser and Wolsey, 1999), (Vanderbeck and Savelsbergh, 2006). The MILP presented in Section 3.3 only has binary integer variables, well routing and SOS2 variables, and accordingly no internal integer solutions in the solution space. For our problem there is no difference between applying the discretization approach and the convexification approach when representing a solution from a subproblem in the master problem, as the set of extreme points in a subproblem is equal to the set of feasible integer solutions (Vanderbeck and Savelsbergh, 2006).

3.4.2 Column generation

After the reformulation the columns in the master problem correspond to extreme points in the subproblems. However, the master problem is solved without tabulating all of the potential columns. Instead columns are generated during the course of the procedure and a restricted version of the master problem (RMP) is solved in each iteration, a method that is termed column generation (Lasdon and Tabak, 1981). In a basic column generation algorithm each of the n subproblems are re-solved each time the RMP is solved using information from the previous solution of the RMP. The objective function in the RMP and in each of the subproblems is dynamic. The information passed on from the RMP, the dual prices, change the objective function of the subproblems, while the new columns generated in the subproblems change the RMP. The changes in the subproblems' objective function entails changing the cost of "using" the resources, gas and water capacities, which are constrained by the common capacity constraints in the RMP. This is to obtain a more "correct" cost of using these resources, as the capacity constraints have been removed from the subproblems. Figure 3.4 is an illustration of the passing of information in the Dantzig-Wolfe algorithm. The black arrows indicate the sequence of events, while the gray dotted arrows indicate information flow. Note that each of the columns corresponds to a certain production setting for the cluster that the subproblem solution (column) corresponds to.

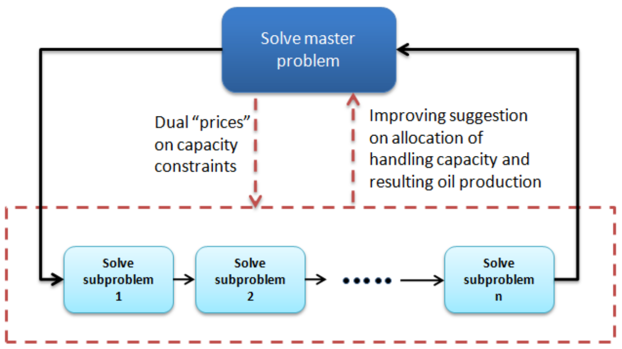

Solve master problem
Dual "prices" on capacity constraints
Improving suggestion on allocation of handling capacity and resulting oil production
Solve subproblem 1
Solve subproblem 2
Solve subproblem n

Figure 3.4: Illustration of a sequential Dantzig-Wolfe algorithm

3.4.3 Restricted master problem

Objective function

Using the convexification approach, the production level of gas, oil and water, represented by phase p in the optimal solution of a subproblem is stored in an array, Q_{icp}^{PROD}, which represents a candidate column c. If the subproblem solution has the potential to improve the master problem solution this array is sent to the master problem. There a non-negative weighting variable, μ_{ic}, that represents the candidate column which is created. $\mathcal{C}(i)$ in (3.23) is the set of candidate columns for cluster i. Having introduced these, the objective function for the RMP solved as an LP, Z_{RMP-LP}, have the following form:

$$\max\ Z_{RMP-LP} = \sum_{i\in\mathcal{I}} \sum_{c\in\mathcal{C}(i)} Q_{ico}^{PROD}\mu_{ic} \tag{3.23}$$

The oil production from a certain cluster is hence a convex combination of the candidate solutions, represented by the index c, which have been sent from the subproblems.

Common constraints

After the variable transformation the common constraints (gas and water handling capacities) will be of the form shown in (3.24). Here, C_p^{TOT} represents the maximum handling capacity of phase p at the platform.

$$\sum_{i\in\mathcal{I}}\sum_{c\in\mathcal{C}(i)} Q_{icp}^{PROD}\mu_{ic} \leq C_p^{TOT} \quad ,p=\{g,\ w\} \tag{3.24}$$

To ensure that the optimal solution of the RMP is a sum of convex combinations of subproblem solutions, the two constraints shown below are also needed. (3.25) and (3.26) are termed the convexity constraints.

$$\sum_{c\in\mathcal{C}(i)} \mu_{ic} = 1 \quad ,\forall\, i \in \mathcal{I} \tag{3.25}$$

$$\mu_{ic} \geq 0 \quad ,\forall\, i \in \mathcal{I},\ c \in \mathcal{C}(i) \tag{3.26}$$

3.4.4 Subproblem

The dual prices for the gas handling capacity constraint, π_g^{CAP}, the water handling capacity constraint (3.25), π_w^{CAP}, and the convexity constraint, π_i^{CONVEX}, are sent to the subproblems and used in the subproblem objective function. Taking these into account, the subproblem objective function can be written:

$$\max\ Z_i = \sum_{l\in\mathcal{L}(i)} (q_{ilo}^{PIPE} - q_{ilg}^{PIPE}\pi_g^{CAP} - q_{ilw}^{PIPE}\pi_w^{CAP}) - \pi_i^{CONVEX}, \ \forall\, i \in \mathcal{I} \tag{3.27}$$

A subproblem solution is only sent to the RMP as a candidate column if its reduced cost in the master problem is positive as it only then has the potential to improve the current master problem solution. The reduced cost is therefore used as the objective function of the subproblem. The first term on the right hand side represents the maximum increase in the RMP objective function by introducing a certain column (subproblem solution) to the master problem basis. The last three terms represent the indirect cost of introducing this column to the RMP. That is, the effects of changing the current optimum master problem basis to accommodate the new basic variable.

The constraints in the subproblems are all the constraints in the original MILP presented in Section 3.3 with the exception of the handling capacity of gas and water.

Obtaining an integer solution and terminating the algorithm

As mentioned the solution space of the MILP presented in Section 3.3 is not convex. Constraint (3.26) will therefore have to enforce binary to guarantee that the master problem provides solutions that are feasible in the original formulation. Two columns, each representing a certain production for a cluster, may have different routings making a convex combination of them an infeasible solution. However, the RMP has to be solved as a LP to obtain the dual prices which drive the DWD algorithm. To find a solution that satisfies the integer requirements, we use a method suggested in Gunnerud and Foss (2009). That is, to solve the RMP with binary requirements on the weighting variables after it has been solved as an LP. We term this heuristic the *DWD-IP heuristic*.

This heuristic's potential of producing a good integer solution is very dependent on the columns present in the RMP when the DWD-algorithm converges to the optimal LP-RMP solution. Adding extra columns for each subproblem in each iteration may make it possible to find an improved integer solution when combined with the DWD-IP heuristic. Only columns that have a positive reduced cost can potentially improve the LP-RMP solution. However, all integer solutions that are encountered when solving the subproblems as MILPs can potentially improve the heuristics ability to provide a good integer solution. After adding these extra columns, the RMP will still be very small and computationally easy to solve compared to the subproblems. Adding several columns for each subproblem in each iteration was used for pure LP problems with the aim of speeding up the solution time in (Barnhart et al., 1998) and (Tebboth, 2001), and can potentially also reduce the solution time for our MILP.

By calculating an upper and lower bound in each iteration the algorithm can be terminated if an acceptable gap has been obtained. As an upper bound, we use the LP objective value of the RMP plus the sum of the subproblems' objective values (Karlof, 2006). The solution provided by the DWD-IP heuristic serve as a lower bound.

3.4.5 Overview of the Dantzig-Wolfe algorithm

To sum up there are three main steps in the DWD algorithm.

1. Heuristic step: Generation of initial columns to solve the RMP for the first time.

2. DWD optimization: Repeat the Dantzig-Wolfe cycle until no subproblem produces a solution with a positive reduced cost.

3. Heuristic step: Solve the RMP with binary requirements on weighting variables to obtain a feasible solution.

As the greatest potential for a lower solution time lies within the 2nd step we have focused on the DWD cycle when implementing a parallel algorithm for the production allocation problem. Different ways of generating initial columns were covered in Tebboth (2001).

3.5 Parallel Dantzig-Wolfe

3.5.1 Basic parallel Dantzig-Wolfe

The block angular structure of the MILP presented in Section 3.3 combined with the DWD formulation we presented Section 3.4 makes a coarse-grained parallel algorithm the most natural first approach in a parallel implementation. By solving each subproblem on a separate CPU a maximum speedup of n, where n is the number of clusters/subproblems, can theoretically be obtained if the solution time of the master problem can be ignored. In our first algorithm, which we term *Basic parallel*, the mapping of tasks to processors is static and we use the barrier synchronization criteria that all subproblems have to be solved to optimality before the master problem is solved and new dual prices are obtained. The potential disadvantage with this algorithm is that if one or more subproblems have solution times that are very different from the other subproblems much of the possible speedup is lost in CPU idling as the RMP has to wait for these bottleneck problems to finish. In addition, with this method the maximum speedup is limited to n, even if the number of available processors is larger than n.

3.5.2 Accelerated feedback Dantzig-Wolfe

The DWD of the production allocation problem has some special characteristics. The RMP has very few constraints and variables compared to each of the subproblems and therefore it has a solution time that is very short compared to the solution time of the subproblems. This makes it possible to solve the RMP several times, each time all subproblems are solved, with a negligible effect on the total solution time. As mentioned, another characteristic is that the subproblems have very different solution times, thus increasing CPU idle time. Together these characteristics make it interesting to investigate more advanced parallelization strategies which better utilize the CPUs, solve the RMP more frequently and decrease scaling issues. As we have mentioned, Gnanendran and Ho (1993) and Tebboth (2001) investigate certain strategies which change the order in which the subproblems and the RMP are solved on each cycle and term a promising approach for large block angular problems the *Accelerated feedback strategy*. In this strategy candidate columns are sent to the RMP when a subproblem terminates. The RMP is then immediately solved and dual values sent to all subproblem that are not currently being solved when it terminates. The reasoning behind this is that updated dual values will be available to the subproblems at an earlier stage and thereby make the problem converge faster. Tebboth (2001) concludes that this is a promising approach for certain block angular structured LP problems. We have developed the Accelerated feedback strategy further to match the MILP production optimization problem and test whether this strategy can improve the performance of the basic parallel algorithm. Reviewing this strategy, we have developed the Accelerated feedback algorithm for two different scenarios:

- The number of CPUs is **fewer** than or equal to the number of subproblems
- The number of CPUs is **greater** than or equal to the number of subproblems

As we will return to, the Accelerated feedback algorithm for each scenario will produce the same result if the number of CPUs is equal to the number of subproblems.

Fewer CPUs than subproblems

When there are fewer parallel processors than the number of subproblems, a queue for the currently unprocessed subproblems created together with rules for which subproblem to next take from the queue and solve. Assuming the time to solve two subproblems on one CPU is the same if they are started concurrently or sequentially, the subproblems are solved with less updated dual values, thereby slowing down the overall convergence if a queue is not implemented. If a queue is in place the master problem can be solved in between the solving of two subproblems and updated dual values sent to the second subproblem before it is started.

Figure 3.5 shows the first iteration of the Accelerated feedback strategy where we have applied four parallel processors on a problem with eight subproblems. As in a basic parallel DWD strategy, the RMP is solved with initial columns and dual values distributed to all subproblems. When the first subproblem terminates, this method resolves the RMP. The updated dual values are distributed to the subproblems not currently being solved and the next available subproblem is started on the CPU which just finished solving the RMP. This continues until no subproblem solution has a positive reduced cost. The algorithm will then be in a state where some of the subproblems have been solved in previous master iterations and with outdated dual values. To guarantee LP optimality of the DWD algorithm these subproblems have to be resolved with the most recent dual values. If one of these subproblems then returns a positive reduced cost, the RMP must be resolved along with all the other subproblems.

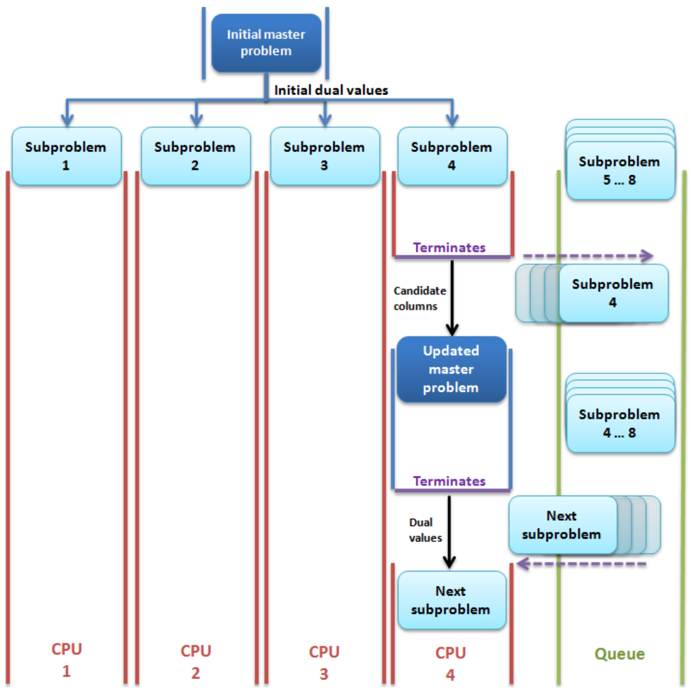


Figure 3.5: First iteration of the Accelerated feedback strategy with fewer CPUs than subproblems

When there are fewer available parallel processors than subproblems, a possible rule to choose which subproblem to be solved next, is to pick the subproblem in the queue that produced the solution with the highest reduced cost in the previous iteration, given that all subproblems have been solved once. As the reduced cost of a subproblem gives the potential increase in the RMP objective function, we expect the subproblem with the highest value in the previous iteration to also produce a good solution in the next iteration. Another approach could be to choose the subproblem that has been solved the least number of times, based on the assumption that each subproblem would require about the same number of iterations.

More CPUs than subproblems

If one has access to more CPUs than subproblems the same subproblem can be solved on multiple CPUs. The extra CPUs are then used to solve subproblems that are already being solved, with updated dual values obtained from resolving the master problem when other subproblems terminate. The advantage with this algorithm is that it offers the potential of scalability for the Accelerated feedback algorithm. An example of an Accelerated feedback algorithm which tries to utilize the extra CPUs is illustrated in Figure 3.6. In this example there are six CPUs available and three subproblems. The subproblems first solved on CPU 1, 2 and 3 are based on a different set of dual values than the subproblems first solved on CPU 4, 5 and 6.

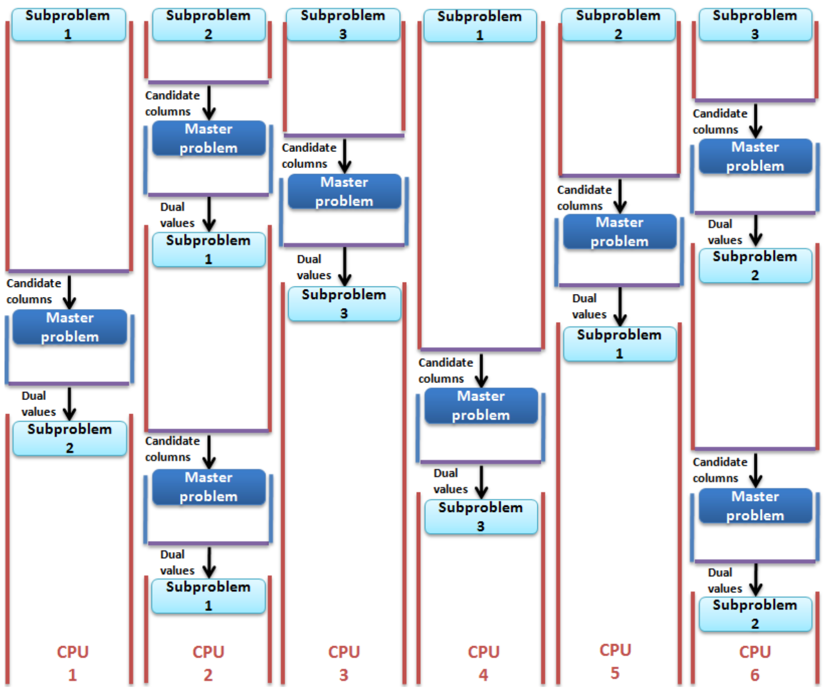


Figure 3.6: First iterations of Accelerated feedback with more CPUs than subproblems

To decide which subproblem to solve next on a CPU, a way of ranking the different subproblems is necessary. In the example below we have used a simple rule where the subproblems are started in the sequence 1,2,3,1,2,3... as CPUs become available. All subproblems have to be solved until they return with a reduced cost of zero to prove optimality. If we assume that the number of iterations until a subproblem returns with an objective value (reduced cost) of zero is about the same for all subproblems, a good rule is to choose the subproblem that has been solved the fewest times. However, this is not necessarily true. Also, the disadvantage with this rule is that it does not take into account the value of the subproblem objective function. To account for this we could choose the subproblem that in its previous iteration produced the solution with the highest reduced cost. To avoid starting the same subproblem on several CPUs with minimal change of the dual values we could disallow values close to values already used on one of the CPUs.

3.5.3 Implementation of parallel algorithms

Petroleum production allocation problems have a limited number of subproblems defined by the number of production clusters. It is therefore harder to efficiently utilize a supercomputer with CPUs in the hundreds or thousands. Given our focus on applications, the implementation was done in Mosel using the Xpress-Optimizer as it allows good utilization of single multi-core PCs through the module `mmjobs`, which supports running multiple models concurrently (Colombani and Heipcke, 2006), (Dash Optimization, 2007).

The parallel algorithm outlined in Section 3.5.2 is based on the algorithm in detail controlling the assigning of CPUs to different subproblems. When using Xpress and the module `mmjobs` threads are distributed automatically over available CPUs. Our initial results showed that this module solves multiple threads on a single CPU more efficiently than when the sequence of threads is user defined if there are fewer CPUs than subproblems. Our parallel implementation therefore initially starts solving all subproblems and there is thus no queue of unsolved subproblems.

When a subproblem terminates with a positive reduced cost, the RMP is immediately solved. If the number of CPUs is greater than the number of subproblems, the subproblem that just terminated is then restarted. If the number of CPUs is less than the number of subproblems, the next subproblem in an ordered sequence is instead started with new dual values obtained from the RMP. If the number of CPUs is greater than the number of subproblems the total number of subproblems that concurrently are being solved is equal to the numbers of CPUs, as described in the previous section.

The algorithm stores the master problem iteration number in which the subproblem was started. If a subproblem terminates with a non-positive reduced cost, the subproblem is recorded in a list and not restarted. The algorithm then iterates through all subproblems on the list and checks if they have been solved with dual values from an iteration prior to the current one. If this is the case for a subproblem, it has been solved with an old set of dual values from the RMP and could possibly produce a positive reduced cost if resolved with the most recent set of RMP dual values. The subproblem is therefore started in a new thread. This procedure is continued until no subproblems return a positive reduced cost and hence all have been solved with the most updated set of dual values.

3.6 Troll case

Is a parallel implementation of a Dantzig-Wolfe algorithm a useful practical tool for solving production allocation problems with a block angular structure? To answer this question the algorithms presented in Section 3.5 were tested on realistic production data from the Statoil operated offshore oil and gas field Troll West, illustrated in Figure 3.7. A computational analysis was performed on semi-realistic data from the Troll West field in Gunnerud and Foss (2009) and on realistic data in Gunnerud et al. (2010). In this paper the problems we solve have two, as opposed to one common constraint as in earlier publications. Although gas handling capacity is currently the most important constraint on the production at Troll West, the water handling capacity is likely to also be an active constraint in the future as the field matures. The problem we solve will thus bear a closer resemblances to future production challenges at the field, though it will also significantly increase the difficulty when employing a DWD algorithm.

Certain characteristics of the Troll field make it very challenging to solve as an optimization problem for the production planning. The oil is located in a thin layer beneath a thick gas cap - a feature that causes the wells to have a highly rate dependent GOR (gas/oil ratio). This, combined with the fact that up to four wells are connected to the same manifold in each cluster, makes well interaction and nonlinear behavior significant factors that need to be accounted for when planning the production.

Our work to create and solve a mathematical model for petroleum production allocation problems was initiated in response to the absence of integrated software that optimizes allocation of gas to all wells at Troll west. Currently, production engineers typically allocate gas to each cluster based on experience and sensitivity analysis - a practice that is believed to be sub-optimal. In the sensitivity analysis, each cluster is optimized at a set gas level through a combination of in-house software and commercial optimization software.

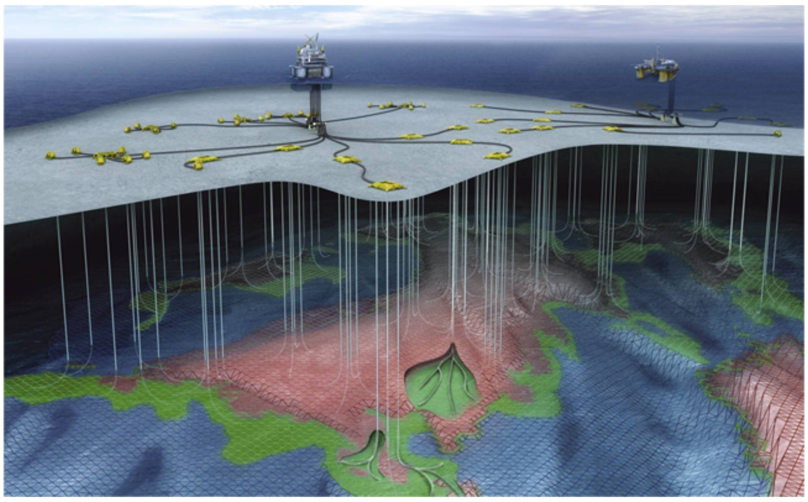


Figure 3.7: The Troll B and C platforms in the Troll west region (http://www.statoil.com accessed 14.11.2010)

3.6.1 Testing objectives

Expanding on the question posed at the beginning of this section, the objectives of the computational study become: to study the speed up of a parallel vs. a sequential DWD implementation on a realistic data set; to compare DWD with a standard solution method; and to investigate the scalability of the Accelerated feedback algorithm.

To isolate the effects of the different elements introduced in our algorithms a number of different solution methods were tested. Their names and short explanations are given below.

Standard	The standard solution strategy solves the original undecomposed problem and uses a branch & bound algorithm to solve the full MILP.
Sequential	The sequential solution strategy is the sequential DWD strategy described in Section 3.4, combined with the DWD-IP heuristic used to produce a feasible solution to the RMP as described in Section 3.4.4.

Basic parallel	The basic parallel solution strategy is described in Section 3.5.1 and solves the subproblems in parallel as opposed to sequentially. It also uses the DWD-IP heuristic to produce an integer solution.
Accelerated feedback	The accelerated feedback strategy is a parallel DWD solution strategy, as described in Section 3.5.2, that tries to decrease the idle time for a computer with multiple CPUs. As for the other DWD strategies, an integer solution is produced by the DWD-IP heuristic.
Extra columns	The extra columns solution strategy is the same solution method as the basic parallel strategy, but where all the integer columns encountered when solving the subproblems are added to the RMP.

3.6.2 Test data

The input data required to solve the model presented earlier mainly describe the system topology such as the number of clusters, manifolds per cluster, pipelines per manifold and wells per manifold. In addition, they describe the well performance curves (WPC) and the pipeline pressure drop as explained in Section 3.2.2.

Generating realistic pipeline and well data

A typical workflow generates pipeline and well data using a reservoir simulator, such as Statoil's Gas Oil Rate Model (GORM) (Mjaavatten et al., 2006), and a multiphase well network optimizer, for instance the General Allocation Package (GAP) developed by Petroleum Experts (Petroleum Experts, 2010). For our purpose, realistic production data is generated using GAP and GORM. WPC data is generated by entering all possible values of wellhead pressure in GAP and thereby simulate through GORM the different flows of gas, oil and water from the reservoir and into the well at different pressures.

Generation of data to represent the pressure-flow relations in the pipelines is performed in a different way. Figure 3.8 shows a screen shot of the GAP user interface and illustrates how pipeline pressure drop is calculated using GAP. Here, all wells except one (indicated by the circle around the triangle) on manifold 2 are closed. For this well, GAP simulates through GORM all combinations of flow rates for the three phases routed from this well through manifold 2, across the pipeline to manifold 1 and up to the first stage separator shown in the upper right corner in the figure. The pressure in each manifold is extracted and the pressure drop in the pipelines from manifold 2 to 1 and from manifold 1 to the separator is calculated.

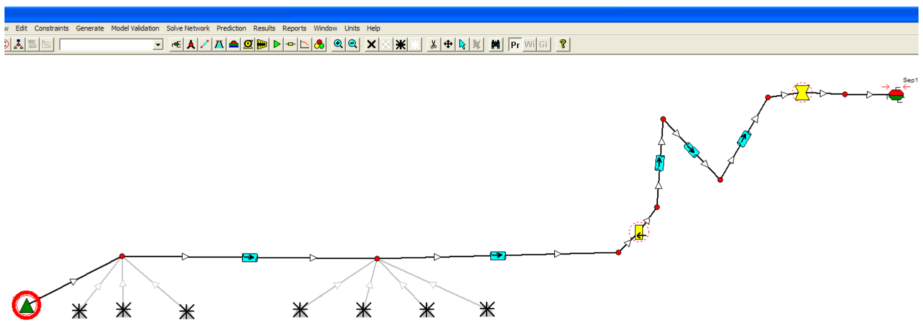


Figure 3.8: Screenshot from GAP showing the topology in a single cluster

Common fixed parameters

We refer to the terms *test case* and *test batch* frequently in the next section. A test batch is a single problem with a specific set of input parameters describing the production system and its capacities. A test batch consists of multiple test cases where each test case represents a run of a single solution method on a certain type of hardware. Certain fixed test batch parameters are common for all test batches and concern the topology of the production system. These parameters and their values are listed below and in Table 3.1.

	Number of:
Clusters	8
Manifolds in each cluster	2
Pipelines at each manifold	2
Wells at each manifold	4
Pipeline pressure drop interpolation points	7

Table 3.1: Common parameters for all test batches

Test parameters

The water and gas handling capacities vary over time at Troll West as they depend on how much is produced at the nearby Fram field which also has pipelines connected to the platforms at Troll West. To reflect this, each solution method is tested in five test batches that have different limits on gas and water.

3.6.3 Hardware and Software Specifications

The following hardware was used during testing:

- 2 CPU: Intel Core2Duo E6700 2,6 GHz, 4GB RAM, Windows XP SP3
- 4 CPU: HP dl140 G3, 2x Intel Core2Duo 5110 1,6 GHz, 8 GB RAM, Rocks Linux v5.3
- 8 CPU: HP dl160 G5, 2x Intel QuadCore E5472 3,0 GHz, 16 GB RAM, Rocks Linux v5.3
- 12 CPU: HP dl165 G6, 2x AMD Opteron 2431 2,4 GHz, 24 GB RAM, Rocks Linux v5.3

Xpress-IVE version 1.19.00 and Xpress Optimizer version 19.00.00 is the optimization software we used, while Petroleum Experts - GAP Multiphase System Optimization version 7.4 and Microsoft Excel 2007 was used for data generation.

3.7 Results

3.7.1 Validation

Semi-realistic production data and results from Gunnerud and Foss (2009) were used to validate the correctness of the implementation of the different solution methods. The solution times for the test batches created from this data were lower than when using realistic production data and they were therefore also more suitable for an initial validation. The results from the validation tests proved the implementations to be able to solve the problem we investigate correctly. As opposed to the results in Gunnerud and Foss (2009) both the gas and water capacities are active in the solutions shown in Table 3.3.

3.7.2 Standard solution method

The standard algorithm was unable to solve any of the test batches presented in 3.6.2 based on realistic production data and the Xpress-Optimizer did not find any integer solutions after 12 hours of solving. To compare its solution time and value with the DWD algorithms we created a new set of test batches that were less computationally demanding to solve. After experimenting we found that both a smaller problem with two clusters and a lower resolution in the piecewise linearization of the pipeline pressure drop was necessary for the standard solution method to be able to provide a solution. The standard solution method solved these test batches in 3500-6000 seconds depending on how constrained the handling capacities were.

The dimensions of the reduced 2-cluster-problem are compared with the dimensions of the full 8-cluster-problem in Table 3.2.

Number of clusters	**2**	**8**
Continuous variables	3880	11317
Binary variables	64	256
SOS2 variables	983	3932
Constraints	424	1720

Table 3.2: Problem characteristics

The results thereby show that the Standard solution method is unable to solve the undecomposed problem using branch & bound for a production allocation problem with a satisfactory resolution in the piecewise linearization. This necessitates and supports the exploration of other solution strategies.

3.7.3 Dantzig-Wolfe general

The Sequential Dantzig-Wolfe algorithm solved the two smaller cluster test batches in 118 - 220 seconds depending on how constrained the water and gas limits were. It terminated with a gap of 3.5 - 4.0% between the LP-RMP and the DWD-IP heuristic solution - a gap that was observed to be very dependent on the number of candidate columns that had been produced at the time the LP-RMP terminated. The sequential DWD method also solved each of the full field size 8 cluster test batches based on realistic production data. The solution times on the different types of hardware are shown in Table 3.3.

The computational burden of solving each of the subproblems with DWD proved to be significantly less than when using the standard solution method. We also observed that the Dantzig-Wolfe algorithms quickly obtained good dual values and that the number of iterations needed therefore was limited. Combined, these two features enable a much lower solution time when using sequential DWD compared to the standard solution method. This again made it possible for us to use DWD to solve the realistic production test batches - test batches which we were unable to provide a single integer solution for by using the standard algorithm.

3.7.4 Comparing Dantzig-Wolfe algorithms

Table 3.3 shows the solution time in seconds for the sequential Dantzig-Wolfe algorithm on the 2, 4, 8 and 12 CPU platforms together with the solution time of the two parallel strategies as a percentage of the sequential solution time for the realistic data cases presented in Section 3.6. For reference, the solution time as a percentage of sequential, if perfect load balancing was achieved, is shown in the last row. The solution time in seconds between the different hardware platforms is not directly comparable as the CPUs are different. The last column shows the gap between the best solution obtained and the optimal LP solution. The extra columns solution method is not shown as this method was only tested on the 8 CPU computer.

		Number of CPUs				
Strategy	**Test batch**	**2 CPUs**	**4 CPUs**	**8 CPUs**	**12 CPUs**	**Optimality gap**
	1	570.3	1014.0	535.0	729.8	0.94 %
	2	647.4	1028.2	542.2	734.7	0.74 %
Sequential	3	508.9	812.8	427.8	584.6	1.50 %
(seconds)	4	1102.8	1780.6	938.7	1272.7	0.40 %
	5	754.2	1212.9	638.9	875.6	0.54 %
	Average	**716.7**	**1169.7**	**616.5**	**839.5**	**0.83 %**
	1	64.5 %	52.6 %	47.5 %	44.7 %	0.94 %
	2	59.5 %	41.9 %	36.6 %	44.2 %	0.74 %
Basic	3	61.1 %	44.6 %	39.4 %	42.0 %	1.50 %
(% of sequential)	4	56.6 %	37.5 %	31.2 %	31.6 %	0.40 %
	5	59.6 %	41.5 %	36.4 %	38.7 %	0.54 %
	Average	**60.3 %**	**43.6 %**	**38.2 %**	**39.0 %**	**0.83 %**
	1	66.0 %	29.3 %	29.7 %	24.5 %	0.57 %
	2	64.8 %	42.6 %	35.5 %	22.5 %	0.57 %
Accelerated	3	72.7 %	45.3 %	39.0 %	34.2 %	1.12 %
feedback	4	53.7 %	33.9 %	22.4 %	16.0 %	0.23 %
(% of sequential)	5	62.9 %	36.4 %	24.2 %	23.0 %	0.48 %
	Average	**64.0 %**	**37.5 %**	**30.2 %**	**22.6 %**	**0.59 %**
Perfect load balancing (% of sequential)		**50.0 %**	**25.0 %**	**12.5 %**	**8.3 %**	

Table 3.3: Solution time for basic and accelerated feedback as a proportion of solution time for sequential for each type of hardware

For all types of hardware, the results show that basic has a substantially lower solution time compared to sequential. Compared with basic, accelerated feedback has a higher solution time with 2 CPUs but a reduced solution time with 4, 8 and 12 CPUs. Neither of the parallel strategies are able to obtain an average solution time proportion closer than 10% to perfect load balancing. The average solution time proportions of the different strategies are also plotted in Figure 3.9.

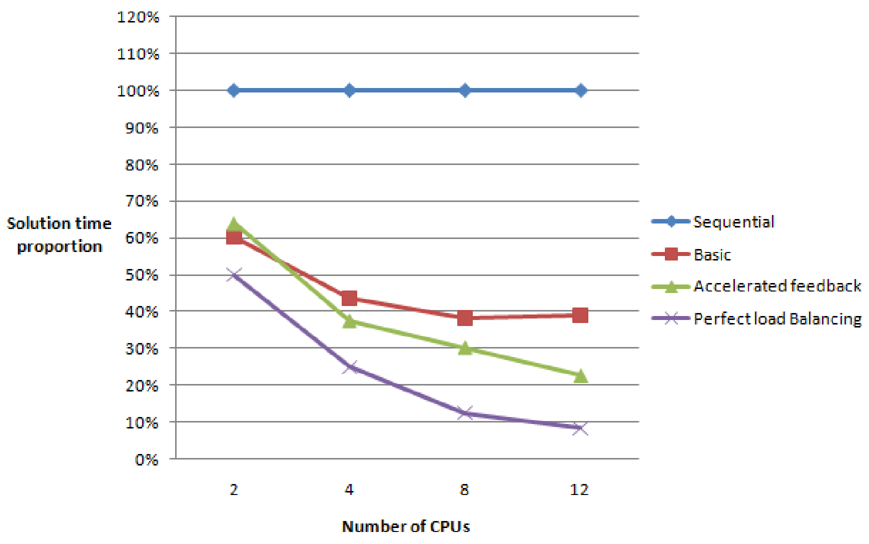


Figure 3.9: Solution time for the different parallel Dantzig-Wolfe strategies as a proportion of the solution time with the sequential Dantzig-Wolfe algorithm

Table 3.4 summarizes the results when comparing the performance of the extra columns method to the basic DWD-algorithm. The extra columns method comes with the benefit of a significant reduction in the optimality gap, from an average over the test batches of 0.825% for basic, to 0.233% for extra columns. If we were satisfied with a gap of less than 1%, the extra columns method and the DWD-IP heuristic would be sufficient to provide a solution with the required quality for the test batches in which extra columns was tested. The solution time seems not to be significantly affected by adding extra columns. This is also consistent with what we observed when validating the extra columns method on semi-realistic production data.

Test batch	**Extra columns (% of Basic)**	**Basic gap**	**Extra columns gap**
1	123%	0.943 %	0.160%
2	78%	0.744 %	0.184%
3	167%	1.500 %	0.571%
4	73%	0.400 %	0.084%
5	91%	0.540 %	0.164%
Average	**107%**	**0.825%**	**0.233%**

Table 3.4: Solution time of extra columns as a proportion of solution time of Basic is the second column in the table. Optimality gap for both methods in each test batch are the third and fourth columns

3.8 Discussion

As mentioned in the introduction, a solution to the petroleum production allocation problem should be provided within hours, preferably within minutes. The long solution time of the standard method for the low resolution test batches clearly does not satisfy this requirement. This method is therefore not considered a possible solution approach. Our DWD algorithms, on the other hand, provide near optimal solutions for the same test batches with much lower solution times. Compared to the sequential DWD implementation, the basic parallel implementation reduces solution time substantially as multiple CPUs are utilized to solve the subproblems concurrently. However, this algorithm cannot decrease the solution time further when the number of CPUs is greater than the number of subproblems. The accelerated feedback strategy is able to increase the efficiency of the CPUs and reduce the CPU idle time and therefore provides the shortest solution time when using hardware with multiple CPUs. This is very promising as hardware with multicore processors is becoming more widely available and petroleum production planners will often have access to such hardware. Our accelerated feedback algorithm is able to provide a solution in about 200 seconds for low resolution problems whereas the standard method is unable to provide a single integer solution within 12 hours. With respect to the optimality gap, we observed that adding all integer feasible solutions that were discovered while solving a subproblem, as columns to the RMP, reduces the expected gap between the LP-RMP and the DWD-IP without significantly increasing the computational load. This algorithm is thus an important step towards being able to solve problems with a realistic accuracy.

Using the basic algorithm we are unable to obtain the same speedup as perfect load balancing. In each iteration of basic DWD, all subproblems have to wait during the serial execution of the RMP and the following distribution of new dual values. Since the solution time of our RMP is close to zero its contribution to CPU idle time is negligible. However, relative differences in subproblem solution time impacts time spent in each iteration since the "bottleneck" subproblem has to finish before the RMP can be solved. These subproblem solution times are very different for our problem, thereby contributing significantly to a higher CPU idle time and a slower solution time.

As shown in Table 3.3, the solution time for test batch 3 on 8 CPUs with Basic is 36.6% of the sequential solution time while for test batch 1 it is only 47.5%. Further investigation showed that for test batch 1 the CPUs were on average idle 69% of the time while for test batch 3 the CPU idle time fraction was 48%. Thus, the subproblem solution times differed more for test batch 1 than test batch 3, reducing the possible gain by utilizing parallelization. The calculations also showed that without the CPUs being idle, the solution time of test batch 1 would have been 14.8% of sequential and 19.0% for test batch 3. This is much closer to the Perfect load balancing limit and illustrates the importance of differences in subproblem solution times. The remaining distance to the Perfect load balancing limit is likely caused by overhead from file writing, memory access and loading/unloading of subproblems.

We observed a decrease in efficiency as the number of CPUs increases. This can also be explained by differences in subproblem solution times. For an 8 CPU computer, seven CPUs are idle while waiting for the last CPU to solve the slowest of the eight subproblems in each iteration. A 2 CPU computer on the other hand utilizes all available CPUs fully up until the point where the two final subproblems are being solved and one of them finishes. Even though the solution time on the 8 CPU platform is lower, the proportion of idle time compared to total solution time is higher and hence the efficiency lower when the number of CPUs increases. However, compared to the standard parallel algorithm the Accelerated feedback algorithm, significantly increases the efficiency of each CPU and thereby also the scalability of the algorithm.

The results in Table 3.3 show increased solution time with 2 CPUs but reduced solution time with 4, 8 and 12 CPUs when using Accelerated feedback compared to basic. As mentioned, Accelerated feedback tries to achieve a reduction in CPU idle time by immediately restarting the subproblems that terminate with a positive reduced cost on the same CPU but with updated dual values from the RMP. Test batch 1 has a substantially lower solution time when using Accelerated feedback compared to basic. This is due to a reduced CPU idle time and a faster convergence of the RMP. Considering the high percentage of CPU idle time using basic it is clear that Accelerated feedback has the potential to reduce the total solution time substantially. However, there are two factors that makes the potential smaller and the main reasons why Accelerated feedback is slower than basic with 2 CPUs.

Since the RMP is resolved every time a subproblem terminates, the reduced cost calculation for columns suggested by the subproblems is based on different dual values. This means that at the first point in time when all subproblems have produced solutions with non-positive reduced cost, those which have not been solved with the most recent dual values have to be resolved. This guarantees optimality of the DWD algorithm. The results show that this final phase contributes towards a smaller reduction in the solution time when going from the basic to the Accelerated feedback implementation. With problems containing more subproblems and a larger number of CPUs the potential of Accelerated feedback is greater as the idle time increases when the number of CPUs increase.

When using Accelerated feedback the master problem is solved and subproblems started without full information, that is, without waiting for all subproblems to finish solving with the currently most updated dual values. This makes each iteration less efficient at bringing us closer to a solution and is therefore also a barrier to reaching the load balancing limit.

3.9 Conclusion

In this article we have shown that petroleum production allocation problems are well suited for parallelization. We have discussed how a basic parallel Dantzig-Wolfe algorithm can be designed for such problems and what its weaknesses are. Furthermore, we have presented more advanced parallel algorithms that reduce the optimality gap, improve efficiency, further reduce solution time and make it possible to scale the algorithm when adding more CPUs. Utilizing a simple parallel Dantzig-Wolfe algorithm on a 12 processor PC we were able to cut the solution time by 61% on average for problems based on realistic production data that resemble future production challenges. By utilizing a more advanced strategy we were able to cut the solution time by 77% on average. Through our computational evaluation we identified that the solution time of the master problem and the difference in solution time of the subproblems are critical factors when evaluating the potential of a parallel algorithm. The master problem has to be solved repeatedly on a single CPU, significantly slowing down the algorithm if the master problem is computationally demanding. Unequal solution times for the subproblems make it harder to obtain an efficient utilization of the available parallel resources, especially if a naive parallel algorithm is implemented, as some subproblems then have to wait for others to finish.

Chapter 4

Handling integer properties with branch & price

This chapter is based on Gunnerud et al. 2010, "Oil production optimization Solved by Piecewise linearization in a Branch & Price framework", as submitted to Computers and Operations Research Journal.

Abstract

This chapter presents a method for optimizing oil production on large scale production networks such as the Troll west field in the North Sea. The method is based on piecewise linearization of all nonlinearities, and on decomposition of the full scale problem into smaller subproblems. Column generation in a branch & price framework is used to solve the decomposed problem. The method differs from most branch & price methods by branching only on continuous quantities and by solving the subproblems using commercial MILP software.

The method is applied to a realistic model of an oil field, the Troll oil and gas field at the Norwegian continental shelf, a petroleum asset with severe production optimization challenges due to rate dependent gas-coning wells. This study shows that the method is capable of solving instances of practical size to proven optimality.

keywords: Oil production planning, oil rim reservoir, mixed integer linear programming, column generation, branch & price.

4.1 Introduction

Development of a field asset requires planning on multiple horizons. On a long-term horizon, typically from one year and up to the field's lifetime, strategic reservoir planning is based on market conditions, field properties and the strategic considerations of the developing company. Decisions related to technology on this horizon include; how to develop the subsea solution, whether to process the fluid onshore or offshore, and how to export the different products produced. The analyses and subsequent development plan seek to maximize the expected net present value of an asset by for instance maximizing oil and gas recovery. The article Nygreen et al. (1998) discusses these issues.

On a medium time horizon, often referred to as tactical reservoir management, the planner seeks to extract as much oil and gas from the reservoir as possible, within the limits set by strategic decisions. For example in Troll oil, the north west part of the Troll field the extraction of gas is limited to ensure higher pressure in the reservoir for easier extraction of oil.

Operational production planning is considered to be on a short time horizon, typically days and weeks. Production optimization involving both the sub-surface part (reservoir and wells) and the surface part (collection and downstream production equipment) of the system, is important on this time scale. These plans list the target production for each well and are made so as to be compatible with the processing facilities. Following the usage in Foss et al. (2009), when this production plan is produced by a optimization model we refer to the problem as a real-time production optimization (RTPO) problem. Production may be constrained by reservoir conditions such as coning effects and/or the production equipment like pipeline capacity or downstream water handling capacity. Decision variables in RTPO include production and possibly injection rates, artificial lift inputs like lift gas rates and electric-submersible pump (ESP) rates, and routing of well streams.

This chapter focus on the RTPO problem and models the complete gathering network with the inlet separator as the downstream boundary. The goal is to maximize the oil production rate subject to the above mentioned constraints on the gas and water rates. In the Troll field the decisions on gas lift and ESP operation are made independently of this optimization and so are not considered in this chapter.

The problem could be modelled as a mixed integer nonlinear program (MINLP). However the nonlinear equations which are needed to accurately describe the flows in pipes and wells are very complex, and including these in the model along with discrete variables for well connections would produce a model for which it would be unrealistic to expect to be able to guarantee global optimality. To overcome this problem we approximate the well and pipe behaviour by a piecewise linearization using special ordered sets of type 2 (SOS2), which will yield a MILP problem. This modular approach allows us to combine methods from any pipe and well simulators with our MILP optimization model.

Unfortunately, in many practical situations the resulting MILP is still too difficult to solve, so in this chapter we develop a branch & price (B&P) decomposition method which can always find the global optimum. The gathering network is decomposed into clusters of wells and for each cluster the resulting subproblem is to find the optimal production of gas, oil and water for given prices on gas and water processing. These are MILPs which can be solved by a standard branch & bound method. The resulting optimal gas, oil and water outputs from the subproblems are incorporated as possible modes of operation into a master problem, which is solved by a specialised branch & bound method. Dual prices from the master problem for the gas and water capacity constraint give the prices used in the subproblems.

The remainder of this chapter is organized as follows: Section 4.2 describes the real problem, Section 4.3 gives the MILP model, Section 4.4 describes the branch & price method, Section 4.5 describes the details from the implementation, Section 4.6 gives computational results and analysis, and Section 4.7 states the conclusions.

4.2 The real-time production optimization problem

4.2.1 Methods and technology

Systems performing RTPO are in use today in upstream petroleum production. There are several commercial products which attempt to solve the RTPO problem, for example REO from Edinburgh Petroleum Services (2010), GAP from Petroleum Experts (2010), and MaxPro from FMC Technologies (2010). These systems have realistic models of wells and pipes. They solve the optimization problem by a combination of linear and nonlinear techniques, but cannot guarantee to find the global optima.

Wang (2003) provides a comprehensive overview of models and solution algorithms for different problems in the industry. He considers both linear and nonlinear formulations, with related techniques for solving them. A survey of the most common concepts and components of an oil production problem are presented in Bieker et al. (2006a), and other work on the subject is presented in Saputelli et al. (2003). Common to the literature mentioned above is a focus on the production chain from the reservoir down to the inlet separator, which is operated at a fixed pressure. This literature provides little information on optimization models and solution algorithms for integrated upstream petroleum production systems. Normally only a part of the problem is addressed or ad hoc rules, which may lead to suboptimal solutions, are applied.

However, the following articles do address problems in the same category as this work. Bieker (2007) solves a problem with simple network topology as a MILP, but without decomposition. Kosmidis et al. (2005) deal with more flexible network topologies that allow routing of fluid streams from wells between different pipelines and to different separators, and solve the problems as an undecomposed MINLP. Foss et al. (2009) and Gunnerud and Foss (2009) use decomposition to solve a problem with a single linking gas constraint. Foss et al. (2009) uses Lagrangian relaxation and Gunnerud and Foss (2009) use Dantzig-Wolf with a heuristic treatment of integrality. Gunnerud et al. (2010) show how the solution time for this method can be improved by solving the subproblems in parallel. These last 3 papers and this chapter use data from the Troll field on the Norwegian shelf.

The current chapter proposes a solution method for an extend version of the problem solved in Gunnerud and Foss (2009). We extend to two common capacity constraints and solve larger systems with more realistic data. The focus is on how to treat the integrality in a non-heuristic way by using column generation in a branch & price framework.

4.2.2 Problem structure

It is common in the offshore oil industry to have a production platform connected to several *clusters* each of which consists of a collection of wells and pipelines connecting the wells to the platform. An illustration of a typical cluster in an oil production system is showed in Figure 4.1. Different clusters may contain different numbers of pipelines, manifolds and wells. A pipeline consists of one or more pipes, each pipe either joining two manifolds or a manifold and the separator.

The flow of oil, gas and water from a well is a function of the wellhead pressure, and the relationship is know as the well performance curve (WPC). The pressure drop across a pipe depends strongly on the flow of oil, gas and water through the pipe, and weakly on the inlet temperature and outlet pressure. The well flow and pipe pressure drop functions are highly nonlinear and accurate models of them requires complex thermodynamic and multi-phase flow calculations. We use commercial simulators for these calculations, and evaluate the functions at grids of points. Based on these

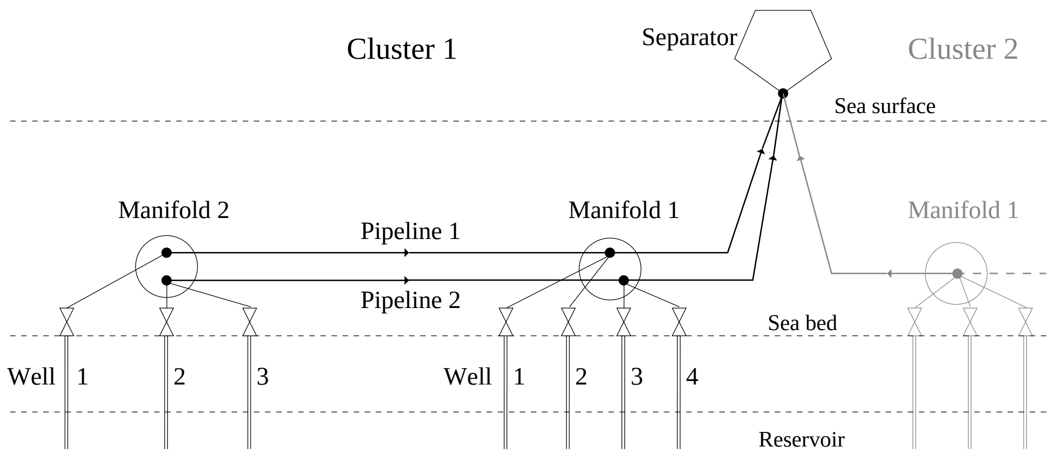


Figure 4.1: Cluster topology

values piecewise linear approximations are defined and included in the optimization model, see Section 4.3.

Each well has a choke valve through which the well's output flows before entering the manifold and one of the pipelines. By operating this valve is it possible to increase the wellhead pressure and thereby reduce the flow. To prevent back flow into the well the model must ensure that whenever the choke is open to any extent the pressure in the manifold does not exceed the pressure in the wellhead. Furthermore a well cannot be connected to more than one pipeline, so when there is more than one pipeline in a manifold there is also a diverter valve which must be set to connect the well to exactly one pipeline. The fluid then flows through the pipeline to the production platform, which has a limited capacity for separating gas and water from the oil.

4.3 Model formulation

The following indexing conventions are used throughout the chapter. Each cluster is identified by a single index i. Each manifold is identified by 2 indices im, the index of the cluster in which it lies and its own index within the cluster. The index of the manifold is its number counting from the separator, which for convenience is treated as a manifold with index 0. Each well is identified by 3 indices imj, 2 indices im to identify the manifold to which it is connected, and the index j of the well within the group of wells connected to that manifold. Each pipeline is identified by 2 indices it, the index of the cluster i where it lies and an index t of the pipeline within the cluster. Each pipe is identified by 3 indices imt, the indexes it of the pipeline of which it is a part and the index m of the manifold at its inlet end.

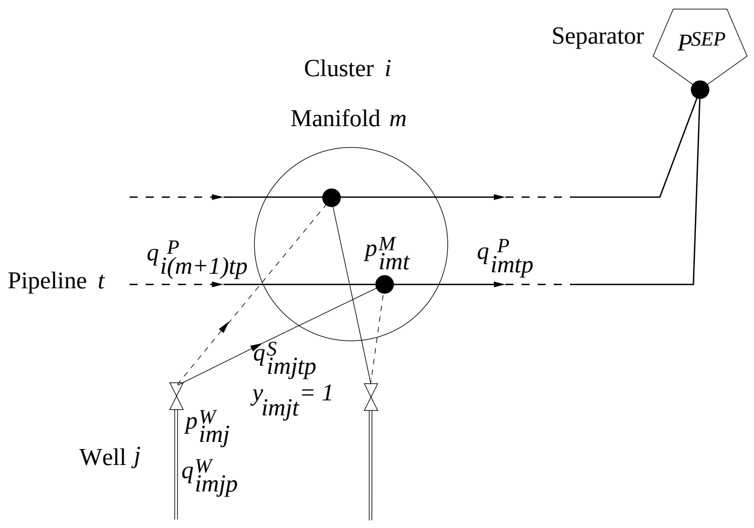


Figure 4.2: Illustration of manifold topology with variables

All quantities used in the model are defined in Tables 4.1 to 4.5 All variables are non-negative, and the main ones are illustrated in Figure 4.2.

Table 4.1: Indices

i	-	Cluster
m	-	Manifold when $m > 0$, and separator when $m = 0$
j	-	Well
t	-	Pipeline
p	-	Phase (gas, oil and water)
k	-	Breakpoint index for the piecewise linearization of the well model
n_p	-	Breakpoint index for the piecewise linearization of the pressure drop in pipes dependent on phase p (n_g for gas, n_o for oil, n_w water)

Objective function The objective is to maximize the total oil flow from all clusters.

$$\max\ z = \sum_{i \in \mathcal{I}} q^{C}_{io} \tag{4.1}$$

Constraints The constraints can be segmented into the following groups:

Capacity constraints (4.2) are the only constraints that connect the clusters. They state that the sum of gas rates and water rates from all clusters must be less than the

Table 4.2: Sets

$\mathcal{I}$	-	Set of clusters
$\mathcal{M}_i$	-	Set of manifolds $1..\mathrm{M}_i$ in cluster i
$\mathcal{J}_{im}$	-	Set of wells connected to manifold m in cluster i
$\mathcal{T}_i$	-	Set of pipelines in cluster i
$\mathcal{P}$	-	Set of phases (g for gas, o for oil, w for water)
$\mathcal{P}^C$	-	Set of capacity constrained phases
$\mathcal{K}_{imj}$	-	Set of indices of breakpoints for the piecewise linearization of the well performance curve (WPC) for well imj
$\mathcal{N}_{imp}$	-	Set of indices of breakpoints for the phase p coordinate in the piecewise linearization of the pressure drop in pipes imt (same for all piplines $t \in \mathcal{T}_i$)

Table 4.3: Parameters

C_p^T	-	Capacity of phase p in the first stage separator
P^{SEP}	-	Pressure at first-stage separator at platform (= p_{i0t}^M)
$P_{imtn_gn_on_w}^D$	-	Pressure drop in pipe imt corresponding to interpolation indices n_g, n_o and n_w in the pipe pressure drop piecewise linearization
P_{imj}^{MAX}	-	Maximum difference between the wellhead pressure and the pressure inside the manifold
P_{imjk}^W	-	Wellhead pressure of well imj for interpolation index k in the piecewise linearization of the well performance curve (WPC)
$Q_{imtpn_p}^P$	-	Flow of phase p in pipe imt interpolation index n_p.
Q_{imjp}^{WMAX}	-	Maximum flow rate of phase p from well imj
Q_{imjpk}^W	-	Flow rate of phase p from well imj corresponding to interpolation index k in the piecewise linearization of the well performance curve (WPC)
M_i	-	Number of manifolds in cluster i

Table 4.4: General variables

p^M_{imt}	-	Pressure in pipeline it in manifold or separator m
p^D_{imt}	-	Pressure drop across the pipe imt
p^W_{imj}	-	Wellhead pressure in well imj
q^C_{ip}	-	Total flow of phase p from cluster i
q^P_{imtp}	-	Flow rate of phase p in pipe imt
q^W_{imjp}	-	Flow rate of phase p from well imj
q^S_{imjtp}	-	Flow rate of phase p from well imj into pipeline t
t^T_{imt}	-	Temperature at inlet of pipe imt
y_{imjt}	-	Equals 1 if well imj is routed to pipeline t, 0 otherwise

Table 4.5: Interpolation variables

γ_{imjk}	-	Weighting variable associated with interpolation index k in the piecewise linearization of the WPC for well imj
$\lambda_{imtn_gn_on_w}$	-	Weighting variable associated with the interpolation indices n_g, n_o, n_w in the piecewise linearization of the pressure drop in pipe imt
η_{imtpn_p}	-	Sum of all $\lambda_{imtn_gn_on_w}$ for a fixed value of n_p. Used in SOS2 sets for coordinate p in the piecewise linearization of the pressure drop in pipe imt

separator's gas handling capacity and water handling capacity respectively.

$$\sum_{i \in \mathcal{I}} q^C_{ip} \leq C^T_p, \quad \forall\, p \in \mathcal{P}^C \tag{4.2}$$

Apart from these capacity constraints, all other constraints are defined locally for each cluster $i \in \mathcal{I}$. For the sake of simplicity this indexing definition will be omitted from the rest of the formulation.

The total flow from cluster i is the sum of flows from the pipes entering the separator.

$$\sum_{t \in \mathcal{T}_i} q^P_{imtp} = q^C_{ip}, \quad \forall\, m = 1,\ p \in \mathcal{P} \tag{4.3}$$

The well model: The output from a well depends on the properties of the reservoir and the wellbore and is characterised by the well performance curve (WPC). This specifies the output of each phase as a function of wellhead pressure. Our model uses the same piecewise linear approximation of the WPC as in Gunnerud and Foss (2009). Let P^W_{imjk} be the k^{th} pressure breakpoint for well imj and let Q^W_{imjpk} be the corresponding flow of phase p. The piecewise linear approximations are shown in (4.4) - (4.7)

$$p^W_{imj} = \sum_{k \in \mathcal{K}_{imj}} P^W_{imjk} \gamma_{imjk}, \qquad \forall\ m \in \mathcal{M}_i,\ j \in \mathcal{J}_{im} \tag{4.4}$$

$$q^W_{imjp} = \sum_{k \in \mathcal{K}_{imj}} Q^W_{imjpk} \gamma_{imjk}, \qquad \forall\, m \in \mathcal{M}_i,\ j \in \mathcal{J}_{im},\ p \in \mathcal{P} \tag{4.5}$$

$$\sum_{k \in \mathcal{K}_{imj}} \gamma_{imjk} = 1, \qquad \forall\, m \in \mathcal{M}_i,\ j \in \mathcal{J}_{im} \tag{4.6}$$

$$\gamma_{imjk} \text{ is SOS2 for } k, \qquad \forall\, m \in \mathcal{M}_i,\ j \in \mathcal{J}_{im} \tag{4.7}$$

Network logic constraints: The following inequality ensures that well imj is connected to at most one pipeline:

$$\sum_{t \in \mathcal{T}_i} y_{imjt} \leq 1, \quad \forall\, m \in \mathcal{M}_i, j \in \mathcal{J}_{im} \tag{4.8}$$

Well and pipe pressure and flow linkage constraints

$$\sum_{t\in\mathcal{T}_i} q^S_{imjtp} = q^W_{imjp}, \qquad \forall\, m \in \mathcal{M}_i,\ j \in \mathcal{J}_{im},\ p \in \mathcal{P} \tag{4.9}$$

$$q^S_{imjtp} \leq Q^{MAX}_{imjp} y_{imjt}, \quad \forall m \in \mathcal{M}_i,\ j \in \mathcal{J}_{im},\ t \in \mathcal{T}_i,\ p \in \mathcal{P} \tag{4.10}$$

$$q^P_{imtp} = \sum_{j\in\mathcal{J}_{im}} q^S_{imjtp} + q^P_{i(m+1)tp}, \qquad \forall\, 1 \leq m < M_i,\ t \in \mathcal{T}_i,\ p \in \mathcal{P} \tag{4.11}$$

$$q^P_{imtp} = \sum_{j\in\mathcal{J}_{im}} q^S_{imjtp}, \qquad \forall\, m = M_i,\ t \in \mathcal{T}_i,\ p \in \mathcal{P} \tag{4.12}$$

$$p^M_{imt} \leq p^W_{imj} + P^{MAX}_{imj}(1 - y_{imjt}), \qquad \forall\, m \in \mathcal{M}_i,\ j \in \mathcal{J}_{im},\ t \in \mathcal{T}_i \tag{4.13}$$

Constraint (4.9) states that the flow from a well has to be equal to the flow into the pipelines. Constraints (4.11) and (4.12) define the flow balance between wells and pipes at each manifold, (4.12) for the furthest out manifolds and (4.11) the others. When the connection between well imj and pipe imt is open, then $y_{imjt} = 1$ so (4.10) does not constrain the flow from the well to the pipe and (4.13) constrains the pressure in the manifold to be no higher than the wellhead pressure. When the connection is closed, then $y_{imjt} = 0$ and (4.10) ensures that there is no flow from the well into that pipeline, and (4.13) is relaxed so that there is no link between the wellhead and pipeline pressure.

Pipe model: The pressure drop $F^P_{imt}(q_g, q_o, q_w, p, t)$ between the inlet and outlet ends of pipe imt can be expressed as a function of the flows of gas q_g, oil q_o and water q_w and the inlet temperature t and outlet pressure p. Over the range where they can vary, the variation of pressure drop with t is not significant and the variation with p is small. We therefore make the following approximation

$$\begin{aligned} F^P_{imt}(q_g, q_o, q_w, p, t) &\simeq F^P_{imt}(q_g, q_o, q_w, P^{REF}_{imt}, T^{REF}_{imt}) + \alpha_{imt}(p - P^{REF}_{imt}) \\ \text{where } \alpha_{imt} &= \frac{\partial F^P_{imt}}{\partial p}(Q^{REF}_{imtg}, Q^{REF}_{imto}, Q^{REF}_{imtw}, P^{REF}_{imt}, T^{REF}_{imt}), \end{aligned}$$

where the REF quantities are fixed and chosen to be in the middle of the expected range for the corresponding variables. $F^P_{imt}(q_g, q_o, q_w, P^{REF}_{imt}, T^{REF}_{imt})$ is a nonlinear function of the 3 variables q_g, q_o and q_w and we approximate it by a piecewise linear function in the same way as in Gunnerud and Foss (2009). The pressure drop variable for pipe imt is p^D_{imt}, so in the model the equation for the approximate pressure drop is

$$p^M_{imt} - p^M_{i(m-1)t} = p^D_{imt} + \alpha_{imt}(p^M_{i(m-1)t} - P^{REF}_{imt}), \quad \forall\, m \in \mathcal{M}_i,\ t \in \mathcal{T}_i \tag{4.14}$$

The piecewise linear function for p^D_{imt} is constructed as follows. Given the pipe's reference pressure P^{REF}_{imt} and temperature T^{REF}_{imt} the simulator is used to calculate

$$P^D_{imtn_gn_on_w} = F^P_{imt}(Q^P_{imtgn_g}, Q^P_{imton_o}, Q^P_{imtwn_w}, P^{REF}_{imt}, T^{REF}_{imt}),$$

for all breakpoints, where $Q^P_{imtpn_p}$ is the flow of phase p at the breakpoint with index n_p. Using this breakpoint data the piecewise linear function is now defined by (4.15) to (4.21) below.

$$p^D_{imt} = \sum_{n_g \in \mathcal{N}_{img}} \sum_{n_o \in \mathcal{N}_{imo}} \sum_{n_w \in \mathcal{N}_{imw}} P^D_{imtn_gn_on_w} \lambda_{imtn_gn_on_w}, \quad \forall\, m \in \mathcal{M}_i,\ t \in \mathcal{T}_i \tag{4.15}$$

$$q^P_{imtp} = \sum_{n_g \in \mathcal{N}_{img}} \sum_{n_o \in \mathcal{N}_{imo}} \sum_{n_w \in \mathcal{N}_{imw}} Q^P_{imtpn_p} \lambda_{imtn_gn_on_w}, \ \forall\, m \in \mathcal{M}_i,\ t \in \mathcal{T}_i,\ p \in \mathcal{P} \tag{4.16}$$

$$\sum_{n_g \in \mathcal{N}_{img}} \sum_{n_o \in \mathcal{N}_{imo}} \sum_{n_w \in \mathcal{N}_{imw}} \lambda_{imtn_gn_on_w} = 1, \ \forall\, m \in \mathcal{M}_i,\ l \in \mathcal{T}_i \tag{4.17}$$

In addition we require that all non-zero $\lambda_{imtn_gn_on_w}$ weights are at vertices of a cube of neighbouring breakpoints. This condition can be enforced by 3 SOS2s, one for each of the gas, oil and water dimensions and defined as follows

$$\eta_{imtgn_g} = \sum_{n_o \in \mathcal{N}_{imo}} \sum_{n_w \in \mathcal{N}_{imw}} \lambda_{imtn_gn_on_w}, \quad \forall\, m \in \mathcal{M}_i,\ t \in \mathcal{T}_i,\ n_g \in \mathcal{N}_{img} \tag{4.18}$$

$$\eta_{imton_o} = \sum_{n_g \in \mathcal{N}_{img}} \sum_{n_w \in \mathcal{N}_{imw}} \lambda_{imtn_gn_on_w}, \quad \forall\, m \in \mathcal{M}_i,\ t \in \mathcal{T}_i,\ n_o \in \mathcal{N}_{imo} \tag{4.19}$$

$$\eta_{imtwn_w} = \sum_{n_g \in \mathcal{N}_{img}} \sum_{n_o \in \mathcal{N}_{imo}} \lambda_{imtn_gn_on_w}, \quad \forall\, m \in \mathcal{M}_i,\ t \in \mathcal{T}_i,\ n_w \in \mathcal{N}_{imw} \tag{4.20}$$

$$\eta_{imtpn_p} \text{ is SOS2 for } n_p, \quad \forall\, m \in \mathcal{M}_i,\ t \in \mathcal{T}_i,\ p \in \mathcal{P} \tag{4.21}$$

It should be mentioned that only 4 neighboring $\lambda_{imtn_gn_on_w}$ are needed to enclose the solution, so 8 neighboring $\lambda_{imtn_gn_on_w}$ will therefore result in several degrees of freedom. These could be removed by including SOS2 sets on the diagonal as described in Williams (2005), however, as this adds complexity, we compensate instead by using more breakpoints.

Fixed separator pressure: To facilitate a compact formulation, the separator is defined as $m = 0$. The pressure P^{SEP} at the separator is fixed and is equal to the outlet pressure p^M_{i0t} of all pipes $i0t$ connected to it, so

$$p^M_{i0t} = P^{SEP}, \quad \forall\, t \in \mathcal{T}_i \tag{4.22}$$

For pipe $i1t$, *i.e.* the one connected to the separator, we take the reference outlet pressure to be $P^{REF}_{i1t} = P^{SEP}$. It follows that when $m = 1$ the last term in (4.14) is zero.

4.4 Branch & Price

In formulation (4.1) - (4.22) it is only the objective (4.1) and the constraint (4.2) that involve variables from more than one cluster. Further, each of these expressions involves only a sum of these variables, so by removing these constraints and instead dealing with them by pricing, the problem decomposes into independent subproblems, one for each cluster and each subproblem is itself a MILP. A common approach for problems of this form is Lagrangian relaxation, however in this chapter we use branch & price (B&P). B&P has the advantages of Lagrangian relaxation, and can in addition guarantee to find the global optimum.

4.4.1 Branch & price overview

In B&P there is a master problem which constructs an optimal solution by selecting feasible solutions (or modes of operation) of subproblems and combining them. The master problem contains constraints for the common resources (gas and water separation capacity in our case). Each column in the master problem corresponds to a mode of operation of a subproblem and its coefficients in the resource constraint rows are the total use of that resource by the subproblem mode. The master problem is solved by branch & bound and the problem at each node of the branch & bound tree is solved by column generation, Desaulniers et al. (2005) chapter 1. Column generation is used because there is an infinite number of possible subproblem modes so it is not possible to generate the master problem explicitly. The master problem at a B&P node is solved iteratively. Each iteration starts by solving a restricted master problem (RMP), which consists of a finite subset of the possible master problems columns. Then the subproblems are solved using the dual values for the common constraints to find the column with the most attractive reduced cost. If there is no attractive column, the master problem at that B&P node is solved. Otherwise, a new column is added to the master problem and the algorithm continue.

The solution of the master problem at a B&P node consists for each subproblem of either a single mode of operation or a convex combination of modes. In most nodes there will be some subproblem with convex combination of modes that is not physically possible. To get round this problem and generate feasible solutions we also solve the RMP as an integer problem to select exactly one operating mode per subproblem.

In most B&P applications the branching is done on a discrete subproblem structure, though in some routing problems branching is done also on one continuous quantity, usually time. In this chapter all the branching is done on the continuous capacity usages.

Also the subproblems in B&P usually have a structure which can be solved effectively by dynamic programming, see chapter 2 and 5 in Desaulniers et al. (2005). However, the subproblems presented in this chapter have a more general structure, so we solve them using a commercial MILP solver, which is efficient and makes the implementation simpler.

4.4.2 Restricted master problem

Let μ_{is} be a variable representing the weighting within the RMP of operating mode s of cluster i, and let Q_{isp}^{MODE} be the total production of phase p in this mode. Also let $\mathcal{S}_i$ be the set of existing modes for cluster i which satisfy the cluster's gas and water production constraints at the current B&P node. Then the RMP is as follows:

$$\max \sum_{i \in \mathcal{I}} \sum_{s \in \mathcal{S}_i} Q_{iso}^{MODE} \mu_{is} \tag{4.23}$$

subject to

$$\sum_{i \in \mathcal{I}} \sum_{s \in \mathcal{S}_i} Q_{isp}^{MODE} \mu_{is} \leq C_p^T, \quad \forall\, p \in \mathcal{P}^C \tag{4.24}$$

$$\sum_{s \in \mathcal{S}_i} \mu_{is} = 1, \quad \forall\, i \in \mathcal{I} \tag{4.25}$$

$$\mu_{is} \geq 0, \quad \forall\, i \in \mathcal{I},\ s \in \mathcal{S}_i \tag{4.26}$$

Equation (4.23) is the objective function, (4.24) represents the constrained common resources, and (4.25) and (4.26) allow a convex combination of all solutions found for a given subproblem.

When this problem is solved, it produces a primal solution, μ_{is}, and a dual solution. If we denote the dual variables for (4.24) by π_p^{CAP} and for (4.25) by π_i^{CONVEX}, the reduced cost, $\overline{c}_{is}$, for μ_{is} can be expressed as:

$$\overline{c}_{is} = Q_{iso}^{MODE} - \sum_{p \in \mathcal{P}^c} \pi_p^{CAP} Q_{isp}^{MODE} - \pi_i^{CONVEX}, \quad \forall\, i \in \mathcal{I},\ s \in \mathcal{S}_i \tag{4.27}$$

4.4.3 Structure of the master problem solution

The RMP is an LP with one equality convexity constraint for each subproblem and $|\mathcal{P}^C|$ common inequality constraints. If we add slack variables for the inequalites, then since there is one basic variable per row, the problem will have $|\mathcal{P}^C|$ more basic variables than clusters. Hence the number of basic variables among the μ_{is} variables is the number of clusters + the number of nonbasic slack variables, which is at most $|\mathcal{P}^C|$. Since the convexity constraints have positive RHS there must be at least one non zero μ_{is} in each cluster, and this must be basic. In addition to these basic variables there can be up to $|\mathcal{P}^C|$ other basic variables among the μ_{is}. For the case in this chapter when $|\mathcal{P}^C| = 2$, there can be either 2 clusters with 2 basic variables each, or 1 cluster with 3 basic variables, or 1 cluster with 2 basic variables, or no clusters with more than one basic variable. Also since all nonbasic variables are 0, if a cluster has a single basic variable, then this must be integer with value 1. If the problem is non-degenerate, then clusters with more than one basic variable will be fractional, however this is not necessary the case if the problem is degenerate. Hence for our problems where $|\mathcal{P}^C| = 2$ one of the following cases must occur: 2 clusters with 2 fractional values in each, 1 cluster with 2 or 3 fractional values, or 0 clusters with fractional values. It follows that except in cases of degeneracy, the RMP solution will contain fractional weights.

4.4.4 Subproblem

If we want to improve the continuous solution of the RMP, we need to find a feasible mode of operation for at least one of the clusters which gives a positive reduced cost according to (4.27). If such a production mode exist, it can be found by solving separately for each cluster, i, the MILP subproblem

$$z_i = \max \ q_{io}^C - \sum_{p \in \mathcal{P}^C} \pi_p^{CAP} q_{ip}^C - \pi_i^{CONVEX} \tag{4.28}$$

subject to (4.3) - (4.22) and

$$Q_{ip}^{MIN} \leq q_{ip}^{C} \leq Q_{ip}^{MAX}, \;\; \forall\, p \in \mathcal{P}^{C} \tag{4.29}$$

where Q_{ip}^{MIN} and Q_{ip}^{MAX} are the lower and upper bounds on q_{ip}^{C} in the current B&P node. Constraint (4.29) will be discussed later in relation to the branching.

As mentioned earlier, these MILP subproblems will be solved by a commercial MILP solver that is able to treat SOS2 automatically. Whenever an optimal subproblem objective, z_i, is positive, we create a new column for the master, with index s say, with entries Q_{iso}^{MODE} in the objective row, Q_{isg}^{MODE} and Q_{isw}^{MODE} in the gas and water rows respectively, and a 1 in the convexity row for cluster i. Here Q_{isp}^{MODE} for each phase p is the optimal value of q_{ip}^{C} in the solution for cluster i. Finally s is added to set S_i.

If no column with a positive reduced cost is found for any of the clusters, then the current solution is the optimal continuous solution to the current node of the B&P tree.

4.4.5 Finding feasible solutions

We noted in Section 4.4.3 that except in cases of degeneracy, the RMP solution will contain fractional weights. If any modes in a cluster that have fractional weights have a different set of nonzero binary or SOS2 variables, then the convex combination is infeasible. This is likely to be the situation at the solution of a B&P node unless the bounds on the flows of the constrained phases are narrow. However relying on branching to reduce these ranges sufficiently to produce feasible solutions would lead to a huge B&P tree.

To avoid this problem, we find an integer feasible solution by solving the RMP (4.23) - (4.26) as a pure binary problem with (4.26) replaced by

$$\mu_{is} \in \{0, 1\}, \quad \forall\, i \in \mathcal{I},\; s \in \mathcal{S}_i \tag{4.30}$$

The set $\mathcal{S}_i$ here consists of all previously generated operating modes, not just those from the current B&P node. This step is done after the solution of each B&P node. In the following, the IP version of the RMP is denoted RMP-IP while the LP version is denoted RMP.

4.4.6 Branching strategy

The lower bound on the optimal solution is given by the best RMP-IP solution, and the upper bound is given by the highest optimal RMP objective value among all active nodes in the B&P tree. To close this gap we need to create new modes of operation for the different clusters so that better IP solutions can be created, and we need to branch to tighten the LP upper bounds by splitting nodes.

In B&P it is normal to branch on the original discrete variables, see Barnhart et al. (1998). Our problem contains two types of discrete variables, the binary variables in (4.8), and the SOS2 variables in (4.7) and (4.21). There are very many of these variables and so it is likely that using them for branching will result in a very large number of branches. The column generation procedure however is time consuming, so only a small number of master B&P nodes can be solved in a reasonable amount of time, and so this approach is unlikely to be successful.

In this chapter, instead of branching on the original integer and SOS variables, we branch on the continuous gas and water flow variables from each cluster by imposing bounds on the available capacities. Columns which conflict with these new bounds are removed from the child nodes.

Our aim in choosing the branching variable q_{ip}^{C} is to maximize the distance between the solution at the parent and the closer of the solutions at its children. This should keep the branch and bound tree balanced and avoid generating new production modes that are close to existing modes. We use the following method:

First identify all μ_{is} with fractional values. There must be at least one otherwise the node would be feasible and it would not have been branched on. As noted in Section 4.4.3 the only cases that can occur are two clusters with 2 fractional variables each, or one cluster with 2 or 3 fractional variables. In the first case we need to select a cluster and a phase and in the other cases only a phase. The following rules are used in all cases:

Let the interpolated production of phase p (gas or water) in the RMP solution of subproblem i be denoted Q_{ip}^{AV}. For each subproblem i such that μ_{is} is fractional for some mode s, calculate

$$D_{isp} = \mid Q_{isp}^{MODE} - Q_{ip}^{AV} \mid, \;\; \forall p \in \mathcal{P}^{C}, s \in \mathcal{S}_i, 0 < \mu_{is} < 1 \qquad (4.31)$$

D_{isp} is the absolute difference between the RMP solution and a column used in the interpolation. Then for each cluster i with a fractional μ_{is} and each phase p find the second largest value of D_{isp} among the 2 or 3 modes s with fractional μ_{is} and denote this by D^*_{ip}. Then find the cluster i^* and phase p^* which give the largest relative values of these D^*_{ip}, *i.e.*

$$i^*, p^* = \arg\max_{i,p} \frac{D^*_{ip}}{Q^{AV}_{ip}} \tag{4.32}$$

Then branch on cluster i^* and phase p^*. Finally set the branching value to Q^{AV}_{ip}, *i.e.* enforce the bound $q^C_{ip} \leq Q^{AV}_{ip}$ in the lower branch and $q^C_{ip} \geq Q^{AV}_{ip}$ in the upper branch.

Choice of master node to evaluate

As can be seen in the results section, Section 4.6, the number of B&P nodes is small and we are able to explore the full tree. We therefore choose best first search for node selection to minimize the number of B&P nodes solved. Another advantage of this compared to depth first search is that it generates a more diverse set of modes early on in the search and this is likely to improve the integer solutions found by RMP-IP.

When there is a choice between two child nodes of the same parent we choose the lower branch. This is likely to produce a mode with lower resource usage than the upper branch and a mode with less resource usage is more likely to generate a feasible solution in combination with other existing modes. This new solution may allow the other twin branch to be eliminated.

4.4.7 Branch & price algorithm

Algorithm 1 shows a pseudo code of the basic version of the B&P algorithm we are using to solve the RTPO problem.

The B&P algorithm starts by generating several initial columns i.e. modes of operating the clusters. This is done by creating a zero production column and a maximum production column for each cluster. This guarantees that the root B&P node is feasible. The next step is to solve the RMP to find dual values on the common capacity constraints. These prices are sent to the subproblems, and all subproblems are solved. Each subproblem which returns with a positive reduced cost has its optimal solution added as a column to the RMP. The column generation procedure is repeated until no new positive reduced cost column is generated, at which stage the master problem at that node has been solved.

Algorithm 1: Branch & price (B&P-1)

```
forall the clusters do
    generate zero and max production columns
create root-node
repeat
    choose node to calculate
    repeat
        solve RMP
        send dual values for common constraints to subproblems
        forall the clusters do
            solve cluster subproblem
            if column has positive reduced cost then
                send new column to RMP
    until no new columns with positive reduced cost found;
    solve RMP as IP
    update bounds
    if IP gap < termination criteria then
        optimal solution found
    else
        if node is feasible and LP > global LB then
            branch and create new nodes
            add new nodes to list of unsolved problems
until optimal solution found;
```

At this stage, the RMP-IP is solved to provide a feasible integer solution, and hence a lower bound for the problem. If the gap between the upper bound and this feasible solution is within the predefined termination criteria, the algorithm terminates. Otherwise the B&P algorithm creates two new problems based on the branching rules described in Section 4.4.6. These two problems are then added to the list of unsolved problems. The next step is then to choose one of these unsolved problems based on the search strategy in Section 4.4.6, gather the set $\mathcal{S}_i$ of existing modes for cluster i which satisfy the cluster's gas and water production limits, and run the column generation procedure again.

We shall compare the following variants of the B&P method

- **B&P-1** is our basic B&P method, B&P-2 and B&P-3 builds on this method. It uses a best first search and branches on the gas or water production from clusters with a fractional solution in the RMP, as described in Section 4.4.6. All subproblems are solved before the RMP is resolved. Each mode which has a positive reduced cost and is the most positive in its subproblem is added to RMP. To create feasible solutions the RMP-IP is solved using all columns so far generated. This is done once per B&P node after its column generation has converged.

- **B&P-2** is the same as B&P-1 with 2 changes. Firstly the RMP is re-solved after each subproblem if a new production mode is found. Secondly when a subproblem fails to produce a column in one iteration it is not solved again until after an iteration which produces no columns. At that stage all the subproblems are again made active and the solution continues using the current dual prices.
- **B&P-3** is the same as B&P-2 except that after each iteration a three step procedure is used to generate a better incumbent. Firstly all subproblems are solved with dual variables equal zero, and upper bounds on gas and water production equal to the RMP-IP solution plus the gap between the RMP-IP solution and the total production capacities (the slack of the two global constraints). Secondly, any of these columns that are better than the column for that subproblem selected in the RMP-IP are added to the RMP. Thirdly if any better columns have been found, the RMP-IP is solved again (which will give a better incumbent).

4.5 Implementation

The data for the piecewise linearization of the well and pipe models can be generated using any suitable simulators. In this chapter we have used two state of the art simulators, GORM, Mjaavatten et al. (2006) (an in house Statoil simulator), for the wells, and GAP, from Petroleum Experts Petroleum Experts (2010), for the pipes. These simulators are used currently in Troll operations. For each well we calculate the gas, oil and water flow for every wellhead pressure, and for each pipe we calculate the pressure drop across the pipe for every combination of gas, oil and water flow. The results are then stored in tables in a form suitable for Xpress.

In the Troll field the parallel pipelines are identical. In such situations swapping the set of wells attached to each pipeline produces an equivalent solution, and this introduces a symmetry which makes branch & bound for each subproblem much harder. We remove this symmetry, by selecting one well in each cluster that is likely to be producing and only allowing it to be routed to (at most) one of the pipelines.

The B&P optimization algorithm is implemented in Mosel in Xpress-IVE FICO (2009) without any parallelization. *"mmjobs"* is used to jump between the sub and master problems, and breakpoint tables together with other topology information are included through data files. The computations were performed on single core of a Intel E5472, 3.0 GHz, 16GB RAM. Both Linux and Xpress were 64 bit versions. The solution times presented do not include time used to simulate well and pipe data. New runs for some of the well models i.e. for the wells that have changed behavior, is done before every optimization run, which in total takes typically a few seconds to a minute. The pipe models don't change so they need to be run only once.

4.6 Results and discussion

To validate our B&P method we have constructed a realistic model based on the structure of Troll B and C and using pipe and well data from these fields. The structure was shown in Figure 4.1 and its size is comparable to the largest subsea production systems in the world. The full model has 8 clusters each containing 8 wells, 2 manifolds and 2 parallel pipelines. A problem of this size cannot be solved as a single MILP, so for comparison purposes we have created 2, 4 and 6 cluster problems by removing clusters from the full 8 cluster problem and reducing the gas and water limits proportionally.

The purpose of the numerical study is to compare the four solution methods, standard, i.e. the problem is solved as one large MILP without decomposition, B&P-1, B&P-2 and B&P-3. Table 4.6 gives the results for each of these methods on four typical test cases.

The first section of Table 4.6 shows the problem characteristics for the standard formulation. The large number of variables per cluster results from the piecewise linearization of the well and pipe models. The well flow piecewise linearizations each have 20 to 100 breakpoints, and the pipe pressure drop linearizations each have 343 breakpoints, corresponding to 7 breakpoints in each of the gas, water and oil flow SOS2s. There are 2 binary variables and one SOS2 per well and three SOS2s per pipe.

We were not able to solve the 4, 6 and 8 cluster problems to optimality with the standard formulation, as each stopped due to insufficient memory. The optimality gaps at failure are large and increase with the number of clusters. However all the B&P variants solved all the problems to high accuracy.

For the 2 cluster problem the standard and B&P-3 methods took the same time while

Table 4.6: Results

No of clusters	2	4	6	8
Standard				
Variables	3880	7809	11553	15505
Constraints	424	850	1275	1720
Solution time [min]	0.7	181	430	623
Oil prod [Sm^3/day]	2803.5	5648.4	8346.2	12035.7
Optimality gap [%]	<0.01	3.17	9.51	10.9
B&P - 1				
Solution time [min]	6.4	123	257	195
Oil prod [Sm^3/day]	2803.5	5674.0	8534.6	12431.5
Optimality gap [%]	<0.01	<0.01	<0.01	<0.01
No. B&P nodes	4	22	14	14
No. subproblems solved	24	192	198	264
No. subprob with 0 red. cost	14	149	158	219
B&P - 2				
Solution time [min]	1.4	119	181	125
Oil prod [Sm^3/day]	2803.5	5674.0	8534.6	12431.5
Optimality gap [%]	<0.01	<0.01	<0.01	<0.01
No. B&P nodes	4	22	13	14
No. subproblems solved	21	175	156	214
No. subproblems with 0 red. cost	11	134	121	177
B&P - 3				
Solution time [min]	0.7	102	192	121
Oil prod [Sm^3/day]	2803.5	5673.6	8535.2	12431.5
Optimality gap [%]	<0.01	<0.01	<0.01	<0.01
No. B&P nodes	1	16	13	13
No. subproblems solved	15	141	173	245
No. subproblems with 0 red. cost	6	97	119	172

B&P-2 was twice as slow and B&P-1 was 8 times as slow. For the 4 cluster problem the standard method took longer to achieve a bound gap of 3.2% (where it ran out of memory) than the B&P methods took to achieve a bound gap of 0.01%. For the 6 and 8 cluster problems the standard method solution times continued to increase significantly despite completing a decreasing proportion of the search. Interestingly the solution time for the B&P methods in going from the 4 to the 8 cluster problems increases less than linearly. For the 8 cluster problem the standard method took more that 3 times longer than any of the B&P methods despite only achiving a bound gap of 10.9% compared to the 0.01% of the B&P methods.

Most subproblems solve in 5 - 120 seconds with 2000 - 100000 branch & bound nodes. However, on some occasions, the subproblem can be much harder to solve, using several thousand seconds and more than a million branch & bound nodes. The RMP solution time is in the order of 0.01 second, while the RMP-IP solution time is usually one magnitude higher, so both these times are negligible compared to the subproblem solution times.

B&P-1 works well, however as can be seen from Table 4.6, all solution times are lower for B&P-2. The number of B&P nodes are similar, however the recalculation of the master solution after each subproblem solved and postponing resolving subproblems whose last reduced cost was zero has succeeded in significantly reducing the number of subproblem solved and the number which have zero reduce cost. By in addition utilizing the spare capacity of the common constraints as done in B&P-3 we see a further improvement in solution time for three of the four problems. Finding better feasible solutions early in the search produces a significant reduction in the number of B&P nodes and this compensates for the extra subproblem that is solved to find the better feasible solutions.

The optimization methods presented in this chapter are intended as a tool for production engineers at oil and gas fields like Troll. Being able to find solutions overnight is of significant value, and all test problems in this study were solved by B&P-3 within 3.25 hours, so comfortably satisfying this criterion.

However there are occasions when faster solution times would be useful, for example if a well or cluster has to be closed down suddenly due to a production problem. Fortunately good suboptimal solutions are found much earlier in the solution process. Figure 4.3 shows the progress for each B&P method running the 8 cluster example, how the gap between the upper and lower bound varies over time. This shows that B&P-3 produces very good solutions within an hour. In Table 4.6 the 6 cluster example is the only one where B&P-3 not the fastest. A similar analysis of the 6 cluster case however shows that B&P-3 has substantially the best early convergence behaviour. For example with a tolerance of 0.07% B&P-3 is more that 2.9

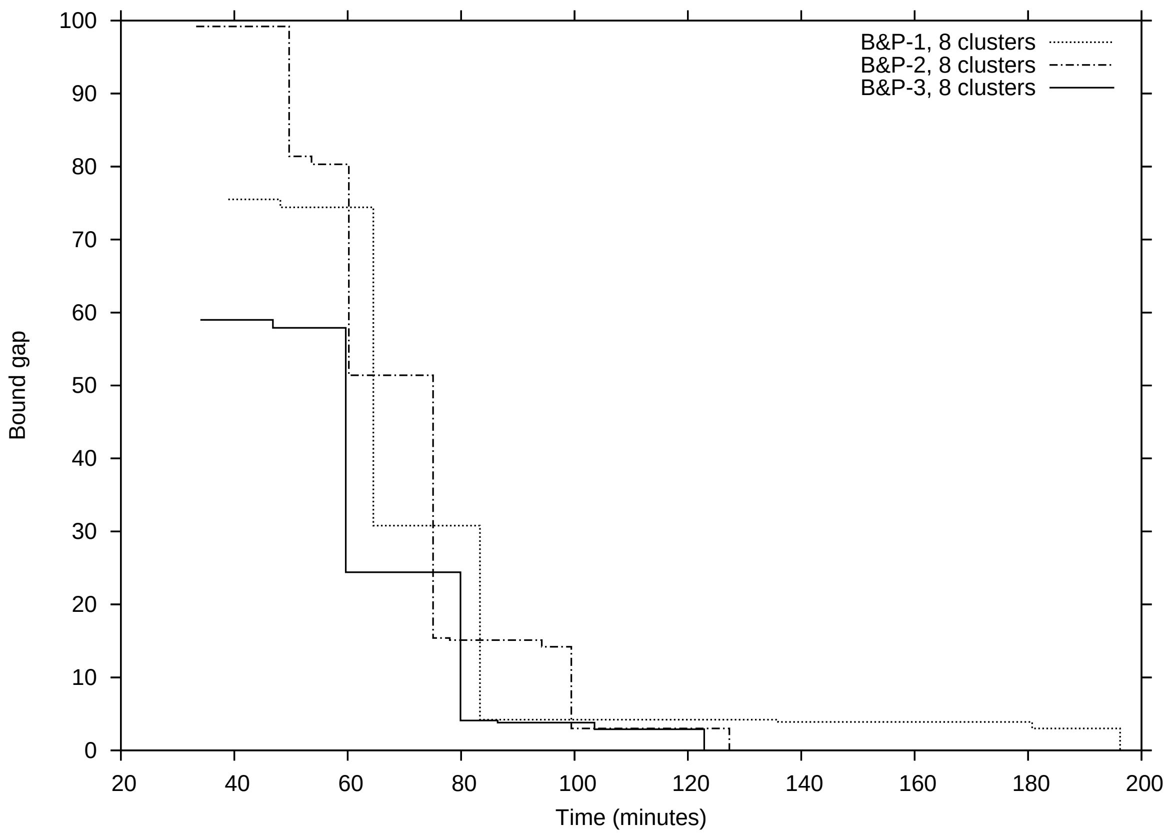


Figure 4.3: Bound gap as function of time for 8 cluster problem

times faster than B&P-1 and B&P-2. It therefore follows that overall B&P-3 is the best and most practical method.

4.7 Conclusion

This chapter presents a method that solves a real-time production optimization problem (RTPO) using column generation in a branch & price framework. The contribution lies in continuous branching on the cluster gas and water production variables, and on the MILP formulation of the subproblems which enables us to use a commercial solver. In addition, we utilize the spare capacity of the common constraints, after solving the restricted master problem, as an IP to create better feasible solutions. In this way we improve the lower bound significantly. Since the master problem solution time is negligible, it is re-solved each time a subproblems returns with a new column. To improve the solution time further, subproblems which once return without generating new columns are skipped until no further columns are generated, at which point the subprblems are solved again to establish the bounds.

The B&P method outperforms the standard implementation, which is only capable of finding the optimal solution of the two smallest cluster problem. The proposed algorithm terminates with no more than 22 B&P nodes for all tested cases, and near optimal solutions were always found in less than 1 hour.

In the Troll case there are 2 parallel pipelines in each cluster and pipelines join only at the separators. In some fields the clusters have different numbers of parallel pipelines and more complex topologies. Our model deals with an arbitrary number of parallel pipelines, however if modified slightly, it could deal with other topologies as well. The computational performance however is unlikely to be affected by this change. For some fields a useful extension would be to include the optimization of gas lift or pump operation within the model. The same solution framework extends to this problem but with more linking constraints between the clusters.

The RTPO problem is a challenge for the production engineers at Troll and other gas and oil producing installations with rate dependent gas, oil and water flow-fractions. The study performed in this chapter was initiated by industrial partners, and has shown the potential to have significant impact when successfully integrated into their operating practices.

Chapter 5

Conclusions and further work

Petroleum plays an important role in our modern society and will continue to do so for several more decades. When realizing this, we should see to utilize our production system in a best possible way. This would mean many things, however the RTPO problem we attempt to solve is an essential component in achieving it.

The complexity of the problem instances and their realism have increased as the methods which have been developed for solving them have improved. In Chapter 2, a semi realistic model of the Troll oil and gas field was used, while more realistic models was developed and solved in Chapter 3 and 4. Since a realistic model of the full field instance of Troll cannot be solved without decomposing it, 2, 4 and 6 cluster problems were created for comparison purposes by removing clusters from the full 8 cluster problem and reducing the gas and water limits proportionally.

The nature of the problem makes it a MINLP problem that is both challenging to describe and solve. The approach has been to piecewise linearize all nonlinearities and thereby create a MILP formulation. It is then possible to apply known and well developed algorithms such as simplex and branch & bound. For the RTPO problem addressed in this work, it means piecewise linear well and pipeline models, as the relation between flow and pressure is highly nonlinear and complex. An additional important effect of this approach is the fact that the underlying wells and pipeline models becomes completely disconnected from the optimization problem. Data points can be created from any reservoir, well and pipeline simulator as an alternative to real production data. Flexible accuracy is also an attribute since the breakpoints can be placed arbitrarily, with longer computation time as the price for higher accuracy.

After establishing a piecewise linear MILP representation of the RTPO problem at Troll, Chapter 2 continues by exploring the possibility of decomposition. Two approached were studied, DWD and LR. For large problems i.e. 6 and 8 clusters, the two decomposition methods are by far faster than the standard non-decomposed implementation. Further, DWD outperforms LR. The reason for this is the mechanism for updating the Lagrangian multipliers. The DWD master problem finds good multipliers with less iteration than for the LR case, and usually converges faster. DWD is also more stable with respect to solution time, it has fewer tuning parameters and works well for different data sets. The experience is that LR is actually quite sensitive to perturbations in the data set. Small change could double the solution time.

As the recent improvements in computational power has mainly been related to parallelization of CPU architecture, and decomposition techniques results in standalone subproblems for each cluster, it was natural to start exploring this to reduce the solution time even further. Chapter 3 describes two different approaches and tests them on 2, 4, 8, and 12 CPUs hardware. By running the simple parallel algorithm on a 12 CPU PC, the solution time was cut by 61% on average. By utilizing a more advanced strategy, the solution time was cut even more, 77% on average. Through the computational evaluation it was identified that the solution time of the master problem and the difference in solution time of the subproblems are critical factors when evaluating the potential of a parallel algorithm.

DWD is developed for LP problems, but can readily be extended to convex ones, however it is not straight forward to incorporate it on a MILP problem. With respect to the Troll RTPO problem it is possible to solve the master problem as an IP and achieve a feasible solution. Even though this is easy to implement and works fairly well on this particular RTPO problem, it does not guarantee convergence. To overcome this challenge, column generation in a branch & price framework is proposed in Chapter 4. This algorithm is an exact method for the RTPO problem at hand, however, it comes with additional computational expenses as a DWD column generation procedure has to be solved in each node. As mentioned in the chapter, the contribution lies in continuous branching on the cluster gas and water production variables. In addition, spare capacity of the common constraints after solving the master problem as an IP is used to find better feasible solutions. Algorithmic adaption to utilize the fact that the master problem solution time is negligible, and that the solution time of the subproblems differs significantly is implemented and tested. The branch & price method converges to global optimality with no more than 22 branch & price nodes for all tested cases.

In the three previous chapters, different approaches to decompose and solve the RTPO problem were tested and discussed. In a practical setting, if one were to implement and solve a real RTPO problem, we recommend the DWD Accelerated feedback strategy presented in Chapter 3. This is an approach which takes advantage of the recent improvements in computationally power, is fairly simple, and on a new workstation with 8-16 cpu's outperforms the sequential approaches presented in Chapter 2. The branch & price approach presented in Chapter 4, has both pros and cons compared to the DWD Accelerated feedback strategy. Given that the deviation between the upper bound found from the decomposed problem solved with DWD, differ significantly from the feasible solution found when solving the RMP as and IP, i.e. the duality gap, the branch & price approach might find significantly better solutions, and hence, the extra computential expenses and implementational complexity may be worthwhile. On the other hand, if the duality gap is relatively small, as is the case for the RTPO instances tested in this thesis, it might not be interesting to implement such an advanced algorithm, as the gain in production is expected to be small. However, by accepting a relatively large duality gap, branch & price acts as a DWD algorithm as long as the root node converges to a solution with a duality gap less than the predefined threshold. In the special cases where this do not occure, it will start to branch and still be able to achieve this threshold, which a DWD algorithm would not be able to. Further, the branch & price approach in Chapter 4 is not parallelized, however the same techniques as outlined in Chapter 3 could be used in each node of the branch & price search tree. In additional it would be possible to solve the open branch & price nodes in parallel, which potentially could make a parallel branch & price implementation, with access to 100 cpu's, on average only 2-3 times more time consuming than the DWD Accelerated feedback strategy presented in Chapter 3.

As the petroleum production chain consists of many parts that is only mildly interconnected, the idea of decomposition seems appealing. Instead of using the inlet separator as downstream boundery and decompose with respect to production clusters, i.e the RTPO problem addressed in this thesis, it would be interesting to include parts of the platform process system as additional subproblems. In some cases it might also be interesting to account for interaction between platforms, also by using decomposition techniques. Hence, decomposition techniques are applicable to large classes of problems in petroleum production optimization.

A key challenge when solving the RTPO problem using the approaches described above, is related to the piecewise linearization of the pipeline pressure drop models. Since pressure drop depends on gas, oil and water flow rates the number of weighting variables increases with the power of three. Twenty breakpoints for each phase will e.g. result in 8000 interpolation weighting variables. In addition there will be $20 * 3 = 60$ SOS2 variables. Reasonable accuracy on the pipeline pressure drop models may mean unacceptable solution times. Therefore, alternative simplifications of the piecewise linearization of the pipelines should be considered. In Chapter 4, an outline of how it is possible take account for absolute pressure without increasing the complexity of the optimization problem is presented. By using the same technique it is also possible to reduce complexity. It could be done by aggregating oil and water flow into liquid flow, and letting the pipeline pressure drop model depend piecewise linearly on gas and liquid, and linearly on oil, water and absolute pressure. This would replace the 8000 interpolation weighting variables with $20 * 20 = 400$. The number of SOS2 variables would then be $20 * 2 = 40$. It will be a huge simplification of the optimization problem, but at the cost of model accuracy. To compensate for this, it is possible to increase the number of two dimensional breakpoints, to let say $40 * 40 = 1600$, and with two sets of SOS2, 40 each. This piecewise linear representation is expected to improve model quality and reduce solution time, compared to the one presented in the thesis.

Another interesting continuation is to develop reduced order nonlinear models for these pressure drops, and in addition the well models, and resort to a MINLP problem formulation, to possibly bypass this challenge completely. By formulating the RTPO problem as an MINLP, other optimization algorithms need to be explored. The nonlinear relation between the pressure at the wellhead and the flow of oil gas and water produced from the well is hard to describe by explicit functions since they are computed in a simulator. One option is to approximate this relation with a polynomial model. The same may be done for the relation between the flow through the pipes and the pressure drop across it. The problem then turns into a MINLP. After solving this MINLP problem the polynomial models may be refitted to locally match the simulators for this solution. The procedure should iterate until hardly no mismatch occur.

The MINLP formulation suggested above reduces the number of variables in the cluster subproblem with eight wells and four pipelines, to an approximately 50. With this small number of variables it might also be possible to include the system dynamics and consider a time varying system. The trend in petroleum production is to a larger and larger extent to use dynamic simulators; it would therefore be interesting to add optimization capabilities also to these simulators.

Engineering companies and end users have shown interest in the developed methods. Hence, the methods are presently considered implemented to enhance existing technology for RTPO.

References

A. Alabi and J. Castro. Dantzig-Wolfe and block coordinate-descent decomposition in large-scale integrated refinery planning. *Computers and Operations Research*, 36:2472–2483, 2009.

G. Amdahl. Validity of the single processor approach to achieving large-scale computing capabilities. In *AFIPS '67 (Spring): Proceedings of the April 18-20, spring joint computer conference*, pages 483–485, 1967.

K. Asanovíc, R. Bodik, B. Catanzaro, J. Gebis, P. Husbands, K. Keutzer, D. Patterson, W. Plishker, J. Shalf, S. Williams, and K. Yelick. *The Landscape of Parallel Computing research: A View from Berkely http://www.eecs.berkeley.edu/Pubs/TechRpts/2006/EECS-2006-183.pdf 29.03.*, 2006.

A. Awasthi, S. Sankaran, M. Nikolaou, L. Saputelli, and G. Mijares. Closing the gap between reservoir simulation and production optimization. *SPE 107463, Digital Energy Conference and Exhibition, 11-12 April, Houston, Texas, U.S.A.*, 2007.

A. Awasthi, S. Sankaran, M. Nikolaou, L. Saputelli, and G. Mijares. Meeting the challenges of real-time production optimization - a parametric model-based approach. *SPE 111853, Intelligent Energy Conference and Exhibition, 25-27 February, Amsterdam, The Netherlands*, 2008.

T. Backx, O. Bosgra, and W. Marquardt. Integration of model predictive control and optimization of processes: Enabling technology for market driven process operation. *ADCHEM conference, 14-16 June 2000, Pisa, Italy.*

C. Barnhart, E. L. Johnson, G. L. Nemhauser, M. W. P. Savelsbergh, and P. H. Vance. Branch-and-price: Column generation for solving huge integer programs. *Operations Research*, 46 (3):316–332, 1998.

E. Beale and J. Tomlin. Special facilities in a general mathematical programming system for non-convex problems using ordered sets of variables. In J. Lawrence, editor, *Proceedings of the Fifth International Conference on Operational Research*, pages 447–454, London, 1969. Tavistock Publishers.

J. Beasley. *Modern heuristic techniques for combinatorial problems*, chapter 6: Lagrangean relaxation, pages 243–303. Halstad Press, 1993. Edited by C.R. Reeves.

H. Bieker, O. Slupphaug, and T. Johansen. Real-time production optimization of offshore oil and gas production systems: A technology survey. *SPE 99446, Journal SPE Production & Operations*, 22:382–391, 2006a.

H. P. Bieker. *Topics in offshore oil production optimization using real-time data*. PhD thesis, Norwegian University of Science and Technology, Trondheim, Norway, 2007.

H. P. Bieker, O. Slupphaug, and T. A. Johansen. Global optimization of multiphase flow networks in oil and gas production systems. In *AIChE Annual Meeting*, San Francisco, 2006b.

C. Blum and A. Roli. Metaheuristics in combinatorial optimization: Overview and conceptual comparison. *ACM Comput. Surv.*, 35(3):268–308, 2003.

P. Bonami, L. T. Biegler, A. R. Conn, G. Cornuejols, I. E. Grossmann, C. D. Laird, J. Lee, A. Lodi, F. Margot, N. Sawaya, and A. Wachter. An algorithmic framework for convex mixed integer nonlinear programs. *Discrete Optimization, doi:10.1016/j.disopt.2006.10.011*, 32, 2008.

S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

C. E. Brennen. *Fundamentals of Multiphase Flow*. Cambridge University Press, 2005. URL `http://caltechbook.library.caltech.edu/51/1/content.htm`. Last accessed 14.11.2010.

R. Cheng, J. Fraser Forbes, and W. San Yip. Dantzig-Wolfe decomposition and plant-wide mpc coordination. *SPE Production and Operations*, 32:1507–1522, 2008.

Y. Colombani and S. Heipcke. Multiple models and parallel solving with Mosel. *Dash optimization Whitepaper*, 2006. URL `http://www.dashoptimization.com/home/downloads/pdf/moselpar.pdf`. Downloaded 02.06.09.

A. J. Conejo, E. Castillo, R. Mínguez, and R. García-Bertrand. *Decomposition Techniques in Mathematical Programming*, chapter 1–3, pages 3–139. Springer Berlin Heidelberg, 2006.

A. Conn, K. Scheinberg, and L. N. Vicente. *Introduction to derivative-free optimization*. SIAM, 2009.

R. Cramer, K.-C. Goh, M. Dolan, and C. Moncur. Data driven surveillance and optimization. *SPE 122554, SPE Digital Energy Conference and Exhibition, 7-9 April, Houston, Texas, USA*, 2009.

Ø. Dahl Hem, A. Svendsen, S.-E. Fleten, and V. Gunnerud. Real options analysis of strategic decisions in offshore petroleum production. *Submitted to real options conference 15-18, Turku, Finland*, 2011.

C. DAmbrosio, A. Lodi, and S. Martello. Piecewise linear approximation of functions of two variables in milp models. *Operations Research Letters, doi:10.1016/j.orl.2009.09.005*, 38:39–46, 2010.

E. Danna and C. Le Pape. *Column Generation*, chapter 4: Branch–and–price heuristics: A case study on the vehicle routing problem with time windows, pages 99–129. Springer, 2005.

G. B. Dantzig and M. N. Thapa. *Linear Programming*. Springer US, 2003a.

G. B. Dantzig and M. N. Thapa. *Linear Programming 2: Theory and Extensions*, chapter 10: Decomposition of Large–Scale Systems, pages 265–321. Springer New York, 2003b.

Dash Optimization. Xpress-Mosel Reference manual, 2007. Release 2.2.

G. Desaulniers, J. Desrosiers, and M. M. Solomin. *Column Generation*. Springer US, 2005.

J. Desrosiers. *Column Generation*, chapter 1: A primer in column generation, pages 1–32. Springer, 2005.

J. Desrosiers and M. E. Lübbecke. *Column Generation*, pages 1–32. Springer US, 2006.

M. Dueñas Díez, K. Brusdal, G. Evensen, T. Barkve, and A. Mjaavatten, editors. *Opportunities and challenges of using sequential quadratic programming (SQP) for optimization of petroleum production networks*, 2005. 15th European Symposium on Computer Aided Process Engineering.

Edinburgh Petroleum Services. *http://www.e-petroleumservices.com/ 22.10.2010*, 2010.

E. Erlingsen, T. Strat, V. Gunnerud, B. Nygreen, and M. Dueñas Díez. Decision support in long term planning of petroleum porduction fields. *Submitted to SPE Inteligent energy conference, Utrecht*, 2012.

O. Fevang, K. Singh, and C. Whitson. Guidelines for choosing compositional and black-oil models for volatile oil and gas-condensate reservoirs. *This paper was prepared for presentation at the 2000 Annual Technical Conference and Exhibition held in Dallas, Texas, 5–8 October, SPE 63087*, 1997.

FICO. Xpress-Mosel User guid, 2009. Release 3.

S.-E. Fleten, V. Gunnerud, O. Dahl Hem, and A. Svendsen. Real option valuation of offshore petroleum field tie-ins. *Submitted to Resource and Energy Economics Journal*, 2010.

FMC Technologies. *http://www.fmctechnologies.com/ 22.10.2010*, 2010.

B. Foss, V. Gunnerud, and M. Dueñas Díez. Lagrangian decomposition of oil-production optimization applied to the troll west oil rim. *SPE Journal SPE-118299-PA, Dec*, 2009.

B. A. Foss and I. J. Halvorsen. Dynamic optimization of the LNG value chain. *Proceedings of the 1st Annual Gas Processing Symposium, Doha, Qatar*, 2009.

I. Foster. *Designing and Building Parallel Programs*. Addison–Wesley, 1995. URL `http://www.mcs.anl.gov/~itf/dbpp/`. Last accessed 10.09.09.

General Algebraic Modeling System GAMS. *http://www.GAMS.com/ 30.03.2011*, 2011.

A. M. Geoffrion. Lagrangean Relaxation for Integer Programming. *Mathematical Programming Study*, 2:82–114, 1974.

S. K. Gnanendran and J. K. Ho. Load balancing in the parallel optimization of block-angular linear programs. *Mathematical Programming: Series A and B*, 62: 41 – 67, 1993.

R. E. Gomory. Outline of an algorithm for integer solutions to linear programs. *Bulletin of the American Society*, 64:275–278, 1958.

A. Grama, G. Karypis, V. Kumar, and A. Gupta. *Introduction to Parallel Computing (2nd Edition)*. Addison Wesley, 2003.

I. Grossmann. *Presentation slide 14, http://200.13.98.241/ antonio/cursos/control/links/ResChallenges.pdf last axcccessed 30.03.2011*, 2011.

V. Gunnerud and B. Foss. Oil production optimization - a piecewise linear model, solved with two decomposition strategies. *Computers & Chemical Engineering Journal, doi:10.1016/j.compchemeng.2009.10.019*, 2009.

V. Gunnerud, B. Foss, B. Nygreen, R. Vestbø and N. C. Walberg. Dantzig-Wolfe decomposition for real-time optimization - applied to the Troll west oil rim. *AD-CHEM conference. July 12-15, Istanbul*, 2009.

V. Gunnerud, B. Foss, and E. Torgnes. Parallel Dantzig-Wolfe decomposition for real-time optimization - applied to a complex oil field. *Journal of Process Control, doi:10.1016/j.jprocont.2010.06.003*, 2010.

V. Gunnerud, B. Nygreen, K. I. M. McKinnon, and B. Foss. Oil production optimization solved by piecewise linearization in a branch & price framework. *Submitted to Computers & Operations Research Journal*, 2011.

J. Hall. Towards a practical parallelisation of the simplex method. *Comput Manag Sci*, 2008.

S. Handley-Schachler, C. McKie, and N. Quintero. New Mathematical Techniques for the Optimisation of Oil & Gas Production Systems. *Soc Petrol Eng J*, 2000.

J. Hauge and T. Horn. The challenge of operating and maintaining 115 subsea wells on the Troll field. *Offshore Technology Conference*, 2005.

V. Hepsø. When are we going to address organizational robustness and collaboration as something else than a residual factor? *Intelligent Energy Conference and Exhibition, 11-13 April 2006, Amsterdam, The Netherlands*, 2006.

J. K. Ho, T. C. Lee, and R. Sundarraj. Decomposition of linear programs using parallel computation. *Mathematical Programming*, 42:391–405, 1988.

J. K. Karlof. *Integer Programming: Theory and Practice*, chapter 4: Decomposition in Integer Linear Programming. CRC Press, 2006.

E. J. Kontoghiorghes. *Handbook of Parallel Computing and Statistics (Statistics, Textbooks and Monographs)*. Chapman & Hall/CRC, 2005.

V. D. Kosmidis, J. D. Perkins, and E. N. Pistikopoulos. A mixed integer optimization for the well scheduling problem on petroleum fields. *Computers & Chemical Engineering Journal*, 29:1523–1541, 2005.

L. S. Lasdon and D. Tabak. Optimization theory of large systems. *Math Program*, 20(1):303–326, December 1981.

M. E. Lübbecke and J. Desrosiers. Selected Topics in Column Generation. *Operations Research*, 53(6):1007–1023, 2005.

B. A. McCarl and T. H. Spreen. *Applied mathematical programming using algebraic systems*, chapter 15: Applied Integer Programming, pages 434–464. Texas A&M University, 2004.

J. E. Mitchell. *Encyclopedia of Optimization*, volume 2, chapter Integer programming: Branch-and-cut algorithms, pages 519–525. Kluwer Academic Press, 2001.

A. Mjaavatten, R. Aasheim, S. Saelid, and O. Gronning. A model for gas coning and rate-dependent gas/oil ratio in an oil-rim reservoir. *Society of Petroleum Engineers, SPE 102390*, 2006.

F. W. Molina. A Survey of Resource Directive Decomposition in Mathematical Programming. *ACM Comput. Surv.*, 11(2):95–104, 1979. doi: http://doi.acm.org/10.1145/356770.356774.

G. Moore. Progress in digital integrated electronics. In *Electron Devices Meeting, 1975 International*, volume 21, pages 11–13, 1975.

G. E. Moore. Cramming more components onto integrated circuits. *Electronics*, 38 (8):114–117, April 1965. URL `ftp://download.intel.com/museum/Moores_Law/Articles-Press_Releases/Gordon_Moore_1965_Article.pdf`. Last accesed 10.09.09.

J. A. Nelder and R. Mead. A simplex method for function minimization. *Comput. J.*, 7, 1965.

G. L. Nemhauser and L. A. Wolsey. *Integer and Combinatorial Optimization*. Wiley-Interscience, November 1999.

J. Nocedal and S. Wright. *Numerical Optimization*. Springer, 2006.

B. Nygreen, M. Christiansen, K. Haugen, T. Bjorkvoll, and Kristiansen. Modeling Norwegian petroleum production and transportation. *Annals of Operations Research*, 82:251–267, 1998.

M. Patriksson. A survey on the continuous nonlinear resource allocation problem. *Eur J Oper Res*, 185:1–46, 2008.

F. Pereira Pinto da Cunha e Alvelos. Branch–and–Price and Multicommodity Flows. *Universidade do Minho*, 2005.

Petroleum Experts. *http://www.petex.com/ 22.10.2010*, 2010.

R. L. Rardin. *Optimization In Operations Research*. Prentice Hall, 2000.

J. Rios and K. Ross. Massively parallel Dantzig-Wolfe decomposition applied to traffic flow scheduling. In *Proceedings of the AIAA Guidance, Navigation, and Control Conference*, 2009.

L. Saputelli, S. Mochizuki, L. Hutchins, R. Cramer, M. Anderson, and J. Muller.

Promoting real-time optimization of hydrocarbon production systems. *SPE 83978, Offshore Europe, 2-5 September 2003, Aberdeen, United Kingdom*, 2003.

L. Saputelli, M. Nikolaou, and M. Economides. Self-learning reservoir management. *SPE 84064, Journal SPE Reservoir Evaluation & Engineering*, 8:534–547, 2005.

L. Saputelli, H. Malki, J. Canelon, and M. Nikolaou. Managing risk and uncertainty in the visualization of production scenarios. *Paper SPE 107562 presented at the Hydrocarbon Economics and Evaluation Symposium, Dallas, 1 - 3 April. doi: 10.2118/107562-MS.*, 2007.

M. Savelsbergh. A Branch–and–Price Algorithm for the Generalized Assignment Problem. *Operations Research*, 45(6):831–841, 1997.

Schlumberger. Acting in Time to Make the Most of Hydrocarbon Resources. *Oilfield Review*, 17(4):4–13, 2005.

M. Tawarmalani and N. V. Sahinidis. *Convexification and global optimization in continuous and mixed integer nonlinear programming*. Kluwer academic publishers, 2002.

J. R. Tebboth. A Computational Study of Dantzig–Wolfe Decomposition. *PhD thesis, University of Buckingham*, 2001.

V. T'kindt, F. D. Croce, and C. Esswein. Revisiting branch and bound search strategies for machine scheduling problems. *J. of Scheduling*, 7(6):429–440, 2004.

E. Torgnes, V. Gunnerud, E. Hagem, M. Rönnqvist, and B. Foss. Parallel Dantzig-Wolfe decomposition of petroleum production allocation problems. *Accepted for the Journal of the Operational Research Society*, 2010.

N. L. Ulstein, B. Nygreen, and J. R. Sagli. Tactical planning of offshore petroleum production. *Eur J Oper Res*, 2005.

F. Vanderbeck. *Column Generation*, chapter 12: Implementing mixed integer column generation, pages 331–358. Springer US, 2006.

F. Vanderbeck and M. W. P. Savelsbergh. A generic view of Dantzig–Wolfe decomposition in mixed integer programming. *Oper Res Lett*, 34:296–306, 2006.

A. Wachter and L. T. Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming*, 106:25–57, 2005.

P. Wang. Development and applications of production optimization techniques for petroleum fields. *Ph. D. Dissertation, Standford University*, 2003.

P. Wang, M. Litvak, and K. Aziz. Optimization of production operations in petroleum fields. *This paper was prepared for presentation at the SPE Annual Technical Conference and Exhibition held in San Antonio, Texas, 29 September to 2 October, SPE 77658*, 2002.

M. R. Wartmann, V. Gunnerud, B. Foss, and E. Ydstie. Distributed optimization and control of offshore oil production: the intelligent platform. *FOCAPO conference, Jun 29 - Jul 2, Boston*, 2008.

H. P. Williams. *Model Solving in Mathematical Programming*. John Wiley & Sons, 1993.

H. P. Williams. *Model Building in Mathematical Programming*. Wiley, 2005.

P. Winker. *Optimization heuristics in econometrics: applications of threshold accepting*. John Wiley and Sons Ltd., Chichester, UK, 2001.

P. Winker and M. Gilli. Applications of optimization heuristics to estimation and modelling problems. *Computational Statistics & Data Analysis*, 47(2):211 – 223, 2004.

L. A. Wolsey. *Integer programming*. John Wiley & Sons, New York, 1998.