Master Thesis

Petroleum Production Planning Optimization
Applied to the StatoilHydro Offshore Oil and Gas Field Troll West

Eirik Hagem and Erlend Torgnes

Trondheim, June 10th, 2009

Norwegian University of Science and Technology
Faculty of Social Sciences and Technology Management
Department of Industrial Economics and Technology Management

NTNU
Norwegian University of Science and Technology
Preface

This master thesis, written within the field of managerial economics and operations research, is a culmination of the work to obtain a Master of Science for the Department of Industrial Economics and Technology Management at the Norwegian University of Science and Technology.

The work has combined the fields of petroleum engineering and operations research, making it possible for us to utilize our different technical backgrounds. This thesis can be seen in connection with the present Ph.D. research conducted by Vidar Gunnerud, and builds on our project thesis, written during the fall of 2008, along with the master theses of Gunnerud & Langvik (2007) and Vestbo & Walberg (2008).

This report is written in \LaTeX using the editor TeXnicCenter. The modeling was performed in Mosel using Xpress-Optimizer as commercial solver.

We want to thank our academic supervisor Professor Mikael Rönnqvist for constructive feedback and inspiring comments. We further want to thank Professor Bjarne Foss for providing us with problem specific knowledge. Ph.D. student Vidar Gunnerud has been of great help spending patient hours discussing problem formulations and solution methods with us. Lars Johan Sandvik also deserves thanks for making the hardware and software necessary to obtain real production data available to us. Finally, we want to extend our thanks to Kristin Hestetun, Vidar Alstad and personnel at StatoilHydro in Porsgrunn for helping us understand the production process at the Troll West field.

Trondheim, June 10th, 2009

Eirik Hagem Erlend Torgnes
Summary

This thesis evaluates different mathematical models for a petroleum production allocation problem and investigates the computational performance of a parallel Dantzig-Wolfe algorithm and Branch & Price applied to such problems. The StatoilHydro operated Troll West field is used as a case since the current allocation procedure used by StatoilHydro is believed to be sub-optimal. The objective when optimizing petroleum production allocation problems is to achieve an ideal allocation of limited resources, typically to maximize weekly oil production. Such problems require that a solution is provided within hours as the production planners should be able to evaluate and reoptimize iteratively. The class of allocation problems discussed in this thesis contains highly interacting well flows, making them computationally hard to solve. They also include nonlinear relations between variables which we handle through piecewise linearization. This converts the problem into a mixed integer linear program (MILP), making it possible to compute upper and lower bounds on the optimal objective function value.

Our work starts by developing a method for extracting production data from the Troll West field. This method is validated by StatoilHydro and the extracted data used to emulate a general petroleum production field. Based on this data, two different mathematical model formulations for a petroleum production system are evaluated. One model uses gas, oil, and water as flow variables while the other uses a different set of variables, utilizing certain fluid characteristics to try to reduce solution time. The latter shows promising results but is less user intuitive, something which StatoilHydro consider very important. The first model formulation is therefore chosen. We further show that a sufficiently accurate model for a full field size MILP requires a high number of integer variables. This leads to a very high solution time, which makes it necessary to reduce the resolution in the piecewise linearization in order to compare different solution methods.

To reduce the solution time, different decomposition methods are evaluated. Dantzig-Wolfe Decomposition is considered the most appropriate method due to the block-angular structure of the problem and the possibilities it creates for parallelization. Both sequential and parallel Dantzig-Wolfe strategies are evaluated. To be able to produce a solution within a specified gap of the global optima, a Branch & Price algorithm with branching on continuous production variables is also investigated.

A solution approach using a standard Branch&Bound algorithm based on the Simplex
method is unable to find a single integer solution for our allocation problem during 12 hours of solving. Our sequential Dantzig-Wolfe implementation solves the same problem in an average of about 600 seconds. The solution time is further reduced to less than 1/3 of this with an advanced parallel strategy. By using a heuristic in the final Dantzig-Wolfe iteration, we are able to produce good integer solutions to the problem. However, these are in most cases further from the optimal solution than what StatoilHydro deem acceptable. Our Branch & Price implementation with branching on the continuous production variables is able to reduce the distance to the optimal solution to within the acceptable limit.

The underlying assumptions of the model combined with the low resolution in the piecewise linearization separates our solution of the problem from a solution that can be directly implemented by petroleum production planners. Our development of a solution method that combines Branch & Price with an advanced parallel Dantzig-Wolfe strategy is nevertheless a step towards being able to create integrated software that can increase oil production for an offshore oil and gas field.
5.7 Model Intuitiveness and Generality ........................................ 82
5.8 Conclusion and Model Choice .............................................. 83

6 Solution Methods ......................................................................... 85
6.1 Problem Characteristics .......................................................... 85
6.2 Selection of Solution Method .................................................... 86
6.3 Dantzig-Wolfe Decomposition .................................................. 87
   6.3.1 Restricted Master Problem ............................................... 88
   6.3.2 Subproblem ................................................................. 90
   6.3.3 Column Management ..................................................... 91
   6.3.4 Termination Criteria ...................................................... 92
   6.3.5 Parallel Solving .......................................................... 93
   6.3.6 Obtaining an Integer Solution ......................................... 96
6.4 Branch & Price ........................................................................ 96
   6.4.1 Tree Search Strategies .................................................. 96
   6.4.2 Branching Rules in Literature ......................................... 97
   6.4.3 Choosing Branching Rule ............................................... 97

7 Implementation ........................................................................... 103
7.1 Implementation of Standard Solution Method .............................. 103
7.2 Implementation of Dantzig-Wolfe Decomposition ....................... 104
   7.2.1 Events in mmjobs ......................................................... 104
   7.2.2 Memory Drivers ........................................................ 105
   7.2.3 File Structure for the Dantzig-Wolfe Implementations ............ 105
   7.2.4 Procedures ............................................................... 106
   7.2.5 Compilation and Initialization of Submodels ....................... 106
   7.2.6 Controlling Information Flow ........................................ 107
7.3 Dantzig-Wolfe Algorithm ......................................................... 109
   7.3.1 Finding an Initial Solution ............................................. 110
   7.3.2 Column Management ................................................... 111
   7.3.3 Implementation of Sequential Dantzig-Wolfe ...................... 111
   7.3.4 Implementation of Parallel Dantzig-Wolfe ....................... 112
   7.3.5 Terminating the Dantzig-Wolfe Algorithm ......................... 114
   7.3.6 Heuristic to Find an Integer Solution ............................... 115
7.4 Implementation of Branch & Price .......................................... 115
   7.4.1 File Structure for the Branch & Price Implementation ............ 115
   7.4.2 Implementation Decisions ............................................. 116

8 Computational Setup .................................................................... 119
8.1 Testing Objectives ................................................................. 119
8.2 Initial Testing ...................................................................... 120
8.3 Test Data ............................................................................... 121
   8.3.1 Fixed Test Batch Parameters ...................................... 121
   8.3.2 Non-Fixed Test Batch Parameters ................................. 122

9
8.3.3 Solution Methods .............................................. 123
8.3.4 Test Batches .................................................. 123
8.4 Hardware and Software Specifications ............................ 125
8.5 Testing Limitations ............................................. 125

9 Results and Analysis ............................................. 127
  9.1 Validation ...................................................... 127
  9.2 Standard Solution Method ...................................... 128
  9.3 Dantzig-Wolfe: General ......................................... 128
  9.4 Extra Columns ................................................ 132
  9.5 Dantzig-Wolfe: Parallel ....................................... 134
  9.6 Branch & Price ............................................... 139
    9.6.1 General Branch & Price .................................. 139
    9.6.2 Cluster Branching Rule .................................. 143
    9.6.3 Phase Branching Rule .................................... 144
  9.7 Summary of Analyses .......................................... 146

10 Conclusion ....................................................... 149

11 Further Work .................................................... 151

A Model of Hagem & Torgnes ....................................... 161
  A.1 Declarations ................................................ 161
  A.2 Mathematical Model ......................................... 164

B Mosel code ........................................................ 173
# List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>Petroleum production planning horizons</td>
<td>19</td>
</tr>
<tr>
<td>1.2</td>
<td>Troll B and C platforms in the Troll West Region</td>
<td>22</td>
</tr>
<tr>
<td>1.3</td>
<td>Gas coning into a horizontal well in a thin oil column</td>
<td>23</td>
</tr>
<tr>
<td>1.4</td>
<td>Troll West well structure</td>
<td>23</td>
</tr>
<tr>
<td>1.5</td>
<td>Pipeline flow regimes</td>
<td>24</td>
</tr>
<tr>
<td>1.6</td>
<td>Software information flow</td>
<td>25</td>
</tr>
<tr>
<td>1.7</td>
<td>StatoilHydro production optimization software</td>
<td>27</td>
</tr>
<tr>
<td>2.1</td>
<td>Piecewise linear approximation</td>
<td>35</td>
</tr>
<tr>
<td>2.2</td>
<td>Illustration of primal block angular structure</td>
<td>38</td>
</tr>
<tr>
<td>2.3</td>
<td>Illustration of the dual block angular structure</td>
<td>40</td>
</tr>
<tr>
<td>5.1</td>
<td>Example of WPC data for single</td>
<td>66</td>
</tr>
<tr>
<td>5.2</td>
<td>Screenshot from GAP showing the topology in a single cluster</td>
<td>67</td>
</tr>
<tr>
<td>5.3</td>
<td>Example of pipeline pressure drop data</td>
<td>67</td>
</tr>
<tr>
<td>5.4</td>
<td>File structure of Xpress implementation</td>
<td>70</td>
</tr>
<tr>
<td>5.5</td>
<td>Communication between the parts of the testing application</td>
<td>71</td>
</tr>
<tr>
<td>5.6</td>
<td>Illustration of pressure in production system</td>
<td>73</td>
</tr>
<tr>
<td>5.7</td>
<td>Example of a test batch</td>
<td>75</td>
</tr>
<tr>
<td>5.8</td>
<td>Solution time and value in gas constrained test batch</td>
<td>77</td>
</tr>
<tr>
<td>5.9</td>
<td>Solution time and value in gas and water constrained test batch</td>
<td>78</td>
</tr>
<tr>
<td>5.10</td>
<td>Plot of real pressure drop data</td>
<td>79</td>
</tr>
<tr>
<td>5.11</td>
<td>Plot of artificial pressure drop data</td>
<td>80</td>
</tr>
<tr>
<td>5.12</td>
<td>Piecewise linearization of WCL</td>
<td>82</td>
</tr>
<tr>
<td>6.1</td>
<td>Illustration of the restricted master problem</td>
<td>89</td>
</tr>
<tr>
<td>6.2</td>
<td>Illustration of sequential solving of subproblems</td>
<td>91</td>
</tr>
<tr>
<td>6.3</td>
<td>Illustration of parallel solving of subproblems</td>
<td>93</td>
</tr>
<tr>
<td>6.4</td>
<td>First iteration of the Accelerated feedback strategy</td>
<td>95</td>
</tr>
<tr>
<td>6.5</td>
<td>B&amp;P tree with root node and the two first child nodes</td>
<td>99</td>
</tr>
<tr>
<td>6.6</td>
<td>Interpolated solution between two columns and a new column</td>
<td>100</td>
</tr>
<tr>
<td>6.7</td>
<td>Illustration of an interpolated solution and how bounds are set</td>
<td>100</td>
</tr>
<tr>
<td>7.1</td>
<td>File structure for implementation of DWD</td>
<td>106</td>
</tr>
<tr>
<td>Section</td>
<td>Title</td>
<td>Page</td>
</tr>
<tr>
<td>---------</td>
<td>----------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>7.2</td>
<td>Communication between master-and submodels</td>
<td>109</td>
</tr>
<tr>
<td>7.3</td>
<td>Flow chart for the implemented Dantzig-Wolfe algorithm</td>
<td>110</td>
</tr>
<tr>
<td>7.4</td>
<td>Flow chart of the Accelerated feedback parallel strategy</td>
<td>114</td>
</tr>
<tr>
<td>7.5</td>
<td>File structure for the B&amp;P implementation</td>
<td>116</td>
</tr>
<tr>
<td>7.6</td>
<td>Flow chart for the implemented B&amp;P algorithm</td>
<td>118</td>
</tr>
<tr>
<td>8.1</td>
<td>Single cluster topology</td>
<td>122</td>
</tr>
<tr>
<td>9.1</td>
<td>Optimality gap vs. sum of limits on water and gas</td>
<td>129</td>
</tr>
<tr>
<td>9.2</td>
<td>Development of LP optimality gap and solution time</td>
<td>131</td>
</tr>
<tr>
<td>9.3</td>
<td>Solution time for the different parallel DWD-strategies</td>
<td>136</td>
</tr>
<tr>
<td>9.4</td>
<td>Development in upper and lower bound for B&amp;P</td>
<td>140</td>
</tr>
<tr>
<td>9.5</td>
<td>Solution time B&amp;P normal vs B&amp;P random cluster</td>
<td>143</td>
</tr>
<tr>
<td>9.6</td>
<td>Solution time B&amp;P normal vs B&amp;P random phase</td>
<td>145</td>
</tr>
</tbody>
</table>
Nomenclature

B&B - Branch & Bound
B&C - Branch & Cut
B&P - Branch & Price
BD - Benders Decomposition
DWD - Dantzig–Wolfe Decomposition
GAP - General Allocation Package
GOR - Gas/Oil–Ratio
GORM - Gas/Oil–Ratio Model
IP - Integer Program
IPR - Inflow Performance Ratio
PDP - Production Data Portal
LD - Lagrange Decomposition
LP - Linear Program
LR - Lagrange Relaxation
MILP - Mixed Integer Linear Program
MINLP - Mixed Integer Nonlinear Program
MIP - Mixed Integer Program
NLP - Nonlinear Program
RMP - Restricted Master Problem
RTPO - Real Time Production Optimization
SOS2 - Special Ordered Sets of type 2
SQP - Sequential Quadratic Programming
TB - Test Batch
VLP - Vertical Lift Performance
WCL - Water Cut Linking constraint
WPC - Well Performance Curve
Chapter 1

Introduction

This chapter presents the petroleum production allocation problem. Using the production system at the Troll West field as a case, it explains how the integrated oil and gas company StatoilHydro currently plans the production at the field. The previous work performed on formulating and solving mathematical programming problems for such problems will also be covered, together with an explanation of why there is a continued interest in working on these problems.

The production system at Troll West consists of a platform and a set of wells grouped in clusters, each connected to a first stage separator on the platform through two pipelines. Ideal allocation of available gas capacity to wells in the production clusters is the core consideration when trying to maximize production at Troll West. The reason for this is that the total production is currently constrained by the gas handling capacity at the platform. StatoilHydro has stated that the objective of their production planning is currently to maximize weekly oil production. However, extracting oil efficiently at the Troll West field is complicated as the oil is located in thin layers beneath a thick gas cap - a feature that causes the wells to have a highly rate dependent GOR (gas/oil ratio). The characteristics of Troll West resemble the characteristics of general petroleum production fields.

The work to create and solve a mathematical model for the production system at Troll West was initiated in response to the absence of integrated software that optimized allocation of gas to the entire field. Currently, StatoilHydro production engineers allocate gas to each cluster based on experience and sensitivity analysis. This practice 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 StatoilHydro software and commercial optimization software. Although gas handling capacity is currently the most important constraint on production at Troll West, StatoilHydro has indicated that the water handling capacity is likely to be an active constraint in the future. The model and solution approaches presented in this thesis hold the potential for providing better solutions to problems in both these areas and also for other comparable fields. Similar to the goal of StatoilHydro,
the objective of the production allocation problems we will investigate is to maximize the weekly oil production.

Some general requirements apply to optimization models for the production allocation problem. A production planner should have the possibility of evaluating a solution produced by the model and then adjust the handling capacity limits and reoptimize. The time to obtain a solution should therefore not exceed a few hours. StatoilHydro have stated that they will aim to implement a solution produced by the optimization model if the oil production is higher than the present production at Troll West. In addition, they have indicated that we should be able to guarantee a solution value within a specified limit of the global optimum for our model. These requirements are also believed to be general requirements regarding petroleum production allocation problems.

This thesis builds on the work of Gunnerud & Langvik in Gunnerud and Langvik (2007). Gunnerud & Langvik began by formulating a Mixed Integer Nonlinear Program (MINLP) for the production optimization problem at Troll West. Using piecewise linearization they linearized their model, obtaining a Mixed Integer Linear Program (MILP) which they solved using Lagrangian Decomposition (LD). This model formulation results in a very large MILP that is computationally demanding to solve due to the fact that the piecewise linearization introduces a large number of variables to handle the nonlinearities of the original MINLP. In Vestbø and Walberg (2008) the work in Gunnerud and Langvik (2007) was continued. Vestbø & Walberg concentrated their efforts on investigating different decomposition methods to solve a MILP of the type presented in Gunnerud and Langvik (2007). The problem formulation that was introduced in Gunnerud and Langvik (2007) was therefore kept largely unchanged in their implementation. The work of Vestbø & Walberg was continued in Hagem and Torgnes (2008) where the potential of a new model formulation for a production allocation problem was explored. Their aim was to create a model with the same solution accuracy, but that was less computationally demanding to solve.

The first objective of this thesis is closely related to the work performed in Hagem and Torgnes (2008). The model presented in Hagem and Torgnes (2008) will be compared with the model in Vestbø and Walberg (2008) with regards to solution time and accuracy in solution value. Hagem & Torgnes performed tests to uncover this based on artificial production data but were unable to reach a conclusion regarding which model performed better. They stated that if they were to choose a single model to continue work on, more testing of the two different models should be carried out using real production data. The first objective of this thesis is therefore to try to overcome previously met difficulties of obtaining real production data and use this to decide which model to proceed with.

Irrespective of the particular model that is implemented, the problem is likely to be computationally demanding to solve. Thus, it is interesting to investigate solution approaches that can utilize the special block structure of production allocation problems to reduce the solution time. In addition to LD, Benders Decomposition (BD) and Dantzig-Wolfe Decomposition (DWD) are investigated in Vestbø and Walberg (2008). Although the
results that Vestbø & Walberg obtained were not promising with respect to using BD as a solution method, they did indicate that both LD and DWD have the potential to significantly reduce the solution time. As a continuation of this work, the second objective of this thesis is to develop a decomposition method that makes use of the problem’s structure to enable parts of the problem to be solved in parallel. Using a decomposition method and algorithm to solve a MILP does not always provide an integer solution. As the problem contains many integer variables, a natural extension of the second objective is therefore to investigate algorithms that make it possible to obtain solutions that satisfy the integer requirements of the problem.

1.1 Petroleum Production Optimization

The petroleum industry is a heavy user of operations research, and optimization techniques have essentially been applied in all parts of the petroleum industry value chain. A good starting point for readers interested in optimization methods for allocation problems in the oil and gas industry is the Ph.D. thesis Wang (2003) by Wang. Another good starting point is the article (Bieker et al., 2006). The article by Bieker gives an overview of the information flow necessary for a petroleum production optimization problem. However, the main part of the article is a survey of articles and methods for optimization of well allocation, gas lift and gas/water injection.

1.1.1 Production Allocation Problems

Production allocation problems are problems in which determination of optimal production rates, lift gas rates and well connections, subject to multiple constraints, are the central decisions. In his Ph.D. thesis, Wang focuses on the value chain from the platform and further upstream. He starts by looking at efficient ways of simulating the production system including the complex multiphase flow in the pipelines. He then continues by investigating solution methods for the modeled system. He notes that there is a big difference in solving problems where well flows interact, compared to problems in which they do not. When well interaction is insignificant, the performance of each well can be evaluated individually and the problem solved as a separable programming problem which objective and constraint functions are sums of functions of one variable (Wang et al., 2002). Specifically, when well interaction can be ignored, a number representing the performance of each well can be calculated and the wells prioritized according to this number until the production capacity is reached.

Production systems where well flow interaction is significant require a different formulation and a different solution approach. Wang suggests solving problems like these using sequential quadratic programming (SQP) - a solution approach that was also explored for petroleum production problems in Dueñas Díez et al. (2005) and in Wang et al. (2002). A brief introduction to SQP is given in section 2.1.
The optimization procedures developed by Wang were coupled with models for multiphase flow in reservoirs and pipelines and tested on both short and long term production optimization problems. The Prudhoe Bay field in Alaska served as the short term test case, while two newly developed fields in the Gulf of Mexico served as long term reservoir development studies. The Prudhoe Bay field is a mature field and its production is constrained by gas and water handling capacity, similar to the Troll West field. Wang concludes in his thesis that the field model and solution methods developed and tested on synthetic and real cases have distinct advantages compared to conventional, often more heuristic based, approaches for petroleum production optimization. Among his recommendations for further study is a suggestion that more work should be done on how to efficiently solve problems for which well flow interaction is important.

In Kosmidis et al. (2005), they present a MINLP with the objective function to maximize the daily profit from an offshore oil and gas field. More specifically, the objective of Kosmidis, Perkins and Pistikopolous paper was to develop a formal optimization model and algorithm that considers the whole production system, which they comment, up until their time of writing, had not been done. The decisions variables central to their problem include the operational status of wells (open or closed), the allocation of a well to a specific manifold, determination of well oil rates and allocation of gas to gas lift wells. An important feature of the problem they try to solve is that the interacting flow effect between wells is significant, something which makes separable programming an impossible solution approach. Instead, the MINLP is solved using a sequential linear programming (SLP) procedure (Zhang et al., 1985).

In Güyagüler and Byer (2008) an optimization procedure that uses a MILP to model a petroleum production allocation problem is introduced. The nonlinear flow rate behavior of the multiphase flow in the pipelines is in the suggested solution approach linearized using a piecewise linear function. To optimize the fit of the piecewise linear function, the position and number of breakpoints is decided through solving a least squares approximation. Güyagüler and Byer (2008) suggests a new approach to make the proposed optimization method more robust. They claim that the reason optimization has not received more widespread active use in the petroleum industry is that more traditional rule based methods are better at consistently providing a solution. To make the optimization framework they introduce more robust, the authors suggest using an objective function that allows a solution to break certain constraints, at the cost of incurring a penalty. This allows the procedure to always suggest a solution, and makes it possible to set a price on breaking a constraint.

1.1.2 Planning Horizon

When solving optimization problems in the petroleum industry it is important to be aware of the assumptions and limitations that follow the time horizon of the problem being solved. With this in mind, we look at the different planning horizons that are
used in the petroleum industry and place the production allocation problem we are investigating within the existing framework.

Different frameworks that divide planning in petroleum production have been introduced, see Ulstein et al. (2005), Foss et al. (2009) or Schlumberger (2005). The name of the different planning levels are often different, but the levels are generally divided along the same lines with regards to time horizon and scale. Here, we will use the terms Strategic Decisions, Field Optimization, Real Time Production Optimization and Operator Optimization to represent the planning at the different levels. The length of the planning horizon for each of these levels is equivalent to the hierarchy in figure 1.1.

![Figure 1.1: Planning horizons for different aspects of petroleum production, (Schlumberger, 2005)](image)

**Strategic Decisions**

Decisions with a time horizon of two years and longer are here classified as strategic decisions. These decisions address long term planning and concern the goals and visions of a project. An example of several petroleum optimization problems with a very long time horizon is presented in Nygreen et al. (1998). At the strategic level we find decisions regarding field development such as choice of technology and export options, investment strategies and reservoir recovery optimization. Through the planning at this level the bounds that should ensure long term optimality are determined. The bounds of planning problems with shorter time horizons, like pipe dimensions and handling capacity, are also set in this period. An example of a MILP for a long time frame is presented in Haugland et al. (1988). This paper presents a model for the early evaluation of petroleum fields that can be used as decision support when making decisions concerning first stage separator capacity, a drilling program or the production.
Field Optimization

At the next level, we find decisions with a time horizon of about three months to two years. In this range, decisions regarding field optimization are taken. This includes decisions to optimize reservoir drainage and scheduling of well interventions. As mentioned earlier, an example of the first is how StatoilHydro currently focus their effort at Troll West on oil rather than gas extraction to maintain a high reservoir pressure. Depending on the life cycle of a field, decisions at this level might also include a drilling program.

Real Time Production Optimization

Real Time Production Optimization concern decisions with a time horizon of days to a few months. Foss et al. (2009) suggests defining mathematical optimization at this level as Real Time Production Optimization (RTPO). The system layout together with the reservoir properties are considered fixed at this stage. Decision variables at this stage include production and possibly gas injection rates and routing of well streams. The objective at this stage is often related to maximizing daily or weekly production rates. The problem and models investigated in this thesis belong to this planning horizon.

Operator Optimization

At the lowest level we find the planning horizon that stretches from a few seconds to a day. Schlumberger (2005) refers to this as operator optimization. In this short time frame, the operator makes adjustments continuously. If decisions at this stage are updated automatically, we call it a closed loop, while if they are done with operator approval, we call it an open loop. Decisions at this stage can be well control decisions like increasing drilling mud weight to prevent a blow out or decreasing drilling mud weight to avoid formation fracturing and loss of well control.

Multiple Horizon Planning

Some research has also been performed on combining planning in several time periods to obtain an integrated model. Saputelli et al. (2002) suggest using a hierarchy of time scales to separate the levels over which decision making is performed to render a complex problem solvable. The multistage structure they introduce combines optimization levels such as multilateral selection, well location and portfolio optimization. A different approach to handle multiple time horizons is to use stochastic programming and a multistage model.

A success story with respect to developing and implementing a multiple horizon planning tool is the decision support tool GasOpt, developed by SINTEF (Rømo et al., 2009). The tool is based on a single-period steady-state model of the gas network on the Norwegian...
Continental Shelf and is used for short-, medium and long-term planning. Among the decisions that the tool supports is planning and handling of downtime and maintenance in addition to more long term decisions on when and where to expand the network.

**Scope of Current Optimization Problems**

The production plans currently used at Troll West are created by StatoilHydro engineers on a weekly basis. They concern decisions at the RTPO level and the reservoir conditions are therefore considered as constant over the planning period. StatoilHydro have stated that the objective of creating these production plans is to maximize the weekly oil production. Targets in the plans are set for the weekly production of oil, gas and water from each well. However, the production engineers at Troll West are free to make changes if they deem it necessary. In addition, a production technology group onshore together with the production engineers at the platforms discuss daily measures that they both have proposed (Dueñas Díez, Marta., 2008a).

The models presented in Gunnerud and Langvik (2007) and Vestbø and Walberg (2008) both concern decisions with a time horizon that place them within RTPO. The reason they look at decisions at this stage is partly that the models and solutions are visioned as replacement tools for the production planning methods and software currently used by StatoilHydro.

The new model that was introduced in Hagem and Torgnes (2008) also concerned planning at the RTPO level, as they were looking for ways to improve on the MILP presented in Vestbø and Walberg (2008). The assumptions and limitations that follow the time horizon of the models presented in Hagem and Torgnes (2008) and Vestbø and Walberg (2008) are covered in chapter 3.

**1.2 The Troll Oil and Gas Field**

The offshore oil and gas field Troll West is located about 80 km west of the west coast of Norway. The Troll West field is regarded by many as the cornerstone of Norwegian gas production since the field contains about 60% of the total gas reserves on the Norwegian Continental Shelf. The three platforms at the field, Troll A, B and C, are currently all operated by StatoilHydro. The focus of the applied aspect of this thesis is the oil extraction at the field, which is controlled from the Troll B and C platforms, shown in figure 1.2. StatoilHydro hopes to continue to produce oil at Troll West until 2030, and they expect gas production to continue until about 2060 (StatoilHydro, n.d.).
The water depth at the Troll West field is around 300m and the oil in the region is located about 1600m below the surface. The special feature of the Troll West field is the thin layers of oil, 13-27m high, located beneath a thick gas cap. This is a characteristic of what are referred to as oil rim fields. The effect of this feature is a highly rate dependent gas/oil-ratio for many of the wells at the Troll West. The thin oil layer and the thick gas cap above, lead to a phenomenon called gas coning, which is illustrated in figure 1.3 from Hauge and Horn (2005). When extracting oil from the thin oil column the thickness of the oil layer will be reduced in a small area around the horizontal well, creating a cone shape around the well. This happens partly also because of the high permeability of the reservoir. If the thickness of the oil layer is reduced so much that gas breaks through to the well the GOR changes dramatically. As a result of this and the overall gas and water handling constraints, the interaction between well flows become significant, and thus the problem more challenging to solve, as mentioned in section 1.1. For more on modeling of gas coning see Hauge and Horn (2005) and Konieczek (1990).
1.2. **THE TROLL OIL AND GAS FIELD**

The oil at Troll West is extracted through in excess of 110 horizontally drilled wells. Each are together with at most three other wells connected to a manifold at the sea floor. Each manifold is again connected to a first stage separator through two identical pipelines - a production line and test line. Under normal production both of these pipelines are used. The fact that there are two identical pipelines allows the wells to be routed to either of the two lines. Often high producing wells are routed to one pipeline and low producing wells to the other. At Troll West, two and two manifolds are connected to the separator through the same pipelines and are said to be a part of the same cluster. Troll B has 9 clusters named from D to L, and Troll C has 8 clusters named M, N, O, P, Q, S, X and Y. The figure in 1.4 illustrates the layout of a single cluster.
the flow rate of the different phases, the pipe characteristics, and the angle of the pipe (Nydal, 2007). An illustration of the large differences between flow regimes is shown in figure 1.5.

Figure 1.5: Different flow regimes for different pipe angles for multiphase flow, $\theta$, (Gomez et al., 2000)

StatoilHydro use what are referred to as Inflow Performance (IPR) curves to give information about the flow rates into the wellbore for different wellhead pressures. These curves are together with what are commonly called Vertical Lift Performance (VLP) curves typically used to model the interaction between the reservoir and a well (Dueñas Díez et al., 2005). The VLP curves account for the hydrodynamic and friction effects in the well (Dueñas Díez et al., 2005). In Gunnerud and Langvik (2007), the VLP and IPR curves are combined into a single curve, a Well Performance Curve (WPC), which is used to represent the performance of a well. As the WPCs describe the multiphase flow mentioned above, they are highly nonlinear.

Modeling the pressure drop in pipelines is very complex and in reality a function of a number of variables including among others temperature, absolute pressure, and the flow rates of the phases of fluid. Models exist that produce a pressure drop given a certain flow regime. However, the flow regime in the pipelines at Troll West is not constant and creating a correct pressure drop function is very challenging (Hauge and Horn, 2005). In the prior works Gunnerud and Langvik (2007), Vestbe and Walberg (2008) and Hagem and Torgnes (2008) the pressure drop was modeled with an approximate nonlinear pressure drop function of three variables. This function was then linearized using piecewise linearization. As we illustrate in chapter 2, the linearization of this nonlinear function is the biggest contributor to the size of the problem.

Water and gas at the Troll West field are currently considered byproducts of oil production. The reason gas is also considered a byproduct is that StatoilHydro wants to postpone the gas production until they have extracted as much oil as possible. This is because it is easier to extract the oil if they can maintain a high reservoir pressure.
1.3 Troll West Production Optimization

The combination of a huge sub-sea network, limited gas and water handling capacity with a GOR for several wells that is highly rate dependent, makes the production optimization at Troll West complex. Gas have to be allocated to a large set of wells where the production rates of the different wells strongly vary with changes in their allocated gas rate. The fact that there are two pipelines from the manifolds at sea floor and up to the separator introduces binary routing decision variables to the problem. This routing decision is important because the pressure, and thereby production rates of a well, is in part determined by the other wells that are routed to the same pipeline. Introducing these binary variables transforms the original problem from a nonlinear program (NLP) to a mixed integer nonlinear program (MINLP), which is generally considered as one of the most challenging optimization problem classes.

During the last two decades, petroleum companies have begun to embrace the use of optimization software for various parts of their operations. Currently, there are several companies that specialize in creating software claiming to cover all parts of petroleum production (Petroleum Experts Ltd., n.d.b) (Weatherford International Ltd., n.d.). The production optimization problem at Troll West is currently handled by StatoilHydro using a combination of in-house and commercial software. In figure 1.6, which is a simplified diagram from Hanto and Dueñas Díez (2008), we see an overview of the communication between the software that StatoilHydro use to optimize oil production at Troll West. The whole structure of software is quite complex, so only a brief explanation of the main parts will be given. Though the combination of software that StatoilHydro use is unique, the functions and information that the software provide are generally necessary for petroleum production optimization.

![Simple Overview](image)

Figure 1.6: Simple overview of software information flow. (Hanto and Dueñas Díez, 2008)
CHAPTER 1. INTRODUCTION

1.3.1 Gas Oil Rate Model

The Gas Oil Rate Model (GORM) is a well/reservoir simulator developed in-house by StatoilHydro. It was created to model gas coning in the horizontal wells at the Troll West field (Mjaavatten et al., 2006). The Inflow Performance Curves (IPR) are produced from GORM. They contain information on the flow rates into the well of water, gas and oil, given the pressure in the well (Hauge and Horn, 2005). StatoilHydro has tuned GORM to generate one model for each individual well at Troll West.

1.3.2 Production Data Portal

The Production Data Portal (PDP) is a web portal where all historic and current production data is stored. Data is logged every fourth second in the system and the system can be used to compare planned and historic production rates (Gunnerud and Langvik, 2007).

1.3.3 General Allocation Package

The General Allocation Package (GAP) is the software that StatoilHydro utilize to optimize well routing and oil production for single clusters. It is a part of the IMP software suite by Petroleum Experts and was developed as a multiphase optimizer of well networks. GAP communicates with several other software packages, among them GORM and the database IP21. From IP21, which is also called the weekplan, GAP obtains the latest information of how the production system has been run (Hanto and Dueñas Díez, 2008). The GAP optimization engine utilizes algorithms for both linear and nonlinear problems (Petroleum Experts Ltd., n.d.a).

1.3.4 Scheduler

Scheduler has been developed by StatoilHydro to manage runs of time-demanding tasks. It automates the passing of information between several of the programs that StatoilHydro uses in short-term production optimization. It is through Scheduler that GAP is used to find the optimal well routing. By calling GAP repeatedly to find production rates for all possible routings, Scheduler finds the optimal routing using a complete enumeration. GARM, shown in figure 1.6, is used to visualize results from Scheduler, as Scheduler does not contain a visualization feature (Hanto and Dueñas Díez, 2008). Scheduler also has a feature that allows it to call GAP to generate sensitivity curves for the production from each cluster at different gas levels.
1.3.5 Software Integration

The current practice at StatoilHydro combines all of the above software packages to optimize the production at Troll West. In essence, GORM generates information about the performance of wells, GAP optimizes each cluster of wells and Scheduler finds the optimal routing. Figure 1.7, which is a modified figure from a presentation by Marta Dueñas Díez at the SIAM 08 Conference on Optimization, shows an overview of the different levels where each software is used. Two software suites shown in the figure that we have not mentioned, MaxPro and Flowmanager developed by the FMC Corporation, are also in use at the Troll West field. However, the connection GORM/GAP/Scheduler has been in use longer and undergone more testing compared GORM/MaxPro/Scheduler.

![Figure 1.7](image)

Figure 1.7: Overview of the software used by StatoilHydro for production optimization, (Dueñas Díez, Marta., 2008)

Under the supervision of Marta Dueñas Díez, students on a summer project at StatoilHydro in 2008, including one of the authors of this thesis, looked at extending the current software suite. The expansions looked at the potential of including a program that analyzes sensitivity curves for clusters from GAP to optimize allocation of gas over all clusters. The feature developed is today used as an off line decision support tool. The allocation of gas to each cluster is done by StatoilHydro engineers based on experience and by using GAP/Scheduler/GARM as decision support.
1.4 Previous Literature on the Troll West Field

As mentioned earlier, this thesis is closely related to the master thesis "Production Planning Optimization for the Troll C field" by Gunnerud & Langvik in 2007, the master thesis "Decomposition of optimization problems in petroleum production - applied to the Troll C field" by Vestbø & Walberg in 2008 and the project thesis "Optimization problems in petroleum production – Applied at Troll West" by Hagem&Torgnes. For completeness we will therefore give a brief description of the work done and the results obtained in these theses. The assumptions and limitations of these optimization models will be discussed further in chapter 3, where we introduce the various models. The work done in the mentioned master theses has been the basis for two articles, Gunnerud, Foss, Nygreen, Vestbo and Walberg (2009) and Foss et al. (2009), that will be published in 2009.

1.4.1 Gunnerud & Langvik [2007]

In Gunnerud and Langvik (2007) the authors look at a new way of maximizing oil production at the Troll C platform. As explained earlier, the special properties of the Troll West field together with the size of the field makes production optimization for the entire field challenging. Since the method used by StatoilHydro need an expert user to allocate gas to each cluster, Gunnerud & Langvik saw a potential in developing a new solution approach to optimize the entire field.

The problem formulation that Gunnerud & Langvik introduce has a planning horizon of one week. As mentioned earlier, this is the same planning horizon currently used by production engineers at the field. Since Gunnerud & Langvik wanted to compare their answers with the current best solution, this was an important assumption.

The MINLP by Kosmidis et al. (2005) that we mentioned in section 1.1 formed a starting point for the model formulation in the thesis of Gunnerud & Langvik. Similarly to Kosmidis, Gunnerud & Langvik’s initial modeling of the production system at Troll West results in a nonlinear model. Using Special Order Sets of type 2 (SOS2), which are covered in chapter 2, they linearize the nonlinear parts of the model using a piecewise linear function, turning the model into a MILP. To obtain a better distribution of the data points used in the piecewise linearization they develop an algorithm which they name the Dynamic Interpolation algorithm (DInA). The DInA is an iterative algorithm that aims to increase the accuracy of the piecewise linearization.

By linearizing their MINLP, they are able to make use of solution methods specialized for linear problems, such as Simplex, decomposition methods and B&B. However, a linearized model results in a much larger problem. To solve the MILP they created, Gunnerud & Langvik develop an algorithm which they term the Total Optimization Algorithm (TOA). At the core of this algorithm we find a Lagrangian relaxation of the gas handling constraint. As the gas handling constraint is the only constraint that ties
the clusters together to a field optimization problem, relaxing this constraint decomposes the problem into one subproblem for each cluster. They argue that this leads to an intuitive decomposition, which makes the algorithm easier to understand and accept for production engineers. To find the Lagrange multiplier for the gas handling constraint they make use of the Subgradient optimization method, explained in Beasley (1993).

The results that Gunnerud & Langvik obtained on artificial production data were promising and the solution time turned out to be shorter than they expected. However, they encountered problems when testing the TOA on real data. This was mainly due to what they described as a numerical instability that resulted in the Xpress-Optimizer not solving the LP-relaxation in some of the nodes in the B&B tree. After making changes to the TOA they were able to obtain acceptable results, though with a larger gap between the upper and lower bound in some cases.

Though the results obtained from tests on real data proved inferior to the current solution, Gunnerud and Langvik (2007) conclude that the TOA that they developed show promising results as they succeeded in developing a method that automatically allocates gas to the different clusters. They further conclude that the method has potential if further improvements are made.

1.4.2 Vestbø & Walberg [2008]

In Vestbø and Walberg (2008), the work that Gunnerud & Langvik started is continued and expanded on. Vestbø & Walberg reformulate Gunnerud & Langvik’s MILP and investigate three different decomposition methods: BD, LD and DWD. They also saw a potential in combining these methods. However, they felt it natural to investigate the different methods separately first.

Few changes are made from Gunnerud & Langvik’s MILP to the model presented in Vestbø and Walberg (2008). The most important change that Vestbø & Walberg introduce is the division set of wells into two sets; wells that can be closed (have a production rate of zero), and wells that cannot be closed. They also make changes to the interpolation of the WPCs so the production rates have to be above a certain limit for the wells that are open. They introduced this change after discussions with StatoilHydro about the solutions that Gunnerud & Langvik obtained. During these discussions it was uncovered that it was undesirable to allow a solution where the production was below a certain limit for some of the wells. The reasons for this were both operational and technical considerations, among them, concerns regarding the difficulties of restarting production from closed wells. The other changes that Vestbø & Walberg made to Gunnerud & Langvik’s MILP are more cosmetic and are introduced to make it easier for the reader to understand the model.

Vestbø & Walberg report that the standard Branch&Bound algorithm based on the Simplex method is a potential method for solving medium sized problem versions of the full Troll West problem. However, for a problem of a scale equivalent to the Troll
West field, they did not obtain a satisfactory solution using this optimization procedure in Xpress. Therefore, they conclude that a different approach, like a decomposition method, is necessary to solve their MILP within reasonable time.

By relaxing the gas and water handling constraints, the production optimization problem naturally decomposes into one problem for each cluster. However, using Benders Decomposition does not make it possible to utilize this model feature. As a result of this, Vestbø & Walberg suspect that Benders Decomposition might not be a good solution approach for their MILP. During the implementation they encounter some challenges. To start Benders Decomposition, an initial solution is needed. Finding this prove difficult and they are unsuccessful in finding a good way of doing this manually or by construction of a heuristic. This, together with the structure of the problem, lead Vestbø & Walberg to the conclusion that Benders Decomposition is an inappropriate solution method for this problem.

When implementing Lagrangian Decomposition Vestbø & Walberg, as Gunnerud & Langvik, decide to relax the handling constraints as this creates an intuitive decomposition of the model. They see that the solution they obtained is highly influenced by how well they are able to tune the Lagrange multiplier. However, after proper tuning they obtain good results. When testing on different datasets they see that a retuning of the Lagrange multipliers is needed for new input data. Determining the Lagrange multiplier requires user knowledge about the Subgradient method in addition to knowledge of the underlying problem. As StatoilHydro has stressed the importance of a robust solution method, Vestbø & Walberg therefore conclude that the implemented Lagrangian decomposition had less potential than the last decomposition method they implemented.

The method that is shown the greatest attention in the thesis of Vestbø & Walberg is Dantzig-Wolfe Decomposition (DWD). Vestbø & Walberg argue that DWD, which often works well on problems with a block angular structure with few common constraints, is well suited for their MILP. The results that they obtain confirm this belief. DWD provides solutions of good quality and with short solution times for all the test cases except for cases where water handling capacity is an active constraint. On this test case, they are unable to obtain convergence using DWD or LD. Vestbø & Walberg state that the main disadvantage by applying DWD to a MILP is the problems related to generating appropriate columns and finding integer solutions. They suggest that this issue might be better handled by implementing a Branch&Price algorithm to solve the MILP.

Vestbø & Walberg conclude in their thesis that DWD is the most relevant decomposition method of the methods they test. They base their conclusion on the finding that the DWD, through their test cases, proved to be the most robust and generic method in addition to having a short solution time. They end by recommending that further research should be performed on DWD and Branch & Price. They also suggest that there might be a potential in using parallel processing given the structure of the problem, which is structured so that it can decomposed in to subproblems that can be solved in parallel.
1.4. PREVIOUS LITERATURE ON THE TROLL WEST FIELD

1.4.3 Hagem & Torgnes [2008]

In Hagem and Torgnes (2008) the work done in Vestbø and Walberg (2008) and Gunnerud and Langvik (2007) is continued. A new model formulation is introduced to try to achieve the same solution accuracy with fewer interpolation points in the piecewise linearization of the pipeline pressure drop. The new model attempts to exploit the similarities in fluid characteristics of water and oil to simplify this linearization. Together with the original model, the new model in the report is applied to a large set of test cases, each consisting of a single cluster. The results obtained show that solution quality, and especially solution time, for both models depended significantly on the level of the handling capacity. Further, the results show that for both models only a low resolution in the piecewise linearization is necessary to obtain an accurate solution for the test cases and that given a certain accuracy, the model developed by Vestbø & Walberg produce solutions faster. The new model is shown to have a shorter solution time when using a high resolution in the linearization, which StatoilHydro indicated is likely required when using real data. The observations are explained through an analysis of the function used to generate the artificial production data. The analysis is based on theory on linearization of nonlinear models, including chains of SOS2. Given the small differences between the two tested models Hagem and Torgnes (2008) recommend that further testing should be performed using real pressure drop data to determine which model performs better.
Chapter 2

Theory

This chapter presents the most relevant theory for optimization problems of the type that form the basis of this report. As mentioned in chapter 1, the problems that we are interested in originally have the form of a mixed integer nonlinear program (MINLP), but have a special structure that can be utilized when solving the problem. The theory that is presented in section 2.1 relates to how to solve a MINLP, including how to reformulate a MINLP to a mixed integer linear program (MILP). The theory presented in 2.2 relates to the first objective of this thesis and gives an introduction on how to evaluate a mixed integer linear model. The later sections present different solutions methods for MILPs. The chosen solution method is covered more thoroughly in chapter 6.

2.1 Mixed Integer Nonlinear Programs

MINLPs are generally considered difficult to solve. The reason for this is that they combine the difficulties of their problem subclasses: mixed integer programs (MIP) and nonlinear programs (NLP). MIPs belong to the class of NP-complete problems and it is natural to expect that a problem that combines this class with solving NLPs is challenging to solve. The algorithms available for solving a MINLP typically try to take advantage of solution methods specifically designed for each of the two subclasses. Some of the more popular algorithms include the Outer Approximation Algorithm (Duran and Grossmann, 1986), Generalized Benders Decomposition (Geoffrion, 1972), Branch & Bound (B&B) and Extended Cutting Planes methods (Westerlund and Petersson, 1995), (Floudas, 2000). The basics of these algorithms typically consist of an approach in which a sequence of NLPs are solved. An example of such a modular technique is the approach presented in (Hehua et al., 2004).

B&B, covered in section 2.4.1, is the approach that has the largest history of being applied to solve MINLPs. If a B&B algorithm is used to solve a MINLP, a sequence of NLPs has to be solved. However, to solve a general NLP is often computationally challenging.
2.1.1 Solution Methods for Non Linear Programs

Methods for solving NLPs include a great range of algorithms. The algorithms available for solving a specific problem are often very specialized as there is such a great variety of different nonlinear problems. As the original problem investigated in this report is a constrained nonconvex nonconcave optimization problem the range of applicable solution methods is somewhat more limited. The most used methods for solving general constrained optimization problems include sequential linear and quadratic programming methods (Wilson, 1963), (Fletcher, 1987), reduced-gradient methods (Gill, 1981), methods based on augmented Lagrangians (Bertsekas, 1982) and exact penalty functions (Krogstad, 2007). Sequential quadratic programming (SQP) is generally considered the most efficient method of solving nonlinear constrained problems (Murray, 1997).

Sequential Quadratic Programming

SQP is an iterative procedure that utilizes a second-order approximation of the Lagrangian function of a problem. The quadratic formulation of the problem is a local approximation of the real problem and consists of a quadratic objective function and linear equality and/or inequality constraints. SQP can be used both within a trust-region and a line search framework. In a line search framework, the algorithm proceeds by first calculating a search direction. If we are trying to maximize the original problem, a function is then solved that maximize the quadratic approximation in the search direction. When a new iteration point in the direction that was searched has been reached, a new local approximation is constructed and the algorithm proceeds to the next iteration given that a set of optimality conditions has not been fulfilled.

SQP is a generalization of Newton’s method for unconstrained problems as it uses a quadratic approximation of the Lagrangian function, steps in a direction it believes the optimum lies, and then creates a new approximation of the original model when a new iteration point has been reached. The main difference between Newton’s method and SQP is that for constrained nonlinear problems the Taylor approximation of the original problem cannot be used, as the model problem also needs to incorporate the constraints of the original problem. Instead, the Lagrangian function is used and the constraints of the problem thereby taken into account. For a more thorough introduction to SQP see Nocedal and Wright (1999).

2.1.2 Linearization of Nonlinear Models

The algorithms and software available for solving linear optimization problems far exceed the methods available for solving nonlinear models. When possible, an attractive way of solving nonlinear problems is to reformulate them so they only contain a linear objective function and linear constraints. Such a reformulation can be obtained using a
piecewise linearization (Williams, 1999). Although piecewise linearization originally only allows linearizing functions of one variable, the method can also be extended to functions of several variables, see Beale (1980). The disadvantage of extending that concept to multiple dimensions is that this can considerably increase the size of the problem as well as its computational difficulty.

In order to apply piecewise linearization to general nonconvex nonconcave nonlinear functions it is necessary to introduce Special Ordered Sets of Variables of type 2 (SOS2), first introduced in Beale and Tomlin (1969). A SOS2 set is a set of variables in which at most two neighboring variables are allowed to be greater than zero. With neighbors we mean two adjacent variables in the ordering in which the set is defined. In one dimension a set of neighbors is simply the weighting variables associated with the endpoints of each line in the piecewise linear curve, shown in the plot to the left in figure 2.1 as the points marked with green squares (Williams, 1999). The graph to the right illustrates what would happen if the variables associated with each breakpoint were not required to be SOS2. The piecewise linearization would then be a very poor approximation to the original blue smooth function.

![Figure 2.1: Piecewise linear approximation with (left) and without (right) SOS2 constraints](image)

The restriction that a set of variables belong to a SOS2 set can be modeled using integer variables. The disadvantage of this method is that it introduces complicating variables that can often make the problem harder to solve. If an optimization problem is formulated and solved using the optimization software Xpress-MP no binary requirements are set on the SOS2 variables (Dash Optimization, 2002). This often reduces the computational burden both in computer time and storage (Dash Optimization, 2004).

Before linearizing certain parts of a model it is important to consider how important to the solution the part that is being linearized is. Having considered this, the next step is to evaluate what resolution is needed to obtain the required accuracy. If approximating a function of higher dimensions this evaluation becomes increasingly important as the size of each dimension has a bigger effect on the size of the problem.
2.2 Model Building

A reduction in the solution time of a problem can often be obtained through a reformulation. This is especially true for discrete optimization problems as they can have the same optimal integer solution for different problem formulations (Williams, 1999). For an introduction to some typical reformulation techniques the reader is referred to the first chapter in Appa et al. (2006). Although the methods presented there were originally designed for pure integer programs, the methods have also proved useful in modeling and solving MILPs. If a reduction in the solution time can be obtained, it is usually through reducing the size of the problem or by obtaining a tighter LP formulation. The reason that a tighter LP formulation can often reduce the solution time is that solving a sequence of LPs is usually a part of the solution method for solving MILPs.

The different reformulation techniques presented in Appa et al. (2006) utilize a mathematical understanding of the solution space of a model. A reduction in solution time can however also sometimes be gained by studying the physical properties of a problem and recasting the problem in a new set of variables.

Another factor important to consider when modeling a system is if changes can be handled easily by the model. When a problem is formulated through a mathematical model one should keep in mind that if the model is to be used for similar, but not identical problems, it should be general enough that significant modification is not necessary. If likely extensions to the original problem can be handled without significantly increasing the solution time, this is a feature of a good model and extends its usefulness.

In general, there is always a tradeoff between accuracy and speed when creating a mathematical model. It is therefore important to work closely with the end users, as they ultimately are the ones that are able to decide the level of accuracy required.

2.3 Decomposition Methods

When constructing a mathematical model for a multi divisional system that consists of a number of subsystems the result can often be a very large model (Bradley et al., 1977). If there are few variables and constraints that connect the different divisions, the whole problem may be naturally partitioned into a number of independent subproblems, which hopefully are more manageable. If one can solve each of the subproblems and coordinate them to ensure that the optimal solution of the subproblems constitute an optimal solution of the main problem, one can handle the large main problem by the method commonly termed decomposition. For a general introduction to decomposition as a method see chapter 12 in Bradley et al. (1977).

A range of different decomposition methods have been suggested for handling the coordination between the subproblems and what is termed the master problem. A common distinction between the methods is based on how information is passed from the master
2.3. DECOMPOSITION METHODS

problem to the subproblems. Information is passed on either using a resource-directive way or a price-directive way (Molina, 1979). This distinction is generally equivalent to the distinction between primal and dual decomposition methods. Primal methods are methods where feasibility is preserved throughout the algorithm, while dual methods do not return a feasible answer before reaching the optimal solution. The most famous dual decomposition methods are Lagrange decomposition and Dantzig-Wolfe Decomposition, while Benders Decomposition is the primal decomposition method on which most work has been done (Holmberg, 1994).

2.3.1 Lagrange Relaxation

Lagrange Relaxation is a procedure to obtain good upper bounds for a problem (maximization problem), as lower bounds usually can be found using a heuristic. The starting point when using the methods is a modified version of the original problem one is trying to solve, which is known as the Lagrangian Relaxation. The main idea is to remove complicating constraints from the original problem, but to punish the objective function for breaking the constraints that have been removed. This is done by moving the relaxed constraints into the objective function and then multiplying them with a Lagrange multiplier (Fisher, 1981), (Beasley, 1993). How this is done for a typical optimization problem is illustrated in equation 2.1.

\[
\begin{align*}
\text{max } cx & \quad \text{max } cx + \lambda(h - Hx) \\
\text{s.t. } Ax = b & \quad \text{s.t. } Ax = b \\
Hx \leq h & \quad x \geq 0 \\
x \geq 0 & \quad \lambda \geq 0
\end{align*}
\]

(2.1)

Given that \( \lambda \) has to be greater than 0 and that a constraint has been removed going from the formulation on the left to the formulation on the right in 2.1, we can derive by inspection that the Lagrangian Relaxation will always be an upper bound on the original problem if we are maximizing (Hillier and Lieberman, 2005). By adjusting the Lagrange multiplier through a series of iterations, the hope is that a better upper bound can be found by finding the "correct price" of breaking the relaxed constraint(s).

The central decisions in an implementation of Lagrange Relaxation are which constraint(s) to relax and how to update the Lagrange multipliers. Which constraint(s) that should be relaxed is problem specific. The aim should be to obtain an Lagrangian Relaxation that easily can be solved.

Two general techniques are available for finding the Lagrange multipliers - subgradient optimization and multiplier adjustment (Beasley, 1993). Subgradient optimization is an
iterative procedure that can be viewed as a method that attempts solve the dual to the Lagrangian relaxation. That is, it tries to maximize the lower bound value obtained from the Lagrangian Relaxation. Multiplier adjustment is a heuristic approach that generates a starting set of Lagrange Multipliers, tries to improve them in some systematic way and then repeats the procedure if improvements are made.

Lagrangian Decomposition (LD) is a special form of Lagrangian Relaxation that tries to isolate sets of constraints that can be solved independently if some constraints were ignored (Gass and Harris, 2001). Separating these sets increase the total size of the problem as it creates the need for variables that link the different subsets. However, the hope is that solving each subproblem independently and then coordinating the subproblems is easier than solving the entire problem.

2.3.2 Dantzig–Wolfe Decomposition

The decomposition method that later became known as Dantzig-Wolfe decomposition (DWD) was first introduced in Dantzig and Wolfe (1960) and Dantzig and Wolfe (1961). Much work has been done since the 1960’s on applying the method to different problems and a good starting point for DWD and computational experience with DWD is the Ph.D. thesis Tebboth (2001). DWD has shown the greatest potential when applied to linear problems with a primal block angular structure, illustrated in figure 2.2.

![Figure 2.2: Illustration of primal block angular structure](image)

The essentials of using DWD to solve such a problem is a division of the problem into a single "master problem" and \( n \) independent "subproblems", one for each sub block \( A_1, A_2, \ldots, A_n \) as shown in figure 2.2. The master problem contains only a few more constraints than the number of common constraints in the original problem but contains a large number of columns. In DWD-formulations for LPs the columns in the master problem correspond to corner solutions in the convex subproblems. However, the master
problem is solved without tabulating all of these columns. Instead the columns are generated during the course of the procedure and a restricted version of the master problem (RMP) solved, a method that is termed column generation (Lasdon and Tabak, 1981). In between each time the RMP is solved, each of the subproblems are solved using information from the previous solution of the RMP. The objective function in each of the subproblems is dynamic. The information passed on from the RMP, simplex multipliers or shadow prices, change the objective function of the subproblems. The change in the subproblems’ objective function entails changing the cost of "using" the resource(s) that are constrained by the common constraint in the RMP. This is to obtain a more "correct" cost of using these resource(s), as there are no restrictions on them in the subproblems.

In the traditional version of the method, all subproblems are solved to optimum and new columns passed on to the RMP if their reduced costs are greater than zero (Barnhart et al., 1996). New columns that have been sent from the subproblems are, together with existing columns in the RMP, combined in different ways to produce new solutions. The algorithm terminates when no new columns with a positive reduced cost can be found. The maximum number of iterations needed to reach optimum is equal to the number of corner solutions in the subproblem. In practice, significantly fewer iterations are required (Lundgren et al., 2003). For a thorough treatment of Dantzig-Wolfe decomposition and the different decisions that need to be addressed in an implementation of the algorithm see Lasdon and Tabak (1981) and Tebboth (2001).

Parallel Solving of Subproblems

Standard DWD solves, in each iteration, the RMP first and then all subproblems sequentially. The fact that the subproblems are independent allows the assignment of each to different solution processes to solve them in parallel to obtain algorithmic efficiency (Ho et al., 1988). Early research used "supercomputers" consisting of many small computers combined to a computational cluster of processors. In recent years, the development of multicore processors has made parallel computing much more accessible as it allows for parallel computation using a single computer.

Gnanendran and Ho (1993) measures the performance of parallel DWD by speedup and efficiency. Speedup \( (S_p) \) is the ratio of the total sequential solution time \( (T_s) \) to the total parallel solution time \( (T_p) \) as shown in equation 2.2.

\[
S_p = \frac{T_s}{T_p} \quad (2.2)
\]

Speedup will be high when the total sequential solution time is high compared to the total parallel solution time. Efficiency \( (E_p) \) is the Speedup obtained per processor, as shown in equation 2.3.

\[
E_p = \frac{S_p}{p} \quad (2.3)
\]
Several factors impact on Speedup and Efficiency. Low computational effort in the RMP compared to the subproblems will have a positive effect. This occurs when there are few common constraints in the problem. The relative solution times of the subproblems will also significantly affect speedup and efficiency since the RMP, in standard implementation, has to wait for the slowest subproblem to finish (Gnanendran and Ho, 1993). To counter this, Tebboth (2001) offers different strategies concerning the order in which the subproblems are solved in each iteration and whether all problems should be solved on every cycle.

2.3.3 Benders Decomposition

Bender Decomposition (BD) was originally designed for mixed integer programs (Benders, 2005). Since its introduction it has shown the greatest results for problems which are in the form of the dual to the block angular structured problems for which DWD was designed (Molina, 1979). This structure, shown in figure 2.3, is usually termed the dual block angular structure (Bradley et al., 1977).

![Figure 2.3: Illustration of the dual block angular structure](image)

These problems typically have a set of complicating variables, \(A_0\), that if fixed, makes the problem much easier to solve. Analogue to DWD, the reformulation when using BD creates a master problem and a set of subproblems. The complicating variables are placed in these subproblems. Given a fixed set of values for the complicating variables, the subproblems find values for the simple variables. The dual variables of these solutions are then used to create new feasibility or optimality constraints which are subsequently added to the master problem. A new solution is then found by solving the master problem. From this new solution, the fixed values of the complicating constraints that are sent to the subproblems are determined. Given that the subproblems are bounded and have a feasible solution for each variable this iterative procedure will eventually lead the master problem to converge to optimum (Dantzig and Thapa, 2003).
2.4 Solving Mixed Integer Linear Programs

A general mixed integer linear program (MILP) is the minimization or maximization of a linear function subject to linear constraints where one or more of the variables have to be integer. A general maximization problem belonging to this class of problems is on the form shown in equation 2.4

\[
\begin{align*}
\text{max } & \quad cx + hy \\
\text{s.t. } & \quad Ax + Hy = b \\
& \quad x \geq 0, y \in \{0, 1\}
\end{align*}
\]

The Simplex and the Interior point method are general methods that have been proven to work very well for general linear programs. In contrast, to LPs, which can be solved efficiently in most cases and also in polynomial time in the worst case, integer programming problems are NP hard and in many practical situations no efficient algorithm exists. If a general polynomial time algorithm to solve pure IPs, here termed IPs, was found it would open up the possibility of solving a large class of problems efficiently, including the infamous traveling salesman problem. However, most scientists believe that such an algorithm does not exist (Williams, 1999). As IPs are a subclass of MILPs, it is necessary to look at which methods are available for solving IPs, to solve MILPs. Most exact methods for solving IPs today fall in to one of the following two categories Williams (1999):

- Cutting Plane Methods
- Enumerative Methods (with Branch & Bound methods as a special subclass)

Cutting Plane Methods have since they were first introduced in Gomory (1958) been implemented in many different variants, however the general idea persists. The integrality requirements are relaxed and the problem is solved as an LP. If the solution of the relaxed problem has integer values for the variables that were originally required to be integer, the problem is solved. If not, additional constraints (cutting planes) are added that make prior non integer solutions infeasible. This procedure continues iteratively until an optimal integer solution is found.

Enumerative Methods are usually applied to pure binary integer programs and typically utilize a tree search to only examine some solutions while systematically ruling out other solutions as suboptimal or infeasible. For a further description see chapter 4 in Garfinkel and Nemhauser (1972). Algorithms based on B&B are the enumerative methods that have shown greatest results and are currently by far the most used methods for solving problems where integer variables play an important role (Williams, 1999). The central
CHAPTER 2. THEORY

concept of Branch and Bound and some of its variants are explained further in the following sections.

2.4.1 Branch & Bound

The B&B algorithm centers around solving a sequence of LP relaxations using a tree structure. Through repeatedly solving a changing version of the original problem, the goal with the tree search is to gradually obtain better lower and upper bounds. When the difference between the upper and lower bound is less than a user defined limit, the algorithm terminates.

The solution of the LP relaxation of the original problem forms the root node of what is termed the B&B search tree. The value of this solution gives the first upper bound (maximization problem) on the original problem. During the B&B procedure there are two central decision steps that are repeated.

The first decision step involves choosing which variable to branch on. The variables that normally are branched on are the variable(s) that were required to be integer in the original problem but are not in the node we are currently looking at, termed the parent node. The act of branching involves solving two new LP relaxations to create two child nodes in the B&B tree. Each of the two new problems have an extra constraint on the variable that we chose to branch on. In the one child problem we add a constraint that forces the variable to be lower or equal to the rounded down value of the value it had in its parent node. For example, if the variable $x_1$ was originally required to be integer and had a value of 3.2 in the parent node, a constraint is added so that it is required to be 3 or less in the new node and nodes further down in the B&B tree on the same branch. In the second child node a constraint is added, which forces the variable to be greater or equal to the rounded up value of the value it had in its parent node. If any of the two new solutions are integer and higher than the current lower bound, they are the new lower bound, and are not branched on further. The unbranched node in the tree with the highest solution value is the upper bound on the original problem. This is because nodes further down in the tree will always have lower objective function values, as the feasible region becomes smaller the further we go down in the tree. If after solving the two new problems there are any noninteger solutions that have solution values that are lower than the current lower bound, the corresponding nodes are said to be bounded, and are not branched on further. If a node has a solution that is infeasible we also do not continue branching, as all children of this node will also be infeasible. How to choose which variable to branch on and the criteria that the branching rule should fulfill are covered in section 6.4.2.

The second decision step concerns which node to branch on. There are generally three different strategies that are used. In the depth first strategy branching is done on the active (nonbounded, noninteger) node with the lowest level in the tree. This is done to find a feasible solution fast to try to use this to cut off (bound) sections of the B&B
2.4. SOLVING MIXED INTEGER LINEAR PROGRAMS

In the best first strategy, the active node with the highest upper bound value is branched on in the hope that this is also where the feasible solution with the highest value lies. The breadth first strategy involves selecting the active node with the highest level in the B&B tree. If there are several candidates, it may select arbitrarily among them or choose the node with the highest upper bound value. Usually a combination of the different strategies are used. The combination is however dependent on what one wants to achieve with the B&B search; to find and prove the optimal solution fast, to find a "good solution" fast and then terminate the search, or to find several integer solutions. For more on how to choose a strategy see T’kindt et al. (2004) and Williams (1993).

2.4.2 Branch & Cut

Branch and Cut (B&C) is a hybrid solution method that combines the cutting plane method with B&B. If B&C is used to solve a MILP, an initial solution is first created by solving the problem as an LP using simplex. If the solution obtained has fractional values for one or more of the variables that are required to be integer, 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. When no such constraints can be found and the solution is still infeasible, the B&C algorithm moves on to using B&B to try to find the optimal integer solution. For a more thorough introduction to B&C the reader is referred to Mitchell (2001).

2.4.3 Branch & Price

Branch and Price (B&P) is a combination of B&B and column generation, and was specifically designed to handle IP models containing a large number of variables (Barnhart et al., 1996). Similarly to in a DWD, sets of columns are in B&P left out of the RMP problem because there are too many columns to handle efficiently and because the necessary reformulation of the problem leads to a model where most of the associated variables are equal to zero in an optimal solution. To check if an LP solution is optimal, a subproblem, termed the pricing problem, is solved to try to locate columns that have a positive reduced cost (maximization problem). If such columns are found, they are added to the RMP and the RMP reoptimized. Branching is done if no columns with a positive reduced cost can be found and the solution does not satisfy the integrality requirements. Although DWD and B&P have many common features, only applying B&B to the RMP in DWD while generating columns from the subproblems to obtain an integer solution does not guarantee a feasible or optimal solution (Savelsbergh, 1997). After branching, a column may exist that would have a positive reduced cost but that is not currently present in the RMP. B&P handles this by applying column generation in every node of the B&B tree to produce new columns to the RMP (Barnhart et al., 1996).

B&P is an exact method and will therefore, as with a pure B&B search, in the end find
the optimal integer solution (Danna and Le Pape, 2005). Within B&P there is however a large set of different strategies available for searching after the optimal solution. To start the B&P procedure an initial feasible RMP solution is needed as the pricing subproblems require information about the dual values of a RMP solution. Once the initial RMP solution has been obtained the remaining decisions revolve around how to branch, how to manage columns and what to branch on. The different branching strategies presented in section 2.4.1 with respect to how to branch are also valid for a B&P search. Similarly, the column management strategies available for B&P are the same as for DWD, and were discussed in section 2.3.2. How to choose what to branch on is more problem specific and is covered in section 6.4.3.

2.4.4 Heuristics

Heuristics are solution methods that try to obtain a good solution fast but that offer no formal proof of a offered solutions distance to the global optima. These methods are often used when there are no good exact solution methods available. There are generally two classes of heuristics, construction heuristics and local search heuristic (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. 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 out the best solution possible without any long term search considerations, thereby risking getting stuck in a local optima. Diversifying elements in a heuristic search in many ways try to do the opposite, which is to search in areas of the solution space which are unexplored. Heuristics should typically have a balance between elements that diversify and intensify the search to try find good solutions fast, but avoid getting stuck in local optima.

A metaheuristic refers to a master strategy that modifies other heuristics to obtain a balance between intensification and diversification to produce solutions beyond those that are normally generated in local search algorithms. Metaheuristics usually add a random element to the procedure or do not allow one to step to solutions that have been visited recently. This is done to avoid that the heuristic gets stuck in local optima. For more on heuristics and metaheuristics see Winker (2001).
Chapter 3

Model

This chapter presents the background for our formulation of the mathematical model describing the production allocation problem. An introduction to the most important parts of the model is followed by the assumptions made during modeling. The mathematical model for the mixed integer linear program (MILP) is presented in its entirety in the final section.

3.1 Introduction

As described in section 1.4.1, Gunnerud and Langvik (2007) first introduce a MINLP describing the upstream production system at the Troll West field. In their model, they represent the pipeline flow using the 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., 2000). Modeling with these three phases as flow variables, they solve an optimization problem that aims to maximize oil production while not exceeding the gas handling capacity in the first stage separator.

The pressure drop in the well pipelines is, as a function of gas, oil and water, highly nonlinear as described in section 1.2. The same is true for the pressure drop in the pipelines both between the manifolds and between the last manifold and the first stage separator. As explained in section 1.4.1, Gunnerud and Langvik (2007) approximate both the nonlinear Well Performance curves (WPC) and the pressure drop in the pipelines using piecewise linearization, thus reformulating the model into a MILP. This model and the work of Vestbø and Walberg (2008) is the basis for the model presented in section 3.3. Introducing this piecewise linearization increases the problem size substantially and a satisfactory accurate model for the full size field problem might have a total of about 270 000 variables. About 216 000 of these will be SOS2 weighting variables in the piecewise linearization. In addition there are about 3800 regular constraints. This has a substantial impact on the solution time of the model. In Hagem and Torgnes
(2008), a different model formulation is developed. This formulation utilizes certain fluid characteristics and has the potential of reducing the total number of interpolation points in the piecewise linearization, thus reducing the amount of variables and model solution time.

During this thesis, a thorough evaluation and testing of both these model formulations, using real production data from StatoilHydro, is performed. The new model formulation from Hagem and Torgnes (2008) showed in some cases promising results with regards to solution time and solution value. However, the model of Vestbø and Walberg (2008) was more user intuitive and simpler to understand. The latter makes it easier to implement more advanced solution methods such as decomposition. Therefore, we have chosen to proceed with a model formulation that is similar to the model introduced in Vestbø and Walberg (2008). This is presented in its entirety in section 3.3.

The differences between this model and the model of Hagem and Torgnes (2008) is presented in chapter 4 and the entire model of Hagem and Torgnes (2008) can be found in appendix A. The background and reasoning for the choice of model is presented in chapter 5.

### 3.2 Common Assumptions and Limitations for Both Models

#### 3.2.1 General Assumptions

Hauge and Horn (2005) states that the data produced by the Gas Oil Ratio Model (GORM) and subsequently passed on to GAP is recalculated for all wells on Troll West every 2-3 weeks. Since this is the current practice by StatoilHydro, we have also considered the reservoir to be in steady state over the weekly time period we are optimizing. As a result, the planning horizon for the model presented in section 3.3 belongs at the RTPO level, as it is defined in Foss et al. (2009). The pipelines from the manifolds and up to the platform are assumed to be identical in size and flow properties. This entails some symmetry in the solution space addressed in the final part of section 3.3. Both models are also deterministic, meaning uncertainty in parameters or data is not considered beyond a basic assessment of error sources.

#### 3.2.2 Absolute Pressure

Variations in the pressure drop that are attributed to differences in the absolute pressure are not considered in our model. Gunnerud and Langvik (2007) describe pressure drop in a pipeline as a function of the flow of different phases in that pipeline. As the separator pressure is fixed, there is no need to include absolute pressure when modeling the pressure drop between the separator and the manifold closest to it. However, between two manifolds further upstream in the production system, the absolute pressure will be
3.2. COMMON ASSUMPTIONS AND LIMITATIONS FOR BOTH MODELS

relevant to the pressure drop model. In this part, the absolute pressure drop will affect the flow as the pressure at neither of the manifolds is constant. Whether a pressure drop of for example 20 bar is between 150 and 130 bar or 90 and 70 bar will have an effect on the flow (Nydal, 2007). Including this extra variable in the pressure drop model means introducing a new set for the absolute pressure and summing over this extra variable is shown in equation 3.1 and 3.2.

\[
q_{\text{PIPE}}^{\text{limp}} = \sum_{n_g \in N_{\text{img}}} \sum_{n_o \in N_{\text{imo}}} \sum_{n_w \in N_{\text{imw}}} \sum_{d \in D} Q_{\text{DROP}}^{\text{imn}_{\text{pd}} \lambda_{\text{imngnwnwd}}} \quad \forall m \in M(i), \ l \in L(i), \ p \in P \tag{3.1}
\]

\[
p_{\text{DROP}}^{\text{iml}} = \sum_{n_g \in N_{\text{img}}} \sum_{n_o \in N_{\text{imo}}} \sum_{n_w \in N_{\text{imw}}} \sum_{d \in D} P_{\text{DROP}}^{\text{imn}_{\text{nwnwd}} \lambda_{\text{imngnwnwd}}} \quad \forall m \in M(i), \ l \in L(i) \tag{3.2}
\]

Due to limitations on the scope of our master thesis and after recommendations from the personnel at the Department of Engineering Cybernetics, this factor was not included in our model in section 3.3. StatoilHydro considers this an important simplification and the effect of this should be investigated (Alstad, V., 2009). For general production allocation problems this factor should be included. Including this extra variable in the linearization would increase the number of interpolation points substantially as it would add a fourth dimension to the linearization of the pressure drop. The resolution needed in the interpolation of this variable would be dependent on the number of manifolds. As the number of manifolds per cluster does not exceed two at the Troll West field, this limits the inaccuracy caused by this assumption.

3.2.3 Temperature

Temperature variations are not considered in the models. Temperature mainly affects the viscosity and the amount of gas present in the oil. Since the sea floor temperature is stable at 4°C, the fact that temperature is held constant in the model will have a very limited effect on the results (Alstad, V., 2009).

3.2.4 Piecewise Linearization

The most important simplifications concern the piecewise linearization of both the WPC and the pipeline pressure drop model. These are simplifications of the real life behavior of the reservoir and production system.
Well Performance Curves

As described in section 1.2, the WPCs are nonlinear. Using the same procedure as Gunnerud and Langvik (2007), we have approximated the WPCs using piecewise linearization. How coarse this approximation is, and how much it affects the solution, is decided by the resolution of the linearization and the actual shape of the WPCs.

Pressure Drop in Pipelines

To handle the nonlinear pressure drop described in section 1.2, Gunnerud and Langvik (2007) also made use of a piecewise linearization. This has a significant impact on their model as they linearize a function in three dimensions. Using chains of SOS2 variables to linearize a model over several dimensions can lead to a very large problem.

The piecewise linearization of the pipeline pressure drop can be described as a table with data points depicting pressure drop in bars for a given gas, oil and water flow. An example table for pressure drop interpolation in pipelines between two manifolds and from the separator to the manifold closest to it is shown in table 3.1. In this example, gas and oil are held constant while the amount of water is increased which increases the pressure drop.

<table>
<thead>
<tr>
<th>Pressure drop</th>
<th>Gas</th>
<th>Oil</th>
<th>Water</th>
</tr>
</thead>
<tbody>
<tr>
<td>47.23</td>
<td>2142.86</td>
<td>1285.71</td>
<td>642.86</td>
</tr>
<tr>
<td>49.99</td>
<td>2142.86</td>
<td>1285.71</td>
<td>964.29</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

Table 3.1: Example table for pipeline pressure drop interpolation

The more data points we use, the closer we approximate the nonlinear pressure drop. Each data point has a nonnegative SOS2 weighting variable assigned to it, meaning that the resolution and accuracy of the linearization will be the most important driver for the total amount of variables and thus the solution time of the problem. An important point is that the data points are distributed uniformly over the area linearized.
3.3 Mathematical Model

3.3.1 Declarations

Sets

\( \mathcal{I} \) \quad \text{Set of clusters}
\( \mathcal{M}(i) \) \quad \text{Set of manifolds in a cluster}
\( \mathcal{J}(i,m) \) \quad \text{Set of wells connected to a given manifold in a cluster}
\( \mathcal{H}(i,m) \) \quad \text{Subset of } \mathcal{J}(i,m). \text{ Wells that cannot be shut down}
\( \mathcal{L}(i) \) \quad \text{Set of pipelines in a cluster}
\( \mathcal{P} \) \quad \text{Set of phases (g for gas, o for oil and w for water)}
\( \mathcal{K}(i,m,j) \) \quad \text{Set of interpolation coordinates for the piecewise linearization of the well performance curve (WPC) for wells connected to a given manifold in a cluster}
\( \mathcal{N}(i,m,p) \) \quad \text{Set of interpolation coordinates for the piecewise linearization of the pressure drop in the pipelines connected to a manifold in a cluster, dependent on which phase}

Indices

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

Data

\( C_{\text{TOT}}^p \) \quad \text{Maximum capacity of phase } p \text{ in the first–stage separator at platform–level}
\( P_{\text{DROP}}^{imn_p n_p} \) \quad \text{Function for pressure drop in a pipeline from manifold } m \text{ to the next manifold or separator in cluster } i \text{ corresponding to interpolation coordinates } n_p \text{ for all phases, in the piecewise linearization of the pressure drop in pipelines}
\( P_{\text{MAX}}^i \) \quad \text{Maximum pressure before choke for all wells in cluster } i
\( P_{\text{MIN}}^{imj} \) \quad \text{Minimum pressure before choke for well } j \text{ at manifold } m \text{ in cluster } i
CHAPTER 3. MODEL

\( P_{\text{SEP}} \)
Pressure at first stage separator at platform (= \( P_{\text{M0}} \))

\( P_{\text{WPC}}^{\text{imjk}} \)
Pressure at manifold level before choke in pipeline from well \( j \) connected to manifold \( m \) in cluster \( i \) in the piecewise linearization of the well performance curve (WPC) with interpolation coordinate \( k \)

\( Q_{\text{DROP}}^{\text{imnp}} \)
Flow rate from manifold \( m \) in cluster \( i \) corresponding to interpolation coordinate \( n_p \) for that particular phase \( p \) in the piecewise linearization of the pressure drop in pipelines

\( Q_{\text{MAX}}^{\text{imjp}} \)
Maximum flow of phase \( p \) from well \( j \) in cluster \( i \) up to manifold \( m \)

\( Q_{\text{WPC}}^{\text{imjp}} \)
Flow rate of phase \( p \) from well \( j \) in manifold \( m \) in cluster \( i \) corresponding to interpolation coordinate \( k \) in the piecewise linearization of the well performance curve (WPC)

**Variables**

\( P_{\text{iml}}^{\text{M}} \)
Pressure at manifold level in pipeline \( l \) for flow from manifold \( m \) in cluster \( i \), \( m \in \{0\} \cup M(i) \)

\( P_{\text{iml}}^{\text{DROP}} \)
Pressure drop across a section of pipeline \( l \) with flow from manifold \( m \) in cluster \( i \) to either the next manifold or the separator

\( P_{\text{imj}}^{\text{WELL}} \)
Pressure at manifold level before choke in pipeline from well \( j \) connected to manifold \( m \) in cluster \( i \)

\( q_{\text{imlp}}^{\text{PIPE}} \)
Flow rate of phase \( p \) in pipeline \( l \) from manifold \( m \) to manifold \( m-1 \) or separator in cluster \( i \)

\( q_{\text{imjlp}}^{\text{WRP}} \)
Flow rate of phase \( p \) routed to pipeline \( l \) from well \( j \) connected to manifold \( m \) in cluster \( i \)

\( q_{\text{imjp}}^{\text{WELL}} \)
Flow rate of phase \( p \) from well \( j \) to manifold \( m \) in cluster \( i \)

\( x_{\text{imj}} \)
Binary variable. Equals 1 if well \( j \) at manifold \( m \) in cluster \( i \) is closed, 0 otherwise. This variable is not defined for \( j \in H(i, m) \)

\( y_{\text{imjl}} \)
Binary variable. Equals 1 if well \( j \) at manifold \( m \) in cluster \( i \) is routed to pipeline \( l \), 0 otherwise

\( \gamma_{\text{imjkl}} \)
Weighting variable associated with each interpolation coordinate \( k \) in the piecewise linearization of the WPC for well \( j \) connected to manifold \( m \) in cluster \( i \)

\( \lambda_{\text{imln}}n_{n_{\text{n}},n_{\text{n}}} \)
Weighting variable associated with the interpolation coordinates for \( n_g, n_q \) and \( n_w \) in the piecewise linearization of the pressure drop in pipelines for manifold \( m \) in cluster \( i \) and pipeline \( l \)

\( \eta_{\text{imlpmn}}^{\text{SUB}} \)
Variable connected to the modeling of the SOS2 sets for the piecewise linearization of the pressure drop in pipelines for the interpolation coordinate for pipeline \( l \) connected to manifold \( m \) in cluster \( i \) in interpolation set \( n_p \) dependent on phase \( p \)
3.3.2 Model Formulation

Objective Function

The stated goal of StatoilHydro is to maximize the total amount of oil produced at the platform. This is expressed in the objective function as the sum of oil flow in the pipelines from the last manifold up to the first stage separator for all clusters.

\[
\max Z = \sum_{i \in I} \sum_{l \in L(i)} q_{\text{PIPE}}^{imlp}, \ m = \{1\}, \ p = \{\text{oil}\} \quad (3.3)
\]

Constraints

The constraints can be divided into handling capacity, linearization of well performance curves, mass balance, pressure requirements, linearization of pressure drop, closing of wells, removal of symmetry and requirements on variables.

Handling Capacity  The platform’s handling capacity is the amount of gas and water the first stage separator can handle. The gas constraint is currently the only binding constraint at the Troll West, but StatoilHydro has indicated that the water handling capacity might also become an active constraint in the future when the field matures. The total water or gas produced is the sum of water or gas flow in the pipelines from the last manifold up to the separator for all clusters.

\[
\sum_{i \in I} \sum_{l \in L(i)} q_{\text{PIPE}}^{imlp} \leq C_p^{\text{TOT}} \ m = \{1\}, \ p = \{\text{gas, water}\} \quad (3.4)
\]

As the constraints covered in the rest of the chapter apply for all clusters, \(i\), we have not written \(\forall \ i \in I\) when writing out all the constraints.

Linearization of Well Performance Curves  The linearization of the WPCs is expressed in the constraints below. As described in section 1.2, the wellhead pressure will decide the flows of gas, oil and water from the well to the connected manifold. The \(\gamma\) variables are modeled as SOS2.

\[
p_{\text{WELL}}^{imj} = \sum_{k \in K(i,m,j)} p_{\text{WPC}}^{imjk} \ \gamma_{imjk} \ \forall \ m \in \mathcal{M}(i), \ j \in \mathcal{J}(i,m) \quad (3.5)
\]
\[ q_{imjp}^{WELL} = \sum_{k \in K(i,m,j)} Q_{imjpk}^{WPC} \gamma_{imjk} \quad \forall \ m \in \mathcal{M}(i), \ j \in \mathcal{J}(i,m), \ p \in \mathcal{P} \quad (3.6) \]

\[ \sum_{k \in K(i,m,j)} \gamma_{imjk} = 1 \quad \forall \ m \in \mathcal{M}(i), \ j \in \mathcal{J}(i,m) \quad (3.7) \]

\[ \gamma_{imjk} \geq 0 \quad \forall \ m \in \mathcal{M}(i), \ j \in \mathcal{J}(i,m), \ k \in K(i,m,j) \quad (3.8) \]

\[ \gamma_{imjk} \text{ is SOS2} \quad \forall \ m \in \mathcal{M}(i), \ j \in \mathcal{J}(i,m) \quad (3.9) \]

**Routing of Wells** This constraint was introduced to enforce the separation of wells in to a set of wells that can be closed and a set of wells that cannot be closed. \( x_{imj} \) is only defined for the set of wells that can be closed. This ensures that the wells that cannot be closed still produce, and are routed to one of the two pipelines.

\[ x_{imj} + \sum_{l \in \mathcal{L}(i)} y_{imjl} = 1 \quad \forall \ m \in \mathcal{M}(i), \ j \in \mathcal{J}(i,m) \quad (3.10) \]

**Mass Balance** The mass balance constraint in equation 3.11 forces the flow from a well to be equal to the sum of all flows routed to the different pipelines from that well (\( q_{imjlp}^{WRF} \)). Equation 3.12 ensures that the flow from a well to a particular pipeline is zero if the corresponding routing variable is equal to zero. Equation 3.13 ensures that the flow of gas, oil or water in a pipeline has to be equal to the sum of flow for that particular phase from all wells routed to that pipeline plus the potential flow from a neighboring manifold further upstream in the cluster.

\[ \sum_{l \in \mathcal{L}(i)} q_{imjlp}^{WRF} = q_{imjp}^{WELL} \quad \forall \ m \in \mathcal{M}(i), \ j \in \mathcal{J}(i,m), \ p \in \mathcal{P} \quad (3.11) \]

\[ q_{imjlp}^{WRF} \leq Q_{imjl}^{MAX} \quad \forall \ m \in \mathcal{M}(i), \ j \in \mathcal{J}(i,m), \ l \in \mathcal{L}(i), \ p \in \mathcal{P} \quad (3.12) \]

\[ q_{imlp}^{PIPE} = \sum_{j \in \mathcal{J}(i,m)} q_{imjlp}^{WRF} + q_{i(m+1)lp}^{WRF} \quad \forall \ m \in \mathcal{M}(i), \ l \in \mathcal{L}(i), \ p \in \mathcal{P} \quad (3.13) \]

In equation 3.13, the last term is not included for manifold \( m \in \mathcal{M}(i) \), because it only receives flow from the connecting wells and not a previous manifold, since no such exist.
3.3. MATHEMATICAL MODEL

Manifold Pressure Requirements  The following constraint enforces that the pressure in pipeline $l$ downstream from manifold $m$ in cluster $i$ has to be lower than the maximum pressure before the choke in all wells connected to manifold $m$ in cluster $i$ routed to pipeline $l$. $P_{i}^{\text{MAX}} - P_{imj}^{\text{MIN}}$ is used as the smallest possible Big M.

$$p_{iml}^{M} \leq p_{imj}^{\text{WELL}} + (P_{i}^{\text{MAX}} - P_{imj}^{\text{MIN}}) (1 - y_{imjl}) \ \forall \ m \in M(i), \ j \in J(i,m), \ l \in L(i) \ (3.14)$$

Linearization of Pressure Drop  As described in the beginning of this chapter, the pressure drop in the pipelines is modeled in three dimensions as a function of gas, oil and water. The linearization of the pressure drop covered in section 3.2.4 makes it necessary to introduce the constraints 3.15 - 3.23.

$$q_{imlp}^{\text{PIPE}} = \sum_{n_{g} \in N_{img}} \sum_{n_{o} \in N_{imo}} \sum_{n_{w} \in N_{imw}} q_{imlnp}^{\text{DROP}} \lambda_{imln_{g}n_{o}n_{w}} \ \forall \ m \in M(i), \ l \in L(i), \ p \in P \ (3.15)$$

$$p_{iml}^{\text{DROP}} = \sum_{n_{g} \in N_{img}} \sum_{n_{o} \in N_{imo}} \sum_{n_{w} \in N_{imw}} p_{imln_{g}n_{o}n_{w}}^{\text{DROP}} \lambda_{imln_{g}n_{o}n_{w}} \ \forall \ m \in M(i), \ l \in L(i) \ (3.16)$$

$$\sum_{n_{g} \in N_{img}} \sum_{n_{o} \in N_{imo}} \sum_{n_{w} \in N_{imw}} \lambda_{imln_{g}n_{o}n_{w}} = 1 \ \forall \ m \in M(i), \ l \in L(i) \ (3.17)$$

$$\lambda_{imln_{g}n_{o}n_{w}} \geq 0 \ \forall \ m \in M(i), \ l \in L(i), \ n_{g} \in N_{img}, \ n_{o} \in N_{imo}, \ n_{w} \in N_{imw} \ (3.18)$$

$$\eta_{imlnp} = \sum_{n_{o} \in N_{imo}} \sum_{n_{w} \in N_{imw}} \lambda_{imln_{g}n_{o}n_{w}} \ \forall \ m \in M(i), \ l \in L(i), \ p = \{\text{gas}\}, \ n_{g} \in N_{img} \ (3.19)$$

$$\eta_{imlnp} = \sum_{n_{g} \in N_{img}} \sum_{n_{w} \in N_{imw}} \lambda_{imln_{g}n_{o}n_{w}} \ \forall \ m \in M(i), \ l \in L(i), \ p = \{\text{oil}\}, \ n_{o} \in N_{imo} \ (3.20)$$
\[ \eta_{imlpn_p} = \sum_{n_g \in N_{img}} \sum_{n_w \in N_{imo}} \lambda_{imln} n_g n_w \] 
\[ \forall m \in M(i), \ l \in L(i), \ p = \{\text{water}\}, \ n_w \in N_{imw} \] 

\[ \eta_{imlpn_p} \geq 0 \quad \forall \ m \in M(i), \ l \in L(i), \ p \in P, \ n_p \in N_{imp} \] 

\[ \eta_{imlpn_p} \text{ is SOS2} \quad \forall \ m \in M(i), \ l \in L(i), \ p \in P \] 

**Pipeline Pressure Requirements** The pressure drop in a pipeline from one manifold to another manifold, or up to the separator, needs to be equal to the difference in manifold pressure between the two. The pressure at the first stage separator is fixed.

\[ p^M_{(m-1)i} = p^M_{mi} - P^{DROP}_{iml} \quad \forall \ m \in M(i), \ l \in L(i) \] 

\[ p^M_{0l} = P^{SEP} \quad \forall \ l \in L(i) \] 

**Requirements on Variables** Binary requirements must be put on the variable \( x_{imj} \), which represents whether a well is open or closed. The same is true for \( y_{imjl} \) which contains information on whether a well has been routed to a certain pipeline or not.

\[ x_{imj} \in \{0, 1\} \quad \forall \ m \in M(i), \ j \in J(i,m) \] 

\[ y_{imjl} \in \{0, 1\} \quad \forall \ m \in M(i), \ j \in J(i,m), \ l \in L(i) \] 

Nonnegativity constraints apply to all other variables.

**Closing of Wells** If a well is open, the last data point corresponding to zero flow in the interpolation of the WPC curve cannot be used, that is, it needs to have zero weight. This is to prevent a well from producing no flow. Closing down wells might lead to problems opening them again and StatoilHydro therefore prefers that the wells produce a small amount of flow even though they are not a part of the set of producing wells in the optimal solution. This was discussed in section 1.4.2.

\[ \gamma_{imjk} - x_{imj} \leq 0 \quad \forall \ m \in M(i), \ j \in J(i,m), \ k \in Max\{K(i,m,j)\} \]
3.3. MATHEMATICAL MODEL

**Removal of Symmetry**  Assuming the pipelines between manifolds are identical, an additional constraint to remove symmetry is. In a two pipeline example, if not equal there will always be more wells routed to one of the pipelines than the other. If there are more than two pipelines, pipeline \( l - 1 \) should have more or equal the amount of wells routed to it compared to pipeline \( l \).

\[
\sum_{j \in J(i,m)} (y_{imj(l-1)} - y_{imj}) \geq 0 \quad \forall \ m \in \mathcal{M}(i), \ l \in \mathcal{L}(i)/\{1\} \quad (3.29)
\]
Chapter 4

Model Changes

This chapter aims at presenting the main differences between the model of Vestbø and Walberg (2008), presented in section 3.3, and the changes introduced in the model of Hagem and Torgnes (2008). It also explains the reason the changes introduced in the model of Hagem & Torgnes are complicating.

4.1 Introduction

As described in section 2.2, a reduction in model solution time can sometimes be achieved by changing the formulation of the mixed integer linear programme (MILP). For our problem, the amount of variables in the MILP piecewise linearization can likely be reduced by changing the flow variable representation in the model. Therefore, Hagem and Torgnes (2008) proposed a different flow variable representation than the model of Vestbø & Walberg presented in the previous chapter. To investigate the solution times of both model under different circumstances, test runs were performed using artificial production data with second order approximated functions to describe the pressure drop in the pipelines. Hagem and Torgnes (2008) did not reach any conclusive results regarding which model to proceed with, and a detailed testing scheme using real production data was needed. This testing scheme is described in chapter 5.

4.2 Model of Vestbø & Walberg

As explained in section 3.1, Vestbø and Walberg (2008) represents flow using the phases gas, oil and water. The objective function in equation 4.1 meets StatoilHydro’s goal of maximizing the total amount of oil reaching the platform from all pipelines to the first
stage separator from all clusters.

\[
\max \ Z = \sum_{i \in I} \sum_{l \in L(i)} q_{imlp}^{\text{PIPE}}, \ m = \{1\}, \ p = \{\text{oil}\} \tag{4.1}
\]

The platform handling capacity is preserved by the constraint in equation 4.2 limiting the total flow of gas and water from all pipelines up to the platform.

\[
\sum_{i \in I} \sum_{l \in L(i)} q_{imlp}^{\text{PIPE}} \leq C_p^{\text{TOT}}, \ m = \{1\}, \ p = \{\text{gas, water}\} \tag{4.2}
\]

4.3 Model of Hagem & Torgnes

As stated in section 3.2.4, the solution time of the MILP is largely dependent on the resolution of the piecewiselinearization in the pipeline pressure drop model. By resolution we mean the total number of interpolation points in the piecewise linearization. Each interpolation point has a distinct nonnegative SOS2 weighting variable assigned to it as shown in section 2.1.2. Therefore, increasing the resolution means increasing the number of variables in the model. Because the interpolation for both models are performed in three dimensions, decreasing the resolution on one axis will have a substantial effect on the total amount of variables.

For example, 30 interpolation points on one of the axes would result in \(30 \times 30 \times 30 = 27000\) variables, whereas reducing one of the axes to 5 interpolation points would result in \(30 \times 30 \times 5 = 4500\) variables. This reduction is only for a single cluster and for a full field size problem with eight clusters the amount of variables would be reduced from \(27000 \times 8 = 216000\) to \(4500 \times 8 = 36000\). Revisiting the example in the introduction to chapter 3, the 270 000 variables in the problem would be reduced to 90 000 variables thus reducing solution time substantially.

However, reducing the total number of interpolation points in the pressure drop model might, depending on the shape of the nonlinear function that is approximated, reduce the accuracy of the solution. In their Xpress implementation, neither Gunnerud and Langvik (2007) nor Vestbø and Walberg (2008) are able to use different numbers of interpolation points for the phases gas, oil and water. Since the flow component of one phase might affect the pressure drop more than the others, it is interesting to investigate the impact of introducing a feature that allows varying the resolution in a single dimension (Gunnerud, V., 2008). Reducing the resolution in one dimension while keeping the resolution in the other two might not reduce the accuracy of the solution. To amplify this effect, it is especially interesting to look at alternative ways of describing the flow.

The new model introduced in Hagem and Torgnes (2008) therefore suggests a transformation of the variables describing flow in the pipelines. It uses gas, liquid and water cut
instead of gas, oil and water to describe the flow. As opposed to having distinct variables for oil and water, the sum of the two are treated as one component, liquid. To still be able to extract the amount of oil and water in the liquid, a water cut variable is introduced. This is calculated as the percentage of water present in the liquid (oil+water) as shown in equation 4.3.

\[ Watercut = \frac{\text{Water}(Sm^3/day)}{\text{Oil}(Sm^3/day) + \text{Water}(Sm^3/day)} \] (4.3)

The reasoning behind this reformulation is that water and oil behave similarly in the pressure drop model. The hypothesis is that this will enable a reduction in the number of interpolation points describing the water cut dimension without loss of solution accuracy. This will reduce the total number of SOS2 weighting variables in the MILP. The approach was originally suggested by personnel at FMC, and we were further encouraged by staff at the Department of Engineering Cybernetics to investigate the effect this reformulation had on solution time and solution accuracy.

The flow variable reformulation entails some nonlinearities in the mathematical model. Since oil and water are no longer distinct variables, an extra piecewise linearization needs to be introduced to calculate oil and water flow in the objective function and water handling constraint. As opposed to the model presented in section 3.3, the model of Hagem & Torgnes also assumes a constant water cut in each well which is the same assumption that StatoilHydro makes (Dueñas Díez, Marta., 2008a).

### 4.3.1 New or Changed Declarations

The changes explained above entails a more complicated objective function and some additional constraints. The following section describes the new or changed declarations introduced in the model of Hagem & Torgnes compared to the model of Vestbø & Walberg presented in section 3.3.

**Sets**

- \( S \) Set of substances containing gas and liquid for use in linearization of pressure drop in pipelines
- \( P \) Set of phases containing gas and water for use in RHS of gas and water handling constraint

**Indices**

- \( s \) Substance
- \( p \) Phase
CHAPTER 4. MODEL CHANGES

Data

\( f_{i,m,j}^{WELL} \)  
Water cut for well \( j \) connected to manifold \( m \) in cluster \( i \)

Variables

\( f_{i,m,l}^{PIPE} \)  
Water cut in pipeline \( l \) from manifold \( m \) to manifold \( m - 1 \) or first stage separator in cluster \( i \)

4.3.2 Nonlinearities

Objective Function  
This objective function corresponds to equation 3.3 in section 3.3. The oil production is still maximized. Adding together oil and water in a single variable for liquid makes it necessary to calculate the oil flow by multiplying the liquid variable with the water cut variable. This multiplication of variables results in a nonlinear objective function.

\[
\max Z = \sum_{i \in I} \sum_{l \in L(i)} q_{i,m,l,s}^{PIPE}(1 - f_{i,m,l}^{PIPE})
\]

\[s = \{\text{liquid}\}, m = \{1\}, \]

(4.4)

Water Handling Constraint  
This constraint corresponds to equation 3.4 in section 3.3. When extracting water flow from the liquid variable in the water handling constraint, the same nonlinearity as in the objective function arises. This is shown in equation 4.5.

\[
\sum_{i \in I} \sum_{l \in L(i)} q_{i,m,l,s}^{PIPE} f_{i,m,l}^{PIPE} \leq C_{p}^{TOT}
\]

\[p = \{\text{water}\}, s = \{\text{liquid}\}, m = \{1\} \]

(4.5)

4.3.3 Handling Nonlinearities

To avoid the previously mentioned nonlinear expression, the water flow for each pipeline can be expressed as the flow from each well routed to that pipeline multiplied with the
well water cut. The assumption that each well has constant water cut over the period we are modeling makes this expression linear.

\[ q_{\text{PIPE}} f_{\text{PIPE}} = \sum_{j \in \mathcal{J}(i,m)} q_{\text{WRP}j\text{m}} f_{\text{WELL}j} + q_{\text{WRP}i(m+1)} f_{\text{WELL}i(m+1)} \]

\[ s = \{\text{liquid}\}, m = \{1\}, \forall l \in \mathcal{L}(i) \] (4.6)

\[ q_{\text{PIPE}} f_{\text{PIPE}} = \sum_{j \in \mathcal{J}(i,m)} q_{\text{WRP}j\text{m}} f_{\text{WELL}j} \]

\[ s = \{\text{liquid}\}, m = \{2\}, \forall l \in \mathcal{L}(i) \] (4.7)

Equation 4.6 and 4.7 describe clusters with two manifolds. If a cluster includes more than two manifolds additional constraints for the pipelines between these manifolds also needs to be included. These constraints will be similar to equation 4.6 with additional elements for these pipelines.

**Linearized Objective Function** Replacing the water cut variable in the objective function by the RHS in equation 4.6 results in the linear objective function shown in equation 4.8. The total amount of oil flow to the separator is then expressed as the sum of liquid in the pipelines minus the sum of liquid from each well multiplied with the water cut for that well.

\[ \max Z = \sum_{i \in I} \left( \sum_{l \in \mathcal{L}(i)} q_{\text{PIPE}il} f_{\text{PIPE}il} \right) - \sum_{j \in \mathcal{J}(i,m)} q_{\text{WRP}j\text{m}} f_{\text{WELL}j} + q_{\text{WRP}i(m+1)} f_{\text{WELL}i(m+1)} \]

\[ s = \{\text{liquid}\}, m = \{1\} \] (4.8)

In the water handling constraint, water flow is also expressed by multiplying the liquid for each well with the water cut for that well and then summing over all well flows. To capture the flow from wells in other manifolds and from other clusters it is necessary to sum over all the manifolds in a cluster and over all clusters.

**Linearized water handling constraint**

\[ \sum_{i \in I} \sum_{l \in \mathcal{L}(i)} \sum_{j \in \mathcal{J}(i,m)} q_{\text{WRP}j\text{m}} f_{\text{WELL}j} + q_{\text{WRP}i(m+1)} f_{\text{WELL}i(m+1)} \leq C^{\text{TOT}}_w \]

\[ s = \{\text{liquid}\}, m = \{1\} \] (4.9)
Nonlinear Water Cut Linking Constraint  To link the water cut in the pipeline pressure drop model with the well water cut and flow, it is necessary to add an extra water cut linking constraint (WCL) for each manifold in each cluster. Equation 4.10 shows how a nonlinear version of this constraint would look for the manifold closest to the platform. Relaxing this constraint would allow the water cut in the pressure drop linearization to take a more beneficial value for the pipeline flow rather than matching the water cut from the wells. This constraint can also be viewed as a "mass balance" equation for the water cut variable.

\[
q_{\text{PIPE} iml} - q_{\text{WELL} iml} = \sum_{j \in J(i,m)} WRP_{imjls} F_{imj}^{\text{WELL}} + q_{WELL i(m+1)jls} F_{i(m+1)j}^{\text{WELL}} + \epsilon_{\text{WCL} iml}
\]

\[s = \{\text{liquid}\}, m = \{1\}, \forall l \in L(i) \quad (4.10)\]

The left hand side in equation 4.10 is nonseparable as it contains a product of two variables. Piecewise linearization in one dimension can only be applied on separable functions. To linearize this function we therefore first chose to reformulate the function into a separable form as described in Williams (1999). This reformulation results in a left hand side that consists of two separable quadratic functions. These two functions can then be linearized using piecewise linearization.

Linearized Water Cut Linking Constraint  Equations 4.11 and 4.12 are the transformed WCL constraints.

\[
v_{\text{WCL} iml1} - v_{\text{WCL} iml2} = \sum_{j \in J(i,m)} WRP_{imjls} F_{imj}^{\text{WELL}} + q_{WELL i(m+1)jls} F_{i(m+1)j}^{\text{WELL}} + \epsilon_{\text{WCL iml}}
\]

\[s = \{\text{liquid}\}, m = \{1\}, l \in L(i) \quad (4.11)\]

\[
v_{\text{WCL} iml1} - v_{\text{WCL} iml2} = \sum_{j \in J(i,m)} WRP_{imjls} F_{imj}^{\text{WELL}} + \epsilon_{\text{WCL iml}}
\]

\[s = \{l\}, m = \{2\}, l \in L(i) \quad (4.12)\]

The \(v_{\text{WCL}}\) variables are approximated using piecewise linearization shown in equation 4.13 and 4.14.

\[
v_{\text{WCL} iml1} = \sum_{r \in R} U_{imr}^2 \mu_{iml1r} WCL
\]

\[\forall m \in M(i), l \in L(i) \quad (4.13)\]
4.3. MODEL OF HAGEM & TORGNES

\[ v_{iml2}^{WCL} = \sum_{r \in R} U_{im2r}^{WCL} \mu_{iml2r}^{WCL} \]

\[ \forall m \in M(i), \; l \in L(i) \]  \hspace{1cm} (4.14)

These are again connected to the left hand side of equation 4.10 through the piecewise linear \( u_{iml}^{WCL} \) variables with the same \( \mu_{iml}^{WCL} \) weighting variables.

\[ u_{iml1}^{WCL} = \sum_{r \in R} U_{im1r}^{WCL} \mu_{iml1r}^{WCL} \]

\[ \forall m \in M(i), \; l \in L(i) \]  \hspace{1cm} (4.15)

\[ u_{iml2}^{WCL} = \sum_{r \in R} U_{im2r}^{WCL} \mu_{iml2r}^{WCL} \]

\[ \forall m \in M(i), \; l \in L(i) \]  \hspace{1cm} (4.16)

The \( u_{iml}^{WLC} \) are finally connected to the left hand side of equation 4.10 through equation 4.17 and equation 4.18.

\[ u_{iml1}^{WCL} = \frac{q_{lims}^{PIPE} + f_{iml}^{PIPE}}{2} \]

\[ s = \{ \text{liquid} \}, \; \forall m \in M(i), \; l \in L(i) \]  \hspace{1cm} (4.17)

\[ u_{iml2}^{WCL} = \frac{q_{lims}^{PIPE} - f_{iml}^{PIPE}}{2} \]

\[ s = \{ \text{liquid} \}, \; \forall m \in M(i), \; l \in L(i) \]  \hspace{1cm} (4.18)
Chapter 5

Model Testing

This chapter aims to describe the testing of the two different mathematical models presented in chapter 3 and 4. All tests use real production data as input. The first sections in the chapter describe the test input data and explain the difference between real production data and artificial production data. The process with which the real production data are generated is also described. This process is described in some detail due to the limited work previously performed on this, making the description valuable to a potential future user. The data generated by this method is first utilized to compare the models in Vestbø and Walberg (2008) and Hagem and Torgnes (2008), but will also be used on a larger scale for a full field size problem when testing decomposition methods and Branch & Price as presented in chapter 6.

The final sections of this chapter describe the model testing scheme, the results obtained from this and a conclusion regarding why we have chosen to proceed with the model of Vestbø & Walberg presented in section 3.3 and not the model of Hagem & Torgnes. The latter model is also presented in its entirety in appendix A.

5.1 Real Production Data

5.1.1 Importance of Testing Using Real Production Data

The input data required to solve the models described in section 3.3 and 4.3 mainly describe the system topology such as the number of well 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 1.2. The previous work in Gunnerud and Langvik (2007), Vestbø and Walberg (2008) and Hagem and Torgnes (2008) have used second order functions to approximate the Well Performance Curves (WPC) and the pipeline pressure drop. These simple nonlinear functions will not represent all possible variations in the real pipeline and well flow regime.
for petroleum fields. As described in section 1.2, these relationships are very complex and difficult to approximate. Personnel from StatoilHydro have also indicated that the artificial production data used may be too simple and that the real WPCs and pipeline pressure drop might not be as well-behaved (Dueñas Diez, Marta., 2008a). The real production data will likely require a greater resolution in the piecewise linearization to obtain a satisfactory solution quality. Since the solution time and behavior of the models and decomposition methods are highly dependent on the underlying data, it is hard to draw conclusions without observing how the models behave when real production data is used.

5.1.2 Real Data Generation

Well Performance Curves Generation Section 1.3 explains that StatoilHydro currently generates pipeline and well data using Gas Oil Rate Model (GORM) and General Allocation Package (GAP). For our purpose, real production data can be generated using GAP accessed with VBA. We therefore create an Excel VBA Macro that is able to read input data from a spreadsheet and run GAP simulations based on this input. WPC data is generated by entering all possible values of wellhead pressure in GAP and thereby simulate the different flows of gas, oil and water. This is performed for each well in the cluster.

Figure 5.1 shows an example of real WPC data for a single well. The displayed data have been modified from their original form to avoid disclosing specific StatoilHydro data.

![WPC data example](image)

Figure 5.1: Example of WPC data for a single well

Pipeline Pressure Drop Generation Generation of data for the pipeline pressure drop is performed in a different way. Figure 5.2 shows a screenshot of the GAP user
interface and illustrates how pipeline pressure drop is calculated created using GAP.

Here, all wells except one well, indicated by the circle around the triangle, on manifold 2 are closed. For this single well, GAP simulates 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 in the upper right corner. 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 5.2: Screenshot from GAP showing the topology in a single cluster

Figure 5.3 shows an example of pipeline pressure drop for certain flow combinations in each pipeline. Again, the displayed data have been modified from their original form to avoid disclosing specific StatoilHydro data.

<table>
<thead>
<tr>
<th>Pipe m1 - Sep</th>
<th>Gas</th>
<th>Oil</th>
<th>Water</th>
<th>Pipe m2 - m1</th>
<th>Gas</th>
<th>Oil</th>
<th>Water</th>
</tr>
</thead>
<tbody>
<tr>
<td>11.95</td>
<td>1428.57</td>
<td>321.43</td>
<td>321.43</td>
<td>3.61</td>
<td>1428.57</td>
<td>321.43</td>
<td>954.29</td>
</tr>
<tr>
<td>15.47</td>
<td>1428.57</td>
<td>321.43</td>
<td>642.88</td>
<td>6.88</td>
<td>1428.57</td>
<td>321.43</td>
<td>954.29</td>
</tr>
<tr>
<td>24.28</td>
<td>1428.57</td>
<td>321.43</td>
<td>964.29</td>
<td>6.47</td>
<td>1428.57</td>
<td>321.43</td>
<td>954.29</td>
</tr>
<tr>
<td>26.63</td>
<td>1428.57</td>
<td>321.43</td>
<td>1285.71</td>
<td>6.98</td>
<td>1428.57</td>
<td>321.43</td>
<td>1285.71</td>
</tr>
<tr>
<td>26.75</td>
<td>1428.57</td>
<td>321.43</td>
<td>1607.14</td>
<td>7.28</td>
<td>1428.57</td>
<td>321.43</td>
<td>1607.14</td>
</tr>
<tr>
<td>30.94</td>
<td>1428.57</td>
<td>321.43</td>
<td>1928.57</td>
<td>7.74</td>
<td>1428.57</td>
<td>321.43</td>
<td>1928.57</td>
</tr>
<tr>
<td>33.07</td>
<td>1428.57</td>
<td>321.43</td>
<td>2250.00</td>
<td>8.14</td>
<td>1428.57</td>
<td>321.43</td>
<td>2250.00</td>
</tr>
<tr>
<td>13.51</td>
<td>1428.57</td>
<td>642.86</td>
<td>321.43</td>
<td>4.23</td>
<td>1428.57</td>
<td>642.86</td>
<td>821.41</td>
</tr>
<tr>
<td>22.07</td>
<td>1428.57</td>
<td>642.86</td>
<td>642.86</td>
<td>6.39</td>
<td>1428.57</td>
<td>642.86</td>
<td>642.86</td>
</tr>
<tr>
<td>26.04</td>
<td>1428.57</td>
<td>642.86</td>
<td>964.29</td>
<td>6.61</td>
<td>1428.57</td>
<td>642.86</td>
<td>964.29</td>
</tr>
<tr>
<td>28.03</td>
<td>1428.57</td>
<td>642.86</td>
<td>1285.71</td>
<td>7.02</td>
<td>1428.57</td>
<td>642.86</td>
<td>1285.71</td>
</tr>
<tr>
<td>30.24</td>
<td>1428.57</td>
<td>642.86</td>
<td>1607.14</td>
<td>7.44</td>
<td>1428.57</td>
<td>642.86</td>
<td>1607.14</td>
</tr>
<tr>
<td>32.54</td>
<td>1428.57</td>
<td>642.86</td>
<td>1928.57</td>
<td>7.85</td>
<td>1428.57</td>
<td>642.86</td>
<td>1928.57</td>
</tr>
<tr>
<td>34.38</td>
<td>1428.57</td>
<td>642.86</td>
<td>2250.00</td>
<td>8.29</td>
<td>1428.57</td>
<td>642.86</td>
<td>2250.00</td>
</tr>
<tr>
<td>23.72</td>
<td>1428.57</td>
<td>964.29</td>
<td>321.43</td>
<td>6.47</td>
<td>1428.57</td>
<td>964.29</td>
<td>321.43</td>
</tr>
<tr>
<td>25.87</td>
<td>1428.57</td>
<td>964.29</td>
<td>642.86</td>
<td>6.72</td>
<td>1428.57</td>
<td>964.29</td>
<td>642.86</td>
</tr>
<tr>
<td>28.21</td>
<td>1428.57</td>
<td>964.29</td>
<td>964.29</td>
<td>7.24</td>
<td>1428.57</td>
<td>964.29</td>
<td>964.29</td>
</tr>
<tr>
<td>30.46</td>
<td>1428.57</td>
<td>964.29</td>
<td>1285.71</td>
<td>7.74</td>
<td>1428.57</td>
<td>964.29</td>
<td>1285.71</td>
</tr>
<tr>
<td>32.75</td>
<td>1428.57</td>
<td>964.29</td>
<td>1607.14</td>
<td>8.19</td>
<td>1428.57</td>
<td>964.29</td>
<td>1607.14</td>
</tr>
<tr>
<td>34.76</td>
<td>1428.57</td>
<td>964.29</td>
<td>1928.57</td>
<td>8.66</td>
<td>1428.57</td>
<td>964.29</td>
<td>1928.57</td>
</tr>
<tr>
<td>36.98</td>
<td>1428.57</td>
<td>964.29</td>
<td>2250.00</td>
<td>9.08</td>
<td>1428.57</td>
<td>964.29</td>
<td>2250.00</td>
</tr>
</tbody>
</table>

Figure 5.3: Example of pipeline pressure drop data

During a visit to the StatoilHydro Research Center in Porsgrunn, the methods was
validated by StatoilHydro personnel. The methods of data generation give a real representation of the WPC and pipeline pressure drop in the production system at Troll West (Alstad, V., 2009).

5.2 Testing Scheme for Comparing Models

To make decisions regarding which model should be used during the later stages of this master thesis, a thorough testing scheme needs to be developed. Since the model of Hagem & Torgnes is based on the same problem as the model of Vestbø & Walberg it is relevant to compare solution value and solution times. However, it is important to stress that neither of the solution values for the models are considered as correct answers. They are nevertheless used to identify the advantages and disadvantages of both models.

Solution time and solution value are the most important criteria when testing or comparing algorithms. With this in mind there is one important aspect that should be considered when designing test cases. A single test case should consist of a set of input parameters impacting the solution value and/or solution time of the model. To be able to draw conclusions we first establish reference data for each test case, meaning an accurate solution value with a high resolution in the piecewise linearization. Using this reference data as a benchmark, the resolution in each model can be reduced until the solution value is no longer within a certain acceptable limit of the benchmark. Assessing the performance of the models can be done by identifying which model has the shortest solution time given this acceptable accuracy.

StatoilHydro considers it important to test scenarios relevant for the current production system at Troll West (Dueñas Díez, Marta., 2008a). Since the proposed model is adapted to match certain characteristics for the production at Troll West, this field will also be the main focus during testing. However, some general tests are also performed to determine the applicability of the model for other fields.

As described in 4.3, the purpose behind the new model is to try to utilize fluid characteristics to reduce the size of the piecewise linearization of the pressure drop. Consequently, the testing will focus on the parameters affecting the number of interpolation points used in this linearization. As described in section 1.4.1 and 1.4.2, different decomposition methods for solving problems with multiple well clusters have been investigated. Since the changes introduced in the model of Hagem & Torgnes concern the piecewise linearization of each cluster, the tests are run on a single cluster problem only. This is to simplify the testing process and the analysis.
5.3 Testing Implementation

5.3.1 Choice of Modeling Language

Both models are implemented using the Mosel language and solved using the Xpress-Optimizer solver (Dash Optimization, 2007a). AMPL was also considered a candidate implementation language but was rejected after weighing the advantages and disadvantages of using AMPL and the open source solver COIN-OR compared to Mosel and Xpress-Optimizer (Bell Laboratories, 2009). The initial reason COIN-OR was considered as an approach was that it is open source and can easily be run on multiple processors. There were however some uncertainty concerning whether it can effectively handle SOS2 sets in more than two dimensions, which is central for this problem. Through the department of Industrial Economics and Technology Management, we have access to a cluster consisting of high-end computers with processors of up to 8 CPUs. As one of the focus areas later in this master thesis is to utilize parallel processing, the fact that this cluster runs Xpress was the main reason for choosing Mosel as the implementation language.

5.3.2 Xpress Implementation

Appendix B contains all Xpress code written during implementation. Comments throughout the code have been added to facilitate understanding. The model of Hagem & Torgnes is only implemented for a single cluster even though the mathematical model in chapter 3 can handle multiple clusters.

5.3.3 Hardware Specifications

Hardware
- Intel Core 2 Duo E6700 2.6 GHz, 4GB ram, Windows XP Service Pack 3

Software
- Data generation
  - Petroleum Experts - GAP Multiphase System Optimization version 7.4
  - Microsoft Excel 2007
- Optimization
  - Editor: Xpress-IVE Version 1.19.00
  - Solver: Xpress Optimizer Version 19.00.00
5.3.4 File Structure

The file structure of the Xpress implementation is shown in figure 5.4. Input data is read in from three .dat files providing necessary information regarding cluster data "case_data.dat", WPC data "data_well.dat" and pipeline pressure drop data "data_pipe.dat". Xpress solves the models and writes the results to results.dat.

Figure 5.4: File structure of Xpress implementation describing the data passed to and from the Mosel files

5.3.5 Testing Application

To be able to run a large set of test cases and observe trends in solution value and solution time, an automated testing application is created. This application is similar to the one developed in Hagem and Torgnes (2008) but is also more complex since it includes interaction with GAP. The basic structure of how this application works is shown in figure 5.5. Up to 500 different test cases can be entered into the "Test case data" spreadsheet. Due to instability in Xpress solution time, two equal instances of each test case are run to extract average solution time. For each test case, the test case input is written to two different spreadsheets which again calls the GAP simulation software to calculate WPCs and pipeline pressure drop as described in section 5.1.2. These calculations are based on Troll West cluster data provided by StatoilHydro. With a high resolution in the piecewise linearization, this operation is very time consuming.

The GAP calculated datasets are then written to "case_data", "data_well" and "data_pipe" .dat files in a distinct folder named after the test case number in the "Test summary" spreadsheet. Following this, the Xpress-Optimizer is called to solve model of Hagem & Torgnes and the model of Vestbo & Walberg based on the generated input data. The results for each test, including the most important solution variables, are stored in a
result file. This procedure is repeated for all the test cases defined in the "Test case results" spreadsheet, where all the results are gathered in the same spreadsheet.

5.4 Model Testing Setup

To observe the initial behavior of the models and help us in further testing, we first create a large set of test cases in which we vary the gas and water handling constraints. This was performed to find the levels of gas and water that make the handling capacity constraints binding in the solution.

After running all these test cases and inspecting the results, we continued by setting up
test cases that were tuned for testing different problem characteristics. After discussions with StatoilHydro and staff at Engineering Cybernetics, a list of criteria was developed which the test case solutions should fulfill. This also required a more thorough examination of physical properties of the production system. The main criteria we established are given below.

- Both the water and gas handling capacity or just the gas handling capacity should be active constraints in the solutions

Currently the gas handling capacity in the first stage separator limits the production at Troll West. Other fields also have the water handling capacity as an active constraint, and it is expected that this will happen at Troll West as the reservoir matures (Dueñas Díez, Marta., 2008a).

Although we have not further investigated cases where neither the gas or water handling capacity are active constraints, the large set of initial test cases included such scenarios as a general test of the models.

- The solution should have at least four wells open, including at least one connected to the second manifold.

This is to better emulate realistic solution characteristics of the Troll West field and to ensure that the pressure drop model between the first and second manifold has an effect on the solution. StatoilHydro engineers are reluctant to close wells due to the reasons described at the end of section 3.3. As a result they may choose to produce flow from wells giving a solution the models would regard as suboptimal (Gunnerud, V., 2008). A larger number of open wells will thus be a more realistic solution.

- At least one and preferably two of the choke valves that control the pressure in the well should be fully open.

The impact of fully open wells can most easily be understood through an example as shown in figure 5.6.
5.4. MODEL TESTING SETUP

The pressure at the separator, \( P^{SEP} \), has a fixed value. As long as the pressure for a certain well, \( P^{WELL} \), is kept constant, the flow from the well is also constant as the size of \( P^{WELL} \) determines the flow from the well. Different pressure drop models can give different values for the pressure drop. Given that a new model gives a different pressure drop, we will explain why the criteria that at least one well is fully open should be fulfilled for the new model to have an effect on the flow.

If no wells are fully open, the following relationship is fulfilled for all wells in the system.

\[
P^{SEP} < P^M < P^{WELL}
\]

If the pressure drop, \( \Delta P \), increases by using a new model, but increases less than the difference between \( P^{WELL} \) and \( P^M \), the same flow can be maintained by increasing the choke valve opening, thereby keeping \( P^{WELL} \) constant. The increased pressure drop from using a different pressure drop model will then not have an effect on the flow.

If on the other hand we have the relationship in equation 5.2 fulfilled for at least one well, a model giving a different pressure drop will have an effect on the solution. The well in which this relationship holds, is called a fully open well.

\[
P^{SEP} < P^M = P^{WELL}
\]

If the new pressure drop model gives an increased \( \Delta P \), the current solution cannot be maintained, as \( P^M \) would then be greater than \( P^{WELL} \). To satisfy the pressure conditions \( P^{WELL} \) needs to be decreased as the choke valve cannot be adjusted. If at least one well is fully open a model change will thereby have an effect on the flow.
As the solutions vary slightly from case to case in a set of tests, these requirements cannot be guaranteed for all cases. They will however be fulfilled for the high resolution test cases and also most of the other test cases.

5.4.1 Common Production System Parameters

During testing we hold a number of parameters constant. These parameters concern the topology of the well clusters and are set to resemble the production system at Troll West.

Number of clusters: 1

Number of manifolds per cluster: 2

<table>
<thead>
<tr>
<th></th>
<th>Manifold 1</th>
<th>Manifold 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>No of pipelines</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>No of wells</td>
<td>4</td>
<td>4</td>
</tr>
</tbody>
</table>

Table 5.1: Topology description of the cluster used during testing

This system layout is similar to the one shown in figure 1.4.

5.4.2 Test Batches

With the criteria listed above, we create two test batches. The first test batch consists of test cases with only the gas handling capacity as a binding constraint in the solution. The second test batch consists of test cases with both gas and water handling capacity as binding constraints in the solution.

Within each test batch, we start by observing solution value as a function of the number of interpolation points in the model of Vestbø & Walberg. As described in section 5.2, by comparing with the benchmark solution value it is possible to find the resolution that gives a sufficiently accurate solution value. We further want to observe which resolution is needed in the model of Hagem & Torgnes to achieve the same accuracy. Observing differences in solution times at these resolutions will give us a basis for comparing the performance of the models.

Figure 5.7 shows an example of a test batch. The first test case is the benchmark solution used for solution value comparison. Test cases 2 – 11 tests development in solution value as the number of water cut interpolation points in the model of Hagem & Torgnes is increased. Test cases 12 – 21 tests development in solution value and solution time as the number of interpolation points are changed in this model and the model of Vestbø & Walberg. To isolate this effect, the number of water cut variables is kept fixed at 5.
5.5. MODEL RESULTS

5.5.1 Clarifications

The results presented in this section show the effect of changing the number of interpolation points and water cut points for the tested models. As explained in section 4.3.1, the piecewise linearization of pipeline pressure drop has gas, liquid and water cut as variables in the model of Hagem & Torgnes, and gas, oil and water as variables in the model of Vestbø & Walberg. When referring to changing the number of interpolation points in the model of Hagem & Torgnes, we mean changing the number of points in the liquid and gas interpolation dimensions, and in the model of Vestbø & Walberg changing the number of points in the gas, oil and water interpolation dimensions. Changing the number of water cut points then means changing the number of points in the water cut interpolation dimension in the model of Hagem & Torgnes.

The results will also often refer to distance from benchmark solution as a measure of solution accuracy. If the gap between the solution and the benchmark solution is positive, this means that the test case oil production is lower than the benchmark oil production.

![Table of test case input and interpolation points](image)

Figure 5.7: Example of a test batch. Each test case consists of a number of parameters that can be varied.
5.5.2 Initial Testing Results

Compared to the tests using artificial production data in Hagem and Torgnes (2008), initial testing showed that using real production data was more complex in a number of ways. Firstly, each test case run was much more time consuming than when using artificial production data. As stated earlier, simulating pressure drop for each flow combination using GAP is also very time consuming.

During initial testing, we also had some difficulties obtaining equal solution values using the two models. At most, the high resolution solution values for the two models differed by about 0.82%. However, they showed signs of converging towards the same value, but the resolution needed to obtain this value was too great to allow further investigation. Test cases with more than $30 \times 30 \times 30 = 27,000$ interpolation points for gas, oil and water for example took more than 72 hours to solve in addition to 12 hours of pressure drop data simulation. Running multiple tests with this resolution level was therefore too time consuming.

To confirm our observation that the models converged towards the same solution value, several other tests were also performed to ensure that the two models behaved identically with respect to modeling the pressure drop. Unlike the results in Hagem and Torgnes (2008), the accuracy of the solution is very dependent on the resolution in the piecewise linearization of the water cut linking constraints presented in section 4.3.3. At least 20 interpolation points is needed to provide an accurate solution and this contributes to a substantial increase in solution time.

Even though the models do not produce exactly the same solution values, the tests showed that the pressure drop linearization in the two models behave similarly and that they converge towards the same solution value. The test cases presented below are thus suitable to investigate the solution time and solution value performance characteristics of the two models.

As described in section 5.4, the pressure drop model has no effect when no wells are fully open. This is because the pressure drop model does not impact the solution. The testing confirm this as the models provided equal solution values when no wells were fully open in the solution. These test cases are therefore uninteresting for comparing the two models.

5.5.3 General Results - Gas Constrained

Graph 1 and 2 in figure 5.8 show how solution time and solution value changes as the number of water cut interpolation points are increased in the model of Hagem & Torgnes. Graph 1 indicates that solution time is unstable and varies from 70 seconds to about 1300 seconds. Graph 2 shows that the solution value is stable at around 6% gap to benchmark solution regardless of the number of water cut points. The number of interpolation points is kept constant at 15.
Graph 3 and 4 show how solution time and solution value changes as the number of interpolation points are increased in both the model of Hagem & Torgnes and the model of Vestbø & Walberg. The number of water cut points is kept constant at 5. Graph 4 indicates that the model of Hagem & Torgnes consistently provides a lower gap than the model of Vestbø & Walberg. With 20 interpolation points, the model of Hagem & Torgnes has a gap of 1.09% to its benchmark solution while the model of Vestbø & Walberg has a gap of 2.8%. Graph 3 shows that the model of Hagem & Torgnes also has a consistently lower solution time. At 20 interpolation points this model has a solution time of 260 seconds compared to 358 seconds for Vestbø & Walberg’s model. Since the gap to the benchmark is lower at 20 points for the model of Hagem & Torgnes, it is reasonable to assume that the difference in solution time would be even greater if the model of Vestbø & Walberg had to achieve the same gap.

Figure 5.8: Graphs showing the solution time and solution value for both models in the gas constrained test batch. The blue graph represents the model of Hagem and Torgnes (2008) while the red graph represents the model of Vestbø and Walberg (2008)

5.5.4 General Results - Gas and Water Constrained

Graph 5 and 6 in figure 5.9 show how solution time and solution value for the model of Hagem & Torgnes changes as the number of water cut points are increased. Graph 5 indicates that solution time is also very unstable and varies from 200 to 1000 seconds. Graph 6 indicates that solution value is stable at 2% gap regardless of the number of water cut points.
CHAPTER 5. MODEL TESTING

Graph 7 and 8 show how solution time and solution value changes as the number of interpolation points are increased in both the model of Hagem & Torgnes and the model of Vestbø & Walberg. Graph 8 shows in this case that the model of Hagem & Torgnes has a higher gap than the model of Vestbø & Walberg. Solution time displayed in graph 7 is also higher for the model of Hagem & Torgnes except for the 20 interpolation points test case where the solution time suddenly decreases from 738 seconds to 209 seconds.

Figure 5.9: Graphs showing the solution time and solution value for both models in the gas and water constrained test batch. The blue graph represents the model of Hagem and Torgnes (2008) while the red graph represents the model of Vestbø and Walberg (2008)

5.6 Model Analysis

5.6.1 Solution Value

Tests on artificial production data in Hagem and Torgnes (2008) reveal that accurate solution values can be achieved with a very low resolution in the piecewise linearization of the pipeline pressure drop. The testing presented above shows that solutions using real pressure drop data requires a much higher resolution to reach the same accuracy.

Figure 5.10 shows how the pressure drop in the pipeline from the last manifold and up to the separator behaves with changing liquid and gas flow. The water cut is fixed at
By inspecting the solution values more closely, we observed that the flow in some of the pipelines lie in an area in proximity to the uneven area in the bottom left corner of figure 5.10. Even though the solution is not in the uneven area, the fact that it is close will likely affect the time it takes the Xpress-Optimizer to choose the right weighting on the SOS2 variables and also reduce the accuracy of the solution. The piecewise linearization therefore might need a higher resolution to approximate the real pressure drop. The fact that the two models use different flow variables for the interpolation dimensions and places the points differently will therefore have an impact on the solution value. This can explain both the need for high resolution and the 0.8% difference in solution values in the models.

StatoilHydro has indicated that the uneven parts in the bottom right corner of figure 5.10 might not be realistic solution values. Hence, to further reduce solution time, preprocessing of the pressure drop data could be performed. Removing the less well-behaved parts in the area where the solution will not lie will likely reduce the level of resolution needed in the piecewise linearization and the effort required when choosing SOS2 weighting points in the solution process. In addition, a larger amount of interpolation points could be concentrated in the areas where it is expected the solution will lie. This will
improve the accuracy while keeping the model size constant.

Figure 5.11 shows how the pressure drop varies using artificial production data as used in Gunnerud and Langvik (2007), Vestbø and Walberg (2008) and Hagem and Torgnes (2008). This graph is much smoother and easier to approximate with piecewise linearization. The number of interpolation points will therefore not impact the solution value significantly. This leads to the two models producing equal solution values which were observed in Hagem and Torgnes (2008).

![Figure 5.11: Artificial production data: Plot of approximated pressure drop data for pipeline from manifold to separator with water cut of 73%](image)

Another very important observation when using real data is the fact that the solution value is largely independent of the number of water cut points. This confirms the assumption that water and oil behave similarly in the pressure drop model and that this relationship therefore can be described with few interpolation points.

### 5.6.2 Solution Time

The Xpress-Optimizer makes use of Branch&Bound (B&B) when solving the models, and it was observed that the Xpress performed much more branching for integer solutions when solving on real production data.

As more SOS2 constraints are relaxed in the root node of the branching tree when the number of interpolation points increase, it might be harder for B&B to reduce the optimality gap fast if the procedure is not successful when choosing its first branches. This occurrence becomes even more frequent for the model of Hagem & Torgnes when the resolution of the WCL is increased as we will explain in section 5.6.2.
5.6. **MODEL ANALYSIS**

**Effect of Linearizing the Water Cut Linking Constraint**

As commented in section 5.5, increasing the number of interpolation points in the piecewise linearization of the water cut linking constraint (WCL) in the model of Hagem & Torgnes increased solution time, and we observed from Xpress-Optimizer that the number of branching moves in the B&B procedure increased significantly. After further analysis of our reformulation, this is believed to be a result of a special situation arising when we linearize a function that is a product of two variables and thereby partly convex, partly concave.

As mentioned in 4.3.3, we reformulate the LHS, \( q_{lims}^{\text{PIPE}} f_{int}^{\text{PIPE}} \), in equation 4.6 in two steps as it is a function of two variables. The reformulation includes utilizing the following relationship:

\[
\begin{align*}
    u &= \frac{x + y}{2} \\
    v &= \frac{x - y}{2} \\
    xy &= \left(\frac{x + y}{2}\right)^2 - \left(\frac{x - y}{2}\right)^2 = u^2 - v^2
\end{align*}
\]  

This was done to obtain a separable function. After the reformulation the LHS is on the form of a Hyperbolic parabolic function \( z = u^2 - v^2 \), which can be linearized by linearizing each of the two quadratic functions \( u^2 \) and \( v^2 \). The Hyperbolic paraboloidal function is partly convex, partly concave. Because of this, using piecewise linearization to approximate the function is problematic. It can be more easily understood by thinking of the problem in the context of the two quadratic functions that make up the function. Irrespective of whether one wants the sum of the functions maximized or minimized one will want to maximize one of the quadratic functions while minimizing the other, as they have opposite signs. Assuming we want the sum \( u^2 - v^2 \) maximized, we would want \( u^2 \) as large as possible and \( v^2 \) as small as possible. This increases the necessary branching as we approximate the function using piecewise linearization, and Xpress-Optimizer handles the SOS2 variables using B&B. In the B&B procedure the SOS2 constraint, that only two neighboring break points on the piecewise linear curve can be chosen (allowed to be greater than zero), is relaxed. As the procedure given the described situation wants to make \( u^2 \) as large as possible, the two first SOS2 variables that will become active in the linearization of \( u^2 \) are the variables associated with the two break points furthest apart, highlighted in red in figure 5.12. Choosing these returns the highest value for \( u^2 \). However, as this is an infeasible solution, more branching becomes necessary as one is guaranteed that the first solution will not be feasible. The same situation can be shown to arise whether we are minimizing or maximizing. Utilizing the relationships in equation 5.3 to handle the expression \( q_{lims}^{\text{PIPE}} f_{int}^{\text{PIPE}} \) is therefore expected to contribute towards increasing the solution time.
There are several other ways in which the LHS in equation 4.6 could have been reformulated. The function could be transformed into a separable function using logarithms. One then utilizes that $\log(xy) = \log(x) + \log(y)$, to obtain a separable function. However, this is also not always straightforward as Beale (1975) experienced. A third option, that does not require a separable function, is to linearize the expression using the same principle we used when linearizing the pressure drop model for the pipelines; that is, to extend the concept of piecewise linearization and SOS2 sets to several dimension using chains of SOS2 sets, as we discussed in section 2.2.

5.6.3 Model Performance Comparison

As described in section 5.5, the model of Hagem & Torgnes produces more accurate solutions with a lower solution time when only gas is constrained. When both gas and water are constrained, this model provides a consistently higher gap. The solution time is also higher and more unstable compared to the model of Vestbø & Walberg. Based on this it is hard to assess the performance of the models without taking the characteristics of a certain oil field into account. For a field such as Troll West, where the gas handling capacity for the moment is the only binding constraint, the results point to the model of Hagem & Torgnes performing better. That is, given an acceptable gap to the benchmark solution, it has a lower solution time. As stated in section 5.4, future production scenarios for Troll West include both binding gas and water handling constraints where the model formulation of Vestbø & Walberg would perform better.

5.7 Model Intuitiveness and Generality

StatoilHydro has stressed that intuitiveness is very important when evaluating the models (Dueñas Díez, Marta., 2008a). A model should be easy to understand and the process of
obtaining a solution transparent. With this in mind, the changes introduced in the model of Hagem & Torgnes is complicating. Both the objective function and the water handling constraint become harder to understand since flow of oil and water are not explicit variables. The water cut linking constraint and its linearization further complicates the model by increasing the number of constraints and introducing an extra piecewise linearization. Therefore, with respect to model intuitiveness the model of Vestbø & Walberg is considered better.

As described in section 2.2, another important aspect to consider is which model can handle changes more easily. It is likely that the model of Hagem & Torgnes will handle the introduction of absolute pressure as described in section 3.2 better. This is because a linearization of the pressure drop in the pipelines gains an extra dimension, giving this model an advantage in terms of the total number of variables. On the other hand, implementing this might be more difficult with this model since the piecewise linearization is more complicated to understand.

As explained in section 4.3, the model of Hagem & Torgnes assumes constant water cut for each well. This makes this model more problem specific and less general as this is not necessarily an assumption that is valid for other fields.

5.8 Conclusion and Model Choice

Accurate solution values with few water cut points means that water and oil do behave similarly in the pipelines. Even though this leads to a reduction in the number of interpolation variables, the gain in solution time is offset by the complicating changes introduced in the model of Hagem & Torgnes. Our analysis of the linearization of the water cut linking constraint introduced in Hagem and Torgnes (2008) showed that a different way of linearizing the constraint would probably perform better. A different linearization of this could therefore be interesting to investigate further. Although the model of Hagem & Torgnes outperforms the model of Vestbø & Walberg when only gas is constrained, it is slower and less accurate when both gas and water are constrained.

Model intuitiveness weighs heavily when choosing which model to proceed with. The fact that the model of Hagem & Torgnes is more complicated to understand and introduces an extra piecewise linearization makes it less likely to be used by StatoilHydro. Further, we will scale the model of choice to represent the entire field with multiple clusters. This problem will be solved using decomposition methods, and we consider the model of Vestbø & Walberg more suitable since it is simpler to understand, modify and test. In addition, it is interesting to compare the results from this master thesis with the results in Vestbo and Walberg (2008), which means that the flow representation should be kept the same.

Thus, based on these arguments we chose to proceed with a flow formulation using gas, oil and water as used in the model of Vestbø & Walberg.
Chapter 6

Solution Methods

This chapter presents the different solutions methods that were considered for solving the full field size production allocation problem formulated in chapter 3. The first part of the chapter concerns different decomposition methods to utilize the special structure of the problem. In the second part, the formulation of the chosen decomposition method is presented together with a discussion of how parallel processors can be utilized to reduce the solution time of the problem. In the last section we discuss different solution methods for finding an integer solution to the problem.

6.1 Problem Characteristics

The problem formulation given in chapter 3 leads to a problem with a block angular structure as described in section 2.3.2. The common constraints are the handling capacity of gas and water at the first stage separator. Each cluster of wells correspond to one of the blocks, $A_1$-$A_n$, in figure 2.2, and can be solved independently if the common capacity constraints are ignored. As explained in section 3.2.4, the size of the problem is very much dependent on which resolution is used in the piecewise linearization of the pressure drop. Tests on real pressure drop data from Troll West, described in section 5.6, show that a fairly high resolution is needed to obtain a satisfactory accuracy. Solving a single cluster on real data therefore takes considerably more time than solving a problem on artificial production data. Both Gunnerud and Langvik (2007) and Vestø and Walberg (2008) reported that using a standard solution method to solve a full field size problem on artificial production data was not an attractive approach as the gap between the upper and lower bound was still large after 12 hours of running the solver. As the problem takes considerably longer to solve using real data, solving a full field size problem with a standard solution method is not considered a possible solution approach. Given the block-like structure of the mixed integer linear program (MILP), a decomposition approach as described in section 2.3, is an interesting solution approach.
6.2 Selection of Solution Method

As mentioned in section 1.4, Lagrangian Decomposition (LD), Benders Decomposition (BD) and Dantzig-Wolfe Decomposition (DWD) was investigated in Gunnerud and Langvik (2007) and Vestbø and Walberg (2008). Our evaluation of the different methods is partly based on a review of the results they obtained, and a review of the theory behind the different methods, as presented in section 2.3. One of the objectives of this thesis is to investigate solution methods suitable for parallel solving. Whether a solution method supports this is therefore an important part of the evaluation. Throughout our cooperation with StatoilHydro the importance of having a robust solution method has been stressed. This will therefore also be an important evaluation criteria.

As mentioned in section 2.3.3, BD requires one to decide on a set of variables to fix to create a master problem and a subproblem. If one was to use BD to solve the production allocation problem a central decision would therefore be the choice of complicating variables to fix in the subproblem. The most natural variables to fix are the integer variables in the problem which are the routing variables and the variables that are SOS2 sets. As discovered in Vestbø and Walberg (2008), fixing these variables leads to a master problem and subproblem that is very hard to solve, in addition to making it hard to find initial solutions. The structure of the problem is not on a form on which BD has most commonly been used. Additionally, a BD formulation of the problem has a master problem and a single subproblem. BD can therefore not be used to utilize parallel solving. Combined, the mentioned issues render BD an unattractive solution approach for the MILP presented in chapter 3.3, and it will therefore not be investigated further.

A LD of the production allocation problem requires, as explained in 2.3.1, one to decide on a set of constraints to relax. The most natural constraints to relax are the common capacity constraints. For a general allocation problem relaxing these can lead to unbounded subproblems. However, the production from each well is for our problem limited by the pressure drop difference that can be obtained from a well to the manifold it is connected to. Relaxing the common capacity constraints leads to n independent subproblems, where n is the number of clusters in the problem. The subproblems can then be solved in parallel, using one solution process for each subproblem. As we mentioned in 1.4.2 some challenges arise when implementing LD for the MILP presented in chapter 3. A heuristic to update the Lagrange multipliers is highly problem specific and requires in depth knowledge of the problem and the solution method. However, for problems in which the multiplier is well tuned, Vestbø and Walberg (2008) reported good results.

Given the block angular structure of the problem and that it was for problems with this structure that DWD was designed, it is natural to consider DWD, presented in section 2.3.2, as a solution method. Central to DWD is the choice of constraints to include in the master problem and subproblems. By placing the capacity constraints in the master problem, the subproblems can, in the same way as with LD, be solved independently and in parallel. However, one cannot find a solution to the MILP presented in chapter
6.3. DANTZIG-WOLFE DECOMPOSITION

Choosing the handling capacity constraint as the common constraint and placing it in the master problem is a natural choice when implementing DWD for the MILP presented in chapter 3. Colombani and Heipcke (2006) state that one should ideally have as few common constraints as possible. By placing the handling capacity constraint in the master problem we ensure that there are maximum two common constraints, one representing gas capacity and a potential second, representing water capacity. Decomposing the problem in this way also makes it easier for planners to understand the model, as it becomes conceptually similar to the way in which StatoilHydro optimize the production at the field today - that is each subproblem of the entire field problem correspond to one well cluster.

To allow the separation of the problem in to a master problem and a number of subproblems, a redefinition of the problem variables is necessary. How such a variable transformation is done depends on certain properties of the problem. If the solution space given by the constraints is convex, a solution can be described by a convex combination of the
extreme points. Such a variable transformation is called the convexification approach (Vanderbeck, 2006). However, as the model presented in chapter 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 solutions of the subproblem. This is called the discretization approach and is based on the fact that integer polyhedra can be generated from a finite set of feasible integer solutions plus a non-negative integer linear combination of extreme rays (Wolsey and Nemhauser, 1999), (Vanderbeck and Savelsbergh, 2006).

For general MILPs the discretization approach is used for the integer variables, while the convexification approach is used for the continuous variables. As the MILP presented in chapter 3 only has binary integer variables (well routing and SOS2 variables) there are however no internal integer solutions in the solution space. There will therefore be no difference between applying the discretization approach and the convexification approach to the MILP (Vanderbeck and Savelsbergh, 2006).

6.3.1 Restricted Master Problem

Objective Function

Using the convexification approach, the production level of gas, oil and water in the optimum mixed integer solution of a subproblem is stored in an array $Q^{PROD}_{icp}$. This array represents a candidate column. 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, $\mu_{ic}$, that represent the candidate column is created. $C(i)$ in equation 6.1 is the set of candidate columns for cluster $i$. Having introduced these, the objective function for the restricted master problem (RMP) can be formulated.

$$\max Z_{RMP-LP} = \sum_{i \in I} \sum_{c \in C(i)} Q^{PROD}_{icp} \mu_{ic} , p = \{o\}$$

(6.1)

The oil production for a certain cluster is in this formulation thereby a convex combination of the candidate solutions, represented by the index $c$, that have been sent from the subproblems.

Common Constraints

As discussed in the previous section, the common constraints in the RMP will be the handling capacity constraint of gas and potentially water if this also is a limited resource. After the variable redefinition, these constraints will be on the following form:

$$\sum_{i \in I} \sum_{c \in C(i)} Q^{PROD}_{icp} \mu_{ic} \leq C^{TOT}_p , p = \{g, w\}$$

(6.2)
To ensure that the optimum solution to the RMP is a sum of convex combinations of subproblem solutions the following two constraints are needed.

\[ \sum_{c \in C(i)} \mu_{ic} = 1 \quad \forall \; i \in I \]  

(6.3)

\[ \mu_{ic} \geq 0 \quad \forall \; i \in I, \; c \in C(i) \]  

(6.4)

As we mentioned earlier there are no guarantees that the RMP solution will be a feasible solution to the original problem as a convex sum of integer solutions will not necessarily produce an integer solution. Figure 6.1 illustrates how the RMP would appear if the there were three clusters (three blocks in the block-diagonal part) and three candidate columns for each cluster (three columns in each block). The number of non-integer weighting variables, \( \mu_{ic} \), in the RMP solution depend on the number of constraints in the RMP. If the problem is only constrained on gas capacity there will be a maximum of two non-integer weighting variables. If the problem is constrained on gas and water, as illustrated in figure 6.1, there will be a maximum of four non-integer weighting variables, and consequently a maximum of two cluster that have solutions that are infeasible with respect to the MILP presented in 3.3. The reason this occurs is that the number of constraints in the RMP determines the size of the basis. The non-basic variables are fixed to 0 or 1 in an optimal solution while the basic variables can take any value between 0 and 1. If a better solution can be obtained by interpolating between two columns, two clusters will therefore as illustrated in figure have their associated weighting variables at fractional values. The reason the RMP has to be solved as a LP is that this is the only way that the dual values that drive the DWD procedure can be obtained.

\[ \begin{bmatrix} Q_{11g}^{\text{PROD}} & Q_{12g}^{\text{PROD}} & Q_{13g}^{\text{PROD}} \\ Q_{11w}^{\text{PROD}} & Q_{12w}^{\text{PROD}} & Q_{13w}^{\text{PROD}} \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} Q_{21g}^{\text{PROD}} & Q_{22g}^{\text{PROD}} & Q_{23g}^{\text{PROD}} \\ Q_{21w}^{\text{PROD}} & Q_{22w}^{\text{PROD}} & Q_{23w}^{\text{PROD}} \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} Q_{31g}^{\text{PROD}} & Q_{32g}^{\text{PROD}} & Q_{33g}^{\text{PROD}} \\ Q_{31w}^{\text{PROD}} & Q_{32w}^{\text{PROD}} & Q_{33w}^{\text{PROD}} \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} \mu_{ic} \\ \mu_{ig} \\ \mu_{iw} \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0.3 \end{bmatrix} \begin{bmatrix} C_{g}^{\text{TOT}} \\ C_{w}^{\text{TOT}} \end{bmatrix} \]

Figure 6.1: Illustration of a fractional solution in the restricted master problem
6.3.2 Subproblem

The dual values for the gas constraint, \( \pi^{CAP}_g \), the water constraint, \( \pi^{CAP}_w \), and the convexity constraint, \( \pi^{CONVEX}_i \), are sent to the subproblems and used in the subproblem objective function. The dual values of the capacity constraints are the price that each cluster have to "pay" for using one unit of the limited resource (gas and potentially water). Taking these into account, the subproblem objective function will be on the following form:

\[
\max Z_i = \sum_{l \in \mathcal{L}(i)} (q^{PIPE}_{imlo} - \pi^{CAP}_g q^{PIPE}_{imlg} - \pi^{CAP}_w q^{PIPE}_{imlw} - \pi^{CONVEX}_i) \quad \forall \ i \in \mathcal{I}, \ m = \{1\} \tag{6.5}
\]

The subproblem constraints are equations 3.5-3.29, which are all the constraints in the original MILP with the exception of the handling capacity of gas and water.

As mentioned in section 2.3.2, a subproblem solution is only sent to the RMP as a candidate column if its reduced cost, \( \bar{r}_i \), is positive. According to Tebboth (2001), the reduced cost to subproblem solution can be formulated as:

\[
\bar{r}_i = \sum_{l \in \mathcal{L}(i)} (q^{PIPE}_{imlo} - q^{PIPE}_{imlg} \pi^{CAP}_g - q^{PIPE}_{imlw} \pi^{CAP}_w - \pi^{CONVEX}_i) \quad \forall \ i \in \mathcal{I}, \ m = \{1\} \tag{6.6}
\]

This is equal to the objective function of the master problem with the exception of the extra RMP convex constraint dual value. The first factor 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 factors 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 described standard DWD-algorithm solves the subproblems in sequence as shown in figure 6.2.
6.3. DANTZIG-WOLFE DECOMPOSITION

Figure 6.2: Illustration of sequential solving of subproblems

6.3.3 Column Management

As described in section 2.3.2, the method of column generation is used to generate new columns to the RMP. Within DWD there are however several different column management strategies. Before starting the DWD algorithm one first needs to decide who to generate the initial set of columns. To be able to create a heuristic that can produce a set of initial solutions a thorough understanding and analysis of the problem one is trying to solve is needed. Studies have shown that this can be worth the effort as the DWD algorithm can often be accelerated by starting with a good set of pre-computed columns, given that they can be generated cheaply (Tebboth, 2001). A strategy to create columns solutions for allocation problems of the type modeled in section 3.3 is covered in section 7.3.1.

Once a method for initializing the algorithm has been chosen the next step is to choose a strategy for managing columns during the DWD-algorithm. A strategy often used is to send more than just the optimal solution of a subproblem as a candidate column to the RMP and to remove uninteresting columns from the RMP. In such a strategy, the $n$ best integer solutions encountered could be added when solving the subproblem and columns from the RMP removed if they have a very high unfavorable reduced cost (Desrosiers, 2005). A strategy can also be employed that takes advantage of the fact that the subproblems do not necessarily have to be solved to optimum. If the subproblems are solved in parallel the RMP can for example be resolved and new dual information sent to all subproblems when the first solution with a positive reduced cost is found. This does not change whether the algorithm will converge or not. Another strategy suggested in Larsen (2004) is to keep track of how long a column is part of the basis and to remove it from the RMP if it has not been part of the basis for a certain number of iterations.

As DWD typically has been used for pure LPs, different column management strategies are mostly aimed at reducing the solution time. Adding more columns than just the column representing the best integer solution can however also be used to try to generate
better integer solutions when the original problem is a MILP. As we explain further in section 7.3.6, this might improve the integer solution because the number of integer columns in the RMP, the size of the column pool, then will be greater.

Since the first implementations of DWD, the results have been positive with respect to reducing the solution time by generating multiple columns from each subproblem on each cycle of the DWD-algorithm (Tebboth, 2001). Here this method of adding extra columns will be referred to as "Extra columns". This strategy of adding the $n$ best integer solutions encountered when solving the subproblems as MILPs is believed to require minimal extra computation as it just requires that the encountered integer solutions are stored. It will increase the size of the RMP. However, the solution time of the RMP solved as an LP is, after adding these columns, still thought to be insignificant compared to solving the MILP subproblems for the production optimization problem. This strategy can be especially efficient if the subproblems are computationally expensive to solve (Barnhart et al., 1996). The implementation of this column proposal strategy is presented in section 7.3.2.

6.3.4 Termination Criteria

Decomposing a problem using DWD and utilizing column generation is essentially the same as using a specialized simplex method in which only a few of the non-basic variables are handled at any point, while the remaining variables are fixed to zero (Tebboth, 2001). The different ways in which the DWD-algorithm can terminate is therefore comparable to when the simplex algorithm terminates. If a feasible integer solution cannot be found for one of the subproblems, the whole problem is infeasible. The proof of optimality with respect to solving the RMP as an LP is the same as for the simplex algorithm: when no more variables (columns) with a positive reduced cost can be found, the current solution cannot be improved and optimum has been reached.

Instead of continuing the algorithm until a solution can be proved to be optimal, an alternative is to terminate the algorithm if the gap to the optimal solution, $Z^*_{RMP-LP}$, is smaller than a user defined limit. An upper bound and an optimality gap then needs to be calculated in every iteration. The upper bound, $Z^{UB}_{RMP-LP}$, can be calculated using the sum of the reduced cost from the subproblems in the last iteration, $rc_i$, and the corresponding objective function value of the RMP, $Z_{RMP-LP}$ (Tebboth, 2001).

$$Z^{UB}_{RMP-LP} = \sum_{i \in I} rc_i + Z_{RMP-LP} \geq Z^*_{RMP-LP} \qquad (6.7)$$

$$Z^{UB}_{RMP-LP} = \sum_{i \in I} \sum_{I \in \mathcal{L}(i)} (q_{imlo}^{pipe} - q_{imlg}^{pipe} \pi^g_{CAP} - q_{imlw}^{pipe} \pi^w_{CAP}) - \pi_i^{CONVEX} + Z_{RMP-LP}$$
This calculation requires all the subproblems to be solved to optimum. As the problem investigated in this thesis is a MILP the above calculated optimality gap will only apply to the relaxed LP version of the problem. To calculate the right optimality gap an IP solution needs to be created in each iteration. This is explain further in section 7.3.6.

6.3.5 Parallel Solving

The independent subproblems can, as described in section 2.3.2, be solved in parallel to reduce solution time. The basic idea is to utilize independent solution processes, such as CPUs, concurrently. Tebboth (2001) states that the RMP should be solved in a master process that also coordinates the solution of subproblems in independent slave processes. The master process starts by solving the RMP based on an initial set of columns. It then distributes dual values to all subproblems and startes the processes that solve them. Figure 6.3 illustrate the steps in every iteration.

After all subproblems have terminated, the candidate columns are sent to the RMP and the process repeated if the optimality criteria has not been satisfied. Tebboth (2001) describes the strategy of solving all subproblems before the RMP is solved as the "Basic strategy".

As described in section 2.3.2, Speedup ($S_p$) is the ratio of the total sequential solution time ($T_s$) to the total parallel solution time ($T_p$) and efficiency ($E_p$) is the ratio of the speedup to the number of parallel processes ($p$). Gnanendran and Ho (1993) also defines a measure termed load balancing as the percentage of process busy time to the total time in the processes. Perfect load balancing occurs when all processes are utilized 100%. The Speedup and Efficiency in this state is shown in equation 6.8 and equation 6.9.
\[ S_p = \frac{T_s}{T_p} = \frac{1}{\frac{1}{p}} = p \]  
\[ E_p = \frac{S_p}{p} = \frac{p}{p} = 1 \]

The nature of decomposition methods makes these theoretical limits unreachable. In the Basic strategy, the slave processes running the subproblems are not started before the master process is finished. The master process also has to wait for the final subproblem, termed the "bottleneck" problem, to finish. To achieve Perfect load balancing, identical solution times for all subproblems and zero solution time for the RMP are required. Since no problem then will wait for others to finish, full utilization of the parallel processes is obtainable. This is not the case for the production optimization problem where the subproblem solution times vary greatly due to different cluster characteristics. As stated above, this means that many of the slave processes will be idle while the RMP waits for the final slave process to finish solving the "bottleneck" subproblem. The Speedup and Efficiency will therefore be lower than the theoretical limit.

The DWD formulation of the problem has some special characteristics. The RMP has few constraints and variables and therefore a solution time very close to zero. This makes it possible to solve the RMP more frequently without affecting the total solution time. Based on this, it is interesting to investigate more advanced parallelization strategies that better utilize the slave processes and solves the RMP more frequently. Tebboth (2001) mentions certain strategies which change the order in which the subproblems and the RMP are solved on each cycle. A promising approach for our problem is called the "Accelerated feedback strategy". It is described as a strategy where the subproblems and the RMP are constantly being solved. Candidate columns from subproblems are sent to the RMP when each problem terminates and the RMP is immediately solved and sends out dual values to all subproblem that are not 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. We have developed the Accelerated feedback strategy further to match the production optimization problem.

Reviewing this strategy, it is important to consider how many parallel processes are available compared to the number of subproblems. When the number of parallel processes are greater than or equal to the number of subproblems, the Accelerated feedback strategy resolves the RMP every time a subproblem terminates and restarts it with the updated shadow price immediately. However, when there are fewer parallel processes than the number of processes, a queue for the currently unprocessed subproblems have to be created together with rules for which subproblem should be taken out of the queue and solved next.

In the example we here will use there are four parallel processes working on a RMP with eight subproblems. Figure 6.4 shows the first iteration of the Accelerated feedback
strategy. As in a basic parallel DWD strategy, the RMP is solved with initial columns and dual values are distributed to all subproblems. When the fastest subproblem terminates, the strategy resolves the RMP. The updated dual values are distributed to the subproblems not currently being solved and the next available subproblem is started in the slave process. This continues until no subproblem has a positive reduced cost. The algorithm will then be in a state where some of the subproblems have been solved in previous master iteration 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. Section 7.3.4 gives a more detailed description of how this is implemented.

![Figure 6.4: Illustration of the first iteration of the Accelerated feedback strategy](image-url)
6.3.6 Obtaining an Integer Solution

As explained in section 6.3.1, the optimum DWD solution is unlikely to satisfy the integer requirements of the problem. Finding a feasible integer solution to the original problem therefore requires more than a pure DWD-procedure. Given the structure of the problem a DWD of the problem is however a good starting point for finding an integer solution. One way of utilizing the DWD decomposition to find an integer solution is to first find the optimal DWD-LP solution and then make use of the columns that are in the RMP when the optimal LP solution is found. By resolving the RMP with binary requirements on the weighting variables an integer solution can be found using B&B and Simplex. As it is the RMP with a limited set of columns that is resolved, this will be a heuristic procedure with no guarantees of optimality, though an optimality gap can be calculated after the algorithm has terminated. In chapter 7.3.6, this heuristic is discussed further with respect to finding integer solutions to the MILP presented in section 3.3. If the GAP between the best IP solution and the LP solution is satisfactory small after running the heuristic the algorithm can be terminated. If the gap is significant when the heuristic terminates an exact solution procedure can be explored.

6.4 Branch & Price

Similarly to the heuristics outlined in the paragraph above, a DWD is the starting point for using Branch and Price (B&P) as a solution method. As mentioned in chapter 2, B&P was originally developed for solving large IPs with a block angular structure. More recently positive results have however also been reported on using the method to solve large MILPs (Degraeve and Jans, 2003). In B&P, as in DWD, the most central decision when trying to solve a large problem is determining how to decompose the problem. The arguments that were used in favor of creating a DWD of the problem by placing the capacity constraints in the RMP are also valid when choosing how to decompose the problem to solve it using B&P. Once the decision of how to decompose the problem has been taken the most central decision in an implementation of B&P is to decide which tree search strategy to use and which rule to decide what variable or constraint to branch on. These two topics are covered in the following two sections.

6.4.1 Tree Search Strategies

Possible tree search strategies for B&B were discussed in section 2.4.1. These rules are also applicable to the tree search that is done in B&P. The purpose behind starting a B&P algorithm after the RMP has been solved to LP optimality is to be able to guarantee a solution within a specified gap. In B&P, this is usually obtained by combining a depth first strategy, to obtain integer solutions, with a best first strategy to explore promising nodes. However, as integer solutions can be found very fast in each node of the tree
using the heuristic presented in the previous section, a pure best first strategy will be employed.

### 6.4.2 Branching Rules in Literature

Amor and Valério de Carvalho (2005) mentions three criteria that a branching rule should fulfill. First, the branching rule should not introduce intractable changes in the structure of the subproblems. That is, the optimization problem should generally remain the same throughout the optimization. Second, the branching rule should divide the solution space into two mutually exclusive parts, a so called dichotomy which will eliminate symmetry. Symmetry is detrimental to a B&B search. In a tree with symmetry, the same solution may exist in different forms in several nodes, something which would reduce the value of branching. The last criterion Amor and Valério de Carvalho (2005) mentions is that the branching rule should also balance the set of solutions evenly among the branches to ensure that both branches are useful.

Intuitively, branching on the weighting variables using so called 0/1-branching is an attractive approach. It is simple to implement in the RMP and clearly cuts of infeasible parts of the solution space. This branching strategy can however often lead to an unbalanced B&B tree which again leads to excessive enumeration (Ryan and Foster, 1981). This is because the 0-branch will have a limited effect on finding a solution as most columns essentially have their associated weighting variable at value 0 in the optimal solution. The subproblems have no information of which columns that have had their associated weighting variables set to 0 in the RMP. Another problem that arises with this branching strategy is therefore that subproblems will tend to reproduce columns that have been sent to the RMP in prior iterations. Summarized, 0/1-branching is believed to work poorly for the production optimization problem.

A different approach frequently suggested for problems with a block angular structure is to branch on the original integer variables indirectly using a sum of variables (Barnhart et al., 1996). The large number of integer variables in the original problem in the form of routing and SOS2 variables however makes this very challenging for the production optimization problem. To ensure a feasible integer solution a large number of constraints would need to be added. This would make it necessary to solve a large number of nodes. Finding the optimal DWD-LP solution in each node is however not trivial and requires significant computation for a full field size problem. Adding constraints to ensure the feasibility of the SOS2 variables would also not be trivial and would require adding a set of constraints for each branching.

### 6.4.3 Choosing Branching Rule

Given the issues that the traditional branching rules presented in the previous section leads to, we suggest branching on the production volume of gas and water in each cluster.
This strategy is inspired by a branching strategy first introduced in Gunnerud, Nygreen, McKinnon and Foss (2009). The constraints necessary to enforce this branching rule can be introduced implicitly to the RMP when new child nodes are created. Not adding the branching constraints explicitly comes with the advantage of not introducing new dual variables that have to be dealt with in the pricing problem (Barnhart et al., 1996). The columns with a production volume that violates the branching constraints, marked by the red and green squares in the root node in figure 6.5, are then removed from the RMP in the child nodes. Column 3 in the red square is removed from the RMP in the green "Up"-child node while columns 1 and 2 are removed from the RMP in the red "Down"-child node. To ensure that the removed columns are not again introduced to the RMP, a new constraint, $C_{ip}^{\text{MAX}}$ or $C_{ip}^{\text{MIN}}$, is also added to each subproblem that is branched on. This will at the most increase the size of the subproblem basis by four variables as a $C_{ip}^{\text{MAX}}$ and $C_{ip}^{\text{MIN}}$ constraint can potentially be added for each phase. However, given the large size of the subproblems, this is believed to have a small impact on the solution time. Removing columns from the RMP and introducing new constraints to the subproblems will lead to a new set of candidate columns, represented by the columns in bold in figure 6.5, column 10-15, being sent to the RMP in the next iteration.

As opposed to branching on an integer variable, branching on a continuous variable will not significantly reduce the upper bound. The objective of the suggested branching rule is instead to reduce the optimality gap mainly by increasing the lower bound. As we explain further in the next paragraphs, our hypothesis is that the optimal solution to the MILP presented in section 3.3 is very close to the optimal LP RMP solution. This enables us to mainly search for better integer solutions and to partly ignore the search for lower upper bounds. The heuristic introduced in section 6.3.6 and further described in section 7.3.6, that resolves the RMP as an IP, will be used to produce an integer solution after the DWD-algorithm has converged to an optimal LP RMP solution. The solution that the heuristic produces is what is referred to as the "DWD-IP heuristic" solution in figure 6.5.
6.4. BRANCH & PRICE

Figure 6.5: Illustration of the three first nodes in the B&P tree, including the root node and the two first child nodes

The position of the squares in figure 6.6 represents the level of oil production in a certain subproblem solution, also known as a candidate column. Our hypothesis is that if a optimum LP RMP solution has a production level for one cluster that is an interpolated solution between two or more columns, in figure 6.6 represented by columns \( a \) and \( b \), then the optimum IP RMP solution will have a production level that is close to the interpolated solution, marked by \( X \) in figure 6.6. The hypothesis is that setting the branching bound to the production level in the optimum LP master solution will likely in the next iteration lead to the subproblem in the DWD algorithm suggesting a new column, represented by column \( c \) in figure 6.6, that has a production volume close to the interpolated optimum LP solution. This new column could then potentially increase the IP master solution and reduce the optimality gap.
CHAPTER 6. SOLUTION METHODS

Figure 6.6: Illustration of an interpolated solution between two columns, represented by the teal squares, and a new column sent from a subproblem, represented by the pink square.

Figure 6.7 illustrates how the branching bounds, that are added to the subproblems and used to remove columns from the RMP, are obtained. $C_{ip}^{\text{MAX}}$ and $C_{ip}^{\text{MIN}}$ are set to the production level in the interpolated solution, marked by the $X$ in the figure. As illustrated in figure 6.5, $C_{ip}^{\text{MAX}}$ is used to remove columns that are larger, meaning a higher gas or water production, than the bound in the "Down" node. Similarly $C_{ip}^{\text{MIN}}$ is used to remove columns that have a production that is lower than the bound in the "Up" node. The pink square in figure 6.7 is an example of the production level of gas in a new column that has been sent from the subproblem that was branched on.

Figure 6.7: Illustration of an interpolated solution and the effect of adding a bound.

This branching rule is thought to satisfy two out of Amor and Valério de Carvalho (2005)’s three criteria for a good branching rule. It is a dichotomy of the solution space defined by
the constraints in the MILP presented in section 3.3, and therefore eliminates symmetry in the B&B tree. In addition, the master and subproblems will remain mainly the same optimization problem as constraints are introduced implicitly to the RMP and only a few new constraints are added to the subproblems. The two child nodes that are created during branching are not thought to be equal with respect to reducing the optimality gap and the criterion of Amor and Valério de Carvalho (2005) that the solution set should be evenly divided among subtrees is not satisfied. Given that we believe that the optimum IP solution is close to the interpolated solution, the node in which we branch down on is, according to our hypothesis, more likely to produce a good integer solution that will increase the lower bound. This is because the optimum IP solution in a node will always have a production volume that is equal or lower to the optimum LP solution in the same node, and because according to our hypothesis it is likely that in the IP solution the limited resources will be allocated to clusters in a similar way as in the LP solution. This can however be utilized in the tree search by giving a preference to nodes in which we branch down if two nodes are equal with respect to the best first strategy.

Three central decisions in the suggested branching rule are: which cluster should be branched on, which phase should be branched on and what should the branching bound be set to when two new child nodes are created. The latter was covered in the paragraph above. What guides these decisions is our overall main objective with the branching: to obtain better integer solutions. If the production optimization problem is only constrained on one resource, the two first decisions only have one option each. As commented in 6.3.1, a single capacity constraint in the RMP leads to a fractional solution for at most one cluster, which then will be the cluster that is branched on. Similarly, if the production volume is only constrained on one phase (gas/water), this will be the phase that is branched on.

Choosing Which Cluster to Branch on

For a production optimization problem that is constrained on both gas and water, rules are needed to decide what to branch on. The cluster that is branched on will be a cluster that has fractional values for the weighting variables in the optimum LP solution. If there are two clusters with this feature we suggest branching on the cluster in which the difference between the LP solution, marked by X in figure 6.6, and the lower of the two columns that the interpolation is between, marked by \(b\) in the same figure, is largest with respect to oil production. It is believed that this is the better choice because such a cluster will benefit the most from having a new column introduced, represented by the pink square, that will likely have a production volume that is close to the interpolated LP RMP solution for that cluster. To investigate the importance of choosing the right cluster to branch on and to verify this rule it is compared with randomly choosing which cluster to branch in section 9.6.2.
Choosing Which Phase to Branch on

The choice of which phase to branch on is decided after the decision of which cluster to branch on has been made. We believe that branching on the phase in which the difference in production between the two columns that have fractional weighting variables is largest, will be a good branching rule. Two characteristics of how the problem is solved are thought to contribute towards this being a good choice.

First, in section 6.4.3 we explained the likely effect of branching on continuous production variables. Branching on phase $p$ in cluster $c$ will according to our hypothesis lead to the column suggested from subproblem $c$ in the next iteration having a production of phase $p$ close to or on the bound sent to the subproblem.

Second, the heuristic used to create an IP solution for the RMP only utilizes the columns that are currently present in the RMP. As maximum one column can be chosen in an IP solution of the RMP, we believe it is more likely that the column with the lowest production, out of the two columns that an interpolated solution was created from, will be chosen. This is because the interpolated solution is a part of an optimal LP solution that in most cases has a total production equal to the maximum handling capacity of the different phases. A solution containing the upper column is therefore more likely to be infeasible. If the difference between the interpolated columns is large, a higher production will potentially be "lost" by choosing the column with the lowest production. We believe that most can be gained from creating a new column that has a production close to the interpolated solution for the phase that was branched on. This is a simplified explanation that builds on several assumptions. The performance of this branching rule will therefore be compared with a B&P algorithm that randomly chooses which phase to branch on.
Chapter 7

Implementation

The three main strategies presented in the previous chapter, sequential Dantzig-Wolfe, parallel Dantzig-Wolfe and Branch& Price were all chosen to be implemented for the production allocation problem presented in section 3.3. The sequential DWD code is the starting point for the two other implementations and will be used to measure the potential speedup of the parallel DWD code. The B&P-algorithm is implemented to be able to guarantee solutions within a specified optimality gap to the production allocation problem. In addition to the three main solution methods a standard search strategy, described in the next section, was implemented to serve as a basis for comparison.

The first section in this chapter covers the implementation of the standard solution procedure and the different DWD solution strategies, while the second section covers the implementation of the B&P algorithm presented in section 6.4. The complete Mosel code for the global, DWD and B&P solution strategies can be found in appendix B.

7.1 Implementation of Standard Solution Method

The standard implementation of the problem solves the MILP presented in section 3.3 using the Simplex and Branch & Bound algorithms embedded in the Xpress-Optimizer. Hence, our standard implementation mainly contains definitions of datasets, variables, constraints and the objective function as presented in section 3.3. These definitions are followed by instructions to the solver to maximize the objective function. The Xpress-Optimizer then solves the LP relaxation of the problem and initiates Branch & Bound if the solution is not integer. The results are written to a result file including the values of the most important variables and the total solution time of the model.
7.2 Implementation of Dantzig-Wolfe Decomposition

In contrast to the global solution procedure, a DWD of the problem renders it necessary to run several models that can communicate with each other. The submodels needs to communicate with the master model, and information about dual values and subproblem solutions needs to be sent from and stored in the different models. Given that one of the objectives of this thesis is to develop an algorithm suitable for parallel solving of the subproblems, it is also necessary that the DWD-subproblems can be solved in parallel in the implementation framework chosen. The Xpress module \texttt{mmjobs} described in chapter 7 in Dash Optimization (2007b) and in Colombani and Heipcke (2006) is central to a DWD implementation of the problem in Mosel. This module enables the concurrent running of several Mosel files and the passing of information between them.

An alternative to using \texttt{mmjobs} to manage information between models is to use a Message Passing Interface created in a C++ framework. This interface is typically used in a cluster environment where multiple computers work together to solve optimization problems. It allows a master process to send input files and run Xpress models on slave computers. Information is passed by writing to and reading from text files. This framework thus allows for a DWD implementation where separate computers solve the subproblems and the RMP. This approach is transparent since the user can specifically assign submodels to certain solver processes at a specific time. More specific parallel strategies can thus be developed and also tuned to match the characteristics of the problem and only the number of computers connected in the cluster limits the number of subproblems that can be solved concurrently.

The production allocation problem has a limited number of subproblems defined by the number of production clusters in the problem. For the problems that we investigate the number of subproblems will rarely exceed eight. These problems can be solved on a single computer with eight CPUs using \texttt{mmjobs}. Therefore, we decided to implement \texttt{mmjobs} and not the more complex C++ framework since there was no need to connect multiple computers together to solve the DWD.

7.2.1 Events in \texttt{mmjobs}

\texttt{mmjobs} utilizes an event system to manage when and how information should be sent between models. Events are characterized by a value and a class. Every model has an associated event queue, and events may be exchanged between a submodel and its parent model using these queues. The events in the queue are managed using a FIFO (First In - First Out) policy (Colombani and Heipcke, 2006). \texttt{send} and \texttt{wait} are two central events for the communication between the master- and submodels. Wherever there is a \texttt{wait} event in one of the models, the model will pause until it has received a value from a \texttt{send} event in the other model. The \texttt{send} and \texttt{wait} events can thereby be used to control the progress of, and communication between, the master-and submodels.
7.2. IMPLEMENTATION OF DANTZIG-WOLFE DECOMPOSITION

7.2.2 Memory Drivers

In addition to the event system, and essential part of being able to run one model from another, and several models in parallel are two IO drivers that are defined by the module mmjobs: the shmem driver and the mempipe driver. The shmem driver is meant for one-to-many communication and would be the only driver necessary if the submodels where solved consecutively. That is, in the case of one model writing and one model reading. To allow several models to concurrently write to memory the mempipe driver is needed. It is meant for many-to-one communication but could be used instead of the shmem driver, though it is more complicated to use.

The shmem shared memory driver is used in a similar way as reading from and writing to physical files. The exchange of data between different models is carried out through initializations blocks and the send and wait events are used to coordinate when information should be initialized to and from the memory driver. The sending of dual values from the RMP to the subproblems is one-to-many communication. As the shmem driver is easier to use and understand, this information is sent from the master model to the submodels using shmem initialization blocks.

The mempipe shared memory driver works in the opposite way compared to the shmem driver. First a "pipe" needs to be opened before it can be written to. That is, initializations from has to be called before initializations to. If several submodels are running in parallel there will be several submodels that want to send information to the master model at the same time. To handle this many-to-one communication the mempipe is necessary.

7.2.3 File Structure for the Dantzig-Wolfe Implementations

The different sequential and parallel DWD strategies have all been implemented in separate Mosel files. This simplifies comparison of the different models and renders the code easier to read. Figure 7.1 is an illustration of the Mosel files and procedures used in the sequential DWD implementation of the production optimization problem. The file structure is basically the same for all the different DWD solution strategies. The master_problemHT_sequential file is the file in which the RMP is solved and that coordinates when and how the subproblems, represented by sub_problemHT, are solved. For simplicity all the declarations are gathered in declarationsHT and included in both the master and sub files. The sub_constraintsHT file contains all the DWD-subproblem constraints of the model presented in section 3.3. That is, all the constraints of the MILP presented in section 3.3 except the handling capacity constraints of water and gas. The dotted lines represent information being sent between the master model and the submodels. The dual values of the master problem constraints are sent to the subproblems while information about which subproblem was solved and its potential to improve the RMP is sent the other way.
Figure 7.1: File structure for the sequential DWD implementation. The other DWD algorithms have the same structure.

7.2.4 Procedures

The text inside the two squares in figure 7.1, headed by master_problemHT_sequential and sub_problemHT, represents the different procedures in the Mosel files. These procedures were written to make it easier to understand the code and to avoid having to write the same sequence of code lines repeatedly. Each of them is thoroughly commented in the code in appendix B. Introducing them have no influence on the running time of the algorithm.

7.2.5 Compilation and Initialization of Submodels

The subproblems are compiled, loaded and run from the master file using the set of commands shown on page 107.

In the forall loop, one submodel is loaded and started for each DWD-subproblem using the load and the run commands. The same submodel Mosel file is used for all clusters. However, the model is dependent on an input parameter, i, which determines which subproblem that is run and what data should be used when data is loaded from input files in the submodels.
7.2. IMPLEMENTATION OF DANTZIG-WOLFE DECOMPOSITION

The wait event pauses the master model until a message/event has been received from the submodel that was started with the run command on the line above. At the start of the submodel file there is a send(SUB_READY, 0) event, which sends a value, SUB_READY, telling the master model that the submodel is running. The dropnextevent command then removes this event from the event queue so that the queue becomes empty and ready for new events. After the last SUB_READY message has been received and dropped the master model and all the submodels are running.

Result:= compile("g","sub_problemHT.mos")

forall(i in Cluster) do
  load(Sub(i),"sub_problemHT.bim")
  MODEL_ID(getid(Sub(i))) := i
  run(Sub(i),"CLUSTER= " +i)
  wait
  dropnextevent
end-do

7.2.6 Controlling Information Flow

Master Model

Information also needs to be sent between the master-and submodels during the DWD-algorithm. The code lines below are a simplification of the Mosel code for the sequential-DWD implementation, and illustrates in general terms how information is sent and received in the master model when running submodels in parallel.

The send(Sub(i), SOLVE_PROBLEM, 0) event in the first line in the forall loop sends a value, SOLVE_PROBLEM, which is interpreted in the submodels as a command to start solving the DWD-subproblems. The master model then pauses on the wait event until an event is received from the submodel that was started.

forall(i in Cluster) do
  send(Sub(i), SOLVE_PROBLEM, 0)
  initializations from "mempipe:sol"
  CLUSTER
  IMPROVED
  end-initializations
  wait
  master_ev:= getnextevent
  wait
  if(getclass(master_ev)=PROBLEM_SOLVED) then
    process_sub_result
  else
    FINISHED:= true
exit(0)
end-if
end-do

Submodel

The following Mosel code is a simplified version of the code in the submodel that receives and processes the SOLVE_PROBLEM command value:

repeat
  wait
  sub_ev:= getnextevent
  MESSAGE:= getclass(sub_ev)
  PROBLEM_TYPE := MIP
  if MESSAGE=SOLVE_PROBLEM then
    initializations from "raw:"
    HANDLING_CAP_DUAL as "shmem:HANDLING_CAP_DUAL"
    CONVEX_DUAL as "shmem:CONVEX_DUAL"
    end-initializations
    maximize(Oil_Production_Sub(CLUSTER))
  elif MESSAGE=TERM_PROBLEM
    EXIT:=true
  end-if
  get_solution_status
  if(SOLUTION_STATUS = OPTIMAL) then
    send(PROBLEM_SOLVED,0)
  else
    send(SUB_INFEASIBLE,0)
  end-if
  if (MESSAGE<>TERM_PROBLEM) then
    process_solution_sub
  end-if
until EXIT

After the subproblem has terminated, the submodel sends a value, depending on the solution status of the subproblem, to the master model using a send event. For example, if a feasible optimum solution was found for the subproblem, a PROBLEM_SOLVED message value is sent. In the master model, the characteristic value of the send event is obtained from the attached event queue using the getnextevent command. Depending on the value of the message, the master model then executes a number of commands and procedures. If the subproblem was solved successfully the information about the subproblem’s solution is obtained through a mempipe initializations block that was opened for writing before the master model paused on the first wait event in the Mosel code on page 108.
A flow chart illustrating how the event system and memory drivers are used to control the flow of the algorithm is shown in figure 7.2. The numbers are a representation of how the algorithm progresses. Arrows with same number represent commands/procedures that occur at about the same time. The black arrows represent an event being sent between the models. An important thing to notice is that the initializations from block is executed before the initializations to block.

Figure 7.2: Illustration of how events are used to control the communication between master-and submodels

### 7.3 Dantzig-Wolfe Algorithm

Figure 7.3 gives a general overview of the DWD-algorithm as it was implemented in the sequential- and parallel-DWD code attached in appendix B. After the subproblems have been compiled, loaded and run as described in section 7.2.5, a set of initial columns are created. Once an initial set of columns has been created, the RMP is solved and the dual values of the master problem constraints sent to the submodels. The subproblems are then solved and the solution status of each problem checked. If one of the subproblems does not return a feasible solution, the entire problem is infeasible and the algorithm terminates. If all the subproblems find a feasible solution, the algorithm proceeds. The subproblems potential to improve the RMP solution is then examined. If no subproblems have a reduced cost greater than zero, an optimal LP solution has been reached and a heuristic to find an integer solution is started after terminating the submodels. This heuristic is described in detail in section 7.3.6. However, if any subproblem solutions that can improve the RMP are found, they are sent to the master model as candidate columns. The RMP is then resolved and the optimality gap recalculated using the procedure.
described in section 6.3.4. If the gap is larger than a user defined limit, the main DWD-cycle is repeated as shown in figure 7.3. If not, the subproblems are terminated and the heuristic to find and integer solution started. A more detailed overview of how the algorithm functions for each of the different DWD-strategies is given in section 7.3.4.

7.3.1 Finding an Initial Solution

In section 6.3.3 it was mentioned that a set of initial columns need to be created to solve the RMP in the first iteration. The number of initial columns needed to produce a basic feasible solution and dual values for the master problem constraints is equivalent to the number of constraints in the RMP. If the problem is constrained on both gas and water, the total number of columns needed is equivalent to two plus the number of subproblems. To make sure that enough columns are produced, two initial columns are created for each subproblem. Dual values for all the constraints will only be produced if all the constraints are active. This can be ensured by creating a set of starting columns that can be combined to give a total production that violate the gas and water handling capacity limits. The first initial column for each subproblem is set to have a production of 110

Figure 7.3: Flow chart for the implemented Dantzig-Wolfe algorithm
zero for all phases. The second column for each subproblem is created by solving each of the subproblems with constraints on the water and gas production. These constraints are set to a value that is slightly larger than the total allowable production of gas and water divided by the number of subproblems. The optimal initial RMP solution should then be restricted on both gas and water and have a production for some of the well clusters that is an interpolation between two initial columns. Ideally, both initial columns for each subproblem should be columns that are good candidates for columns that would be selected in the optimum RMP solution. However, columns that have a production of zero for all phases are very cheap to create compared to solving the subproblems to obtain a better initial candidate column. It is therefore believed that using the mentioned strategy will be more efficient. If a minimum production is required for all clusters in the final solution the zero production column cannot be chosen in the final RMP and has to be removed once the column pool in the RMP has reached a reasonable size.

### 7.3.2 Column Management

The standard way of finding columns that potentially can improve the RMP solution, is to solve the subproblems to optimum and then send a column representing this solution to the RMP. This strategy was the first strategy that was implemented and is used as the standard strategy.

As discussed in section 6.3.3, a speedup of the DWD-algorithm can often be obtained by employing the Extra column method. This means adding more than just a column representing the best subproblem solution. The Xpress `mmxprs` solver utilizes a B&B search to solve the mixed integer subproblems (Dash Optimization, 2007a). A computationally cheap way of obtaining more candidate columns in every subproblem iteration is to store the integer solutions that are encountered in the B&B search. This is achieved by using the Mosel command `setcallback(XPRS_CB_INTSOL,"store_int_sols")`. This command starts the procedure `store_int_sols` every time a MIP solution is found in the B&B-tree. This procedure stores the production of each phase in the MIP solution that was found. When the optimal subproblem solution is found, this and the other stored intermediate solutions are sent to the master model as candidate columns. Adding the intermediate columns that do not have a positive reduced cost will not influence the convergence speed of the DWD-algorithm. The reason these are also added is explained in section 7.3.6. Adding extra columns to the RMP in every iteration will not change that fact that the RMP is computationally very easy to solve compared to the subproblems. No procedure to remove columns from the RMP was therefore needed.

### 7.3.3 Implementation of Sequential Dantzig-Wolfe

The Mosel code presented on page 7.2.6 is, as mentioned in section 7.3.3, the code for the sequential DWD-implementation. The sequence of `send` and `wait` events makes the code
sequential. A \texttt{SOLVE\_PROBLEM} value is sent to the first submodel with a \texttt{send} event that then starts the solving of the corresponding subproblem. While the first subproblem is being solved, the master model is paused on a \texttt{wait} event. When the subproblem that was started terminates, a \texttt{send} event is used to continue the master model. Information about the subproblem solution is then stored and the process repeated for the next submodel.

### 7.3.4 Implementation of Parallel Dantzig-Wolfe

Our Mosel code can quite easily be changed from a sequential to a parallel implementation. However, this can only be done because the \texttt{mempipe} memory driver is used when the submodels send information to the master model. As mentioned in section 7.2.2, the \texttt{mempipe} driver facilitates many-to-one-communication. If the submodels were only to be solved sequentially, the \texttt{shmem} driver would be sufficient as only one submodel writes information to memory at the same time. When the right memory drivers are in place, the only change needed to make the submodels run in parallel is to move the \texttt{send(Sub(i),SOLVE\_PROBLEM,0)} event to its own \texttt{forall(i in Cluster)}-loop as illustrated in the code on page 113. When using \texttt{mmjobs}, each model loaded via the module is executed in a separate thread. Thus, on a computer with multiple CPUs, the threads are distributed over the available CPUs by the operating system and therefore run in parallel if the information flow between the master-and submodels is coordinated properly (Colombani and Heipcke, 2006).

#### Basic Strategy

The basic parallel strategy was explained in section 6.3.5. This solves all subproblems in parallel and waits for the slowest subproblem to finish before the RMP is solved and the next iteration is started. In the code below, the first forall loop starts all subproblems before moving to the first \texttt{wait} statement in the second forall loop. The master process then stops and waits for the fastest subproblem to terminate. The results from this are processed and the master process then waits for the next subproblem to terminate. This is repeated until the slowest subproblem has terminated.

```mosel
forall(i in Cluster) do send(Sub(i),SOLVE\_PROBLEM,0)
forall(i in Cluster) do
  initializations from "mempipe:sol"
  CLUSTER
  IMPROVED
  end-initializations
  wait
  master_ev:=getnextevent
```
if(getclass(master_ev)=PROBLEM_SOLVED) then
    process_sub_result
else
    FINISHED:= true
    exit(0)
end-if
end-do

Hence, the time in every iteration will be equal to the running time for the slowest subproblem, as opposed to in the sequential implementation, where it is equal to the sum of running times for all subproblems.

Accelerated Feedback

The accelerated feedback method is described in section 6.3.5 and illustrated in figure 6.4. This is a theoretical description and is based on the user manually assigning CPUs to different subproblems in the implementation. However, the mmjobs module used in our DWD implementation requires us to modify this strategy. As explained, the number of available CPUs must be taken into account when implementing accelerated feedback. This is not necessary when using mmjobs. This module distributes threads automatically over available CPUs and solves multiple threads on a single CPU more efficiently than when the sequence of threads is user defined.

Hence, the accelerated feedback implementation initially starts all subproblems regardless of the number of CPUs. The following steps in the algorithm are shown in figure 7.4. When the a subproblem terminates with a positive reduced cost, the RMP is immediately solved and the same subproblem restarted with the new dual values obtained from the RMP. The algorithm also stores the master problem iteration number in which the subproblem was started.

If a subproblem terminates with a nonpositive reduced cost, the subproblem is queued and not restarted. The algorithm then iterates through all queued subproblems and check if they have been solved in a master problem iteration earlier than 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 master problem dual values. The subproblem is therefore started in a new thread. This procedure is continued until no subproblems return a positive reduced cost and all have been solved with the newest set of master problem dual values.
7.3.5 Terminating the Dantzig-Wolfe Algorithm

There are basically three different ways in which the main DWD-cycle will be terminated in our implementation. The first, and standard criterion in the DWD-implementation, is that the cycle is terminated if no subproblems are able to find a solution with a positive reduced cost. The second criterion, as explained in 6.3.4, is to calculate an optimality gap and terminate the cycle if the gap is less than a user defined limit. This can be used to reduce the solution time of the DWD-algorithm if a gap larger than zero is satisfactory. To calculate an optimality gap in each iteration the upper bound calculation needs to be combined with the DWD-IP heuristic as an integer solution then is needed in each iteration. The last way in which the DWD-cycle can terminate is if the number of iterations exceed a specified number of maximum iterations. The optimality gap can then be found from the upper bound calculation presented in section 6.3.4.
7.3.6 Heuristic to Find an Integer Solution

As explained in section 6.3.6, the optimal DWD solution will most likely not satisfy the integer requirements of the original problem. However, the optimal DWD LP RMP solution is a good starting point for obtaining an integer solution. We therefore implemented a heuristic that resolves the RMP with SOS1 restrictions on the weighting variables. This is termed the DWD-IP heuristic. The solution this heuristic produces is the best solution that can be obtained based on the columns in the RMP when the DWD-procedure terminates. The DWD-procedure is believed to require a fairly limited number of iterations and the number of columns in the RMP is hence believed to be very small when procedure terminates. Compared to the size and solution time of the subproblems, the solution time of the DWD-IP heuristic will therefore be very low. For simplicity, the columns are ordered by the order in which they were found and a unique ordering of columns is used for each subproblem. A better way would probably be to order the columns according to increasing oil production. However, given the computational ease of solving the RMP as an IP, this was not implemented.

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. The procedure described in section 7.3.2, of adding extra columns for each subproblem in every iteration to obtain a speedup, could also make it possible to find a better integer solution when combined with the DWD-IP heuristic. Only columns that have a positive reduced cost could potentially improve the LP RMP solution. However, all the integer solutions that are encountered when solving the subproblems as MILPs could 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. If the subproblems provide a very large number of integer solutions, a limit could be put on the number of intermediate columns to suggest from each subproblem.

7.4 Implementation of Branch & Price

The B&P algorithm is implemented to guarantee that a solution of satisfactory quality can be produced for the MILP presented in section 3.3. The reasons and background for the most important decisions taken in the implementation of the algorithm were discussed in section 6.4. We will here only describe the file structure and briefly cover some of the changes that were made from the description of the B&P algorithm in section 6.4.

7.4.1 File Structure for the Branch & Price Implementation

Figure 7.5 is an illustration of the Mosel files and procedures used in the B&P implementation of the production allocation problem presented in section 3.3. The file structure
is similar to structure used for the DWD-implementation. The main difference is the extra procedures in the `BP_master_problemHT` file that coordination the creation and solving of nodes. `C_MAX` and `C_MIN` represent the bounds that are sent from the master model to the submodel that correspond to the subproblem that is branched on.

![Figure 7.5: File structure for the B&P implementation](image)

### 7.4.2 Implementation Decisions

An overview of the steps in the algorithm is given in figure 7.6. For a more detailed description the reader is referred to the B&P Mosel code with its comments, which can be found in appendix B.

No constraints have been added in the root node. The root node in the implemented B&P algorithm is therefore equivalent to solving the problem using the basic parallel DWD-algorithm. If the gap between the DWD-IP heuristic and LP solution of the RMP in the root node is less than a specified gap, the algorithm is terminated before any branching is started. As presented in chapter 1, there is a maximal acceptable gap between our proposed solution and the optimal solution for the problem we consider. Through discussions with StatoilHydro, it was decided that the maximum acceptable gap should be set to 0.1% (Alstad, V., 2009). The termination criteria was therefore set to this level.

The main deviation from a "normal" B&P implementation is that child nodes are created
immediately after a node has been solved. In addition, the cluster and phase to branch on is for these nodes decided before which node to branch on next has been selected. This was done for practical reasons when implementing the algorithm and does not have an effect on the speed of the algorithm.

To avoid that two solutions can exist in both the "Up" and "Down" branch when creating two new child nodes, the new bounds were respectively set to slightly higher and slightly lower than the interpolated solution. By not setting the bounds to the same value, a part of the solution space is cut away. However, if the bounds are set very close this will be an insignificant small section of the solution space.
Figure 7.6: Flow chart for the implemented B&P algorithm
Chapter 8

Computational Setup

A computational setup and evaluation is necessary to verify the correctness and to observe the performance of our implementation of the different solution methods presented in chapter 6. The development of concrete objectives, managing variables with significant impact and a thorough testing plan are mandatory features of a successful computational setup (Crowder et al., 1979). A computational setup should be developed before implementing the model in a solver environment. This chapter therefore briefly outlines the testing objectives along with a description of how testing is designed to meet them. The final part consists of a description of the different test batches along with a brief summary of the solution methods that are tested. This computational setup leads to a computational evaluation, and the results from this is presented and analyzed in chapter 9.

8.1 Testing Objectives

The performance of a solution method can be measured by its ability to provide accurate solution values with a low solution time. One of the goals of this thesis is to evaluate the performance of several different solution methods for the petroleum allocation problem. All tests performed during the computational evaluation will reflect the objectives as they are listed below.

- Compare the performance of Dantzig-Wolfe Decomposition (DWD) to the standard solution method
- Compare the performance of the sequential and parallel DWD strategies
- Investigate the need for and the performance of Branch & Price (B&P)

The computational evaluation should also validate the correctness of the DWD and B&P implementations.
CHAPTER 8. COMPUTATIONAL SETUP

Solution time and solution value are the most important criteria when testing or comparing the performance of algorithms. Testing the solution time and solution value of the sequential, and different parallel DWD solution strategies will provide us with useful information regarding the performance of each method when these methods are compared with a standard solution strategy. It is especially interesting to observe which strategy provides the lowest solution times and under which circumstances this occurs. The development in solution time for each parallel strategy as the number of CPUs available in the solver hardware changes, will also be important when evaluating the performance of the strategies.

The B&P algorithm is interesting to test in the cases in which the gap between the LP RMP and DWD-IP heuristic solution is not considered satisfactory when the DWD algorithm terminates. As branching on a continuous variable is a very little used branching strategy, the potential of the branching scheme put forward in section 6.4.2 is very interesting to test. The objective of the B&P testing will be to establish whether the algorithm can provide an optimal solution faster than the standard solver, and potentially whether the extra time to obtain a more accurate solution is worth the effort compared to the solution methods that do not guarantee an optimal solution within a specified limit.

8.2 Initial Testing

The model testing phase, described in chapter 5, provides important information regarding the tuning of certain test parameters for the computational evaluation. The results concerning the required resolution in the piecewise linearization of the pipeline pressure drop are particularly interesting. Section 5.5 explains that for a single cluster problem, at least 30 points per pressure drop interpolation dimension is needed to provide accurate solution values. The solution time with this level of resolution is very high. On average, our DWD implementation solves single cluster problems 50-60 times and Branch & Price solves them up to 400 times. A high resolution for each cluster therefore contributes to a very high solution time for the entire problem. For the tests presented later in this chapter, the resolution in the piecewise linearization was set to a much lower level making it possible to test the performance of the different solution methods within reasonable time. This reduces the accuracy of the solution value and separates it from a solution that can be implemented in a real production system. However, the low resolution tests will still be able to reveal the characteristics of the solution methods applied to production allocation problems.

Following this, tests that revealed which level of gas and water handling capacity that made the handling capacity constraints binding in the optimal solution were performed. An unconstrained handling capacity problem is of little interest and through consultations with StatoilHydro and personnel at Department of Engineering Cybernetics, we decided that both gas and water should be binding constraints in the solution. It is expected that
this will happen at Troll West as the field matures and it is also a more general scenario applicable to other petroleum fields (Dueñas Díez, Marta., 2008a). Results obtained on tests of problems that are constrained on both phases are also believed to be applicable for problems only constrained on one phase as problems of the former kind have the same characteristics but are more computationally challenging to solve.

Especially with respect to testing the potential of the B&P algorithm it is more relevant to test problems where both the gas and water constraints are binding. This is because the results obtained in Vestbø and Walberg (2008) using artificial production data indicate a higher difficulty in providing a good integer solution for problems that have both gas and water as binding constraints.

8.3 Test Data

In the following testing phase, we will refer to the terms test case and test batch frequently. A test batch is a single problem with a specific set of input parameters describing the production system and its capacities. Crowder et al. (1979) states that the same problem should be solved by each solution method investigated by the experiment. Hence, each test batch (TB) will consist of multiple test cases where each test case represents a run of a single solution method on a certain type of hardware. For each test case, the solution time and the gap between the upper ($Z_{UB}^{RMP-LP}$) and lower bound ($Z_{RMP-LP}$) is stored. These were defined in section 6.3.4.

Test problems can generally be divided into two categories - randomly generated and hand selected. Randomly generated problems are often not based on real-world behavior, and the parameters are sampled within the space of all problems. Hand-selected problems are usually based on real-world behavior, and the parameters are sampled within the space of all real-world problems (Crowder et al., 1979). The objective of this thesis is to evaluate different solution methods applied to production allocation problems, and not to evaluate the solution methods on a more general set of problems. Hence, realistic problems that resemble those of a petroleum production field are be created. Test batches are be hand-selected and the parameters tuned to realistic values. The test batch input data are based on two different types of parameters, fixed and non-fixed.

8.3.1 Fixed Test Batch Parameters

The 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 8.1.

Number of well clusters: 8

Number of manifolds in each well cluster: 2
Number of pipeline pressure drop interpolation points: 7

<table>
<thead>
<tr>
<th></th>
<th>Manifold 1</th>
<th>Manifold 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>No. of pipelines</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>No. of wells</td>
<td>4</td>
<td>4</td>
</tr>
</tbody>
</table>

Table 8.1: Common parameters for all test batches

Figure 8.1 illustrates the topology of a single well cluster. Seven similar clusters are also connected to the first stage separator.

Figure 8.1: Illustration of the topology for a single well cluster, including wells, manifolds and first stage separator

The Well Performance Curves (WPC) and pipeline pressure drop data are examples of other fixed parameters. As described in chapter 5, these data have been created from realistic production data provided by StatoilHydro.

8.3.2 Non-Fixed Test Batch Parameters

The non-fixed test batch parameters are changed between each test batch. As described in the beginning of this chapter, these parameters should have significant impact on the problem with regards to solution value and solution time. They should also be parameters that will vary in realistic production scenarios. The levels of the gas and water handling capacities are examples of non-fixed parameters. Both Troll West and other oil fields are exposed to changing handling capacity limits. Changing these levels will also move the pressure drop to a different area in figure 5.3 depicting pipeline pressure drop shown in section 5.1.2. This creates a different problem structure and changes both the solution time and solution value of each subproblem.
8.3. TEST DATA

8.3.3 Solution Methods

The different solution methods tested in the test batches are presented in table 8.2.

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standard</td>
<td>The Standard solution strategy was described in section 7.1, and uses a Branch &amp; Bound algorithm based on the Simplex method to solve the MILP.</td>
</tr>
<tr>
<td>Sequential</td>
<td>The Sequential solution strategy is the sequential DWD strategy, described in section 7.3.3, combined with the DWD-IP heuristic to produce an integer solution that solves the RMP as an IP, described in section 7.3.6.</td>
</tr>
<tr>
<td>Basic</td>
<td>The Basic solution strategy is described in section 7.3.4, and solves the subproblems in parallel as opposed to sequentially. It also uses the DWD-IP heuristic to produce an integer solution.</td>
</tr>
<tr>
<td>Accelerated feedback</td>
<td>The Accelerated feedback strategy is a parallel DWD solution strategy, as described in 7.3.4, that tries to decrease the idle time for a computer with multiple CPUs. Similarly for the other DWD strategies, a integer solution is produced by the DWD-IP heuristic.</td>
</tr>
<tr>
<td>Extra columns</td>
<td>The Extra columns solution strategy is the same solution method as the Basic strategy but where all the integer columns encountered when solving the subproblems are added. This was described in section 7.3.2.</td>
</tr>
<tr>
<td>Branch&amp;Price normal</td>
<td>The Branch &amp; Price (B&amp;P) normal solution strategy is the solution strategy that was described in section 6.4 and in section 7.4. It utilizes the rule for branching on phase and cluster that we believe work best.</td>
</tr>
<tr>
<td>Branch&amp;Price random cluster</td>
<td>The B&amp;P random cluster solution strategy is the same strategy as B&amp;P normal strategy but where which cluster to branch on is chosen randomly among the two clusters that are represented by an interpolation between two columns. If only one cluster is represented by an interpolated solution, the algorithm chooses to branch on this cluster.</td>
</tr>
<tr>
<td>Branch&amp;Price random phase</td>
<td>Similarly to B&amp;P random cluster, the B&amp;P random phase solution strategy is the same strategy as the B&amp;P normal strategy, except that which phase to branch on is chosen randomly.</td>
</tr>
</tbody>
</table>

Table 8.2: Overview of the different solution strategies that are tested

8.3.4 Test Batches

A total of ten test batches were created. These were created to test the robustness of the solution methods as changing the gas and water limit changes the properties of the
problem. Each test batch represents a problem with certain limits on the gas and water production. An overview of the gas and water limit for each test batch is given in table 8.3.

<table>
<thead>
<tr>
<th></th>
<th>Gas limit</th>
<th>Water limit</th>
</tr>
</thead>
<tbody>
<tr>
<td>TB1</td>
<td>16000</td>
<td>16000</td>
</tr>
<tr>
<td>TB2</td>
<td>18000</td>
<td>16000</td>
</tr>
<tr>
<td>TB3</td>
<td>20000</td>
<td>18000</td>
</tr>
<tr>
<td>TB4</td>
<td>21000</td>
<td>15000</td>
</tr>
<tr>
<td>TB5</td>
<td>22000</td>
<td>19000</td>
</tr>
<tr>
<td>TB6</td>
<td>24000</td>
<td>20000</td>
</tr>
<tr>
<td>TB7</td>
<td>25000</td>
<td>18000</td>
</tr>
<tr>
<td>TB8</td>
<td>26000</td>
<td>21000</td>
</tr>
<tr>
<td>TB9</td>
<td>28000</td>
<td>22000</td>
</tr>
<tr>
<td>TB10</td>
<td>30000</td>
<td>23000</td>
</tr>
</tbody>
</table>

Table 8.3: Overview of the different test batches and the level of the gas and water limit in each of them

Table 8.4 gives an overview of which solution strategies that are tested in each test batch. The X’s in the table represents a single test case. The does not illustrate that the four first strategies each are tested on 2CPU, 4CPU and 8CPU hardware, described in section 8.4. This was done because these are the solution strategies that actively utilize multiple CPUs. To avoid just testing the most constrained problems, these strategies were spread out over the test batches. The other solution strategies were only tested on the 8CPU hardware. The reason the B&P normal algorithm was run in more test batches is because the variance in the solution time was observed to be higher for different test batches during the initial testing. As the last two strategies contain randomness, they are each run ten times for each test batch and the average used for both solution time and optimality gap. Producing a solution time and gap therefore required many more runs for each of these strategies, and limited the testing of these two strategies to a smaller set of batches. Due to some instability in solution times for the Accelerated feedback strategy, a total of 5 runs were performed for each test case run on this strategy. The solution time was calculated as an average of the solution time from each test case run.
### 8.4 Hardware and Software Specifications

In addition to the software and 2CPU hardware used in the model testing phase and presented in section 5.3.3, the following hardware was used during testing.

**Hardware**

- 4 CPU: 2x Intel Core 2 Duo 5110 1.6 GHz, 4GB RAM, Linux
- 8 CPU: 2x Intel Quad Core E5472 3.0 GHz, 16GB RAM, Linux

### 8.5 Testing Limitations

The testing scheme presented in this chapter aims to create underlying data resembling a realistic and general petroleum production environment. As described in section 8.3, varying the gas and water constraints is believed to reveal the problem characteristics we are interested in when evaluating different solution methods. However, some limitations should be considered when evaluating the results. To increase the generality of the tests, it would be interesting to create test batches with different topologies in each well cluster different total number of clusters. More specifically, the number of wells connected to each manifold and also the total number of manifolds in a cluster could be changed. In addition, well types with other production characteristics could also be tested. These features would impact the total solution time of the problem, reflect a more general and realistic full field size problem and make it possible to investigate the robustness of the solution methods further. However, generating real data for different cluster topologies and well types proved to be very difficult due to limitations in the production data.

---

**Table 8.4: Overview of the different test batches and the solution strategies tested in each test batch**

<table>
<thead>
<tr>
<th>Solution strategy</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standard</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sequential</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Basic</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Accelerated feedback</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Extra columns</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>B&amp;P normal</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>B&amp;P random cluster</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>B&amp;P random phase</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---
received from StatoilHydro. Although test batches with different cluster topologies and well types could easily have been created using artificial production data, we chose to focus on using real production data since the previous work performed using this was limited.
Chapter 9

Results and Analysis

This chapter presents the most important results and analyses from the computational evaluation. This evaluation is based on the computational setup presented in chapter 8. Each subsection is dedicated to one of the solution methods described in table 8.2. The testing objectives and criteria are first presented followed by the test results and analyses. The most important findings are summarized in the final section together with a discussion regarding their practical application.

The analyses mainly focus on areas with a large impact on solution time and solution value, and where our contribution has been the greatest. In brief, this includes the areas of sequential and parallel Dantzig-Wolfe Decomposition (DWD) methods, column management and Branch&Price (B&P) with branching on continuous variables.

9.1 Validation

To validate the correctness of the implementation of the different solution methods, extensive testing was first performed using test batches with artificial production data. The solution times for these test batches were lower than when using real production data, and they were therefore more suitable for initial validation than the real production data test batches described in section 8.3.4. The solution values of the Standard solution method was compared with the results in Vestbø and Walberg (2008) to verify correctness. The results from all these validation tests proved the implementations to be correct and able to solve the problem we investigate.
9.2 Standard Solution Method

Test objectives

The goal of testing the Standard solution method is to observe if the real production data test batches can be solved using this method. The solution times of the Standard method can then be compared with the solution times for DWD and B&P.

Results

The standard solution method was unable to solve any of the test batches presented in section 8.3.4. The Xpress-Optimizer did not find any integer solutions after 12 hours of solving and was terminated due to lack of computer memory. To validate the correctness of the standard solution method and compare its solution time and value with our implementation of DWD, we created a new set of test batches that were less computationally demanding to solve. First, a lower resolution in the piecewise linearization of the pipeline pressure drop was tried. The computer still ran out of memory after 12 hours of solving, and we therefore created test batches where the number of production clusters were reduced to two. The Standard solution method solved these test batches in 3500-6000 seconds depending on how constrained the handling capacities were.

Analysis

When using artificial production data, we are unable to solve the full field size problem using the Standard method. As stated in section 5.6, the problem is computationally harder to solve when using real production data. The full field size problem with eight clusters and real production data will therefore be impossible to solve within reasonable time using the Standard solution method. This makes decomposition even more interesting as a solution method.

9.3 Dantzig-Wolfe: General

Test objectives

The objective of these tests is to uncover if the test batches can be solved using DWD and to observe the general behavior of the DWD method when applied to the production allocation problems we investigate. The change in gap between the LP restricted master problem (RMP) and DWD-IP heuristic solution between test batches is also interesting to investigate. This gives an indication of how close to the optimal solution the DWD-method generally can provide solutions for our problems. By inspecting the DWD results,
we also want to assess how the initial columns we create affect the convergence of the DWD-algorithm. The development of the gap between the LP solution and the LP-upper bound \( (Z_{RMP-LP}^U) \) provides information regarding how fast the RMP converges.

### General Results

The Sequential DWD method solved the two cluster test batches specially created for the Standard solution method in 118 - 220 seconds depending on how constrained the water and gas limits were. It terminated with a gap of 3.5 - 4% between the LP RMP and the DWD-IP heuristic solution. The Sequential DWD method then solved each of the full field size test batches presented in section 8.3.4. The solution times on the different types of hardware are shown in table 9.3, and will be further analyzed in section 9.5. The gap between the optimum LP RMP solution and the solution produced by the DWD-IP heuristic was calculated for each batch. This was then plotted against the sum of the limit of gas and water and the result can be observed in the graph in figure 9.1.

![Graph](image)

**Figure 9.1:** Optimality gap vs. sum of limits on water and gas in the test batches

### General Analysis

The computational burden of solving each of the subproblems in DWD proved to be significantly less than using the Standard solution method. It was also observed that the DWD method quickly obtained good dual values, and the number of iterations needed was therefore limited. Combined, these two features enable a much lower solution time when using DWD compared to the Standard solution method. This also enabled DWD to solve the real production test batches that the Standard solution method was unable to provide a single integer solution for.

As shown in figure 9.1, the optimality gap generally decreases when the problem becomes
less constrained. For an unconstrained problem, the LP RMP and DWD-IP heuristic solutions are identical as each cluster has a limited production and the columns corresponding to this production will be chosen in both the LP and IP solution. Generally, given that the total solution space is constant, the more possible integer combinations of columns are available, the smaller the gap will be between the LP RMP and DWD-IP heuristic solution. As long as the columns are spread out over the solution space, an increased number of combinations will enable the IP solution to come closer to the LP RMP solution. When the problem becomes more constrained, the number of feasible integer combinations of columns decrease. The possible integer combinations of columns will then be a more discrete representation of the space spanned out by the convex combination of columns in the master problem. The gap will then be larger between the LP RMP and DWD-IP heuristic solution.

Initial Solutions: Results

The method for creating a set of starting columns was discussed in section 7.3.1. Initial columns are mainly created to obtain a full basis in the RMP to be able to produce a feasible LP RMP solution. However, such columns are also created to avoid that the first dual values that are sent to the subproblems are zero to speed up the algorithm. The objective of creating a good set of initial columns, as opposed to creating any columns, is to obtain a more "correct" dual price for the capacity constraints early, to decrease the number of iterations needed to reach an optimal LP RMP solution.

When first testing on the real production data in the test batches it was observed that to obtain non zero dual values in the first iteration it is not sufficient to create columns simply by setting a bound on the subproblems that is slightly higher than the total capacity divided by the number of subproblems. The maximum production for some clusters was less than the bound that was sent to it. The sum of the production of all columns in the first master iteration was therefore less than the capacity limit and the dual values sent to the capacity constraints zero in the first RMP solution. To remedy this, the capacity constraint used in the subproblems to generate a starting set of columns was adjusted up so that a set of dual values larger than zero is sent to the subproblems after the RMP is solved for the first time. After making this adjustment the solution time decreased for the test batches which previously received dual values with a value of zero in the first iteration. An interesting observation is that the optimality gap increased for these test batches.

Initial Solutions: Analysis

Starting with a good set of initial columns that reduce the total number of iterations has an adverse effect on the optimality gap as it reduces the number of columns in the final RMP. However, one should still try to obtain a good set of starting columns as they are
better, more predictable and controlled ways of obtaining good integer solutions that do not slow down the convergence of the DWD-algorithm.

Termination Criteria: Results

To observe the behavior of the Sequential DWD method, we recorded the gap between the LP solution of the RMP and the upper bound \( Z_{UB_{RMP-LP}} \) in each iteration and the solution time development for each test batch. \( Z_{UB_{RMP-LP}} \) is calculated by adding the LP solution of the RMP with the sum of the reduced cost from all subproblems. This was described in 6.3.4. Figure 9.2 shows that this gap is reduced quickly and is close to zero in the third iteration.

![Figure 9.2: Development of LP optimality gap and solution time as the number of master problem iterations increase in the DWD-algorithm](image)

Termination Criteria: Analysis

As shown in figure 9.2, the gap in the third iteration of DWD is 0.23%. A gap close to zero in the third or fourth iteration is a consistent feature for all test batches. This indicates that the DWD-algorithm converges fast, and that the columns in the RMP in the final iterations remain fairly unchanged. Although the gap between the LP solution and \( Z_{UB_{RMP-LP}} \) is low in a given iteration, this does not necessary mean that the gap between the DWD-IP heuristic solution and \( Z_{UB_{RMP-LP}} \) will be low. This gap is the guaranteed maximum distance to the global optimum since the DWD algorithm has not yet terminated. As mentioned in section 7.3.5 this requires that the DWD-IP heuristic solution is calculated in every iteration. Given that the gap between this and \( Z_{UB_{RMP-LP}} \) is lower than the acceptable limit this will enable the algorithm to terminate in an earlier iteration.
CHAPTER 9. RESULTS AND ANALYSIS

9.4 Extra Columns

Test objectives

The objective of these tests are to observe the reduction in solution time and gap between LP RMP and DWD-IP heuristic solution when the Extra columns strategy is used. Comparing the solution time and gap of Extra columns to Basic is therefore interesting. To evaluate the frequency in which extra columns are added, it is also interesting to observe the size of the column pool.

The strategy of adding more columns to the RMP in each iteration of the subproblems was, as discussed in section 7.3.6, implemented both to try to reduce the solution time of the DWD algorithm and to improve the integer solution that is obtained by the DWD-IP heuristic after the DWD-algorithm has converged.

Results

Table 9.1 summarizes the results when comparing the performance of the Extra columns method to the Basic DWD-algorithm. Though the results differ from batch to batch, the trend is that adding extra columns give approximately the same solution time. The average of all the test batches is a 7% increase. A relatively unchanged solution time is consistent with what was observed when validating the Extra columns method on artificial production data. 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 the only solution method needed for the test batches in which Extra columns was tested.
9.4. **EXTRA COLUMNS**

<table>
<thead>
<tr>
<th>Test batch</th>
<th>Extra columns ( % of Basic)</th>
<th>Basic gap</th>
<th>Extra columns gap</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>123%</td>
<td>0.943 %</td>
<td>0.160 %</td>
</tr>
<tr>
<td>3</td>
<td>78%</td>
<td>0.744 %</td>
<td>0.184 %</td>
</tr>
<tr>
<td>4</td>
<td>167%</td>
<td>1.500 %</td>
<td>0.571 %</td>
</tr>
<tr>
<td>6</td>
<td>73%</td>
<td>0.400 %</td>
<td>0.084 %</td>
</tr>
<tr>
<td>7</td>
<td>91%</td>
<td>0.540 %</td>
<td>0.164 %</td>
</tr>
<tr>
<td><strong>Average</strong></td>
<td><strong>107%</strong></td>
<td><strong>0.825%</strong></td>
<td><strong>0.233%</strong></td>
</tr>
</tbody>
</table>

Table 9.1: 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.

From table 9.2 we see that though the column pool on average triples in size, the time it takes to solve the RMP with the DWD-IP heuristic is still very small compared to the total solution time. The average solution time for the DWD-IP heuristic was 0.09 seconds for Extra columns while the average total solution time for the algorithm was 235 seconds for the five test batches in table 9.2. The solution time for the LP RMP was observed to be less than 0.001 seconds in all iterations for both solution strategies.

<table>
<thead>
<tr>
<th>Test batch</th>
<th>Columns</th>
<th>Basic Sol. time DWD-IP</th>
<th>Extra columns Columns</th>
<th>Sol. time DWD-IP</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>38</td>
<td>0.04sec</td>
<td>145</td>
<td>0.11sec</td>
</tr>
<tr>
<td>3</td>
<td>45</td>
<td>0.02sec</td>
<td>124</td>
<td>0.03sec</td>
</tr>
<tr>
<td>4</td>
<td>39</td>
<td>0.03sec</td>
<td>123</td>
<td>0.06sec</td>
</tr>
<tr>
<td>6</td>
<td>55</td>
<td>0.03sec</td>
<td>160</td>
<td>0.18sec</td>
</tr>
<tr>
<td>7</td>
<td>46</td>
<td>0.03sec</td>
<td>160</td>
<td>0.07sec</td>
</tr>
<tr>
<td><strong>Average</strong></td>
<td><strong>45</strong></td>
<td><strong>0.03sec</strong></td>
<td><strong>142</strong></td>
<td><strong>0.09</strong></td>
</tr>
</tbody>
</table>

Table 9.2: Overview of the number of columns in the final RMP for Basic and Extra columns and the solution time of the final RMP solved as an IP for both methods.

**Analysis**

As mentioned in the previous section, the results indicate that adding all columns encountered when solving the subproblems gives about the same the total solution time but decreases the optimality gap when using the DWD-IP heuristic to produce an integer solution. As the column pool in the final RMP is larger for the method in which extra columns are added in each iteration, it is not surprising that a better solution is found with the DWD-IP heuristic.

Adding more columns was thought to increase the convergence speed as it was expected that a good representation of the subproblems would be built up faster by adding more
columns with a positive reduced cost in early iterations. A large proportion of the extra columns added in later iterations have a reduced cost less than zero, and therefore do not improve the convergence speed of the DWD-algorithm. By inspecting the results further, we observe that the LP RMP solution is consistently slightly higher in the first RMP solution when adding extra columns. However, this difference is gradually reduced from iteration to iteration, each time the RMP is solved. It appears that the extra columns with positive reduced cost added in early iterations generally are little used in later iterations and in the final solution. Most columns selected in the optimal LP solution were added in later iterations when the dual values reached a value close to the value they have in the optimum solution. As the MILP solution space in the subproblems is believed to be very dense, the columns that were suggested in early iterations, based on "wrong" dual values, therefore have a limited effect on helping to find the correct dual values. The difference in solution time between Extra columns and Basic, that can be observed in each batch in table 9.1, is a result of the subproblems solution time varying with small differences in the dual values sent to them from the RMP.

To focus the method of adding extra columns on obtaining better DWD-IP heuristic solutions, the intermediate columns should be stored each time a subproblem is solved and only added after the final iteration of the DWD-algorithm. This is because the DWD-IP heuristic is only used when the DWD method has converged and extra columns will only then be useful for obtaining a better DWD-IP heuristic solution.

9.5 Dantzig-Wolfe: Parallel

Test Objectives

The goal of these tests is to observe the performance of the different DWD strategies compared to the Sequential solution method. A comparison of the solution times for the different solution methods is therefore necessary.

The proportions of Sequential to Basic and Accelerated feedback solution times indicate the gain from introducing parallel solving of subproblems. It is not interesting to compare solution time for test cases on different types of hardware since these have unequal processor speeds. As explained in section 6.3.5, Speedup and Efficiency are also important measures regarding performance, and are therefore also calculated of each test batch. Speedup is defined as the inverse of the solution time proportion and Efficiency is the Speedup per CPU used.

The individual subproblem solution times and the number of times they are solved also provides us with information when evaluating the solution methods. Unequal or equal subproblem solution times will partly explain Speedup gained from parallelization since, depending on the parallelization strategy, fast subproblems will have to wait for slow subproblems to produce candidate columns.
9.5. **DANTZIG-WOLFE: PARALLEL**

Results

Table 9.3 shows the solution times of Sequential on 2, 4 and 8 CPUs. This solution time is used as a benchmark and the solution times of all the parallel strategies including Perfect load balancing are shown as a percentage of this.

<table>
<thead>
<tr>
<th>Strategy</th>
<th>Number of CPUs</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Test batch</td>
<td>2 CPUs</td>
<td>4 CPUs</td>
<td>8 CPUs</td>
</tr>
<tr>
<td>Sequential (Seconds)</td>
<td>1</td>
<td>570,348</td>
<td>1013,96</td>
<td>535,03</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>647,41</td>
<td>1028,19</td>
<td>542,217</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>508,9</td>
<td>812,807</td>
<td>427,815</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>1102,84</td>
<td>1780,59</td>
<td>938,68</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>754,242</td>
<td>1212,85</td>
<td>638,853</td>
</tr>
<tr>
<td></td>
<td>Average</td>
<td><strong>716,748</strong></td>
<td><strong>1169,6794</strong></td>
<td><strong>616,519</strong></td>
</tr>
<tr>
<td>Basic (% of sequential)</td>
<td>1</td>
<td>64.49 %</td>
<td>52.56 %</td>
<td>47.52 %</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>59.52 %</td>
<td>41.85 %</td>
<td>36.58 %</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>61.08 %</td>
<td>44.64 %</td>
<td>39.38 %</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>56.56 %</td>
<td>37.50 %</td>
<td>31.23 %</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>59.61 %</td>
<td>41.49 %</td>
<td>36.37 %</td>
</tr>
<tr>
<td></td>
<td>Average</td>
<td><strong>60.25 %</strong></td>
<td><strong>43.61 %</strong></td>
<td><strong>38.22 %</strong></td>
</tr>
<tr>
<td>Accelerated feedback (%</td>
<td>1</td>
<td>66.04 %</td>
<td>29.31 %</td>
<td>29.72 %</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>64.84 %</td>
<td>42.60 %</td>
<td>35.54 %</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>72.65 %</td>
<td>45.26 %</td>
<td>39.04 %</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>53.69 %</td>
<td>33.91 %</td>
<td>22.40 %</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>62.89 %</td>
<td>36.37 %</td>
<td>24.17 %</td>
</tr>
<tr>
<td></td>
<td>Average</td>
<td><strong>64.02 %</strong></td>
<td><strong>37.49 %</strong></td>
<td><strong>30.17 %</strong></td>
</tr>
<tr>
<td>Perfect load balancing (%</td>
<td>1</td>
<td>50.00 %</td>
<td>25.00 %</td>
<td>12.50 %</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>6</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>7</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Average</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 9.3: 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, with solution time proportions of 60.25, 43.61 and 38.22% for 2, 4 and 8 CPUs respectively. Compared with Basic, Accelerated feedback has a higher solution time for 2 CPUs but a reduced solution time for 4 and 8 CPUs. The final row in the table shows the Perfect load balancing solution time proportion. 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 9.3.
Testing showed that the Accelerated feedback strategy was unstable with regards to both solution time and the DWD-IP heuristic solution value. This only occurred in tests where the number of CPUs were lower than the number of subproblems. The instability was therefore highest in the 2 CPU case and moderate in the 4 CPU case, while there were no instability in the 8 CPU solutions.

Table 9.4 shows the average Speedup and Efficiency as a function of the number of CPUs. Speedup increases for both strategies when moving from the 2 CPU to the 4 CPU processor. The gain in Speedup when moving from the 4 CPU processor to the 8 CPU processor is lower for both strategies. Since Speedup increase becomes less with more CPUs, Efficiency also declines.

<table>
<thead>
<tr>
<th>Number of CPUs</th>
<th>2 CPUs</th>
<th>4 CPUs</th>
<th>8 CPUs</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Average speedup</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sequential</td>
<td>1,00</td>
<td>1,00</td>
<td>1,00</td>
</tr>
<tr>
<td>Basic</td>
<td>1,66</td>
<td>2,29</td>
<td>2,62</td>
</tr>
<tr>
<td>Accelerated feedback</td>
<td>1,56</td>
<td>2,67</td>
<td>3,31</td>
</tr>
<tr>
<td><strong>Average efficiency</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sequential</td>
<td>1,00</td>
<td>1,00</td>
<td>1,00</td>
</tr>
<tr>
<td>Basic</td>
<td>0,83</td>
<td>0,57</td>
<td>0,33</td>
</tr>
<tr>
<td>Accelerated feedback</td>
<td>0,78</td>
<td>0,67</td>
<td>0,41</td>
</tr>
</tbody>
</table>

Table 9.4: Average Speedup and Efficiency for the two parallel strategies for each type of hardware
Analysis

Regardless of the number of CPUs, the solution times of the Basic parallel strategy are much lower than for the Sequential strategy. This is true for all test batches. Basic distributes the subproblems over the available CPUs and solves them in parallel, thus decreasing the time spent in each iteration of the algorithm. As the number of CPUs increase, the number of subproblems solved concurrently increase which again leads to a reduction in the solution time. However, Basic is not able to obtain the same speedup as Perfect load balancing. As described in section 6.3.5, the serial/parallel nature of decomposition problems makes this limit unreachable. This is a result of CPU idle time, which again arises as a result of two main factors: The forced serial solving of the RMP and the relative differences in subproblem solution times. 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 times impact time spent in each iteration since the "bottleneck" subproblem have to finish before the RMP can be solved. These subproblem solution times are very different for our problem, thereby contributing substantially to CPU idle time.

It is not trivial to explain why Basic does not obtain the same solution time proportion as Perfect load balancing. The solution process when using mmjobs in Xpress is not transparent and this makes a deep analysis of the results difficult. To try to explain the results, we quantified the impact of relative differences in subproblem solution times. The analysis was performed for the 8 CPU test cases. For all the test batches, we recorded the solution times of each subproblem in every iteration. To show the total impact of different subproblem solution times, the time the CPUs were idle in each iteration was calculated. The CPU idle time for a subproblem in a particular iteration is the difference in subproblem solution time and the solution time for the slowest subproblem. The total CPU idle time is the sum of all these idle times in all iterations. The proportion of CPU idle time compared to the total solution time was used as a measure for how much the subproblem solution times differed.

As shown in table 9.3, test batch 3 has 36.58% of Sequential solution time while test batch 1 only had 47.52%. Our calculations showed that for test batch 1 the CPUs were, on average, idle 69% of the time. For test batch 3, the CPU idle time percentage was 48%. This means that the subproblem solution times differed more for test batch 1 than test batch 3, thus 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 14.82% of sequential and 19.02% for test batch 3. This is much closer to the Perfect load balancing limit and shows the importance of differences in subproblem solution times. The remaining distance to the Perfect load balancing limit of 12.5% is likely caused by overhead from file writing, memory access and loading/unloading of subproblems.
Difference in subproblem solution times can also explain why efficiency declines as the number of CPUs increase. This is shown in table 9.4. For an 8 CPU computer, all subproblems will have to wait for the slowest of the eight subproblem to finish in each iteration. A 2 CPU computer on the other hand utilizes both 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 processor is lower, the proportion of idle time compared to total solution time is higher and hence the efficiency will be lower.

The results in table 9.3 show increased solution time with 2 CPUs but reduced solution time for both 4 and 8 CPUs when using Accelerated feedback. Section 6.3.5 explained that the background for implementing Accelerated feedback was to reduce the CPU idle time. Accelerated feedback tries to achieve this 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 reduced solution time when using Accelerated feedback. This is due to reduced CPU idle time and faster convergence of the RMP. Test batch 3 shows almost no change in solution time when Accelerated feedback is used. This is partly due to the fact that it has a lower proportion of CPU idle time. A detailed analysis of subproblem solution times and iterations is much harder with this strategy since the subproblems are solved in no particular order and the RMP is solved after each subproblem terminates. Considering the high percentage of CPU idle time in Basic, we would suspect that Accelerated feedback has the potential to reduce the total solution time substantially. However, we observe in the results that the number of iterations in Accelerated feedback is greater than in Basic. Since the RMP is resolved every time a subproblem terminates, the reduced cost of the subproblems will be based on different dual values. This means that the first time all subproblems return nonpositive reduced cost, those that have not been solved with the most recent dual values have to be resolved. This is to guarantee optimality in the DWD algorithm. The results for our test batches show that there may be a high number of iterations in this final phase, hence contributing to increased solution time.

The solution time varies more using Accelerated feedback compared to Basic. Accelerated feedback provides a higher solution time in the 2 CPU tests, in addition to the solution time and solution value generally being more unstable. Without specific knowledge about the inner procedures of \textit{mmjobs}, it is difficult to determine the exact reason for this instability when using Accelerated feedback. It does not occur in the 8 CPU cases, but only when there are fewer CPUs than subproblems. With eight subproblems solved simultaneously on 2 or 4 CPUs, several threads are executed per CPU. It is reasonable to assume that there are small variations in the message passing order between the master and slave threads for each test run. This leads to subproblems sending candidate columns at different times to the RMP, thus creating other basic solutions and different dual values during the solution process.
9.6 Branch & Price

9.6.1 General Branch & Price

Test objectives

The goal of these tests is to evaluate the performance of B&P on the production allocation problem and to compare its performance to using DWD on the same problem. This includes an evaluation of different strategies within B&P. The root node gap between LP and DWD-IP heuristic solution and solution time compared to the total solution time for the B&P normal strategy is an important evaluation criteria. The root node solution is identical to the solution produced by the Basic DWD strategy. Comparing the gap and solution time will determine whether it is worth to solve the problem using B&P normal compared to using DWD with the DWD-IP heuristic. The solution time of the B&P normal strategy compared to the solution time of the Standard strategy will also be an important criteria for evaluating the performance and potential of the B&P implementation as these two strategies are the only ones that potentially can guarantee a global or close to global optima.

The number of nodes needed to be solved to reach an optimal solution together with the development in optimality gap explains how the B&P proceeds. As for the other solution strategies, the robustness is evaluated by the consistency of the results.

Results

The results of running B&P normal on TB1-TB10 are shown in table 9.5. As we see, the B&P normal algorithm was able to solve all the problems in less than an hour of running time with the termination gap set to 0.1%. Though the increase in solution time differs greatly, the algorithm was robust in being able to provide a satisfactory accurate solution for a full field size problem where both gas and water are active constraints. The algorithm progressed beyond the root node in all batches except for TB 10, where the optimality gap was less than 0.1% after the DWD-IP heuristic terminated in the root node. This is why the optimality gap and solution time is identical for the root node and the whole B&P search for TB10. The number in the column in table 9.5 that has "Nodes" as the headline is the number of nodes that were solved before the optimality gap was less than 0.1%.
140

CHAPTER 9. RESULTS AND ANALYSIS

<table>
<thead>
<tr>
<th>Test batch</th>
<th>Root node</th>
<th>B&amp;P normal</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Sol. time</td>
<td>Gap</td>
</tr>
<tr>
<td>1</td>
<td>148.57sec</td>
<td>0.943 %</td>
</tr>
<tr>
<td>2</td>
<td>170.41sec</td>
<td>0.718 %</td>
</tr>
<tr>
<td>3</td>
<td>157.18sec</td>
<td>0.744 %</td>
</tr>
<tr>
<td>4</td>
<td>148.30sec</td>
<td>1.500 %</td>
</tr>
<tr>
<td>5</td>
<td>167.70sec</td>
<td>0.188 %</td>
</tr>
<tr>
<td>6</td>
<td>224.05sec</td>
<td>0.400 %</td>
</tr>
<tr>
<td>7</td>
<td>204.50sec</td>
<td>0.540 %</td>
</tr>
<tr>
<td>8</td>
<td>254.27sec</td>
<td>0.159 %</td>
</tr>
<tr>
<td>9</td>
<td>183.03sec</td>
<td>0.144 %</td>
</tr>
<tr>
<td>10</td>
<td>206.86sec</td>
<td>0.100 %</td>
</tr>
<tr>
<td><strong>Average</strong></td>
<td><strong>186.49sec</strong></td>
<td><strong>0.544%</strong></td>
</tr>
</tbody>
</table>

Table 9.5: Test batch results for B&P normal and the root node in B&P normal. The root node is equivalent to solving with Basic DWD.

An interesting observation is that the solution time of B&P normal is positively correlated with the optimality gap in the root node. For the first four test batches the gap is on average about 1% and the solution time increases on average by a factor of 9.9. For the last 6 test batches the gap is on average 0.26% and the solution time increases on average by a factor 1.3.

Figure 9.4 shows a plot of how the upper and lower bound change from iteration to iteration, that is each time a new node is solved, for TB1 and TB4. The x-axis represents the number of nodes that have been solved and the y-axis represents the oil production. TB1 and TB4 required the most nodes to be solved in B&P normal and therefore best illustrate what was also seen to occur in the other test batches: The upper bound is fairly unchanged throughout the B&P search and the closing of the optimality gap occurs because the lower bound increases.

![Figure 9.4: Development in upper and lower bound for B&P as the number of nodes solved increases](image-url)
To explain one observation, we term the cluster that is branched on in a given node, \( c \), and the phase branched on, \( p \). After studying the columns that are added in each node in the tree it was observed that the production of phase \( p \) in columns suggested from cluster \( c \) in most cases has a production that is on the bound on phase \( p \) - a bound that was obtained from the production of phase \( p \) in cluster \( c \) in the optimal LP RMP solution. If no new constraints are added to a subproblem over several child nodes it was also observed that columns with the same production of phase \( p \) was added in several iterations. However, for these columns the production of oil and the phase that was not constrained in cluster \( c \) was different. Summarized, a column is sent to the RMP that has a production of phase \( p \) that lies on the bound sent to the subproblem representing cluster \( c \).

Analysis

This analysis mainly focuses on the aspects of the implemented B&P algorithm that separate it from most B&P algorithms in literature, which is that branching is performed on a continuous variable. The arguments for the different choices taken to implement such an algorithm were discussed in section 6.4. We here explain how the results provide support for the arguments that were put forward in that section.

It is expected that a larger optimality gap leads to a larger total solution time for the B&P algorithm. Better solutions are gradually obtained in the B&B tree and it will therefore take more time to close a higher root node optimality gap.

That a column suggested to the RMP in most cases has a production equal to the bound on gas or water in the subproblem it was sent from confirms our belief about the effects of branching on the gas and water production in a cluster. The reason columns suggested from a subproblem do not always have production equal to the bound is that the dual values in the subproblem objective function change. This is also the reason the production of oil and the phase that is not an active constraint in a subproblem is different for columns suggested from a subproblem that does not receive new constraints over a number of nodes.

The fact that the subproblems are able to provide a column that is very close or on the bound that was sent to the subproblem indicates that the solution space is very dense and that there generally will be a very small optimality gap between the LP and DWD-IP heuristic solutions.

Section 6.4.3 discussed that the likely effect of branching on a continuous variable is a closing of the optimality gap first and foremost as result of an increase in the lower bound. It is not because of a reduction in the upper bound as a result of a reduction of the solution space when we branch. This assumption was used to argument for using a pure best first tree search strategy, to explore potential good nodes, together with a heuristic to produce an IP RMP solution. The assumption about the change in the bounds is clearly verified by figure 9.4.
Branching on a continuous variable, one might expect that the upper bound would be unchanged. However, as can be observed in the figure, the upper bound decreases slightly from the first to the last iteration for both test batches. The reduction of the upper bound is about 0.06% for TB1. This reduction can be explained by what happens when we partition the solution space between two child nodes. Adding constraints to the subproblems does not directly influence the upper bound. This is because adding an upper bound in the subproblem that was branched on in one node and a lower bound in the same subproblem in the other node will not reduce the solution space for this subproblem if one views the combined space that the subproblem in the two child nodes together span out.

Removing columns from the RMP in each node that violate the new constraints implicitly added to the RMP does not reduce the solution space for the IP solution of the RMP. This is because only one column can represent each cluster and any column representing a subproblem solution can be chosen when viewing the combined subproblem in two child nodes. Removing columns from the RMP in two new child nodes in this way is therefore a dichotomy of the IP RMP solution space. However, it is not a dichotomy of the LP RMP solution space. The LP solution in a node will, as discussed in section 6.3.1, in most cases have a production that is represented by an interpolation between two columns for at least one of the clusters. As the upper column of the interpolated columns is removed from the RMP in one node and the lower column in the other node, this interpolated solution will not be possible in the RMP in any of new branches when two new child nodes are created. The LP objective function value of the new RMP will depend on how close to the interpolated solution in the RMP new columns from the subproblem that was branched on are. They are highly unlikely to have a production identical to the interpolated solution and the LP objective function value will therefore be lower in both child nodes as the optimal LP solution is no longer feasible. This is why the upper bound decreases slightly.

That the B&P-algorithm closes the optimality gap mainly by increasing the lower bound makes the algorithm and extra time spent on finding a solution more attractive from a practical point of view. It is more interesting for a user to find better solutions rather than just proving that the solution already obtained, using only DWD and the DWD-IP heuristic, is optimal or close to optimal by decreasing the upper bound.

The positive effect of this branching rule can also be seen if we compare the final IP solution with the bounds that are sent to the subproblems during the B&P search. We then observe that several columns in the optimal IP solution, which did not exist in the root node, have a production of gas or water that is equal to the bound that was sent to the subproblems.

One of the disadvantages of the implemented branching scheme is that new constraints are explicitly added to the subproblems and therefore increase the size of the subproblem basis. The constraints put on the production of water and gas are in reality constraints on a sum of continuous variables, as the production in a cluster is the sum of flow over the
two pipelines up to the separator. By introducing a new variable, \( q_{P}^{\text{PIPE}} \), representing the total production of phase \( p \) in cluster \( c \), the constraint sent to a subproblem could be set directly on the new variable as an upper and lower bound. The number of variables in each subproblem would then increase by one. However, this would avoid increasing the size of the solution basis in the subproblems and would therefore be a better way of adding constraints to the subproblems.

### 9.6.2 Cluster Branching Rule

**Test objectives**

The goal of these tests is to determine the potential of the rule used to decide which cluster to branch on in B&P normal. Comparing the solution time of B&P normal with B&P random cluster will give an indication of this and the importance of choosing the right cluster to branch on.

**Results**

The column diagram in figure 9.5 shows the solution time for B&P normal as a proportion of the solution time of B&P random cluster for TB6, TB4 and TB1. As mentioned in section 8.3.4, the solution time in each test batch for B&P random cluster was calculated using the average over ten runs. If the variance was high or the results unclear this would have been a too few runs to calculate an average over and use as a basis for a conclusion. However, the solution time for B&P random cluster was higher for virtually all cases and the average also clearly higher as shown in the plot in figure 9.5.

![Figure 9.5: Solution time of B&P normal as a proportion of solution time for B&P random cluster for TB6, TB4 and TB1](image-url)
Analysis

The diagram in figure 9.5 clearly illustrates that the implemented branching rule is better than choosing which cluster to branch on randomly. It also illustrates that which cluster to branch on is important for the solution time of the B&P algorithm. The results thereby support our hypothesis put forward in section 6.4.3, that it is better to branch on the cluster where the difference in oil production between the interpolated solution and the lower of the two interpolated columns is the largest.

9.6.3 Phase Branching Rule

Test objectives

The goal of these tests is to determine the potential of the rule used to decide which phase to branch on in B&P normal by comparing the solution time for B&P normal with B&P random phase.

Results

B&P normal and B&P random phase where initially planned to be compared in TB6, TB4 and TB1. The tests run on TB6 and TB4 showed only a small difference in solution time between the two solution strategies. A further inspection of the results in these two test batches showed that only three nodes (including the root node) were solved using B&P normal and in most runs of B&P random phase. When two nodes in addition to the root node are solved, the choice of which phase to branch on is only taken twice. These two test batches were therefore not considered as a good basis for an evaluation of a rule for deciding which phase to branch on. We therefore ran B&P random phase in two new test batches, TB3 and TB2, to obtain an average solution time for B&P random phase. The results of these tests are shown in figure 9.6.
9.6. BRANCH & PRICE

Figure 9.6: Solution time of B&P normal as a proportion of solution time for B&P random phase for TB3, TB1 and TB2

Analysis

The results on comparing the solution time for B&P normal with B&P random phase do not clearly indicate that the rule implemented in B&P normal is better than randomly choosing which phase to branch on. This could either indicate that the rule is not especially good or that which phase we branch on is not very important to the convergence speed of the B&P algorithm. We believe the latter is closest to the truth. To explain why, we use a counter argument. We start by assuming that there is a clear best choice of which phase to branch on in a certain iteration. If this phase is branched on a new column will be produced that in most cases have a production that is equal to the bound that was sent to the subproblem that was branched on. This column will also have production of the phase that was not branched on that is close to interpolated solution from which the bound sent to the subproblem was derived. However, if there was a clear best choice of which phase to branch on the effect of branching on one phase should be clearly different from branching on the other phase. This will generally not be the case, as the production of both phases will be close to the interpolated solution, and there will therefore not be a clear best phase to branch on. This is not an argument against the existence of a best phase to branch on, rather an explanation of why which phase to branch on has limited impact on the convergence speed of the B&P algorithm.

Though there on average is little difference between choosing which phase to branch on randomly, and using the rule implemented in B&P normal, the former is a less robust method and it is our recommendation that the rule described in section 6.4.3 should be used.
9.7 Summary of Analyses

Chapter 1 describes the requirements that should be fulfilled by an optimization model for a production allocation problem. For the solution methods investigated in this chapter to have a practical value for a potential future user, they should be analyzed on the basis of these requirements.

The first requirement is that a solution to the optimization model should be provided within hours. The high solution time of the Standard method for the low resolution test batches clearly does not satisfy this requirement, and this method is therefore not considered a possible solution approach. DWD, on the other hand, provides 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, the Perfect load balancing limit is not reached by the Basic strategy as the unequal solution times of the subproblems lead to a high percentage of CPU idle time for all test batches. The Accelerated feedback strategy is able to reduce the amount of 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 and more available, and petroleum production planners will often have access to such hardware. Neither of the solution methods are able to provide a solution when tested on problems with a satisfactory accurate resolution in the piecewise linearization. However, the DWD method is able to provide a solution in 200 seconds for low resolution problems where the Standard method is unable to provide a single integer solution within 12 hours. DWD is therefore an important step towards being able to solve problems with a realistic accuracy.

The next requirement as described in section 7.4, states that a solution method should be able to provide an integer solution guaranteed to be within 0.1% of the global optimum of our model. Our tests show that the DWD implementation terminates with a higher gap than 0.1% for most test batches. Adding extra columns from each subproblem in every iteration with the Extra columns reduces the optimality gap. However, the gap is still not satisfactory, and B&P is needed to reduce the gap further. B&P with branching on the continuous production variables produces good candidate columns to the RMP but has little effect on reducing the upper bound. The reduction in the optimality gap mainly occurs as a result of an increase in the lower bound. This supports our use of a best first tree search strategy in B&P normal. Choosing the right cluster to branch on has a significant effect on the convergence of the algorithm. Choosing the right phase to branch on is on the other hand of less importance. For all the tested production allocation problems B&P-normal provides solutions within 0.1% of optimality.

The solution methods are robust as they are all able to provide a solution within reasonable time. However, the solution times of the methods, especially B & P, vary greatly within the test batches. Varying the gas and water handling constraints has a significant
impact on the solution time of the subproblems. As explained in section 8.3, changing these bounds affect the flow in each pipeline and consequently the variables connected to the modeling of the partly erratic pressure drop function. To be able to properly assess the robustness of the proposed solution methods, a statistical analysis regarding solution time should be performed with more general tests batches containing different cluster topologies.

Combining the solution methods tested during the course of this thesis leads to an overall recommendation regarding which methods should be applied for solving our model of the production allocation problem. The results indicate that using a Branch&Price algorithm that combines Branch & Bound with DWD Accelerated feedback and Extra Columns in each node will provide the lowest solution while honoring the level of the maximum acceptable gap.

Though the test batches tested in this chapter are believed to be a good representation of the properties of a realistic general petroleum production field, the results should be further verified to confirm the observed trends. Specifically, the results should be verified on data that has a higher resolution in the piecewise linearization. As explained in 8.2, a low resolution was chosen to be able to run multiple tests of different solution methods.
Chapter 10

Conclusion

The purpose of this thesis is twofold. First, to evaluate two different mathematical models for a petroleum production allocation problem using real production data from StatoilHydro. Second, to develop decomposition methods for such problems suitable for parallelization while handling integer requirements on variables and multiple global constraints.

We have successfully developed a new StatoilHydro validated method for generating real production data. This creates a realistic production environment which is the basis for testing the two mathematical models. The first model is based on modeling flow with oil, gas and water as variables. The second model has a different flow representation which utilizes certain fluid characteristics to try to reduce the number of variables and constraints and thereby reduce solution time. Although the second model produces accurate solutions with a lower solution time when only the total production level of gas is limited, it is slower and less accurate when both gas and water production levels are limited. It is also considered to be less intuitive for a potential user and more difficult to implement. Based on this, the first model formulation is chosen as a basis for evaluating different solution methods.

The choice of which solution methods to use will ultimately be based on the requirements stated by the petroleum production planners. Through discussions with personnel at StatoilHydro we developed some general requirements for optimizing production allocation problems. The production planners should have the possibility of evaluating a solution and then adjusting the handling capacity limits and reoptimize iteratively. The time to obtain a solution should therefore not exceed a few hours. This confirms the need for focusing on reducing solution time. With the generated real production data, a sufficiently accurate model of the production system entails a problem with a high resolution in the piecewise linearization we use to model the nonlinearities in the system. This creates a very large number of complicating variables, making the problem impossible to solve with the currently developed solution methods. A lower resolution in the piecewise linearization is therefore used.
Even with this low resolution, the standard Branch&Bound algorithm based on the Simplex method is not able to provide a single integer solution during 12 hours of solving. Therefore, we do not consider this as a possible solution approach for production allocation problems of a scale equivalent to the investigated problems. The block angular structure of the problem and the few common constraints make decomposition methods attractive for reducing the solution time. Dantzig-Wolfe is considered the best decomposition approach as it is particularly suited for parallelization and for problems with this structure. Our sequential Dantzig-Wolfe implementation solves the realistic set of problems where both gas and water capacity constraints are active with an average solution time of 616 seconds. As the problem contains integer variables, a simple IP heuristic is used to quickly obtain good integer solutions using the columns in the optimal DWD master problem as a starting point. A basic parallel strategy, which utilizes hardware with multiple CPUs, reduces the solution time to an average of 38% of the sequential algorithm. A more advanced parallel strategy, which tries to minimize CPU idle time, further reduces this solution time to 30%. Since hardware with multiple CPUs are relatively inexpensive and easily available to petroleum production planners, parallelization is an attractive approach.

StatoilHydro have indicated that our solution method should be able to guarantee a solution within 0.1% of the global optimum for our model. For most of the tested problems, Dantzig-Wolfe with the DWD-IP heuristic does not reach this gap. Adding extra columns to the master problem in each iteration does not affect the solution time but reduces the optimality gap. This feature should therefore be included in the DWD implementation. However, it is still necessary to use an additional solution method to be able to guarantee a gap at the acceptable level. Using a Branch & Price algorithm, with branching on the production level in each production cluster, a solution within the acceptable gap of 0.1% is produced. Branching on these continuous production variables has a little effect on reducing the upper bound but a positive effect on providing potential good columns to the master problem. As this branching scheme does not reduce fractionality in the solutions significantly, an IP heuristic is used to produce an integer solution in each node.

StatoilHydro have stated that they will aim to implement a solution produced by our optimization model if the oil production is higher than the present production at Troll West, and if they believe the underlying assumptions in the model are valid. The low resolution in the piecewise linearization used in our problem makes the solution value inaccurate. In addition, the fact that we ignore absolute pressure for fields with more than one manifold, will likely decrease the accuracy of a solution. These two factors separate the tested optimization model from the currently accepted models used by StatoilHydro, and makes a direct implementation of a solution unsuitable. We have nevertheless demonstrated that parallel Dantzig-Wolfe Decomposition and Branch & Price are powerful solution methods for optimizing a full size field production allocation problem. This moves us closer to being able to create integrated software that could increase oil production from the current practice, which is believed to be sub-optimal.
Chapter 11

Further Work

This chapter presents areas that might be of interest for continued research in relation to the work performed in this thesis. Currently, research on several connected topics is performed through projects connected to the IO-centre at NTNU. The Ph.D. student Vidar Gunnerud at Department of Engineering Cybernetics will, in his thesis "Oil Production Optimization", incorporate and continue some of the work we have performed. Several theses for master students at NTNU are also planned within the same area.

An important aspect of the further work should be to try to close the distance between the solution methods proposed in this thesis and a method that can be used by petroleum production planners. Hence, the underlying assumptions for our proposed optimization model need to be addressed. The inaccuracy of not including absolute pressure in the piecewise linearization of the pipeline pressure drop model should be quantified, and this is currently being evaluated in Gunnerud, Nygreen, McKinnon and Foss (2009). Further, a more robust solution method, taking uncertainty into account, should be investigated since it over time could provide a higher total oil production and a more stable production.

To obtain a good basis for further testing, several different realistic benchmark cases could be created. This would enable researchers to compare and test solution methods on a wider set of problems, thus increasing the confidence in the performance of the methods. To be able to use the Troll West field data for a wider set of problems, more data should be retrieved from StatoilHydro. In addition, a more streamlined way of extracting production data using the General Allocation Package software should be developed. Publicly available data from other fields, such as the Norne field case, can also be used for this purpose.

There are several ways of reducing solution time that do not require further development of solution methods. As the main driver for the problem size is the piecewise linearization of the pipeline pressure drop, a better distribution of interpolation points could reduce the solution time. StatoilHydro have indicated that the pipeline flow in a realistic solution will always lie above a certain limit. We observed that the areas of the pressure drop
data below this limit was highly nonlinear. Removing these areas from the data set could make it possible to use fewer interpolation points to obtain the same accuracy. We believe that this approach has a large potential in reducing the total solution time.

Certain nonlinearities arose from the reformulation of flow variables introduced in the model of Hagem and Torgnes (2008). After further analysis, the method used to handle these nonlinearities was proved to perform poorly. Alternative ways of handling these nonlinearities have been suggested in this report and should be investigated further.

This thesis presented promising results with regards to using parallelization to reduce the solution time of Dantzig-Wolfe. To further reduce solution time, it might be interesting to investigate different and more sophisticated strategies for the order in which the subproblems and master problem are solved. Parallelization could also be utilized to obtain a speedup in several other areas. Additional subproblems could be solved in each iteration with a perturbation of the dual values to obtain potential good columns. Independent nodes in the Branch & Bound tree in Branch & Price could also be solved in parallel.

Most solutions will have four columns with fractional weighting variables. A different branching scheme, branching on two clusters simultaneously and creating four child nodes, could be interesting to investigate. It could also be interesting to assess which of the "Up" or "Down" branches in Branch & Price that have the greatest potential for reducing solution time. To assess this, larger problems that require more branching need to be solved. Finally, further work should be carried out on reducing the variations in the solution time of B&P to increase the robustness of the method.
Bibliography

Alstad, V. (2009), ‘Personal correspondance with Vidar Alstad, Senior Researcher, Research Centre Porsgrunn, StatoilHydro’.


Bell Laboratories (2009), ‘AMPL Homepage’. Last accessed 08.06.09.

URL: www.ampl.com


Dueñas Díez, Marta. (2008a), ‘Personal correspondance with Marta Dueñas Díez, StatoilHydro’.

Dueñas Díez, Marta., ed. (2008b), *How to Optimize a Complex Subsea Oil Production Network in the North Sea?*, SIAM 08 Conference on optimization. Solving a Large-Scale MINLP with a Combination of In-House and Commercial Tools.


Fevang, O., Singh, K. and Whitson, C. (2000), ‘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 1997, SPE 63087*.


Foss, B., Gunnerud, V. and Dueñas Díez, M. (2009), Lagrangian decomposition of oil production optimization - applied to the Troll west oil rim. Accepted for SPE Journal.


Gunnerud, V. (2008), ‘Personal correspondance with Vidar Gunnerud, Ph.D. Student, Engineering Cybernetics, Norwegian University of Science and Technology’.


Hauge, J. and Horn, T. (2005), ‘The challenge of operating and maintaining 115 subsea wells on the troll field’, *Paper prepared for presentation at the 2005 Offshore Technology Conference held in Houston, TX, USA, 2–5 May 2005, OTC 17111*.


Konieczek, J. (1990), ‘The Concept of Critical Rate in Gas Coning and Its Use in Production Forecasting’, This paper was prepared for presentation at the 65th Annual Technical Conference and Exhibition of the Society of Petroleum Engineers held in New Orleans, LA, September 23–26, 1990, SPE 20722.


URL: http://www.math.ntnu.no/ bryn/TMA4180/


Petroleum Experts Ltd. (n.d.a), ‘IPM Brochure’. Downloaded 02.02.09.  
URL: http://www.petex.com/includes/download.php?id=1

URL: www.petex.com


StatoilHydro (n.d.), ‘Statoilhydro homepage’. Last accessed 08.06.09.  
URL: www.statoilhydro.com


Weatherford International Ltd. (n.d.), ‘Weatherford international homepage’. Last accessed 08.06.09. 
**URL:** http://www.weatherford.com/


Appendix A

Model of Hagem & Torgnes

A.1 Declarations

Readers familiar with Gunnerud and Langvik (2007) and Vestbø and Walberg (2008) will notice that this model uses the term tubes instead of pipelines. Because liquid is used as a flow variable, the index and set for tubes (pipelines) is changed to $t$ and $T(i)$ respectively to avoid confusion.

Sets

\begin{align*}
I & \quad \text{Set of clusters} \\
\mathcal{M}(i) & \quad \text{Set of manifolds in a cluster} \\
\mathcal{J}(i,m) & \quad \text{Set of wells connected to a given manifold in a cluster} \\
\mathcal{H}(i,m) & \quad \text{Subset of } \mathcal{J}(i,m). \text{ Wells that cannot be shut down} \\
T(i) & \quad \text{Set of tubes in a cluster} \\
\mathcal{P} & \quad \text{Set of phases (g for gas, o for oil and w for water)} \\
S & \quad \text{Set of substances (g for gas and l for liquid)} \\
\mathcal{K}(i,m,j) & \quad \text{Set of interpolation coordinates for the piecewise linearization of the well performance curve (WPC) for wells connected to a given manifold in a cluster} \\
\mathcal{N}(i,m,s) & \quad \text{Set of interpolation coordinates for the piecewise linearization of the pressure drop in the tubes connected to a manifold in a cluster, dependent on which substance} \\
\mathcal{F}(i,m) & \quad \text{Set of interpolation coordinates for the piecewise linearization of the pressure drop in the tubes connected to a manifold in a cluster dependent on water cut} \\
\mathcal{R} & \quad \text{Set of interpolation coordinates for the piecewise linearization of the water cut linking constraint}
\end{align*}
Indices

- $i$: Cluster
- $m$: Manifold
- $j$: Well
- $t$: Tube
- $p$: Phase
- $s$: Substance
- $k$: Interpolation coordinate for the piecewise linearization of the WPC
- $n_s$: Interpolation coordinate for the piecewise linearization of the pressure drop in the tubes, dependent on substance $s$ ($n_g$ for gas and $n_l$ for liquid)
- $f$: Interpolation coordinate for the piecewise linearization of the pressure drop in the tubes, dependent on water cut
- $r$: Interpolation coordinate for the piecewise linearization of the water cut linking constraint

Data

- $C_{TOT}^p$: Capacity of phase $p$ in the first–stage separator at platform–level
- $P_{DROP}^{imn_gn_lf}$: Function for pressure drop in a tube from manifold $m$ to the next manifold or separator in cluster $i$ corresponding to interpolation coordinates $n_s$ for all substances and water cut, in the piecewise linearization of the pressure drop in tubes
- $P_{MAX}^i$: Maximum pressure before choke for all wells in cluster $i$
- $P_{MIN}^{imj}$: Minimum pressure before choke for well $j$ at manifold $m$ in cluster $i$
- $P_{SEP}^i$: Pressure at first stage separator at platform ($= P_{MAX}^i$)
- $P_{WPC}^{imjk}$: Pressure at manifold level before choke in tube from well $j$ connected to manifold $m$ in cluster $i$ in the piecewise linearization of the well performance curve (WPC) with interpolation coordinate $k$
- $Q_{DROP}^{imns_n}$: Flow rate from manifold $m$ in cluster $i$ corresponding to interpolation coordinate $n_s$ for that particular substance $s$ in the piecewise linearization of the pressure drop in tubes
- $Q_{MAX}^{imjs}$: Maximum flow of substance $s$ from well $j$ in cluster $i$ up to manifold $m$
- $Q_{WPC}^{imjsk}$: Flow rate of substance $s$ from well $j$ in manifold $m$ in cluster $i$ corresponding to interpolation coordinate $k$ in the piecewise linearization of the well performance curve (WPC) with interpolation coordinate $k$
linearization of the well performance curve (WPC)

$U_{im1r}$ Data for interpolation point $r$ for equation 1 for manifold $m$
in cluster $i$ in the piecewise linearization of the water cut linking
constraint

$U_{im2r}$ Data for interpolation point $r$ for equation 2 for manifold $m$
in cluster $i$ in the piecewise linearization of the water cut linking
constraint

$E^{WLC}_{imt}$ Maximum error allowed for the water cut linking constraint for tube $t$
in manifold $m$ in cluster $i$

$F^{WELL}_{imj}$ Water cut for well $j$ connected to manifold $m$
in cluster $i$

$F^{DROP}_{imf}$ Water cut from manifold $m$ in cluster $i$ corresponding to
interpolation coordinate $f$ in the piecewise linearization
of the pressure drop in the tubes

Variables

$P^{M}_{imt}$ Pressure at manifold level in tube $t$ for flow from manifold
$m$ in cluster $i$, $m \in \{0\} \cup M(i)$

$P^{DROP}_{imt}$ Pressure drop across a section of tube $t$ with flow from mani-
fold $m$ in cluster $i$ to either the next manifold or the separator

$P^{WELL}_{imj}$ Pressure at manifold level before choke in tube from well $j$
connected to manifold $m$ in cluster $i$

$q^{PIPE}_{imt}$ Flow rate of substance $s$ in tube $t$ from manifold $m$ to manifold
$m - 1$ or separator in cluster $i$

$q^{TEMP}_{imt}$ Temporary variable for the flow rate of substance $s$ from well $j$
connected to manifold $m$ into tube $t$ in cluster $i$

$q^{WELL}_{imj}$ Flow rate of substance $s$ from well $j$ to manifold $m$
in cluster $i$

$f^{PIPE}_{imt}$ Water cut in tube $t$ from manifold $m$ in cluster $i$

$x_{imj}$ Binary variable. Equals 1 if well $j$ at manifold $m$ in cluster $i$
is closed, 0 otherwise. This variable is not defined for $j \in H(i, m)$

$y_{imjt}$ Binary variable. Equals 1 if well $j$ at manifold $m$ in cluster $i$
is routed to tube $t$, 0 otherwise

$\gamma_{imjk}$ Weighting variable associated with each interpolation coordinate $k$ in the
piecewise linearization of the WPC for well $j$ connected to mani-
fold $m$ in cluster $i$

$\lambda_{imtn}n_{nf}$ Weighting variable associated with the interpolation coordinates for $n_g$,
$n_l$ and $f$ in the piecewise linearization of the pressure drop in
tubes for manifold $m$ in cluster $i$ and tube $t$

$\eta^{SUB}_{imtnsn}$ Variable connected to the modeling of the SOS2 sets for the
### A.2 Mathematical Model

**Objective function**

\[
\max \ Z = \sum_{i \in I} \sum_{t \in T(i)} q_{i\text{mts}}^{\text{PIPE}} - \sum_{j \in J(i,m)} q_{imjls}^{\text{TEMP}} P_{imj}^{\text{WELL}} + q_{i(m+1)jls}^{\text{TEMP}} P_{i(m+1)j}^{\text{WELL}} \\
\begin{cases} 
    1 
\end{cases}, \ m = \{1\}, 
\]

(A.1)

The stated goal of StatoilHydro is to maximize the total amount of oil produced at the platform. This is expressed in the objective function as the sum of liquid in the pipelines minus the sum of liquid from each well multiplied with the water cut in that well.
A.2. MATHEMATICAL MODEL

Constraints

The constraints can be divided into handling capacity, linearization of well performance curves, mass balance, linearization of the water cut linking constraint, pressure requirements, linearization of pressure drop, closing of wells, removal of symmetry and requirements on variables.

Handling capacity  The platforms handling capacity is the amount of gas and water the separator can handle. The gas constraint is currently the only binding constraint at the Troll West, but StatoilHydro has indicated that the water handling capacity might also become an active constraint in the future when the field matures.

Water

\[
\sum_{i \in I} \sum_{t \in T(i)} \sum_{j \in J(i,m)} q_{imjts}^{\text{TEMP}} F_{imj}^{\text{WELL}} + q_{i(m+1)jts}^{\text{TEMP}} F_{i(m+1)j}^{\text{WELL}} \leq C_{p}^{\text{TOT}}
\]

\[s = \{l\}, p = \{w\}, m = \{1\}, \forall \ t \in T(i)\]  

(A.2)

The total water produced is the sum of water from each well expressed as the liquid from each well in all clusters multiplied with water cut for that well.

Gas

\[
\sum_{i \in I} \sum_{t \in T(i)} q_{imts}^{\text{PIPE}} \leq C_{p}^{\text{TOT}}, s = \{g\}, p = \{g\}, m = \{1\}
\]

(A.3)

The total gas produced is expressed as the sum of gas flowing from manifold 1 to the separator for all clusters.

As the constraints covered in the rest of the chapter apply for all clusters, \( I \), we have dropped writing \( \forall \ i \in I \) when writing out all the constraints.

Linearization of Well Performance Curves  The linearization of the Well Performance Curves (WPC) are expressed in the constraints below. As described in section 1.2, the wellhead pressure will decide the flows of liquid and gas from the well to the manifold. The \( \gamma \) variables are modeled as SOS2.

\[
p_{imj}^{\text{WELL}} = \sum_{k \in K(i,m,j)} p_{imjk}^{\text{WPC}} \gamma_{imjk} \ \forall \ m \in M(i), j \in J(i,m)
\]

(A.4)

\[
q_{imjs}^{\text{WELL}} = \sum_{k \in K(i,m,j)} q_{imjsk}^{\text{WPC}} \gamma_{imjk} \ \forall \ m \in M(i), j \in J(i,m), s \in S
\]

(A.5)
\[
\sum_{k \in K(i,m,j)} \gamma_{imjk} = 1 \quad \forall \ m \in M(i), \ j \in J(i,m) \quad (A.6)
\]
\[
\gamma_{imjk} \geq 0 \quad \forall \ m \in M(i), \ j \in J(i,m), \ k \in K(i,m,j) \quad (A.7)
\]
\[
\gamma_{imjk} \text{ is SOS2 } \quad \forall \ m \in M(i), \ j \in J(i,m) \quad (A.8)
\]

Routing of wells This constraint was introduced to enforce the separation of wells into a set of wells that can be closed and a set of wells that cannot be closed. \( x_{imj} \) is only defined for the set of wells that can be closed. This ensures that the wells that cannot be closed produce, and are routed to one of the two pipelines.

\[
x_{imj} + \sum_{t \in T(i)} y_{imjt} = 1 \quad \forall \ m \in M(i), \ j \in J(i,m) \quad (A.9)
\]

Mass balance The mass balance constraints forces the flow from a well to be equal to the sum of all flows routed to the different tubes from that well. They also ensure that the flow of liquid or gas in a tube has to be equal to the sum of flow from all wells routed to that tube plus the potential flow from a manifold further upstream in the cluster. The flow from the manifold furthest upstream, \( \max_{m \in M(i)} m \), will only have to be equal to the sum of the flow from each well, that is the first term on the right hand side in equation A.12.

\[
\sum_{t \in T(i)} q_{imjp}^{\text{TEMP}} = q_{imjp}^{\text{WELL}} \quad \forall \ m \in M(i), \ j \in J(i,m), \ p \in P \quad (A.10)
\]

\[
q_{imjts}^{\text{TEMP}} \leq q_{imjss}^{\text{MAX}} y_{imjt} \quad \forall \ m \in M(i), \ j \in J(i,m), \ t \in T(i), \ s \in S \quad (A.11)
\]

\[
q_{limts}^{\text{PIPE}} = \sum_{j \in J(i,m)} q_{imjts}^{\text{TEMP}} + q_{i(m+1)ts}^{\text{PIPE}} \quad \forall \ m \in M(i)/\{\max m\}, \ t \in T(i), \ s \in S \quad (A.12)
\]

Linearization of water cut linking constraints Since the objective function and the water handling constraint is calculated using the well water cut, an additional constraint needs to be added to make sure these water cuts match the water cut in the pipeline for the pressure drop model. This constraint was shown in equation 4.10. The equations
needed to linearize this constraint are equations A.13 to A.32. Notice equation A.26 which is the only variable in our model that is allowed to be negative.

Equations A.13 and A.14 concern the variable transformation necessary to alter the LHS in equation 4.10 so it becomes a separable function.

\[
u_{WLC}^{imt_1} = \frac{q_{lims}^{PIPE} + f_{lim}^{PIPE}}{2} \\
\text{s} = \{l\}, \forall \ m \in \mathcal{M}(i), \ t \in \mathcal{T}(i) \tag{A.13}
\]

\[
u_{WLC}^{imt_2} = \frac{q_{lims}^{PIPE} - f_{lim}^{PIPE}}{2} \\
\text{s} = \{l\}, \forall \ m \in \mathcal{M}(i), \ t \in \mathcal{T}(i) \tag{A.14}
\]

Equations A.15 through to A.28 are the equations necessary for the piecewise linearization of the resulting LHS, \((u_{WLC}^{imt_1})^2\) and \(-(v_{WLC}^{imt_2})^2\), after the transformation.

\[
u_{WLC}^{imt_1} = \sum_{r \in \mathcal{R}} U_{im1r}^{WLC} \mu_{im1r}^{WLC} \\
\forall \ m \in \mathcal{M}(i), \ t \in \mathcal{T}(i) \tag{A.15}
\]

\[
u_{WLC}^{imt_2} = \sum_{r \in \mathcal{R}} U_{im2r}^{WLC} \mu_{im2r}^{WLC} \\
\forall \ m \in \mathcal{M}(i), \ t \in \mathcal{T}(i) \tag{A.16}
\]

\[
u_{WLC}^{imt_1} = \sum_{r \in \mathcal{R}} U_{im1r}^{2} \mu_{im1r}^{WLC} \\
\forall \ m \in \mathcal{M}(i), \ t \in \mathcal{T}(i) \tag{A.17}
\]

\[
u_{WLC}^{imt_2} = \sum_{r \in \mathcal{R}} U_{im2r}^{2} \mu_{im2r}^{WLC} \\
\forall \ m \in \mathcal{M}(i), \ t \in \mathcal{T}(i) \tag{A.18}
\]
\[\sum_{r \in R} \mu_{WLC}^{imt_1} = 1 \quad \forall m \in M(i), \ t \in T(i) \quad (A.19)\]

\[\sum_{r \in R} \mu_{WLC}^{imt_2} = 1 \quad \forall m \in M(i), \ t \in T(i) \quad (A.20)\]

\[\mu_{WLC}^{imt_1} \geq 0, \ \forall m \in M(i), \ t \in T(i), r \in R \quad (A.21)\]

\[\mu_{WLC}^{imt_2} \geq 0, \ \forall m \in M(i), \ t \in T(i), r \in R \quad (A.22)\]

\[\mu_{WLC}^{imt_1} \text{ is SOS2 }, \ \forall m \in M(i), \ t \in T(i), r \in R \quad (A.23)\]

\[\mu_{WLC}^{imt_2} \text{ is SOS2 }, \ \forall m \in M(i), \ t \in T(i), r \in R \quad (A.24)\]

\[u_{WLC}^{imt_1} \geq 0, \ \forall m \in M(i), \ t \in T \quad (A.25)\]

\[u_{WLC}^{imt_2} \text{ is Free }, \ \forall m \in M(i), \ t \in T \quad (A.26)\]

\[v_{WLC}^{imt_1} \geq 0, \ \forall m \in M(i), \ t \in T \quad (A.27)\]

\[v_{WLC}^{imt_2} \geq 0, \ \forall m \in M(i), \ t \in T \quad (A.28)\]
A.2. MATHEMATICAL MODEL

Linearized water cut linking constraint  Equations A.29 and A.30 are the transformed WCL constrains.

\[
\begin{align*}
\frac{WLC}{WLC} v_{imt1} - \frac{WLC}{WLC} v_{imt2} &= \sum_{j \in J(i,m)} q_{imjts} F_{imj}^WELL + q_{i(m+1)jts} F_{i(m+1)j}^WELL + \epsilon_{imt}^WLC \\
&= s = \{1\}, m = \{1\}, t \in T(i) \quad (A.29)
\end{align*}
\]

\[
\begin{align*}
\frac{WLC}{WLC} v_{imt1} - \frac{WLC}{WLC} v_{imt2} &= \sum_{j \in J(i,m)} q_{imjts} F_{imj}^WELL + \epsilon_{imt}^WLC \\
&= s = \{1\}, m = \{2\}, t \in T(i) \quad (A.30)
\end{align*}
\]

The allowable difference between the LHS and the RHS in equation A.29 and A.30 are defined in A.31 and A.32

\[
-\epsilon_{imt}^WLC \leq \epsilon_{imt}^WLC \leq E_{imt}^WLC \\
\forall \ m \in \mathcal{M}(i), \ t \in T(i) \quad (A.31)
\]

\[
\epsilon_{imt}^WLC \geq 0 \ \forall \ m \in \mathcal{M}(i), \ t \in T \quad (A.32)
\]

The \( \epsilon_{imt}^WLC \) variable in equation A.29 to A.32 was added in the hope of decreasing the solution time of the model. It was thought that allowing a small deviation between the LHS and the RHS in this constraints A.29 and A.30 would not decrease the accuracy of the model while making it easier to find a solution.

Requirements on pressure  The following constraint enforces that the pressure in tube \( t \) downstream from manifold \( m \) in cluster \( i \) has to be lower than the pressure before the choke in all wells connected to manifold \( m \) in cluster \( i \) routed to tube \( t \). \( P_{imj}^{MAX} - P_{imj}^{MIN} \) is used as the smallest possible Big M.

\[
P_m^M \leq p_{imj}^WELL + (P_{imj}^{MAX} - P_{imj}^{MIN}) \ (1 - y_{imjt}) \ \forall \ m \in \mathcal{M}(i), \ j \in J(i,m), \ t \in T(i) \quad (A.33)
\]

Linearization of pressure drop  As described in the beginning of this chapter, the pressure drop in the tubes is modeled in three dimensions as a function of gas, liquid and water cut. The linearization of the pressure drop covered in section 3.2.4 makes it necessary to introduce the constraints A.34 - A.45
\[ q_{imts}^{PIPE} = \sum_{n_g \in N_{img}} \sum_{n_l \in N_{iml}} \sum_{f \in F_{im}} Q_{imtn_nf}^{DROP} \lambda_{imtn_nf} \]
\[ \forall \, m \in \mathcal{M}(i), \, t \in \mathcal{T}(i), \, s \in \mathcal{S} \]  
\hspace{1cm} (A.34)

\[ f_{imt}^{PIPE} = \sum_{n_g \in N_{img}} \sum_{n_l \in N_{iml}} \sum_{f \in F_{im}} F_{imn_f}^{DROP} \lambda_{imtn_nf} \]
\[ \forall \, m \in \mathcal{M}(i), \, t \in \mathcal{T}(i) \]  
\hspace{1cm} (A.35)

\[ P_{imt}^{DROP} = \sum_{n_g \in N_{img}} \sum_{n_l \in N_{iml}} \sum_{f \in F_{im}} P_{imtn_nf}^{DROP} \lambda_{imtn_nf} \]
\[ \forall \, m \in \mathcal{M}(i), \, t \in \mathcal{T}(i) \]  
\hspace{1cm} (A.36)

\[ \sum_{n_g \in N_{img}} \sum_{n_l \in N_{iml}} \sum_{f \in F_{im}} \lambda_{imtn_nf} = 1 \quad \forall \, m \in \mathcal{M}(i), \, t \in \mathcal{T}(i) \]  
\hspace{1cm} (A.37)

\[ \lambda_{imln_nf} \geq 0 \quad \forall \, m \in \mathcal{M}(i), \, t \in \mathcal{T}(i), \, n_g \in N_{img}, \, n_l \in N_{iml}, \, f \in F_{im} \]  
\hspace{1cm} (A.38)

\[ \eta_{imtnsn_s}^{SUB} = \sum_{n_l \in N_{iml}} \sum_{f \in F_{im}} \lambda_{imtn_nf} \]
\[ \forall \, m \in \mathcal{M}(i), \, t \in \mathcal{T}(i), \, s = \{g\}, \, n_g \in N_{img} \]  
\hspace{1cm} (A.39)

\[ \eta_{imtnsn_s}^{SUB} = \sum_{n_l \in N_{iml}} \sum_{f \in F_{im}} \lambda_{imtn_nf} \]
\[ \forall \, m \in \mathcal{M}(i), \, t \in \mathcal{T}(i), \, s = \{l\}, \, n_l \in N_{iml} \]  
\hspace{1cm} (A.40)

\[ \eta_{imtnsn_s}^{FRAC} = \sum_{n_g \in N_{img}} \sum_{n_l \in N_{iml}} \lambda_{imtn_nf} \]
\[ \forall \, m \in \mathcal{M}(i), \, t \in \mathcal{T}(i), \, p = \{f\}, \, f \in F_{im} \]  
\hspace{1cm} (A.41)

\[ \eta_{imtnsn_s}^{SUB} \geq 0 \quad \forall \, m \in \mathcal{M}(i), \, t \in \mathcal{T}(i), \, s \in \mathcal{S}, \, n_s \in N_{ims} \]  
\hspace{1cm} (A.42)
A.2. MATHEMATICAL MODEL

\[ \eta_{imtf}^{FRAC} \geq 0 \quad \forall \ m \in \mathcal{M}(i), \ t \in \mathcal{T}(i), \ f \in \mathcal{F}_{im} \quad (A.43) \]

\[ \eta_{imtsn}^{SUB} \text{ is SOS2} \quad \forall \ m \in \mathcal{M}(i), \ t \in \mathcal{T}(i), \ s \in \mathcal{S} \quad (A.44) \]

\[ \eta_{imtf}^{FRAC} \text{ is SOS2} \quad \forall \ m \in \mathcal{M}(i), \ t \in \mathcal{T}(i), \ f \in \mathcal{F} \quad (A.45) \]

**Pressure requirements** The pressure drop in a tube from one manifold to another manifold, or up to the separator, needs to be equal to the difference in manifold pressure between the two. The pressure at the first stage separator is fixed.

\[ p_{im(i-1)t}^M = p_{imt}^M - p_{imt}^{DROP} \quad \forall \ m \in \mathcal{M}(i), \ t \in \mathcal{T}(i) \quad (A.46) \]

\[ p_{i0t}^M = P_{SEP} \quad \forall \ t \in \mathcal{T}(i) \quad (A.47) \]

**Requirements on variables** Binary requirements must be put on the variable \( x_{imj} \), which represents whether a well is open or closed. The same is true for \( y \) which contains information on whether a well has been routed to a certain tube or not.

\[ x_{imj} \in \{0,1\} \quad \forall \ m \in \mathcal{M}(i), \ j \in \mathcal{J}(i,m) \quad (A.48) \]

\[ y_{imjt} \in \{0,1\} \quad \forall \ m \in \mathcal{M}(i), \ j \in \mathcal{J}(i,m), \ t \in \mathcal{T}(i) \quad (A.49) \]

\[ p_{imt}^{DROP} \geq 0 \quad \forall \ m \in \mathcal{M}(i), \ t \in \mathcal{T}(i) \quad (A.50) \]

\[ p_{imt}^M \geq 0 \quad \forall \ m \in \{0\} \cup \mathcal{M}(i), \ t \in \mathcal{T}(i) \quad (A.51) \]

\[ p_{imj}^{WELL} \geq 0 \quad \forall \ m \in \mathcal{M}(i), \ j \in \mathcal{J}(i,m) \quad (A.52) \]

\[ q_{imts}^{PIPE} \geq 0 \quad \forall \ m \in \mathcal{M}(i), \ t \in \mathcal{T}(i), \ s \in \mathcal{S} \quad (A.53) \]

\[ q_{imjts}^{TEMP} \geq 0 \quad \forall \ m \in \mathcal{M}(i), \ j \in \mathcal{J}(i,m), \ t \in \mathcal{T}(i), \ s \in \mathcal{S} \quad (A.54) \]
\( q_{imjs}^{WELL} \geq 0 \quad \forall \ m \in \mathcal{M}(i), \ j \in \mathcal{J}(i,m), \ s \in \mathcal{S} \) \hspace{1cm} (A.55)

\( f_{int}^{PIPE} \geq 0 \quad \forall \ m \in \mathcal{M}(i), \ t \in \mathcal{T} \) \hspace{1cm} (A.56)

**Closing of wells**  If a well is open, the last data point corresponding to zero flow in the interpolation of the WPC curve cannot be used, that is, it needs to have zero weight. This is to prevent a well from producing no flow. Closing down wells might lead to problems opening them again and StatoilHydro therefore preferably wants the wells to produce a small amount of flow even though they are not a part of the set of producing wells in the optimal solution. This was discussed in section 1.4.2.

\( \gamma_{imjk} - x_{imj} \leq 0 \quad \forall \ m \in \mathcal{M}(i), \ j \in \mathcal{J}(i,m), \ k \in \text{Max}\{K(i,m,j)\} \) \hspace{1cm} (A.57)

**Removal of symmetry**  Assuming the tubes between manifolds are identical, an additional constraint to remove symmetry can be introduced. In a two tube example, if not equal there will always be more wells routed to one of the tubes than the other. If there are more than two tubes, tube \( t-1 \) should have more or equal the amount of wells routed to it compared to tube \( t \).

\[ \sum_{j \in \mathcal{J}(i,m)} (y_{imj(t-1)} - y_{imjt}) \geq 0 \quad \forall \ m \in \mathcal{M}(i), \ t \in \mathcal{T}(i)/\{1\} \] \hspace{1cm} (A.58)
Appendix B

Mosel code

The following pages present the implemented Mosel code. Since the code for some of the strategies are equal, we have only included one set of declarations and constraints for the DWD strategies.

- **Model of Hagem & Torgnes**
  - Declarations
  - Constraints
  - Master problem

- **Dantzig-Wolfe Decomposition**
  - Declarations
  - Constraints
  - Sub problem
  - Master problem - Sequential, Basic and Accelerated Feedback strategy

- **Branch & Price**
  - Declarations
  - Constraints
  - Master problem