New Concepts and Algorithms for Fully Transparent Distribution Management Systems

A dissertation submitted to
ETH Zurich
for the degree of
Doctor of Sciences

presented by

Monika Esther Ruh
Dipl. El.-Ing. ETH
born December 13, 1977
citizen of Schaffhausen (SH) and Buch (SH)

accepted on the recommendation of:

Prof. Dr. Göran Andersson, examiner
Prof. Dr. Christian Rehtanz, co-examiner

2010
Dedicated to my parents in deep gratitude
Preface

This thesis was written during my research activities at the Power Systems Laboratory of ETH Zurich from October 2005 till May 2010. These four and a half years have been a very interesting, demanding and enriching time for me. As without the support and help of many people around me, this thesis would not have been written, I like to express my deep thank to all of them.

My highest thanks goes to Professor Dr. Göran Andersson for being my supervisor and for giving me the opportunity to be a member of his research group. Under his excellent guidance, I was able to learn at lot concerning research and academic work. His scientific support, his precious advices and his encouraging feedback always helped me to progress in my PhD project. Besides, with his kind and friendly manner, he created a familiar atmosphere in his research group so that I really enjoyed to work there. What Professor Andersson meant for me during my PhD studies is thus better described with the German expression for doctoral supervisor, namely “Doktorvater”, what he truly was.

Special thanks goes to Professor Dr. Christian Rehtanz, who agreed to be the co-examiner of this thesis. As a master student, I took the course “Power Systems Dynamic and Control”, which he gave together with Professor Andersson. Actually, it was in this lecture that I first learnt about the importance of ergonomic and user-friendly design for power grid control systems. Hence, it was a special pleasure for me that Professor Rehtanz was the co-examiner of my dissertation.

Especially, I would like to thank Rittmeyer AG for being the partner of my PhD project. Rittmeyer company not only financially supported the project but also helped me a lot with valuable meetings, useful courses, planning workshops and with technical support for the RITOP® software. Besides, the responsible persons of the company were really interested in the progress of my work and the results I got. My deepest thank goes to Andreas Borer, who was my tutor from the company. Throughout all phases of my PhD, he supported my work with helpful
suggestions, good ideas and challenging comments concerning the industrial feasibility of my developed ideas. Even after he became the CEO of Rittmeyer AG at the beginning of 2009, he continued to tutor my research project. Above all, he always took the time for some humorous discussions and cheerful conversations. Special thanks goes also to Christoph Bücheler, Mathias Hartmann, Bernhard Müller and Gerhard Nigg for valuable and stimulating discussions as well as for their great support concerning the RITOP® software.

I am very grateful to all my colleagues of the Power Systems Laboratory. Due to the friendly atmosphere, the nice conversations, the warm companionship and the mutual assistance, I had a very good time in the group. Many thanks goes also to the secretary Rita Zerjeski for all the administrative work she had to do for me. Especially, I like to thank my dear room-mates Marija Zima-Bočkarjova and Martin Kurzidem for the great working atmosphere we had during these more than four years in our office. We had a lot of good and interesting discussions not only about electric power systems but also about many other important topics of life. Mainly during the last year of our PhD studies, we supported each other with valuable advices and friendly encouraging words whenever needed.

Finally, my greatest thank goes to those closest to me: My family. I deeply thank my parents and my twin sister Elisabeth for all their love. I am profoundly grateful to my parents for everything they did for me during the last 33 years. Without their encouraging support, their sympathetic understanding, their deep love and their complete confidence in me – I think this thesis would never have been written. Above all, I am very grateful that they taught me that there are non-material values in life and that they showed my how to have an open mind for all the beauty of the world.

So in deep gratitude, this thesis is dedicated to my parents.

Monika Ruh
Abstract

With the advent of distributed generation during the last years, distribution grids are no longer purely passive load systems, what makes that the operation of them has become more complex. For their control and monitoring, distribution management systems (DMSs) have become more important than ever. So, the general aim of this thesis is to develop a future DMS that will meet the new challenges.

To explore innovation potential of existing DMSs, a state-of-the-art survey is carried out. The major conclusion of this survey is that a DMS with a fully transparent data architecture is really needed. Consequently, the development of such a fully transparent DMS becomes the main objective of this thesis. Since grid models are very important in modern DMSs providing grid analysis functions, a so called transparency matrix is introduced that represents the grid model in matrix form.

As the grid model represented by the transparency matrix is too detailed for many grid analysis functions preferring simple node-branch-grid models, an efficient link between the transparency matrix and the simplified grid model is needed. A condensing algorithm is developed that establishes such an efficient link by condensing the transparency matrix into a so called condensed transparency matrix.

Fully transparent DMSs need an additional procedure that detects the actual topology of the distribution grid, mainly to check if rings are currently existing. Thus, a so called condensed grid topology detection algorithm is developed that detects all existing rings and identifies all grid branches that are lying within such a ring.

In addition, a power flow calculation method that can be used for radial distribution grids as well as for weakly meshed distribution grids is developed. The condensed grid topology detection algorithm provides the power flow calculation method all the needed information, thus all data the power flow analysis depends on are coming from the original data source, namely the transparency matrix. This kind of transparent data flow is exactly what is required for a fully transparent DMS.
Kurzfassung


Komplett durchgängige DMS-Leitsysteme brauchen eine zusätzliche Prozedur, die die aktuelle Topologie des Verteilnetzes ermittelt, hauptsächlich um zu prüfen, ob Ringe gegenwärtig existieren. Daher wird ein sogenannter Condensed Grid Topology Detection Algorithmus entwickelt, welcher alle existierenden Ringe detektiert und alle Leitungen, welche innerhalb eines solchen Ringes liegen, erkennt.
Contents

1 Introduction
   1.1 Motivation for Power Grid Control Systems ................. 1
   1.2 Definition of Power Grid Control Systems .................. 2
   1.3 Distribution Management Systems ......................... 5
   1.4 Thesis Outline ............................................ 5
   1.5 Contributions ............................................. 6
   1.6 List of Publications ....................................... 7

2 Power Grid Hierarchy and Power System Control Structure .... 9
   2.1 Hierarchy of Interconnected Power Grids ................... 9
      2.1.1 Transmission Grid ..................................... 10
      2.1.2 Subtransmission Grids ................................ 14
      2.1.3 Regional Distribution Grids ........................... 15
      2.1.4 Local Distribution Grids ............................... 18
   2.2 Structure of Power System Control Levels ................. 19
      2.2.1 Local Control on Bay Level ............................ 20
      2.2.2 Station Control on Station Level ...................... 21
      2.2.3 Remote or Supervisory Control on Grid Level ........ 21
      2.2.4 Grid Control Level and Hierarchical Grid Level ...... 22
Table of Contents

2.3 Structure of Power Grid Control Centers ........................................ 24

3 Results of the State-of-the-Art Survey of Modern Power Grid Control Systems 27

3.1 State-of-the-Art of Power Grid Control Systems ......................... 27
  3.1.1 Base System Functions ............................................. 28
  3.1.2 Applications ...................................................... 29
  3.1.3 Integration of Base System Functions and Applications ........ 32

3.2 Innovation Potential of State-of-the-Art DMSs .............................. 33
  3.2.1 DMSs with Badly Configurable Data Architectures .......... 33
  3.2.2 DMSs with Ambiguous or Mistakable Visualizations .......... 37
  3.2.3 Grid Analysis Functions Have to Be Explicitly Retrieved .... 39

3.3 Conclusions of the State-of-the-Art Survey .................................. 39

4 A New Conceptual Model for a Fully Transparent DMS 41

4.1 Fully Transparent Data Architecture ........................................ 41
  4.1.1 Criteria for a Transparent Data Architecture ................. 42
  4.1.2 Levels of Transparency ......................................... 44

4.2 The Transparency Matrix .................................................... 45
  4.2.1 Structure of the Transparency Matrix ........................... 46
  4.2.2 Managing Topologies with the Transparency Matrix .......... 51
  4.2.3 Practical Aspects of the Transparency Matrix ................ 54
  4.2.4 The Properties of the Transparency Matrix .................... 57
  4.2.5 The Transparency Level of the Transparency Matrix .......... 59
  4.2.6 Verification of the Transparency Matrix’ Data Correctness .. 59

4.3 The Transparency Configuration Manager .................................... 62
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.3.1</td>
<td>Motivation for the Transparency Configuration Manager</td>
<td>62</td>
</tr>
<tr>
<td>4.3.2</td>
<td>The Principal Idea of the Transparency Configuration Manager</td>
<td>62</td>
</tr>
<tr>
<td>4.3.3</td>
<td>The Advantage of the Transparency Configuration Manager</td>
<td>64</td>
</tr>
<tr>
<td>4.4</td>
<td>Concluding Remarks</td>
<td>64</td>
</tr>
<tr>
<td>5</td>
<td><strong>Power Grid Models and Grid Topology Detection for a Fully Transparent DMS</strong></td>
<td>67</td>
</tr>
<tr>
<td>5.1</td>
<td>Introduction</td>
<td>67</td>
</tr>
<tr>
<td>5.2</td>
<td>The Condensing Algorithm</td>
<td>68</td>
</tr>
<tr>
<td>5.2.1</td>
<td>The Principal Idea of the Condensing Algorithm</td>
<td>68</td>
</tr>
<tr>
<td>5.2.2</td>
<td>The Procedure of the Condensing Algorithm</td>
<td>69</td>
</tr>
<tr>
<td>5.2.3</td>
<td>The Resulting Condensed Transparency Matrix</td>
<td>91</td>
</tr>
<tr>
<td>5.2.4</td>
<td>The Condensing Algorithm as Topology Processor</td>
<td>92</td>
</tr>
<tr>
<td>5.2.5</td>
<td>The Transparency Level of the Condensing Algorithm</td>
<td>95</td>
</tr>
<tr>
<td>5.3</td>
<td>The Condensed Grid Topology Detection Algorithm</td>
<td>96</td>
</tr>
<tr>
<td>5.3.1</td>
<td>Well-Known Published Algorithms for Traversing and Searching of Radial or Meshed Network Structures</td>
<td>97</td>
</tr>
<tr>
<td>5.3.2</td>
<td>Detecting of the Condensed Grid Structure</td>
<td>98</td>
</tr>
<tr>
<td>5.3.3</td>
<td>The Procedure of the Condensed Grid Topology Detection Algorithm</td>
<td>107</td>
</tr>
<tr>
<td>5.3.4</td>
<td>Concluding Remarks</td>
<td>121</td>
</tr>
<tr>
<td>6</td>
<td><strong>Power Flow Calculation Method for a Fully Transparent DMS</strong></td>
<td>123</td>
</tr>
<tr>
<td>6.1</td>
<td>Introduction</td>
<td>123</td>
</tr>
<tr>
<td>6.2</td>
<td>Power Flow Analysis in a Fully Transparent DMS</td>
<td>123</td>
</tr>
<tr>
<td>6.3</td>
<td>Sequence of Solving the Power Flow Equations</td>
<td>125</td>
</tr>
<tr>
<td>Section</td>
<td>Title</td>
<td>Page</td>
</tr>
<tr>
<td>---------</td>
<td>----------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>6.4</td>
<td>The Procedure of the Power Flow Calculation Method</td>
<td>128</td>
</tr>
<tr>
<td>6.4.1</td>
<td>Initializing the Start Values of the Complex Node Voltages</td>
<td>128</td>
</tr>
<tr>
<td>6.4.2</td>
<td>Updating of the Complex Line Currents by a Depth-First Search Cycle</td>
<td>131</td>
</tr>
<tr>
<td>6.4.3</td>
<td>Updating of the Complex Node Voltages by a Depth-First Search Cycle</td>
<td>135</td>
</tr>
<tr>
<td>6.5</td>
<td>Power Flow Calculation for Ring Situations</td>
<td>138</td>
</tr>
<tr>
<td>6.5.1</td>
<td>Principal Approach to Power Flow Calculation for Ring Situations</td>
<td>138</td>
</tr>
<tr>
<td>6.5.2</td>
<td>Data Preparation for Power Flow Calculation with Rings</td>
<td>146</td>
</tr>
<tr>
<td>6.5.3</td>
<td>Updating of the Complex Line Voltages</td>
<td>150</td>
</tr>
<tr>
<td>6.5.4</td>
<td>Updating of the Complex Line Currents for Ring End Nodes</td>
<td>151</td>
</tr>
<tr>
<td>6.6</td>
<td>Concluding Remarks</td>
<td>155</td>
</tr>
<tr>
<td>7</td>
<td>Software Implementation of the Developed Concepts and Algorithms</td>
<td>157</td>
</tr>
<tr>
<td>7.1</td>
<td>Introduction</td>
<td>157</td>
</tr>
<tr>
<td>7.2</td>
<td>Mapping the Transparency Matrix on the Data Base of an Existing DMS</td>
<td>158</td>
</tr>
<tr>
<td>7.2.1</td>
<td>Generation of the Partition Nodes</td>
<td>160</td>
</tr>
<tr>
<td>7.2.2</td>
<td>Generation of the Transparency Matrix</td>
<td>165</td>
</tr>
<tr>
<td>7.3</td>
<td>Implementation of the Transparency Matrix into RITOP®</td>
<td>174</td>
</tr>
<tr>
<td>7.4</td>
<td>Concluding Remarks</td>
<td>176</td>
</tr>
<tr>
<td>8</td>
<td>Discussion and Outlook</td>
<td>177</td>
</tr>
<tr>
<td>8.1</td>
<td>Summary and Discussion</td>
<td>177</td>
</tr>
<tr>
<td>8.2</td>
<td>Outlook</td>
<td>180</td>
</tr>
<tr>
<td>Section</td>
<td>Page</td>
<td></td>
</tr>
<tr>
<td>------------------------------------------------------------------------</td>
<td>------</td>
<td></td>
</tr>
<tr>
<td>8.2.1 User-Friendly and Situation Appropriate Visualization</td>
<td>181</td>
<td></td>
</tr>
<tr>
<td>8.2.2 A Capable of Learning DMS Giving the Best Possible Support</td>
<td>184</td>
<td></td>
</tr>
<tr>
<td>to the Operator</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8.2.3 DMS with a Simulation, Optimization and Planning Tool</td>
<td>184</td>
<td></td>
</tr>
<tr>
<td>8.3 Concluding Remarks</td>
<td>186</td>
<td></td>
</tr>
<tr>
<td>A Composition Models for Interconnected Power Grids</td>
<td>187</td>
<td></td>
</tr>
<tr>
<td>A.1 Mathematical Description of the Interconnected Power Grid</td>
<td>187</td>
<td></td>
</tr>
<tr>
<td>A.1.1 Potential, Actual and Regular Topology</td>
<td>187</td>
<td></td>
</tr>
<tr>
<td>A.1.2 Grid Components</td>
<td>188</td>
<td></td>
</tr>
<tr>
<td>A.1.3 Partitions</td>
<td>189</td>
<td></td>
</tr>
<tr>
<td>A.1.4 Grid Elements</td>
<td>191</td>
<td></td>
</tr>
<tr>
<td>A.1.5 Availability of Grid Components and Partitions</td>
<td>191</td>
<td></td>
</tr>
<tr>
<td>A.2 Partitions Branches and Partition Nodes</td>
<td>192</td>
<td></td>
</tr>
<tr>
<td>B Test Distribution Grids</td>
<td>195</td>
<td></td>
</tr>
<tr>
<td>B.1 Structure of the Test Distribution Grid 1</td>
<td>195</td>
<td></td>
</tr>
<tr>
<td>B.2 Structure of the Test Distribution Grid 2</td>
<td>199</td>
<td></td>
</tr>
<tr>
<td>C Computation Functions of the Transparency Configuration Manager</td>
<td>203</td>
<td></td>
</tr>
<tr>
<td>C.1 Computation of the Dependent Graphical Coordinates</td>
<td>203</td>
<td></td>
</tr>
<tr>
<td>D Modeling of Distribution Grids</td>
<td>207</td>
<td></td>
</tr>
<tr>
<td>D.1 Modeling and Assumptions for the Power Grid</td>
<td>207</td>
<td></td>
</tr>
<tr>
<td>D.2 Attribute Rows for Power Flow Calculation</td>
<td>207</td>
<td></td>
</tr>
<tr>
<td>D.3 Tolerance Constraints for Convergence Checks</td>
<td>209</td>
<td></td>
</tr>
</tbody>
</table>
Table of Contents

Bibliography 211

Curriculum Vitae 217
<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Full Form</th>
</tr>
</thead>
<tbody>
<tr>
<td>AC</td>
<td>Alternating Current</td>
</tr>
<tr>
<td>ADVAPPS</td>
<td>Advanced Applications</td>
</tr>
<tr>
<td>AGC</td>
<td>Automatic Generation Control</td>
</tr>
<tr>
<td>AEW</td>
<td>Aargauisches Elektrizitätswerk</td>
</tr>
<tr>
<td>ATEL</td>
<td>Aare-Tessin AG für Elektrizität</td>
</tr>
<tr>
<td>BFS</td>
<td>Breadth First Search</td>
</tr>
<tr>
<td>BKW</td>
<td>Bernische Kraftwerke AG</td>
</tr>
<tr>
<td>CIM</td>
<td>Common Information Model</td>
</tr>
<tr>
<td>CKW</td>
<td>Centralschweizerische Kraftwerke AG</td>
</tr>
<tr>
<td>CRGS</td>
<td>Control Room Graphics System</td>
</tr>
<tr>
<td>CROM</td>
<td>Control Room Operations Management</td>
</tr>
<tr>
<td>DFS</td>
<td>Depth First Search</td>
</tr>
<tr>
<td>DG</td>
<td>Distributed Generation</td>
</tr>
<tr>
<td>DMS</td>
<td>Distribution Management System</td>
</tr>
<tr>
<td>EGL</td>
<td>Elektrizitäts-Gesellschaft Laufenburg AG</td>
</tr>
<tr>
<td>EHV</td>
<td>Extra-High Voltage</td>
</tr>
<tr>
<td>EKS</td>
<td>Elektrizitätswerk des Kantons Schaffhausen</td>
</tr>
<tr>
<td>EKT</td>
<td>Elektrizitätswerk des Kantons Thurgau</td>
</tr>
<tr>
<td>EKZ</td>
<td>Elektrizitätswerke des Kantons Zürich</td>
</tr>
<tr>
<td>EMS</td>
<td>Energy Management System</td>
</tr>
<tr>
<td>ENTSO-E</td>
<td>European Network of Transmission System Operators for Electricity</td>
</tr>
<tr>
<td>EOS</td>
<td>Energie Ouest Suisse SA</td>
</tr>
<tr>
<td>EWZ</td>
<td>Elektrizitätswerk der Stadt Zürich</td>
</tr>
<tr>
<td>GUI</td>
<td>Graphical User Interface</td>
</tr>
<tr>
<td>HMI</td>
<td>Human Man Interface</td>
</tr>
<tr>
<td>HV</td>
<td>High Voltage</td>
</tr>
<tr>
<td>KCL</td>
<td>Kirchhoff’s Current Law</td>
</tr>
<tr>
<td>KVL</td>
<td>Kirchhoff’s Voltage Law</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>------------------------------------------------------------</td>
</tr>
<tr>
<td>LFC</td>
<td>Load Frequency Control</td>
</tr>
<tr>
<td>LV</td>
<td>Low Voltage</td>
</tr>
<tr>
<td>MV</td>
<td>Medium Voltage</td>
</tr>
<tr>
<td>NOK</td>
<td>Nordostschweizerische Kraftwerke AG</td>
</tr>
<tr>
<td>OMS</td>
<td>Outage Management System</td>
</tr>
<tr>
<td>RAM</td>
<td>Random Access Memory</td>
</tr>
<tr>
<td>RE</td>
<td>Rätia Energie AG</td>
</tr>
<tr>
<td>SCADA</td>
<td>Supervisory Control and Data Acquisition</td>
</tr>
<tr>
<td>TCM</td>
<td>Trouble Call Management</td>
</tr>
<tr>
<td>TP</td>
<td>Topology Processor</td>
</tr>
<tr>
<td>TSO</td>
<td>Transmission System Operator</td>
</tr>
<tr>
<td>UCTE</td>
<td>Union for the Coordination of Transmission of Electricity</td>
</tr>
<tr>
<td>UHV</td>
<td>Ultra-High Voltage</td>
</tr>
</tbody>
</table>
1 Introduction

1.1 Motivation for Power Grid Control Systems

Electric energy has beneficial characteristics which make it an outstanding energy form: It can not only be relatively easily measured and controlled but can also be efficiently and reliably transported as well as converted into other desired energy forms. Since electric energy is a secondary energy form, it has to be generated by conversion of primary energy resources. Hence, an electric power system has to perform the following tasks:

– Exploitation of primary energy provided in naturally available energy carriers.

– Conversion of the exploited primary energy sources to electric energy.

– Transmission and distribution of the produced electric energy to the consumers over sometimes long distances.

Actually, the utilization of primary energy resources like hydro power, nuclear power, wind power, solar energy or fossil fuels is also that efficient because the energy gained in form of electricity can be transmitted and distributed. Due to all these advantages, electric energy can be considered as an ideal form of energy.

But besides all mentioned advantages, electric energy has one grave main disadvantage: It cannot be conveniently stored in large quantities. Because of this physical fact, the most important requirement an electric power system has to meet, is to keep generation and consumption of electric power balanced at all times. A further special physical characteristic of electric energy is the grid-bound transmission from the producing power plants to the consuming loads. Hence, a sufficiently meshed power grid with adequate transmission capacities is a prerequisite for a secure and
reliable electric power supply. Both mentioned characteristics – the grid-bound energy transmission and the non-storability of electricity – have strongly influenced the design of electric power systems concerning grid structure, choice of primary energy sources, location of power plants and amount of needed spinning reserves.

Because of the above-named disadvantages of electric energy, the responsible grid operators and power supply companies can only perform their complex task of permanently guaranteeing a secure and economic grid operation with the help of power system control technology. In fact, as mentioned in [Tiet06a], efficient instrumentation and control technologies for power grids, especially ergonomically designed power grid control systems, become more important as grid operators and power supply companies have to satisfy more stringent requirements and are faced with more complex tasks.

The importance of ergonomically designed power grid control systems was also recognized when investigating the causes of the transmission system blackouts of the years 2003 to 2006. The analysis of these blackouts confirmed that some transmission control systems were obsolete or not consistently ergonomically designed so these systems did not support the human operators as necessarily desired. For example as reported in [Lali05], a mistakable visualization of the actual system state was one of the principal reasons causing the system collapse of the SBB power grid on June 22nd 2005.

1.2 Definition of Power Grid Control Systems

Power system control technology is a special information technology to ensure the operation of electric power grids. The application area of this control technology has according to [RuSu89] two interfaces:

− On the one hand there is the man-machine interface between the power system control technology and the human operator, whose cognitive abilities in reception and processing of sensory stimuli has to be taken into account.

− On the other hand there is the interconnected power grid with its power plants and transmission lines for generation, transmission and distribution of electric energy. All this equipment needs permanent supervision and control to guarantee the power system being in the normal operating state.
To date, there exists no power system control technology or rather no power grid control system that could control the operation of an electric power grid completely automatically. The human operator decides on many of the significant matters: Based on the received information from the control system and with his technical knowledge and working experience, he determines what has to be done or not. In fact, this is according to [ScMe05] exactly what distinguishes process control technology from automation technology: Process control technology focus on the human operator who decides on many of the significant matters since the process to be monitored and controlled works not completely automatically. Of course - and this is no contradiction - there are automatically proceeding subtasks. For instance, many modern power grid control systems provide the possibility of automatic switching orders for repetitive or planned switching actions. Even though the switching procedure runs automatically, it is invoked by the responsible operator who decided to start the switching order program. To be accurate, there are fully automated control processes in electric power systems, namely primary frequency control and secondary frequency control, often called load frequency control (LFC) or automatic generation control (AGC). But this is nothing contradictory to the above, because these frequency control processes, for example accurately described in [Ande04b], are continuous and fast control processes that have to run independently from the operators decisions.

The above described relations between the human operator, the power grid and the power system control technology with its monitoring units, its automatic controllers and its power grid control system are depicted in Figure 1.1. Besides the closed-control loop containing the process, the monitoring units and the automatic controllers, Figure 1.1 shows a second control loop: This second control loop passes through the process and then goes through the monitoring units or else directly into the power grid control system, which is the central component of the whole power grid control technology. The task of this supervisory software system is to display, message, record and log the monitored and measured process values in a way the human operator gets accurately informed about the actual state of the power grid.

The human operator can then decide with the help of the received information and his technical knowledge as well as with his working experience what has to be done or not. Hence, the human operator "closes" this supervisory control loop or better he is very much "in the loop". This fact is the real reason why the cognitive abilities in reception and processing of sensory stimuli of human beings have to be taken into account: It is crucial that the visualization of process states is done by ergonomic principles.
Figure 1.1: The supervisory control is done by the human operator, who receives the needed information by the power grid control system.
1.3 Distribution Management Systems

The knowledge of the importance of the transmission system for a secure and reliable electric power supply caused a transition from traditional supervisory transmission control systems to integrated energy management systems (EMSs) in about the last fifteen years. In contrast, slight attention was paid to distribution systems. However, during the last years, more regard has been paid to distribution systems due to several reasons. As reported in [SiSa00], the following ones are important to mention:

– The advent of distributed generation: Distribution grids are no longer purely passive load systems due to installed small hydro power plants, cogeneration plants, photovoltaic plants or wind power plants. Hence, their operation becomes more complex.

– Changes of the grid topology: Additional cables, circuit breakers and load switches or even distribution substations have to be connected with the existing distribution grid when new loads have to be supplied. According to [SiSa00], changes of the grid topology are more frequent in distribution systems than in transmission systems.

The above mentioned reasons also indicate the importance of modern distribution control systems. During these recent years, in which distribution grids attracted more attention, their control systems migrated from rather traditional supervisory distribution control systems to integrated distribution management systems (DMSs).

1.4 Thesis Outline

The remainder of the thesis is organized as follows:

- **Chapter 2** presents the hierarchy and structure of interconnected power grids. The four different hierarchical levels in which an interconnected power grid is structured, the different power system control levels and the structure of power grid control systems are described.

- **Chapter 3** describes the results of the state-of-the-art survey carried out to explore innovation potential of existing DMSs. A couple of weak points,
which most of the existing DMSs have in common, are revealed by this state-
of-the-art survey. These identified weak points are discussed so to motivate the development of future DMSs.

- **Chapter 4** develops a new conceptual model for a fully transparent DMS. Since grid models are very important for a modern DMS providing grid analysis functions, a so called transparency matrix is introduced with the idea to represent the grid model in matrix form. This transparency matrix has the task to store all data objects of the DMS.

- **Chapter 5** develops two algorithms, namely the condensing algorithm and the condensed grid topology detection algorithm. The condensing algorithm is needed to establish an efficient link between base system functions and advanced applications. The task of the condensed grid topology detection algorithm is to detect the grid topology so that existing rings can be identified.

- **Chapter 6** presents a power flow calculation method that can be used for radial distribution grids and also for weakly meshed distribution grids. This power flow calculation method receives all its needed information from the condensed grid topology detection algorithm and from the condensed transparency matrix, which has the original transparency matrix as date source.

- **Chapter 7** describes how the developed concepts, especially the transparency matrix, can be turned into a real software solution. By creating the partition nodes, the transparency matrix can be generated out of the data base of an existing DMS into its random access memory (RAM). In addition, Chapter 7 shortly reports how the transparency matrix is implemented in the RITOP® process control system.

- **Chapter 8** closes the thesis by providing a short summary and discussion of the developed concepts and algorithms. It then gives an outlook on possible development objectives for future DMSs.

### 1.5 Contributions

The contributions of this dissertation can be summarized as follows:

- A new model for a fully transparent DMS is developed using a so called transparency matrix. This transparency matrix stores all data objects of the DMS and represents the grid model in matrix form.
A so called condensing algorithm is developed that establishes an efficient link between base system functions and advanced applications by condensing the transparency matrix into a condensed transparency matrix.

A condensed grid topology detection algorithm is developed that has the task to detect the grid topology. This algorithm detects all existing rings and identifies all grid branches that are lying within such a ring.

A power flow calculation method that can be used for radial distribution grids as well as for weakly meshed distribution grids is developed. The main advantage of this developed power flow calculation method is that all its needed information is coming from the transparency matrix.

A method with which the transparency matrix can be mapped on the data base of an existing DMS is developed. This method creates first the partition nodes. Then, using the partition nodes, it generates the transparency matrix into the RAM of the DMS.

1.6 List of Publications

The work in this thesis is basically covered by the following publications:

1. Monika Ruh, Göran Andersson, Andreas Borer:
   *A New Concept for a Fully Transparent Distribution Management System*

2. Monika Ruh, Göran Andersson, Andreas Borer:
   *Power System Modeling for a Fully Transparent Distribution Management System*

3. Monika Ruh, Göran Andersson, Andreas Borer:
   *A Configuration Manager for a Fully Transparent Distribution Management System*

Besides the above list, another publication covering general aspects of electric power systems was written:
1. Monika Ruh:

Grundlagen einer sicheren und zuverlässigen elektrischen Energieversorgung
SGA Bulletin Nr. 47: "Automatiksklaven im täglichen Leben",
Schweizerische Gesellschaft für Automatik (SGA), 2007, (in German).
2 Power Grid Hierarchy and Power System Control Structure

2.1 Hierarchy of Interconnected Power Grids

During the second half of the last century, the continuously increasing demand of electric energy led to widely extended interconnected electric power grids, almost exclusively three-phase alternating current (AC) systems, in all industrial parts of the world. The transmission of electric power over longer distances made it necessary to transform the operation voltage to higher voltage values so to reduce occurring power losses in transmission lines. Thus, technical developments surmounting limitations for higher maximum voltages for alternating current were required. Besides the technical efforts of enhancing and interconnecting already existing power grids and of increasing the possible maximum AC voltages, it was started to conceptionally structure interconnected power grids in four different hierarchical levels:

1. Transmission grid
2. Subtransmission grid
3. Regional distribution grid
4. Local distribution grid

These four hierarchical levels differ in their mode of operation, ownership, purchase and equipment for instrumentation and control. According to [RuSu89], the original concept of structuring interconnected power grids into four hierarchical levels was to have only four different voltages, respectively one voltage per grid level. But due to the historical development of electricity supply, there are variations from the ideal scheme, mainly because there are several voltages per grid level.
It would have been uneconomical to replace existing facilities just because they have another operation voltage than the desired nominal voltage of the concerning grid level. Not only that replacement work is expensive and also costly in terms of time, it is often not feasible because of the general public resistance towards construction of new power transmission lines. Still, when new construction work or renovation is anyway necessary, the responsible grid companies try to get it done so that afterwards only one voltage value per grid level exists.

For instance, the interconnected power grid in continental Europe is structured in such four hierarchical levels, which are schematically pictured in Figure 2.1 and are described in detail in the following. Especially, the regional distribution grid and the local distribution grid will be accurately described, since these are the grid levels that are monitored and supervised by distribution management systems (DMSs). The description focus on structures and conditions of power grids in Switzerland, but similar grid structures can be found often in other countries as well.

### 2.1.1 Transmission Grid

On the highest of these four hierarchical levels is the transmission grid with high voltages of 220 kV and extra-high voltages\(^1\) of 380 kV connecting all high and extra-high voltage grids of continental European countries to one widely extended interconnected power grid reaching from the south of Italy to the north of Denmark and from the west of Portugal to the east of Romania. This huge interconnected transmission grid supplies about 450 million people with the needed electric energy and has an installed capacity of about 630 GW. The supervisory coordination of this according to [WaWa08] largest synchronously operated power grid of the world is carried out by so called transmission system operators, abbreviated TSOs, which all were associated in UCTE over more than five decades. With the founding of the new Pan-European body ENTSO-E on December 19th 2008, all former transmission system operators of UCTE became founding members in this new European network of TSOs. All operational tasks and core activities of UCTE are taken over by one regional group of ENTSO-E, namely the one concerning continental Europe.

The national transmission system operator of Switzerland is Swissgrid, which is

---

\(^1\)Due to [Dorf97], voltage values from 34.5 kV up to 230 kV are referred to as high voltage (HV) and such from 345 kV up to 765 kV are referred to as extra-high voltage (EHV). Voltage values higher than 765 kV are denominated as ultra-high voltage (UHV).
Figure 2.1: The four hierarchical levels of an interconnected power grid.
responsible for the Swiss transmission grid with lines of a total length of 6700 km. Since Switzerland has seven nationwide transmission grid operators, Swissgrid is not only responsible for the grid usage and grid coordination for energy transactions around Switzerland but also for transactions between these seven transmission grid operators inside of Switzerland: It acts as a transmission grid coordinator of the Swiss transmission grid, monitoring its current status, ensuring its secure, reliable and cost-effective operation and suggesting measures for safety enhancement. The seven transmission grid operators mentioned above, namely Alpiq (merger of Atel and EOS at the beginning of 2009), Axpo (former NOK till October 2009), BKW, CKW, EGL, EWZ and Repower (former RE till April 2010), are up to the present rather power supply utilities than just transmission grid operators, since they have large-scale power plants as storage hydro power stations or as nuclear power plants providing electric energy to the transmission grid. It is planned that Swissgrid, which is wholly owned by these seven companies, becomes the owner of the entire Swiss transmission grid in 2013, like reported in [Swiss08]. Alpiq, Axpo, BKW, CKW, EGL, EWZ and Repower are then becoming common power supply utilities. The 380 kV and 220 kV transmission lines of the Swiss part of the UCTE grid and the locations of the associated transmission substations are schematically pictured in Figure 2.2, which is based on information of [Bach07].

The electric energy generated by the large-scale power plants is transported on the transmission power lines over long distances of several hundred kilometers towards the HV substations. After [Kers02], these HV substations located near the load centers, are also denoted as bulk power substations. Due to the permanently changing loads and the therewith subsequently changing power generation as well as the electric power trading between the association partners and the changing actual topology (see Subsection 4.2.2) of the interconnected power grid, the power flow on this highest hierarchical level is ongoingly changing and does not have a fixed direction.

The reason that the UCTE grid has besides the 380 kV extra-high voltage lines also 220 kV high voltage lines is the historical development of the European interconnected power grid: 220 kV power lines have been built first, then with the increasing demand of electric energy requiring larger grid capacities and the technical achievement of higher possible maximum AC voltages, 380 kV power lines were built. If possible, the older 220 kV power lines get replaced or rather upgraded to 380 kV voltages. But due to the general public resistance towards construction of new transmission lines, it is often not possible to build new 380 kV transmission lines or to upgrade the older 220 kV transmission lines.
2.1 Hierarchy of Interconnected Power Grids

Figure 2.2: Switzerland’s part of the UCTE grid consists of many 380 kV and 220 kV power transmission lines. The here presented expansion of the Swiss transmission grid is based on information of [Bach07].
2.1.2 Subtransmission Grids

On the second hierarchical level are the regionally bounded subtransmission grids with the task to supply regional areas, large cities or bigger industrial enterprises with electric energy. These slightly interconnected subtransmission grids, sometimes also referred to as high voltage distribution grids or overregional distribution grids, have voltages between 50 kV to 150 kV and are operated by regional power supply utilities, power supply companies of large cities or grid operators of big industrial enterprises. In Switzerland, on the one hand the supraregional power supply utilities are the above mentioned seven transmission grid operators with supra-cantonal supply areas. On the other hand, the supraregional power supply utilities are cantonal power supply utilities responsible for the electric supply of their own Swiss canton as for example AEW, EKS, EKT and EKZ. The power supply of electric energy for subtransmission grids is either done by substation transformers connected to the transmission grid or by smaller power plants as hydro power stations. Neighboring subtransmission grids are normally interconnected with tie lines, but their connecting circuit breakers are only closed in emergency situations and are open in all normal operation situations so to prevent power flows parallel to the transmission grid. As a matter of fact, the direction of the power flow on this second hierarchical level depends on the actual topologies of the subtransmission grids: The power flow has only a unique direction,

- if there is only one vertical connection, that is to say only one transformer connecting the considered subtransmission grid with the superjacent transmission grid, and
- if the connecting circuit breakers between the considered subtransmission grid and its neighboring subtransmission grids are open.

Then, the electric power flows from the transformer of the feeding substation and from the feeding power plants to the substations connecting the subjacent regional distribution grids. If the subtransmission grid has more than one vertical connection with the transmission grid or if one or more of the connecting circuit breakers between the considered subtransmission grid and its neighboring subtransmission grids are closed, than the direction of the power flow is not fixed but depends on the power flow on the transmission level.

As mentioned already above, the aim of conceptionally structuring interconnected power grids into four hierarchical levels was to have one voltage per grid level. But as it is also the case on the transmission level, there are subtransmission grids
with more than one voltage level. For example, some of the cantonal power supply utilities have subtransmission grids with both 50 kV power lines and 110 kV power lines. Besides, there are no standardized high voltage values on the subtransmission level. Hence, the voltage of a power line on the subtransmission level can have almost every value between 50 kV and 150 kV. Still, power lines with voltages of 110 kV are very often used in subtransmission grids and, according to [FlHi05], should be preferred when doing future planning of new subtransmission power lines.

### 2.1.3 Regional Distribution Grids

On the third hierarchical level are the regional distribution grids with medium voltages (MV) between 3 kV to 50 kV as listed in [FlHi05]. The task of these regional distribution grids consists of supplying medium-sized or small towns, industrial enterprises, urban or rural districts with electric energy by transporting the latter from the feeding MV substations towards the local distribution substations. Accordingly, these distribution grids are owned by municipal utilities, power supply companies or grid operators of smaller industrial enterprises. With the increase of distributed generation (DG), regional distribution grids are fed more often also by small power plants like small hydro power plants, cogeneration plants, photovoltaic plants or small wind power plants. The increased distributed generation consequently caused that the concerning regional distribution grids are no longer purely passive load systems, which makes their operation more complex.

The structure of a concrete regional distribution grid depends mainly on the electric load density which is the maximal load power consumption per unit of area measured in MW per km$^2$. Since, according to [HeDe02], MV substations typically have a rated power of 20 MVA to 50 MVA, regions with higher electric load densities need more feeding MV substations per unit of area, e.g. per km$^2$, than regions with lower electric load densities. The choice of the nominal voltage depends both on the electric load density – and hence on the numbers of MV substations per unit of area – and on the meshing degree of the concerning regional distribution grid. In rural areas, where the load density is low and the grid structure mainly radial, the nominal operating voltage is often chosen to 20 kV. In cities, where the load density is high and the grid structure weakly meshed for reliability reasons, the nominal operating voltage is usually chosen to 10 kV.

Another difference between rural and urban distribution grids is the selection of the transmission medium: In suburban and urban regions, medium voltage cables
are predominantly used, where as in rural regions still overhead transmission lines are used, even so the usage of medium voltage cables is increasing. The cables of the regional distribution grid are underground at a depth of about 1 m to 2 m. If the low voltage cables of the local distribution grid are also underground, then the cables of the regional distribution grid are laid below them.

From the feeding MV substations many outgoing feeders are leading towards the local substations supplying the subordinated local distribution grids with the needed energy. In sparsely populated regions, local distribution substations are often only connected by one feeder with one MV substation. The disadvantage of such a radial tree topology is obvious: In case of a failure or a disturbance in one local distribution substation or on one connection line between two local distribution substations, all subsequent local distribution substations are not electrically supplied. Thus in the majority of cases, the feeders lead back to the original MV substation or to another MV substation to provide redundancy in the form of a possible two-sided infeed. To prevent power flows parallel to the subtransmission grid, such a so called ring feeder is not normally closed, but it contains as a cut-off point at least one connecting circuit breaker, which thus is usually open. In case of a disturbance or a failure, this connecting circuit breaker can be closed so that the affected local substations can be fed over the non-affected part of the feeder on the other side of the disturbance or failure. The direction of the power flow on the regional distribution level is therefore given: It flows from the feeding MV substations to the local distribution substations.

Each of these local distribution substations has at least one busbar to which the lines are connected over switchable devices. Hence, it is possible to isolate the point of error in case of failures or disturbances, like for instance a faulted line, which can be isolated by opening the switchable devices at both of its ends. The transformers in the local distribution substations provide the connection with the local distribution level, where most of the consumers obtain their needed electric energy. Consumers like factories or other large-scale industrial consumers are rarely directly connected to the regional distribution grid.

Figure 2.3 shows the possible topology (see Subsection 4.2.2) of a typical regional distribution grid of a small Swiss town. This concrete urban distribution grid has two feeding MV substations each with two infeeds. The possible topology contains many rings, some of them nested and some with lateral branches, and also some radial feeders. By opening the circuit breakers at the cut-off points, all rings can be opened so that the resulting actual topology has desired radial tree structure.
Figure 2.3: Structure of a typical urban distribution grid of a small Swiss town.
2.1.4 Local Distribution Grids

On the fourth and the lowest hierarchical level are the local distribution grids or rather the low voltage grids with a phase voltage of 230 V and a line-to-line voltage of 400 V. By transformers connected to the regional distribution grid, the electric power supply is done in the local distribution substations, which according to [HeDe02] have often rated powers of 250 kVA, 400 kVA or 630 kVA. The final distribution of electric energy is realized on this local distribution level, where the major part of consumers are connected. As already mentioned, only large-scale industrial loads are fed directly on the regional distribution level. There are also local distribution grids with higher nominal voltage values, namely 1000 V and 690 V, to which industrial loads that are too big for the 400 V voltage level but also too small to be fed directly from the regional distribution level are connected. To enable the connection of single-phase consumers, local distribution grids are designed as four single-conductor systems in contrast to the three higher grid levels constructed as three single-conductor systems.

Similar to regional distribution grids, the actual structure of a local distribution grid depends mainly on the parameter load density. In areas with low load densities like in rural regions, there are mainly radial tree structures with many lateral branches supplied by just one local distribution substation. One problem of such a radial tree structure is that switching-on of large loads causes a voltage drop at each load connection point. But the main disadvantage is that in case of failures or disturbances in the source local distribution substation, the whole subsequent distribution grid is affected. One remedial action to solve the problem are mobile emergency power systems. Another possibility is to provide connection lines to a neighboring local distribution grid. Over switchable devices these connection lines are closed in emergency cases and are otherwise open. In regions with very low load densities, both overhead lines and LV cables are used.

In urban areas as in cities or in villages, where the load densities are higher, the preferred transmission medium are LV underground cables. These LV cables are laid along the streets, whereby often both street sides are used. This kind of cable laying can be expanded to a ring line by adding a coupling device connecting both cables looms. In normal operation situations, the concerning switchable device is open so that the actual topology has a radial tree structure.

In regions with high load densities, local distribution grids are realized also as branched ring structures if possible from the street network. With a certain number
of cut-off points in form of switchable devices, this local distribution grid can be operated so that the actual topology has only a radial tree structure.

As a concluding remark it has to be mentioned that the design of a local distribution grid is an optimization task, which is getting more and more important, especially with the request of establishing smart grids (see for instance \[HeDe02\]).

### 2.2 Structure of Power System Control Levels

Before describing the structure of the different power system control levels in detail, the primary task of power system control and the operating requirements have to be mentioned. Even though power system control is only one subtask of power system operation, it is according to \[RuSu89\] already exceptional complex, since it requires the supervision and control of a given power grid section with aid of instrumentation and control equipment within constraints specified by physical laws as well as technical, legal and economic requirements.

As described in Section 1.1, the most important requirement an electric power system has to meet is to continuously maintain the balance between the electric power produced by the generators and the electric power consumed by the loads including the transmission losses. The primary task of power system control is hence to follow instantly a change in power consumption of the loads by a change in power generation so to restore the power balance and therewith to sustain the system frequency.

Besides the mentioned primary task of keeping the load-generation equality, power system control has additional operating criteria, i.e. concerning:

- Quality of operation
- Security of operation
- Economy of operation

To permanently meet the balance between generation and consumption of electricity and to fulfill the other operating requirements, power system control is done by a number of nested control loops operating on different hierarchical levels like for instance described in \[Ande04b\] or in \[Kund94\]. Typically, the controllers in the control loops on the lower levels are placed locally near the devices to be controlled and are characterized by small time constants, whereas the controllers in
Figure 2.4: The four different function levels of computer based power system control of a modern substation.

The control loops on the higher levels are placed remotely to the subprocess to be controlled and are characterized by larger time constants.

Figure 2.4 depicts the layout of modern computer based power system control for medium-sized and larger substations. Like described in [FlHi05] or [BrLo03], four different function levels can be distinguished. The lowest level is the *process level*, which in the field of electric power systems is often also called *switch level*. On this level are the actuators and sensors of the electric energy process to be controlled and monitored, namely the switching and measurement devices of the substation.

### 2.2.1 Local Control on Bay Level

The level directly above the process level is the first control level called *bay level* or *bay control level*, because its bay level functions only provide or work on data of one single bay. Hence, the bay level contains for each bay of the substation the appropriate instrumentation equipment for measuring, metering, protection and control. Besides *bay control units* and *bay protection units*, there is a *bay monitoring display*, which allows to operate the relevant substation bay locally, for
instance during conduction of maintenance work. Hence, the control executed on the bay level is denoted as local control.

2.2.2 Station Control on Station Level

One level higher than the bay level is the second control level named station level or station control level with station level functions providing data across all switchbays of the concerning substation. The human machine interface (HMI) is done in form of a station control system running on a station computer, which is placed in the station control room located in a building near to the substation switchyard. The station control system, often denoted as substation automation system, has a similar functionality like a power grid control system. The substation automation system has only to supervise one single substation with several busbars, transformers, feeders and line bays, whereas a common power grid control system has to supervise multiple substations in the section of the power grid it is concerned with. The responsible operator in the station control room can supervise the substation by using the information provided by the station control system and interact with the substation process or rather the switching devices by using the numerous menus offered by the station control system. Since this sort of control is executed on the station level it is called station control.

2.2.3 Remote or Supervisory Control on Grid Level

The third control level is the grid level or the grid control level, where a section of the power grid can be supervised and controlled by means of a power grid control system, often referred to as grid level control system. This third control level is the lowest function level where control tasks depending on information of the entire grid section to be controlled can be accomplished. By supervisory control and data acquisition (SCADA) technology, for instance accurately described in [Boye04], typical signals like alarms, status indications as well as measured values are gathered out of all substations. In return, the SCADA system sends limited control instructions like making set point changes on distant process controllers or changing of switch settings to the substations, which can be distributed over large geographic areas. Since the distances between the central control location and the individual substations can add up to several hundred kilometers, the data transmission of the gathered measurement information is often done by radio com-
munication links or by fiber optic cables. It has to be remarked that the factor of
distance, which is common to most SCADA systems, makes this control on the grid
control level both supervisory and remote, so it is denoted either as *supervisory control* or *remote control*.

Modern grid level control systems have as one significant advantage that they can
normally remotely execute all control tasks of any substation in the grid section
they are responsible for. Control tasks of each substation can not only be done
with the help of the concerning substation automation system on the station level
but rather be remotely executed by the grid level control system on the grid control
level. Hence, smaller substations have not to be manned during 24 hours per day
and 365 days per year. These smaller substations are normally unmanned unless
an operator or a maintenance crew has to be on-site because of maintenance work
or disturbances.

### 2.2.4 Grid Control Level and Hierarchical Grid Level

The above described grid control level corresponds to the hierarchical grid levels
explained in Section 2.1. Depending on in which hierarchical level the section of
the power grid to be monitored is located, the grid control level has to be di-
vided in several hierarchical grid levels. For example, distribution grid operators
have as grid control level the regional distribution grid. But since regional dis-
tribution grids receive their needed electric energy mainly by transformers from
the subtransmission level, the subtransmission grid operators have priority with
respect to the regional distribution grid operators, for instance concerning load
shedding operations. Hence, a regional distribution grid purchasing part of its
needed electric energy from the subtransmission grid has also the subtransmission
level as an indirect grid control level: This (second) grid control level does not
allow a direct control of the regional distribution grid, but the latter can be indi-
rectly controlled over the regional distribution grid operators with the help of their
supervisory distribution control system. Subtransmission grids on their part pur-
chase often a certain amount of the energy from the superjacent transmission grid,
so the concerning transmission grid operators are superior to the subtransmission
grid operators. Consequently, the transmission level is an additional indirect grid
control level. This (third) grid control level also only allows an indirect control
of the regional distribution grid: Over the subtransmission grid operator the re-
gional distribution grid operator gets the necessary operation instructions, like for
instance load shedding directions.
Figure 2.5: The six function levels of a modern distribution substation. The five control levels are from top to bottom: the transmission level, the subtransmission level, the regional distribution level, the station level, and the bay level.

Accordingly, a substation of a subtransmission grid operator can be structured in five function levels, where above the station level are two grid levels, namely the subtransmission level and the transmission level. For a transmission grid operator, the situation is somewhat simpler, because there is no higher grid level. In this case, there are just four function levels, since there is only one grid control level, namely the transmission level as shown in Figure 2.6.
2.3 Structure of Power Grid Control Centers

The hierarchical levels of the interconnected power grid are represented also in the structure of the individual grid level control centers. Since the electric power flow and the flow of monitor information and control instructions are not independent from each other, it is according to [Schw79] very important that the control levels correspond to the grid levels.

Each transmission grid operator on the transmission level has its own transmission control center. The coordination between all these transmission control centers of an interconnected power grid is done by a few special coordination control centers. In Switzerland, the coordination of the seven transmission grid operators is done in the coordination control center of Swissgrid as already mentioned in Section 2.1.1.

Subordinated to a certain transmission control center are a couple of subtransmission control centers. Since neighboring subtransmission grids are connected over one or more tie lines, the subtransmission control centers have not only to communicate with the superjacent transmission control center but also with each other, for instance before switching actions of connecting circuit breakers are done.

Figure 2.6: A modern transmission substation structured in three control levels, namely the transmission level, the station level and the bay level.
Figure 2.7: The hierarchical control levels of an interconnected power grid.
Below each subtransmission control center, there are multiple regional distribution control centers located on the regional distribution level. Because of interconnections between these regional distribution grids, the concerning regional distribution control centers have to communicate with each other.

Finally, on the level of the local distribution grids, there are so called low voltage control stations. These low voltage control stations do not have the functionality of a grid level control center. But they have some measurement and control devices to measure the electric energy consumed by customers.

Figure 2.7 shows how the described different control centers are placed on the different grid levels of the power grid and how they are connected with each other.
3 Results of the State-of-the-Art Survey of Modern Power Grid Control Systems

3.1 State-of-the-Art of Power Grid Control Systems

To deepen the knowledge of modern power grid control systems concerning their available standard and special functions as well as their possible enhancements, a state-of-the-art survey was carried out. In addition, this state-of-the-art survey enabled to explore the innovation potential of existing power grid control systems and to find out how they should be modified to better suit future requirements. Especially on the market well-known distribution management systems (DMSs) have been analyzed by non-proprietary information contained in publicly available product demos, fact sheets, brochures and booklets. Moreover, also some most commonly used energy management systems (EMSs) have been closely surveyed to get a general impression of modern power grid control systems.

The findings of the state-of-the-art survey showed that generally functions and applications which are contained in power grid control systems of today can – independently if EMSs supervising transmission grids or DMSs supervising regional distribution grids are considered – be divided into two main groups:

1. Base system functions
2. Applications

The former fulfill the basic requirements of the control system like interacting with the power grid. The latter serve to support the human operator and to carry out extended tasks of the control system. In the following subsections, base system functions and applications are briefly described.
3.1.1 Base System Functions

According to [NoWi07], base system functions can be divided into two subgroups:

1. Supervisory control and data acquisition (SCADA) functions
2. Control room operations management (CROM) functions

In the following, typical SCADA functions providing real-time monitoring and control of the distribution grid and typical CROM functions containing all facilities provided to the human operator in the control room are listed.

**SCADA Functions**

Modern DMSs unexceptionally provide the following comprehensive SCADA functions:

- Real-time basic processing of indications, messages, measurements and metered values
- Real-time processing of derived values like limit value monitoring, quality flags and markers
- Manual updates, remote parameter setting and single command control
- Alarm processing and alarm summaries like alarm lists
- Event lists, log books, measurement reports and trends
- Topological interlocking and checks of interlocking conditions

Since the function range of SCADA is not coupled to the physics of the process to be controlled and monitored, SCADA functions provided in modern remote supervisory control systems are similar or even identical independently of their application type.

**CROM Functions**

According to [NoWi07], CROM functions include two major subfunction groups, namely the human machine interface (HMI) and the control room graphics system (CRGS). The preparation and visualization of the data gathered and integrated by the SCADA functions are made by CRGS functions, such as:
3.1 State-of-the-Art of Power Grid Control Systems

- Graphical presentations of process states and process variables with zooming and panning
- Grid topology coloring like feeder coloring or voltage level coloring
- Geographical grid representation with background maps
- Connection of graphical data with process data
- Locating objects on maps
- Navigating capabilities and dynamic diagrams for detailed grid views

Since the CROM functions, particularly the HMI and the CRGS functions, provide the interface of the supervisory control system to the human operator, they have to be strictly ergonomically designed. The importance of the ergonomic design makes that CROM functions of remote supervisory control systems of different application types like water supply, wastewater treatment, gas or electricity are similar. For EMSs and DMSs of today, the function range of the CROM is even more than less identical.

In most of the modern distribution grid control systems, SCADA functions and CROM functions are closely integrated as written in [NoWi07]. Hence, base system functions are often just referred to as SCADA or enhanced SCADA functions.

3.1.2 Applications

The second main group contains the application-oriented calculation tools that serve to support and to aid the human operator. Whereas SCADA and CROM functions are similar or even identical for different supervisory control systems, the functions of this second group strongly depend on the process to be supervised. As accurately described in [NoWi07], applications can be divided into two different modules:

1. Advanced applications (ADVAPPS)
2. Outage management system (OMS)

Unfortunately, the collectivity of all applications is sometimes called DMS. But this is not really accurate, since applications alone do not constitute an independent
grid control system. *Distribution management functions* would be the better denomination. The function range of these advanced applications and OMS functions is briefly described in the following.

**Advanced Applications**

The module of advanced applications contains comprehensive calculation and analysis functions for the operational control of the distribution grid. The following applications for distribution grid management and optimization are contained in many of the state-of-the-art DMSs:

- Distribution power flow calculation
- Distribution short circuit calculation
- Distribution state estimation
- Load modeling and forecasting
- Optimal feeder reconfiguration
- Fault location and fault distance calculation
- Fault isolation and restoration planning
- Volt/Var control

Since some regional distribution grid operators also run small power plants like river power plants, current DMSs also provide a couple of generation management applications determining the operation scheduling times of the generation units:

- Generation scheduling applications
- Generation control applications

For the study of past fault situations, the storage and archiving of incident events are needed. Hence, many of the modern DMSs contain so called historical information applications allowing to archive event and process data:

- Storage and archiving of process data
- Simulation of historical events
3.1 State-of-the-Art of Power Grid Control Systems

To efficiently supply the connected loads, there are specific load management applications guaranteeing the optimal use of electric power purchase contracts:

– Purchase limit monitoring
– Load forecasting

The accurate accounting of the delivered electric energy is statutory duty for municipal utilities participating in the liberalized electricity market. Thus, distribution grid control systems of today have energy accounting applications:

– Recording of energy generation, exchange and load values
– Calculation of generation per plant, total generation, total energy exchange
– Calculation of fees and bills

The here presented list of application functions does not claim to be complete but indicates the various function range of advanced applications. Important to mention is that all application functions only present their calculation results as a sort of advice to the human operator, but leave the decision of what has to be done or not to him. Therewith, it is recognized that the human operator has in combination with his working experience a better technical knowledge of the complexity of the process than a particular calculation tool. Hence, the advanced applications are just advisory tools and are not used as automatic control devices.

**Outage Management System**

The outage management system (OMS) is an application module that supports the human operator at all stages of unplanned and planned outages:

– Locating, isolating and restoring of unplanned outages
– Planning and managing of scheduled outages
– Assigning and dispatching of repair crews
– Outage reporting and recording of outage statistics

If the OMS is designed such that it can manage several trouble calls of customers at the same time, than the module is often denominated as trouble call management (TCM) system. A DMS that contains such a full TCM system is according to [NoWi07] implying the idea of a consumer-oriented DMS.
3.1.3 Integration of Base System Functions and Applications

Besides the carried out state-of-the-art survey of power grid control systems, technical literature about SCADA functionalities, distributed process control systems and especially DMSs and EMSs has been studied to gain background knowledge about the technical development of these systems.

With the improvements in radio technology enabling two-way communication, remote monitoring and supervisory control of widespread technical processes was started to develop at the beginning of the 1960s. Whereas older telemetry systems only allowed to monitor remote facilities, remote-two way operation was now possible for many energy industry companies like electric power utilities, gas or oil pipeline transportation companies. First, the expression “two-way telemetry” was used for this kind of control systems to emphasize the remote supervisory control part, which clearly distinguishes them from traditional telemetry systems. Later about the beginning of the 1970s, the term “SCADA” was created and started to supersede the expression “two-way telemetry” more and more as written in [Boye04].

Together with the development of computer technology at about the same time, electric power system operators started to replace the hard-wired mimic boards of the traditional control rooms by computers performing signal processing and displaying the state of the process on screens or on software-based mimic boards as described in [Cegr86]. Since the first generation of these computer based control systems was exclusively performing SCADA tasks, its systems were often just referred to as SCADA systems.

The further progress in computer technology concerning both hardware and software, which made it possible to perform large numerical calculations on affordable computer systems, caused according to [Cegr86] the advent of a new class of control systems supporting the responsible operators with application functions. At an early stage, these systems have been used only as advisory support tools, often without any possibility to perform calculations with real-time values coming from the SCADA system.

To benefit really from the advantages of numerical computing functions for electric power systems, it was either necessary to provide exchanging functions from the SCADA system to the computers running the applications and vice versa or to integrate SCADA functions and applications in one system. Because of the separate historical development of SCADA systems and application software, it was often
simpler just to establish exchanging functions and to detach SCADA functions and applications in two different systems. But even this solution is technical easier feasible than the integration into one total system, it is not the best one, because it is important to closely integrate SCADA and application functions into one real-time control system, as for instance stated in [Cegr86]. Hence, a considerable effort has been made to develop supervisory computer based control systems that have base system functions and applications closely integrated.

Most of the evaluated state-of-the-art DMSs and the also surveyed modern EMSs have according to their product descriptions and brochures base system functions and applications integrated, so the exchange between the data model for the SCADA functions and the several data models for the various application-oriented algorithms is done within the system. According to [NoWi07], a today available DMS having the full functionality of base system functions and applications has the disadvantage that overlaps between advanced application data models occur. Thus, such a DMS requires a data architecture achieving almost seamless integration. However, one problem remains, namely if there is a configuration change, every single advanced application data model has to be updated accordingly. Considerations about possible improvements of the mentioned problem will be done in the following sections.

### 3.2 Innovation Potential of State-of-the-Art DMSs

The carried out state-of-the-art survey revealed a couple of weak points, which most of the existing and nowadays on the market available DMSs have in common. In the following, these identified weak points are described in detail and are presented by means of a characteristic example if possible. The aim is not to denigrate existing DMSs or their beneficial abilities, but to point out that still some considerable technical innovation potential exists and so to motivate the development of future DMSs.

#### 3.2.1 DMSs with Badly Configurable Data Architectures

Modern DMSs are almost exclusively custom made solutions. Although most of these DMSs have a component-based architecture or even have a data model that is oriented by an industrial standard, they require a high degree of project imple-
mentation work, which could strongly affect the quality of the finished systems. At present, the following tasks are part of the implementation work:

- Linking data objects and graphical elements
- Assigning data to reports
- Connecting process variables of substations with data objects of the control system

This type of work is often error-prone and has to be redone after every small configuration change. Thus, configuration changes are cumbersome and time-consuming. The following illustrating example demonstrates how the workflow of a typical configuration change can look like for a DMS with a badly configurable data architecture.

**Example for a DMS with a Badly Configurable Data Architecture**

In this illustrating example, there is a Substation Omega, which has received a new outgoing feeder to supply the newly built industrial plant named Zeta. Hence, the DMS has to be updated accordingly, what is normally done by the system engineer with the help of the configuration menu.

The user interface of this exemplary DMS configuration menu is pictured in Figure 3.1. Over this user interface, the system engineer can accomplish the required configuration changes. First, he uses the insert function and can then choose the type of the new data object he likes to insert. After choosing “Outgoing Feeder” as new object type, he enters the name “Industrial plant Zeta” for the feeder as can be seen in Figure 3.1. With clicking on the OK-button, the new data object is inserted in the data model of the DMS.

By opening the data object manager, the system engineer can check that the new feeder has been correctly inserted. This is actually the case as can be seen in Figure 3.2 picturing the present user interface of the configuration menu with the opened data manager on the left side. But although the DMS now knows that there is a new outgoing feeder in the Substation Omega, the graphical representation is not automatically updated: Though the graphical object of the new feeder appears highlighted with orange and red colors, it is not connected to the graphical elements representing the busbars of the Substation Omega. Hence, this linking of the
3.2 Innovation Potential of State-of-the-Art DMSs

Figure 3.1: With the help of the configuration menu, the new outgoing feeder is inserted by choosing the object type and entering the name of the data object.

Graphical elements has to be manually done by the system engineer. Exactly this kind of linking of different graphical elements – or in other cases of graphical elements with their data objects – is what makes configuration changes often an error-prone and time-consuming work.

It has to be mentioned that the here presented example just shows one possible workflow of implementation work. Nevertheless, the laborious manner of the presented workflow is characteristic for DMSs with badly configurable data architectures. The technical reason for these shortcomings is namely the in Subsection 3.1.3 described property that within a DMS of today very often several advanced data models exist, which all have to be updated one by one. Consequently, the main reason for the error-prone and time-consuming implementation work seem to be identified: Because of the several data models, several data bases exist that store information like process variables, data objects and attributes.

Hence, one possible improvement of state-of-the-art DMSs is to engineer the data architecture such that the mentioned shortcomings cannot appear. Since in their product descriptions many of the modern DMSs including more than one advanced
Figure 3.2: Although the new outgoing feeder is correctly displayed in the data object manager, its graphical element is not linked with those of the busbars.

data model are denominated as *seamlessly integrated, totally integrated, continuous or consistent* grid control systems, even so they have the mentioned shortcomings, the word *transparency* is chosen to distinguish a DMS with a data architecture that does not have the listed weak points. To clarify exactly in which context the word *transparency* will be used and what is meant with it, a short definition has to be given:

**Definition of Transparency**

The word *transparency* is used to emphasize that a data architecture should be achieved with not only *horizontal* seamless integration of several advanced data models but with more a *vertical* integration of all data objects of the control system into one *unique* data base on which the SCADA data model as well as all advanced data models depend.

Consequently, those DMSs fulfilling all criteria for this kind of *transparent* data architecture are denoted as *fully transparent DMSs*, whereas the criteria for *transparency* have yet to be derived. Similarly, DMSs having not a complete *transparent* data architecture are denoted as *not fully transparent DMSs*. 
3.2.2 DMSs with Ambiguous or Mistakable Visualizations

In search of reasons for operating errors eventually resulting in a partial or even total system collapse, there are often technical deficiencies in supervisory grid control systems found. As already mentioned in Section 1.1, the analysis of the blackout series of the years 2003 to 2006 confirmed that some of the EMSs in charge were not satisfactorily ergonomically designed. The manner how these systems presented information concerning the actual system state, process variables or messages to the responsible human operators was bad or mistakable.

Since EMSs evolved from rather traditional supervisory grid control systems before DMSs underwent the equivalent development by implementing the same or similar functionalities as already used in EMSs, the visualization of process states in many modern DMSs has the same insufficiencies as in the mentioned EMSs. Of course, grave operating errors on the distribution level have not the same impact concerning the number of not supplied loads as those on the transmission level. But all the same, DMSs have to take the cognitive abilities in reception and processing of sensory stimuli of human beings into account. Particularly in critical operation states an unmistakable and absolutely clear visualization is of utmost importance, since this is the decisive precondition for the human operator to be able to react quickly and target-oriented.

Example for a DMS with an Ambiguous and Mistakable Visualization

This illustrative example demonstrates how the actual system state of an overregional distribution grid is not optimally visualized on the overview screen, which is pictured in Figure 3.3.

Given the situation where the line from Substation Theta to Substation Iota is grounded due to maintenance work and the line from Substation Alpha to Substation Theta is one-side switched off because of an unplanned outage in Substation Alpha, Substation Theta is only fed by Substation Kappa. But as can be seen in Figure 3.3, the critical fact that Substation Theta is insecurely supplied (see for instance [Tiet06a] for the definition of the word), is not specially high-lighted: On the one hand, the yellow-green dashed visualization of the grounded line from Substation Alpha to Substation Theta contrasts slightly insufficiently with the green color of the lines in the normal operation state. On the other hand, the one-side switched off line is only marked with a red symbol, whereas the color of the line is
Figure 3.3: Due to the ambiguous and mistakable visualization, the fact that Substation Theta is fed by only one line can not be seen at one look unchanged green because it is still energized from Substation Theta. But worse is that due to the two badly visualized lines the visualization as a whole is mistakable. The fact that Substation Theta is fed by only one line cannot be seen just at one look. The human operator has at least to scan the screen from Substation Alpha over Substation Theta to Substation Iota to realize the insecure power supply of Substation Theta. In an already stressed operating situation, the human operator may not have the time for this sort of scanning, or his look may be catched by other information displayed on the screen.

The described example does not claim to appear exactly in this way in a real supervisory control system, but it shows that a bad visualization of single elements or single events can result in combination with other incidents, even if they are independent, in a mistakable and ambiguous overall visualization, which simply can misguide the responsible human operator. Since an unmistakable and absolutely clear visualization is of utmost importance for the human operator to be able to react quickly and target-oriented, another possible improvement is to develop improved DMSs which visualize information more user-friendly and appropriately to the actual grid situation.
3.2.3 Grid Analysis Functions Have to Be Explicitly Retrieved

As described in Subsection 3.1.2, a wide range of comprehensive calculations and grid analysis functions is provided in state-of-the-art grid control systems. However, many of these grid analysis functions are only activated when the human operator explicitly requests their support. It can thus be assumed that these functions are not activated during disturbances or stressful situations, because the human operators have then to make quick decisions and do not take the time to start these calculations tools. Consequently, the benefits of these grid analysis functions are often not available at the very moment they would be needed the most.

3.3 Conclusions of the State-of-the-Art Survey

The innovation potential detected in the previous section and discussed with two illustrative examples allows finally to derive development objectives for future DMSs remedying the weak points of state-of-the-art systems:

1. Fully transparent data architecture
2. User-friendly and situation appropriate visualization
3. A capable of learning DMS giving the best possible support to the operator

To achieve all listed main objectives, the development of new concepts and algorithms for such an improved DMS has not only to focus on purely technical aspects but also to consider ergonomic design questions.

Since the second and the third main objective depend directly or indirectly on the chosen data architecture of the DMS, the goal of achieving a fully transparent data architecture has to be first addressed. Thus, the advantageous achievements of this fully transparent system architecture can be provided to the algorithms and methods that have to be developed for fulfilling the remaining two objectives. Furthermore, it is very likely that a fully transparent data architecture opens new possibilities to attain the other goals in a more efficient way.

Consequently, the aim to develop a new concept for a DMS with a fully transparent system architecture is considered to be of most importance and for this reason selected to be the main objective of this thesis. Of course, the other objectives,
namely the better situation appropriate visualization and the providing of expert knowledge, are essential as well. But since the concept for a fully transparent DMS and the algorithms needed in addition for this purpose are already comprehensive enough, these other aims are out of scope of this work. However, the first already elaborated ideas how a better visualization can be achieved or how the expert knowledge has to be provided to the operators are presented in Subsection 8.2.

In the following, the concept for such a fully transparent data architecture will be developed. As defined in Subsection 3.2.1, a transparent data architecture should be achieved by integrating all data objects of the control system into one unique data base on which the SCADA data model and the advanced data models depend. Hence, when developing the concept for the structure of this unique data base, it already has to be considered how the advanced applications or rather their advanced data models can acquire with the least possible effort their needed data from this developed structure.
4 A New Conceptual Model for a Fully Transparent Distribution Management System (DMS)

4.1 Fully Transparent Data Architecture

As derived in Section 3.3, the principal aim is to develop a DMS with a fully transparent data architecture. Such a fully transparent system architecture has to facilitate that necessary adaptions to a specific customer installation can be done with minimum effort and by assurance of maximum consistency. Thus, errors done during the project implementation work, commissioning or later during configuration updates can be avoided as far as possible.

Basis of a transparent system architecture has to be a standardized and object-oriented design. Moreover, there has to be one unique data file or one unique data object list in which all data objects of the control system can be stored. As pictured in Figure 4.1, not only a data model belongs to a data object, but also

- visualization elements,
- operating elements,
- alarm elements,
- log and archiving elements as well as a
- grid model.

The grid model is very important because it establishes the basis of all kind of application-oriented algorithms. Based on the grid model, advanced applications like power flow calculations, short-circuit calculations or outage management functions can be executed.
4.1.1 Criteria for a Transparent Data Architecture

In order to assess the transparency concept elaborated in the following, criteria for transparency are needed, as mentioned in Subsection 3.2.1. Careful considerations about how transparency can be judged yield the following six criteria:

1. **Unique data file**
   Basic requirement of every transparent data architecture is the existence of a unique data file or a unique database. This unique data file or unique data base has to contain all data objects of the control system with all their attributes. There should be no other data source having stored information of data objects. Hence, the unique data file has to contain topological grid data, data type information, graphical and geographical coordinates as well as physical values of grid components and grid elements (see Section A.1). Besides, it should store event lists, alarm lists and log book information.

2. **Consistency**
   One of the most important transparency criteria is data consistency: Whether a new data object is built or an existing one is deleted or updated, the con-
sistency of the grid control system should always be guaranteed. Updates of
data objects or their attributes done in one program should update all other
views or programs automatically.

3. **Expandability**
Sometimes, for instance when new loads have to be supplied, the topology of
a power grid has to be changed or enlarged by introducing new substations
or lines. Therefore, it is important that the data base or the data file can be
easily expanded at any time so that the grid control system can be updated
accordingly.

4. **Filterability**
A consequence of the required unique data file or unique data base storing
all data objects of the grid control system is the existence of adequate data
access possibilities so that every program can read and select just the data it
needs. This requirement can only be efficiently fulfilled if the unique data file
or the unique data base is filterable.

5. **Flexibility**
An important transparency criterion is flexibility in the sense that the grid
control system can be easily adapted to a specific customer installation.
Therefore, a comprehensive library with standardized functions will be needed.
Most of the base system and application functions should then be independent
of a specific customer installation. The adaption to such a specific customer
installation, even if it is an unusual one, should be easily feasible.

6. **Interoperability**
According to [IEEE90], interoperability is the ability of two or more sys-
tems or components to exchange information and to use the information
that has been exchanged. Since information between data objects is often
exchanged in a grid control system, one transparency criterion is interopera-
tility in the above defined sense. To meet this interoperability criterion,
information should be exchanged directly between the different data objects
of the grid control system. That is to say, intra-system information exchange
should be done without complicated message functions.

The six transparency criteria are intentionally formulated in a general way. The
reason is that a fully transparent data architecture should not only be an aim for
DMSs or EMSs but also for other supervisory control systems like for instance
process control systems of water or gas supply. All derived transparency criteria
can be taken over for these supervisory control systems as well. Only a few, in most cases obvious, adjustments in the formulations have to be done. If beyond that the transparency concept to be elaborated can also be used for these other application areas, this would prove the universality not only of the transparency criteria but also of the transparency concept itself. Thus, the verified universality would be one beneficial feature of the transparency concept and hereby augment its strength.

4.1.2 Levels of Transparency

Further considerations show that besides the transparency criteria also different levels of transparency can be distinguished:

First level of transparency: The first level of transparency is defined as the extent of data consistency that has to be achieved at any time by a DMS fulfilling all six transparency criteria.

Second level of transparency: The second level of transparency is defined as the extent of transparency that not only includes data consistency but data correctness as well. Like explicated in [Maie04] relating to data base theory, data correctness means that data correspond to the portion of physical reality they represent. The relations, correlations and dependencies which these physical objects have in reality, have to be reflected by the data objects in the data architecture without contradictions or discrepancies.

Third level of transparency: The third level of transparency is defined as the extent of transparency that besides fulfilling the second level of transparency includes data consistency and data correctness of derived data models. Derived data models are those data models which are not directly mapped on the unique data file or on the unique data base, as for instance advanced data models. Hence, they have to be derived by using the unique data file respectively the unique data base as data source. To guarantee data consistency and data correctness for these derived data models at all times, it is necessary that their so called derivation algorithms are efficient and are running automatically after being started.
4.2 The Transparency Matrix

Hence, the concept for a transparent data architecture should at least fulfill the first level of transparency. Since data consistency and data correctness of derived data models are a basic prerequisite for application functions, a fully transparent DMS providing more than base system functions should fulfill the second and the third level of transparency as well. Whereas the first and the second level of transparency has to be either judged as fulfilled or not, it is more difficult to judge if a specific DMS attains the third level of transparency. The reason is that if a derivation algorithm is “efficient” cannot be categorically judged as false or true. Nevertheless, it will be possible to evaluate derivation algorithms once they are developed. An accurate and correct evaluation of such derivation algorithms will then make a proposition how well the third level of transparency can be achieved.

4.2 The Transparency Matrix

The importance of the grid model gave rise to the idea of enhancing the data object list to a so called transparency matrix as pictured in Figure 4.2. All data objects of the control system are stored in this transparency matrix by placing...
one data object in each column. In every element of the column, one attribute of the data object is stored. For example, a specific attribute concerning the grid model is stored in row A or a specific graphical attribute is stored in row B as schematically shown in Figure 4.2. To unambiguously address all data objects and their attributes, the first row comprises the identifiers of the data objects and the first column comprises the names of the attributes.

4.2.1 Structure of the Transparency Matrix

To test, evaluate and subsequently improve the introduced transparency matrix, realistic data objects relating to a regional distribution grid have to be provided to it. For this reason, two exemplary regional distribution grids have been determined, which orientate concerning structure and equipment by a real regional distribution grid of a small Swiss town. Thereby, the smaller of these test distribution grids, the so called Test Distribution Grid 1, is used to evaluate the general feasibility of the transparency matrix. The bigger test distribution grid, denominated as Test Distribution Grid 2, is applied for more comprehensive tests to prove the effectiveness of the transparency matrix. In terms of numbers of substations and lines, Test Distribution Grid 2 covers about 20% of the real regional distribution grid. These at 16 kV operating test distribution grids are both described and shown in Appendix B. Test Distribution Grid 1 is pictured in Figure B.1 and the two grid sections Part A and Part B of Test Distribution Grid 2 are represented in Figure B.4 and Figure B.5 respectively. In addition, the transparency matrix of the entire Test Distribution Grid 1 is shown in Figure B.3.

As can be seen in Figure B.1, Test Distribution Grid 1 has one MV source substation, namely Station A, which contains two infeeds. Besides, Test Distribution Grid 1 comprises six local distribution substations, which all hold a distribution transformer connected with loads at the local distribution level. The interlinking of all substations is done by seven lines, whereas all of them are realized as MV cables. Although Test Distribution Grid 1 is only a small one, there are already more than 50 data objects referring to it in the responsible control system. Hence, for describing the structure of the transparency matrix in more detail, only a part of Test Distribution Grid 1 should be focussed on. Since Station A as the MV source substation with two infeeds, two busbars, one busbar coupler, eight circuit breakers and six earth switches has the most sophisticated configuration of all substations of Test Distribution Grid 1, it is possible to study the structure of the transparency matrix only by means of Station A without loss of generality. So
in the following, the part of the transparency matrix storing all data objects of Station A will be accurately described and explained, whereas the earth switches will not be considered in order not to obscure the presentation.

On the left side of Figure 4.3 the transparency matrix of Station A is mapped, showing three typical attribute rows, and on the right side Station A itself is schematically pictured. There is a graphical data row, in which the graphical coordinates of a specific view can be stored. A specific attribute concerning the data model is the data type of the data object. This attribute is stored in the type data row. The graphical data attribute and the type data attribute are static attributes, since their values normally do not change during the operation of the power grid. Besides these static attributes, there exist more dynamic attributes like for instance the state data attribute, which is gathered by the communication equipment of the basic SCADA system and then stored in the state data row.

To make similarities between different data objects well apparent, they are sorted by type and number in the transparency matrix as can be seen in Figure 4.3. Although this sorting is not absolutely necessary, it makes the transparency matrix even more easy to understand: It can be read like a map. To clearly illustrate how the data objects referring to Station A are sorted in the columns of the transparency matrix, the data object of the Infeed A2, the data object of the Circuit Breaker 2 and the data object of the Line GA are indicated: The data columns in which these specific data objects are stored are highlighted with orange color. In the scheme of Station A the corresponding physical device is encircled with red color.

The most important part of the transparency matrix is the block matrix in the upper part highlighted with a thick dark red frame: This block matrix is a special incidence matrix which has to represent the grid model or rather the physical grid topology such that it can later be used as basis for topological calculations or power flow calculations. In its rows, the special incidence matrix contains the nodes of the distribution grid. Therefore, busbars, which are the real nodes of the power grid, are presented twice in the transparency matrix:

- Once as a data object with all its attributes stored in a transparency matrix column and
- once as a grid node in a row of the incidence matrix.

This could be verified by inspection of the transparency matrix shown in Figure 4.4. Besides the rows with the grid nodes, this special incidence matrix contains rows
Figure 4.3: The transparency matrix on the left side stores all data objects referring to Station A on the right side.
with so called partition nodes, which constitute the junctions between adjacent switches and lines. Their motivation and derivation is explained in Appendix A.2.

For the concrete example of the Circuit Breaker 5 and its adjacent nodes, Figure 4.4 illustrates how their data objects appear in the special incidence matrix. As pictured in the scheme of Station A, the green highlighted Circuit Breaker 5 is connected with a real grid node, namely the Busbar 2, which is highlighted in green too. On its other side, the Circuit Breaker 5 is connected with the partition node of the Infeed A2, which is depicted as a small green circle in the scheme of the substation. The row with this partition node, the row with the data object of the Busbar 2 and the column with the data object of the Circuit Breaker 5 are highlighted with orange color in the incidence matrix. The cell elements where the node rows and the switch column are crossing each other contain integers differing from 0 as can be seen in Figure 4.4. Exactly these so called connectivity integers indicate that two devices of the power grid are adjacent to each other.

Since the transparency matrix by definition contains all data objects of the concerning distribution grid, the grid topology represented by this special incidence matrix is thus the one with the most detailed degree of description. In [NoWi07], this degree of grid description is denominated as the highest device resolution. Because of its matrix form, the grid topology becomes “visible” by just scanning the special incidence matrix. This can also be verified by inspection of the transparency matrix and the scheme of Station A in Figure 4.3 or in Figure 4.4. Furthermore, the presented incidence matrix is characterized mainly by three special properties:

1. Incidence orientation:
   The incidence matrix is a partly unoriented and partly oriented incidence matrix: There are directed branches as well as undirected branches. A directed branch is an electric connection whose physical model is given by the power line equation. To visualize the solutions of the power line equations, each electrically long connection needs to have a counting direction, what will be needed for the power grid models derived in Chapter 5. Hence, transmission lines and cables are represented as directed branches in the incidence matrix. Otherwise, an undirected branch is an electrically short connection between two nodes. Consequently, switchable and coupling devices are represented as undirected branches in the incidence matrix.

2. Connectivity integers:
   The incidence matrix uses not only one integer and its additive inverse to express connectivity between different data objects. As shown in the trans-
Figure 4.4: The special incidence matrix contains in its rows the grid nodes and the partition nodes of Station A.
4.2 The Transparency Matrix

The transparency matrix depicted in Figure 4.3 or in Figure 4.4, the integer 2 and its additive inverse -2 are used besides the integers 1 and -1. The reason for that will be explained in Subsection 4.2.2.

3. Half branches:

Since the transparency matrix has to store also data objects which do not have two-terminal characteristic, half edges or better half branches\(^1\) are needed. Such a half branch is connected to only one node why in its column only one integer expressing connectivity appears. Data objects like busbars, infeeds or also virtual links to meta data objects are stored as half undirected branches in the transparency matrix. Such a meta data object can be a substation data object storing general attributes and information about the concerning substation, as it is the case in the transparency matrix mapped in Figure 4.4.

4.2.2 Managing Topologies with the Transparency Matrix

Each power grid has a limited but large number of possible connections. The collectivity of all possible connections is referred to as possible topology. In a given operation state, only a subset of the possible connections are in use since the usage of all possible connections would not be reasonable nor realistic. The topology resulting of the actually used connections at a particular time is the so called actual topology. In distribution grids, one goal of power system management and control is to maintain a certain grid topology over longer periods of time. Only during disturbances or maintenance work another topology is in use, and directly after their correction or their completion the switchable connections are switched back to the specific grid topology, the so called regular topology.

Since as stated in [RuSu89], the DMS has to manage all these three topologies, the transparency matrix has to express which node and which branch are incident but also if their incidence is actual, regular or just possible. The possible and regular topology can be stored in the incidence matrix by using integers 0, 1 and 2 for undirected branches and integers -2, -1, 0, 1 and 2 for directed branches.

Undirected branches:
The value of an unoriented incidence matrix element \([i, j]\) is defined as follows:

\[
\begin{align*}
0: \text{ node } i \text{ and branch } j \text{ are never incident.}
\end{align*}
\]

\(^1\)Half edges are used in graph theory when it is necessary to have edges with only one end. The definition and concept of half-edges is for instance described in [Sili06].
1: node $i$ and branch $j$ are incident in the *possible topology*:

Branch $j$ has a possible connection with node $i$.

2: node $i$ and branch $j$ are incident in the *possible* and *regular topology*:

Branch $j$ has a possible and in regular operation situations used connection with node $i$.

**Directed branches:**

The value of an oriented incidence matrix element $[i, j]$ is defined as follows:

0: node $i$ and branch $j$ are **never** incident.

-1: node $i$ and branch $j$ are incident in the *possible topology*:

Branch $j$ has a possible connection entering node $i$.

1: node $i$ and branch $j$ are incident in the *possible topology*:

Branch $j$ has a possible connection leaving node $i$.

-2: node $i$ and branch $j$ are incident in the *possible* and *regular topology*:

Branch $j$ has a possible and in regular operation situations used connection entering node $i$.

2: node $i$ and branch $j$ are incident in the *possible* and *regular topology*:

Branch $j$ has a possible and in regular operation situations used connection leaving node $i$.

**Actual Topology Determination Algorithm**

For the determination of the actual topology, the DMS has to consider all switch states. Since the switch states are stored and updated in the state data row of the transparency matrix, the actual topology determination algorithm needs to check this state data row. The procedure of the actual topology determination algorithm can be comprehensibly explained by means of a concrete example, namely with the Circuit Breaker 2, which connects the Infeed A1 with the Busbar 1, and its pair switch the Circuit Breaker 3, which connects the same infeed with the Busbar 2. The part of Station A containing these two switches is sketched in Figure 4.5 and in Figure 4.6. Whereas the Circuit Breaker 2 has the connectivity integer 2, its
4.2 The Transparency Matrix

Figure 4.5: The actual topology is the same as the regular topology due to the switch states.

Figure 4.6: The actual topology differs from the regular topology due to the actual switch states.

pair switch the Circuit Breaker 3 has the connectivity integer $1$. This means that in all regular operation situations the Circuit Breaker 2 is closed and the Circuit Breaker 3 is open.

In the first case, which is pictured in Figure 4.5, the switch state of the Circuit
Breaker 2 is Closed and the switch state of the Circuit Breaker 3 is Open. So, both switches are in their regular state. Concerning these two switches, the actual topology is identical with the regular one. In the second case, which is pictured in Figure 4.6, the switch state of the Circuit Breaker 2 is Open and the switch state of the Circuit Breaker 3 is Closed. Hence, both switches are not in their regular state. As a consequence, the actual topology of the distribution grid differs from its regular topology.

After the completion of the actual topology determination algorithm, the gained knowledge concerning which switchable devices are in their regular state and which are not, can be made available to the responsible human operator. The case where the actual topology of the distribution grid is identical with its regular topology has not to be especially presented. But the contrary case, where the actual topology differs from the regular one, should be presented to the human operator, for example by using an appropriate visualization emphasizing switchable devices that are not in their regular states.

4.2.3 Practical Aspects of the Transparency Matrix

Before the benefits of the transparency matrix are discussed, some of its practical aspects like its possible combination with standard data models and its implementation into a real software solution are discussed.

Transparency Matrix and Standard Data Models

The transparency matrix is a conceptual model for the internal data architecture of a fully transparent DMS. However, for external interaction, communication or data exchange with IT-systems of other utilities like a neighboring DMS or a superior EMS, it is essential that the transparency matrix can be combined with industrial standard data models. This situation is schematically pictured in Figure 4.7.

Such an industrial standard data model representing all main components of an electric power system is the Common Information Model (CIM) standardized within IEC 61970. An introduction to the CIM standard as it applies mainly to distribution systems and their operation is for instance given in [King08].

Since the CIM standard and the transparency matrix both have an object-oriented data structure, the data objects stored in the columns of the transparency matrix
4.2 The Transparency Matrix

Figure 4.7: External interaction, communication or data exchange of a fully transparent DMS with other IT-systems.

can be standardized CIM objects. The grid topology modeled in CIM can be transformed into the incidence matrix of the transparency matrix. Each attribute described in the CIM standard can be stored in a row of the transparency matrix.

In principle, each industrial standard data model that has an object-oriented data structure relating to electric power systems can be stored in the transparency matrix: The data objects stored in the columns of the transparency matrix are complying with the concerning standard, and the rows of the transparency matrix contain the associated attributes as schematically pictured in Figure 4.8. Attributes not belonging to the standard but crucial for fulfilling the transparency concept have to be stored in additional rows like for instance the graphical data rows.

Implementation of the Transparency Matrix

Although the transparency matrix in Figure 4.3 or in Figure 4.4 just stores all data objects of Station A, it is already a $11 \times 19$-matrix, even without the earth switches. Consequently, for a real distribution grid with many local substations and several MV substations, the transparency matrix gets large. That is why the
The presented transparency matrix is more a conceptual model of a fully transparent data architecture than a real software solution to store object data. For the final software implementation, the transparency matrix can be mapped on a data base. Still, the transparency matrix is available: It can be generated out of the data base into the random access memory (RAM) of the DMS, where it can be used for transparency checks after configuration changes or as basis for all application-oriented algorithms to name but a few.

How such an implementation can be realized concretely will be explained and presented in Chapter 7. This realization will prove that the here presented concept of the transparency matrix is not just a theoretical idea but that it can be turned into a real grid control system.
4.2.4 The Properties of the Transparency Matrix

The transparency matrix fulfills all initially derived transparency criteria stated in Subsection 4.1.1:

1. **Unique data file**
   The transparency matrix can be stored in a unique data file or be mapped on a unique data base: The transparency matrix contains all data objects with all their attributes. First of all, it comprises topological grid data, data type information, graphical and geographical coordinates as well as physical values of grid components and grid elements. In addition, it can store event lists, alarm lists or log book information. Hence, there is no need for another data file or data base having stored data object information.

2. **Consistency**
   The transparency matrix forces data consistency: Each program or view uploads its needed information from the transparency matrix. Changes generated in one program are saved to the transparency matrix. Hence, all other programs or views are updated automatically when uploading their required information from the transparency matrix.

3. **Expandability**
   The transparency matrix is expandable: A new data object can be added by inserting a new column. New attributes can be added to the control system by inserting a new row in the transparency matrix.

4. **Filterability**
   The transparency matrix is easily filterable: Columns of not needed data objects or rows of not needed attributes can just be ignored.

5. **Flexibility**
   The transparency matrix is flexible. It can handle standardized types and functions as well as customer-specific ones. If needed, another row containing a particular customer-specific attribute can be inserted in the transparency matrix.

6. **Interoperability**
   The transparency matrix easily enables interoperability. Information can be exchanged directly between the data objects.
The consequence of fulfilling all these six transparency criteria is that the transparency matrix meets the first level of transparency according to the transparency level definition in Subsection 4.1.2. Especially the fulfillment of the criteria consistency and filterability is extremely important since configuration changes can be done with minimum effort and by assurance of maximum consistency. Mainly because of the consistency and the filterability of the transparency matrix, it is guaranteed that the entire data model corresponds completely with the physical grid model stored in form of the incidence matrix. Similarly, all graphical representations like graphical grid diagrams or visualizations of different grid topologies correspond entirely to the data model represented by the transparency matrix.

For instance, if the DMS has to store a new data object due to the installation of a new cable or a new distribution substation, a new column is inserted into the transparency matrix. Since in every element of this column, one attribute of the new data object has to be stored, the configuration procedure is correctly terminated, when each element of the new data column is filled with its required attribute. Hence, the risk that an attribute value is not set or is missing can be excluded. In the contrary case, when a data object of the DMS has to be removed, this can easily be done by just deleting the corresponding column of the transparency matrix. By doing so, the graphical coordinates of the data object gets deleted and its graphical symbol disappears from each view. Thus, it is not possible to have any inconsistency between the data model and graphical presentations like overview presentations or detail views.

Besides fulfilling all transparency criteria, the transparency matrix has further beneficial features:

- The transparency matrix is easy to understand and can be read like a map.

- The transparency matrix, or rather the special form of the incidence matrix, allows the control system to manage the different topologies in an effective way.

- Similarities between different data objects are well apparent in the transparency matrix, namely in the row containing the concerning attribute.

In summary, it can be said that the transparency matrix is a promising concept. Primarily the manner how the physical grid topology is connected with the data objects seems to be the ideal condition for deriving advanced data models efficiently for application functions.
4.2.5 The Transparency Level of the Transparency Matrix

As already mentioned in Subsection 4.2.4, the transparency matrix fulfills all six transparency criteria and thus meets the first level of transparency. As a result, the transparency matrix satisfies all in Subsection 4.1.2 stated minimum requirements for a fully transparent DMS.

If in addition, it can be demonstrated that the transparency matrix also fulfills the second level of transparency or if extra functions can be developed which forces data correctness and hereby get the transparency matrix to meet the second level of transparency, then the concept meets the most important requirements. Only the last requirement of fulfilling the third level of transparency has to be examined separately, since at first at least one derivation algorithm has to be developed. The development of such a derivation algorithm will be done in Chapter 5. So in the following, it is investigated if the transparency matrix can fulfill the second level of transparency alone or if additional functions are needed to achieve data correctness.

4.2.6 Verification of the Transparency Matrix’ Data Correctness

As defined in Subsection 4.1.2, data correctness means that data correspond to the portion of physical reality they represent: Data objects in the data architecture have to reflect the relations, correlations and dependencies of the real physical objects they represent so that contradictions or discrepancies cannot occur. Consequently, for an accurate verification of data correctness, the physical relations and dependencies between the different objects have to be exactly known. If it should turn out that extra functions have to be developed to achieve data correctness, it is necessary that those physical relations can be modeled analytically.

Data correctness is certainly important for all program and views of the DMS, but since information is mainly in form of visualizations provided to the operator, it is probably most important for graphical representations. Moreover, since graphical representations in many state-of-the-art DMSs are manually drawn by the use of graphical editors with symbol libraries but without any functions for checking correctness, there is a risk that contradictions between graphical views and the data model may occur. Hence, it will be first verified if the transparency matrix achieves the second level of transparency concerning graphical representations. If this should not be the case, then it is already clear that extra functions have to be
developed so that the transparency matrix can always guarantee data correctness.

**The Graphical Grid Topology**

As described in Subsection 4.2.1, the graphical data row stores the graphical coordinates of the data objects: For each data object of the distribution grid, its graphical coordinates stored in the transparency matrix define exactly where its graphical object has to be placed on the GUI or on the presentation view. In a specific view, like for instance the actual topology view, all presented graphical objects like the symbols for substations, busbars, cables, load switches and circuit breakers produce together a visual representation of the considered distribution grid. The way how the graphical symbols are tangential, intersect or overlap each other is referred to as the *graphical grid topology*.

This graphical grid topology has to correspond with the physical grid topology stored in form of the incidence matrix in the upper part of the transparency matrix. Hence, the transparency matrix part containing the incidence matrix and the graphical data row of the transparency matrix are not absolutely independent from each other with respect to the information they store. This correspondence
between the physical grid topology and the graphical grid topology is schematically pictured in Figure 4.9. As a consequence of this dependence, it has to be determined, which data can be considered as independent and which data can be considered as dependent. Since the physical grid topology is represented by the incidence matrix, this is taken as the independent data source. It can be shown that some graphical coordinates stored in the graphical data row are independent as well. The remainder of the graphical coordinates are depending on the independent graphical coordinates and on the physical grid topology. Table 4.1 shows one possibility how graphical coordinates can be divided into dependent and independent ones: The coordinates of substations are taken as the independent graphical coordinates, and consequently, the coordinates of busbars, infeeds, circuit breakers, load switches and cables are the dependent coordinates. The solution presented in Table 4.1 is not the only one. There are other possibilities to divide graphical coordinates in dependent and independent ones, for instance the graphical coordinates of busbars can be taken as the independent coordinates. Practical considerations will determine which solution that will be chosen.

As another consequence of data dependence, contradictions and discrepancies can occur, for instance if one of the dependent graphical coordinates is incorrect. Thus, this kind of error affects the data correctness of the DMS and the second level of transparency is not fulfilled in such a case. Because of these detected dependencies between the incidence matrix and the graphical data row, it is shown that the transparency matrix alone can only guarantee to meet the first level of transparency at any time.

To permanently guarantee the second level of transparency as well, special addi-

<table>
<thead>
<tr>
<th>Object type</th>
<th>Graphical coordinates</th>
</tr>
</thead>
<tbody>
<tr>
<td>Substation</td>
<td>Independent graphical coordinates</td>
</tr>
<tr>
<td>Busbar</td>
<td>Dependent graphical coordinates</td>
</tr>
<tr>
<td>Infeed</td>
<td>Dependent graphical coordinates</td>
</tr>
<tr>
<td>Circuit Breaker</td>
<td>Dependent graphical coordinates</td>
</tr>
<tr>
<td>Load Switch</td>
<td>Dependent graphical coordinates</td>
</tr>
<tr>
<td>Cable</td>
<td>Dependent graphical coordinates</td>
</tr>
</tbody>
</table>

Table 4.1: Dependence of graphical coordinates
tional functions are needed. In the case of the graphical grid topology, some appropriate geometric functions are required to determine the values of the dependent graphical coordinates. In addition, a suitable procedure is needed which ensures that the geometric functions are always activated when dependent graphical coordinates have to be set. Section 4.3 will present a feasible realization of activating the necessary computing functions so that data correctness for the transparency matrix can be achieved.

4.3 The Transparency Configuration Manager

4.3.1 Motivation for the Transparency Configuration Manager

As discussed in Subsection 4.2.6, the transparency matrix alone cannot guarantee to meet the second level of transparency in all cases. Because of dependencies between different data objects and their attributes, some special functions are needed to ensure the correctness of the dependent data values. That is why the write access to the transparency matrix has to be strictly prescribed. Special functions and write access can be done best by a manager which

- invokes the relevant functions, so they can compute the dependent values correctly
- and prescribes how data is written into the transparency matrix.

Thus, the manager forces data correctness for the DMS at any time. By doing so, the manager together with the transparency matrix attains the second level of transparency. Consequently, it has been shown that the concept for a fully transparent DMS meets all demanded requirements with exception of the third level of transparency, which has to be still examined. Since this manager is primarily needed during configuration changes, it is named transparency configuration manager.

4.3.2 The Principal Idea of the Transparency Configuration Manager

First, the transparency configuration manager has to guarantee data correctness not only for configuration changes in the graphical view but for all configuration
changes, independently which data objects or which attributes are affected. Thus, the transparency configuration manager consists of several parts with different computation and checking functions. For example besides the graphical part, there exists a part with attribute checking functions, which is used for data import during updates or configuration changes.

The principal idea of the graphical part of the transparency configuration manager is the following: For each new data object with dependent graphical coordinates, the possible values for the graphical coordinates are computed by invoking the so called geometric computation functions. Then, the transparency configuration manager writes the computed values into the corresponding cells of the transparency matrix. The procedure of this graphical part of the transparency manager is schematically depicted in Figure 4.10.

These geometric computation functions logically depend much on the graphical library symbols of the specific DMS. Hence, it is not possible to describe them in a generally valid way. Besides, the aim of this subsection is mainly to present the principal idea of the transparency configuration manager and not to cover concrete implementation aspects. However, as long as the presentation views are
stored in a vector graphic format, it is always possible to develop appropriate geometric computation functions for all of them. As graphical library symbols are mostly composed of simple geometric forms like squares, lines or circles, the corresponding geometric computation functions are more or less straightforward. To be complete and comprehensible, a concrete example of such a computation function is explained in Appendix C.1.

Since the transparency manager depends not directly on the structure of the transparency matrix, it can also be implemented as an additional configuration manager tool in an existing, not fully transparent DMS. Moreover, the implemented transparency configuration manager can help to make an existing DMS fully transparent. Due to the fact that the transparency manager has computing functions for the determination of dependent data, it can use these functions to detect data inconsistencies as well. Some inconsistency errors can even be repaired by the transparency configuration manager: It uses its computing functions and if possible sets the inconsistent data value to a predefined standard value.

### 4.3.3 The Advantage of the Transparency Configuration Manager

The advantage of the transparency configuration manager is that it guarantees data correctness by providing computing functions for the determination of dependent data values. Furthermore, it prescribes strictly how data is stored into the transparency matrix thereby preventing unauthorized writing access. Consequently, contradictions and discrepancies between data objects which are related to each other cannot occur.

Since the transparency configuration manager ensures data correctness at any time, it makes the transparency matrix to fulfill the second level of transparency. Hence, the concept for a transparent data architecture fits all basic requirements for a fully transparent DMS.

### 4.4 Concluding Remarks

The transparency matrix presented in this chapter is a conceptual model for the data architecture of a fully transparent DMS. The importance of such a conceptual model – even if practical implementation aspects are so far not really covered –
must not be underestimated. It is well established that models play an important role in science, engineering technology and their education as well as the research. As stated in [GiBo00], the usefulness of such a model has to be determined in consideration of its intended purpose and use. Since the advantageous features of the transparency matrix have already been evaluated and discussed in detail in Subsection 4.2.4, the conceptual transparency model can so far be evaluated as useful. Still, the third level of transparency has to be proven and – even more important – it has to be shown that the final software implementation can be satisfyingly achieved.

As mentioned in Subsection 4.1.2, at least one so called derivation algorithm is needed to accurately investigate the third level of transparency. In Chapter 5, such a derivation algorithm will be developed for power system modeling using the presented transparency matrix as basis. The derived data model, which is represented by the so called condensed transparency matrix, provides the grid model in the desired form for application-oriented algorithms like power flow calculations.

Subsequently in Chapter 7, the developed conceptual model of the transparency matrix is taken as a design guideline for a future DMS. Based on an existing DMS, the real software implementation of the transparency matrix will be done. If the DMS thereby gets a fully transparent data architecture and if the implementation procedure as such can be done in a convincing and facile manner, then it can finally be stated that the here introduced transparency matrix is a powerful and very useful conceptual model for a fully transparent DMS.
5 Power Grid Models and Grid Topology Detection for a Fully Transparent DMS

5.1 Introduction

Many of the state-of-the-art DMSs analyzed in Section 3.1 provide a wide range of application functions depending on different advanced data models. Like described in Subsection 3.1.3, some of these DMSs have the problem that inconsistencies and contradictions between the advanced data models and the base data model for the SCADA functions could occur. Consequently, these DMSs cannot fulfill the third level of transparency according to the definition in Subsection 4.1.2, since for this level of transparency data consistency and data correctness of derived data models – to which advanced data models belong – has to be guaranteed at all times.

That the conceptual model of the transparency matrix together with the configuration manager fulfills the first and the second level of transparency has already been substantiated in Subsection 4.3.3. As mentioned there, only the third level of transparency has yet to be elaborated. Hence, the main aim of this chapter is to develop a so called derivation algorithm that derives from the transparency matrix an advanced data model in an efficient and reliable way. This will finally allow to judge how well the third level of transparency can be achieved using the transparency matrix as data base.

The aim of Chapter 6 is to develop one advanced application function (see Subsection 3.1.2, where typical advanced applications of modern DMSs are listed). As power flow calculation is one of the most important grid analysis functions provided in a modern DMS, a power flow calculation method for distribution grids will be developed using the derived advanced data model as basis for the grid model. The
derived data model has hence to have an appropriate form for power flow calculations so that the corresponding grid model contains only the actual connections of lines with busbars in operation.

Since most of the power flow calculation methods for distribution grids determine the unknown quantities by iteratively performing downstream and upstream cycles through the grid structure, possible existing rings have to be detected before the calculation is started. Hence, a ring detection algorithm has to be developed that uses the advanced data model or rather the corresponding grid model to identify grid branches that are within a ring in the actual grid topology. The such identified branches allow then that the power flow calculation method can be performed in an almost similar way as in normal cases where the grid topology has a pure radial tree structure.

5.2 The Condensing Algorithm

As introduced, using the transparency matrix as data base, this derivation algorithm has to generate a grid model in an appropriate form for power flow calculation methods: As many other advanced applications, power flow calculation needs only a simplified equivalent of the complex grid model that has the highest device resolution. Thus, since this grid model with the highest device resolution is represented in the special incidence matrix of the transparency matrix, the derivation algorithm has to reduce or rather to condense the latter into a so called condensed transparency matrix. This derived condensed transparency matrix represents the actual topology of the distribution grid in the desired concentrated form: Only lines, infeeds and busbars in operation are considered. Due to the condensed grid model it generates, this derivation algorithm is renamed condensing algorithm.

5.2.1 The Principal Idea of the Condensing Algorithm

The principal idea of the condensing algorithm is to establish for every closed switch a virtual connection between its adjacent grid branch, e.g. line or infeed, and its adjacent grid node, e.g. busbar. In the condensed transparency matrix or rather in the condensed incidence matrix, this virtual link is achieved by replacing a 0 integer with a connectivity integer expressing incidence between the relevant grid node and the relevant grid branch. Hence, the connection between the grid branch and the
5.2 The Condensing Algorithm

Figure 5.1: The condensing algorithm has to establish for every closed switch a virtual connection between its adjacent cable and its adjacent busbar.

grid node established by the closed switch becomes a redundant connection, which can be filtered out. This principal idea of the condensing algorithm is schematically pictured in Figure 5.1.

5.2.2 The Procedure of the Condensing Algorithm

The virtual connection, which has to be established for every closed switch, depends on the type and state of the objects that are adjacent to the concerning switch. Using only those substation configurations which appear in the test distribution grids presented in Appendix B, three different cases have to be distinguished:

1. **Closed single switch**
   This case is the standard one: A closed switch makes one short electric connection between its adjacent grid branch and its adjacent grid node. This is also the case for a closed switch belonging to a switch pair of a double busbar if its partner switch is open.

2. **Closed pair switches**
   This case can appear at substations with double busbar configurations: If both switches of the switch pair are closed, then two short electric connections exist: One between the adjacent grid branch and the first busbar and one between the adjacent grid branch and the second busbar. Hence, the two busbars are electrically connected through both closed switches. As a
consequence, both busbars have the same voltage. They can be condensed to one busbar in the condensed transparency matrix.

3. Closed busbar coupler or closed busbar sectionalizer

The closed busbar coupler or the closed busbar sectionalizer makes one short electric connection between its adjacent busbars. Consequently, the two busbars have the same voltage. They can thus be condensed to one busbar in the condensed transparency matrix.

It has to be mentioned that for substation configurations not appearing in the test distribution grids further cases may have to be distinguished. However, independently if these substation configurations are rather common or rather special ones, the distinction of different cases should be no problem since all substation designs are determined by operational considerations. Hence, the concluding validation of the condensing algorithm can be done even though it is only checked out on Test Distribution Grid 1 and on Test Distribution Grid 2.

The procedure of the condensing algorithm can be structured in four different subprocedures, which are explained in detail in the following.

1. Construction of the Derived Transparency Matrix

This first subprocedure starts with making a copy of the transparency matrix. This duplicate of the transparency matrix is called derived transparency matrix and will be modified during this first subprocedure. Afterwards, it will be used for the construction of the desired condensed transparency matrix.

The subprocedure of constructing or rather modifying this derived transparency matrix is slightly different for the three cases defined above. In Figure 5.2, the process chart for all three cases is schematically depicted using different colored arrows to show which steps of the subprocedure have to be executed in which case.

With the help of one concrete example, namely with the closed Circuit Breaker 6 of Station A of Test Distribution Grid 1 and its partner switch, the open Circuit Breaker 7, all steps of the subprocedure are illustrated. One or two steps are pictured per each figure mapping the transparency matrix part of Station A on the left side and sketching the part of interest of Station A on the right side.

For each data object column of the transparency matrix that is not yet marked as Checked in the corresponding column of the derived transparency matrix (see
5.2 The Condensing Algorithm

Case 1: Closed single switch
Case 2: Closed pair switches
Case 3: Closed busbar coupler or sectionalizer

Step 1: Checking the object type
Step 2: Checking the object state
Step 3: Examining the incidence matrix column
Step 4: Checking the node type
Step 5: Examining the incidence matrix row
Step 6: Checking the adjacent data objects
Step 7: Enhancing the connectivity triangle
Step 8: Checking the node type
Step 9: Marking switch columns as checked

Figure 5.2: The process chart illustrates which steps of the subprocedure have to be executed in which case.
Step (9), the subprocedure executes the following instructions:

Step (1): Check if the object type of the data object stored in the actual column $i$ of the transparency matrix is a switch:

- If yes, go to Step (2).
- If no, mark the corresponding column of the derived transparency matrix as Checked and go to the next data column.

Step (2): Check if the actual object state of the switch is closed:

- If yes, go to Step (3).
- If no, go to Step (9).

Step (1) and Step (2) are pictured together in Figure 5.3.

Step (3): Examine all incidence matrix elements of the actual column. Check if the integers stored in the incidence matrix elements are connectivity integers:

- If yes, store the number of the actual row and note the node type of the node stored in this row of the incidence matrix.
- Then go to Step (4).

Step (4): Check if one of the noted nodes is a partition node:

- If yes, go to Step (5).
- If no, go to Step (9).

Step (3) and Step (4) are illustrated in Figure 5.4.

Step (5): Examine all incidence matrix elements of the row in which the found partition node is stored. Check if the integers stored in the incidence matrix elements of this row are connectivity integers:

- If yes, store the number of the actual column and note the data type of the data object stored in this column of the transparency matrix.
- Then go to Step (6).

Step (6): Check if one of the found adjacent data objects is a cable or another switch:
5.2 The Condensing Algorithm

Figure 5.3: **Step ①** and **Step ②** of the subprocedure check if the object type is a switch and if the object state is closed.

Figure 5.4: **Step ③** and **Step ④** of the subprocedure check if the considered switch is connected to a partition node.
Figure 5.5: Step 5 and Step 6 check if one of the adjacent objects is a cable or a closed switch.

Figure 5.6: Step 7 takes the derived transparency matrix and prepares the virtual link by enhancing the triangle of connectivity integers to a square.
5.2 The Condensing Algorithm

- If there is a second closed switch, set a flag that Step 8 has to be executed.
- If there is a cable, go to Step 7.
- Otherwise go to Step 9.

Figure 5.5 visualizes Step 5 and Step 6 together.

**Step 7:** Consider now the derived transparency matrix. Go to its matrix element \([p, l]\) being located in the column \(l\) in which the cable is stored and in the row \(p\) in which the partition node is stored. Copy the connectivity integer stored in this matrix element to the matrix element \([m, l]\) laying in the same column \(l\) but in the row \(m\) in which the grid node is stored. Doing so, the triangle of the matrix elements \([m, i], [p, i]\) and \([p, l]\) expressing connectivity is enhanced with the matrix element \([m, l]\) to a square with connectivity integers.

- If the flag has been set in Step 6, go to Step 8.
- Otherwise go to Step 9.

The preparation of the virtual link is pictured in Figure 5.6 and the established link is illustrated in Figure 5.7.

**Step 8:** Check if the other closed switch found in Step 6 is connected to a grid node:

- If yes, take again the derived transparency matrix. Go to its matrix element \([p, l]\) being located in the column \(l\) in which the cable is stored and in the row \(p\) in which the partition node is stored. Copy the connectivity integer stored in this matrix element to the matrix element \([n, l]\) laying in the same column \(l\) but in the row \(n\) in which this second grid node is stored. Doing so, given that the second closed switch is stored in column \(j\), the triangle of the matrix elements \([n, j], [p, j]\) and \([p, l]\) expressing connectivity is enhanced with the matrix element \([n, l]\) to a square with connectivity integers. Then store the row number \(n\) of this second node.

- Then go to Step 9.

**Step 9:** Mark the derived transparency matrix column in which the actual switch is stored as Checked.
Figure 5.7: The virtual link in the derived transparency matrix established in Step 7 connects the cable directly with the busbar.

- If a second closed switch has been found in Step 6, mark the concerning column as Checked.
- If a second grid node has been found in Step 4 or in Step 8, store the row number of the first grid node and the row number of the second grid node together with the column number of the actual switch in a so called double busbar list provided for this purpose.
- Then go to the next column of the transparency matrix.

Figure 5.8 shows Step 9, which is the final step of the subprocedure.

After all data columns of the transparency matrix have been examined, the first subprocedure is completed and the condensing algorithm proceeds to the next.

2. DetectingDisconnected Branches and Not to Delete Partition Nodes

The second subprocedure has two tasks to perform. The first task is to determine which grid branches are switched off on all their sides. Since these grid branches are lines or infeeds that are completely disconnected from the power grid, they do not have to be considered in advanced application functions like power flow calculations or short circuit calculations (see Subsection 3.1.2 for other examples...
5.2 The Condensing Algorithm

Figure 5.8: Step 9 marks in the derived transparency matrix the column of the actual switch and the column of its pair switch as *Checked* data objects.

of advanced applications). Hence, they are marked as *Disconnected* when deriving the condensed transparency matrix. It would be possible to keep such a grid branch – being marked as *Disconnected* in an additional attribute row – in the condensed transparency matrix. This would have the advantage that this grid branch can still be accessed with all its attributes if needed. However, to emphasize the idea of the condensing algorithm, completely disconnected grid branches are filtered out.

The second task determines the partition nodes which have to be kept in the condensed grid model. Normally, each partition node has to be filtered out during the derivation of the condensed transparency matrix, as the virtual link directly connecting the grid branch to the grid node makes it redundant. But for the case where a line is switched off at only one side, the partition node at this side appears like a real grid node. Thus, this partition node has to be marked as *Not to be deleted* and to be kept in the condensed transparency matrix. Similarly, the partition node of a switched off infeed has to be included in the condensed transparency matrix.

Figure 5.9 schematically presents the principally different possibilities how infeeds or lines can be disconnected or connected to a single or a double busbar for the following infeed-busbar and line-busbar configurations:

- Infeed with a single switch
- Infeed with pair switches
Line with two single switches

Line with a single switch and pair switches

Principally different in this context means that it is only considered how a line can be disconnected or connected to busbars but not how the line itself is directed. That is why in Figure 5.9 just one line direction is shown per principally different condensed topology. Since the direction of a line is given by a connectivity integer at one side and its additive inverse at the other side, turning the direction of a line would only change the algebraic sign of all connectivity integers in the column of the derived transparency matrix where the considered line is stored.

All infeed-busbar and line-busbar configurations appearing in Test Distribution Grid 1 and Test Distribution Grid 2 presented in Appendix B are contained in the list above. Thus, the first and the second task have to be able to determine if lines have to be filtered out and if partition nodes have to be included or not into the condensed grid model for all possibilities presented in Figure 5.9. Fulfilling this requirement, these tasks will work correctly for both test distribution grids.

Further considerations revealed that both tasks can be simultaneously accomplished when for each data column storing either a line or an infeed the connectivity integers in the incidence matrix part of the derived transparency matrix are examined. To visualize this basic idea in Figure 5.9, the principally different condensed topologies are schematically sketched together with their connectivity integers\(^1\): Next to each partition node and next to each connected grid node, its connectivity integer appearing in the derived transparency matrix column of the considered grid branch is noted. Similarly, an integer 0 is written next to each from the considered grid branch disconnected grid node. By both summing up and counting all appearing connectivity integers, it is possible to perform a case differentiation for each line and each infeed. With the help of this case differentiation, it can be clearly decided which line has to be filtered out and which partition node has to be kept in the grid model.

The second subprocedure has hence to take the derived transparency matrix and to execute for each data object column containing an infeed or a line the following:

1. **Summing up and counting all connectivity integers**

   All incidence matrix elements of the actual column are searched for connectivity integers. The number of found connectivity integers is counted and stored

\(^1\)As mentioned above, only one line direction is shown per principally different topology.
### 5.2 The Condensing Algorithm

#### Infeed with a single switch

| Number of connectivity integers: | N = 1 | N = 2 |
| Sum of connectivity integers:    | S ≠ 0 | S ≠ 0 |
| Resulting case:                  | Case 1 | Case 2 |

#### Infeed with pair switches

| Number of connectivity integers: | N = 1 | N = 2 | N = 3 |
| Sum of connectivity integers:    | S ≠ 0 | S ≠ 0 | S ≠ 0 |
| Resulting case:                  | Case 1 | Case 2 | Case 3 |

#### Line with two single switches

| Number of connectivity integers: | N = 2 | N = 3 | N = 4 |
| Sum of connectivity integers:    | S = 0 | S ≠ 0 | S = 0 |
| Resulting case:                  | Case 4 | Case 5 | Case 6 |

#### Line with a single switch and pair switches

| Number of connectivity integers: | N = 2 | N = 3 | N = 3 |
| Sum of connectivity integers:    | S = 0 | S ≠ 0 | S ≠ 0 |
| Resulting case:                  | Case 4 | Case 5 | Case 5 |

Figure 5.9: Tabular illustration of the principally different possibilities how infeeds or lines can be disconnected or connected to a single or a double busbar.
in the variable \( N \). In addition, the found connectivity integers are summed up and the resulting sum is stored in the variable \( S \).

2. Case differentiation

Based on the values stored in the variables \( N \) and \( S \), a case differentiation allowing to distinguish eight different cases can be done for each infeed and each line:

**Case 1:** \((N = 1) \land (S \neq 0)\)

This case can only appear for a completely disconnected infeed, namely either for an infeed with a single switch that is open or for an infeed with pair switches that both are open. Because of the disconnected infeed, the partition node appears like a real grid node in the condensed grid model. Hence, the derived transparency matrix row of the partition node has to be marked as *Not to be deleted*.

**Case 2:** \((N = 2) \land (S \neq 0)\)

This case appears either for an infeed with a single switch that is closed or for an infeed with pair switches when only one of the switches is closed. Since over the closed switch the infeed is connected to the single busbar or to one double busbar, the partition node is not needed in the condensed grid model. Thus, the derived transparency matrix row of this partition node has to be marked as *To be deleted*.

**Case 3:** \((N = 3) \land (S \neq 0)\)

If the data object is an infeed with pair switches (compare with **Case 5**), then this case can appear when both switches are closed. So, the infeed is connected to both busbars and the partition node is not needed in the condensed grid model. Consequently, the derived transparency matrix row of the partition node has to be marked as *To be deleted*.

**Case 4:** \((N = 2) \land (S = 0)\)

This case appears for lines that are switched off at both sides. Since these lines are completely disconnected from the rest of the distribution grid, they do not appear in most of the advanced application functions. Hence, the derived transparency matrix column of such a line is marked as *Disconnected*. Accordingly, their adjacent partition nodes are not needed any more and their derived transparency matrix rows have to be marked as *To be deleted*. 
Case 5: \((N = 3) \land (S \neq 0)\)
If the data object is a line (compare with Case 3), then this case can appear when the line is at one side switched off and at the other side connected to the busbar adjacent to the closed switch. So, if the line has pair switches at this side, only one of the switches is closed and the other one is open. The partition node at the switched off side appears in this case like a real grid node, which has to be kept in the condensed grid model. Thus, the derived transparency matrix row of this partition node has to be marked as Not to be deleted.

Case 6: \((N = 4) \land (S = 0)\)
Actually, this case is the standard one: A line is at both sides connected to a busbar. This is the case for lines with two single switches when they are both closed. Additionally, this case can appear for lines with a single switch and pair switches when the single switch is closed and only one of the pair switches is closed. Since the line is at both sides connected to a busbar, the partition nodes are not needed in the condensed grid model. Hence, their derived transparency matrix rows have to be marked as To be deleted.

Case 7: \((N = 4) \land (S \neq 0)\)
This case can only appear for a line with a single switch and pair switches when the single switch is open and both pair switches are closed. The partition node by the switched off single switch appears then like a real grid node in the condensed grid model. Thus, to be kept in the condensed grid model, the derived transparency matrix row of this partition node has to be marked as Not to be deleted.

Case 8: \((N = 5) \land (S \neq 0)\)
This case can only appear for a line with a single switch and pair switches when all switches are closed. Since over the closed switches the line is on both sides connected to one busbar respectively to two busbars, the partition nodes are not needed in the condensed grid model. Therefore, their derived transparency matrix rows have to be marked as To be deleted.

After each column of the derived transparency matrix containing either an infeed or a line has been checked, all partition node rows are marked with either Not to
Figure 5.10: After the second subprocedure, the partition node of the pair switches is correctly marked as *To be deleted*.

*be deleted* or *To be deleted* as required. Similarly, the derived transparency matrix columns of the infeeds and lines that are completely disconnected from the rest of the distribution grid are all correctly marked as *Disconnected*. For the above introduced example of the closed Circuit Breaker 6 and its open pair switch the Circuit Breaker 7, Figure 5.10 presents the derived transparency matrix at this stage of the condensing procedure. As can be seen there, the partition node of the pair switches is marked as *To be deleted* as demanded.

### 3. Merging of Multiple Busbars

This third subprocedure has the task to condense or rather to merge busbars which due to short electric connections between them have the same voltage. In Test Distribution Grid 1 or in Test Distribution Grid 2, such a short electric connection can be a closed busbar coupler, a closed busbar sectionalizer or two closed pair switches. Merging a busbar with a second one means that the resulting merged busbar has both the connections of the first and the second busbar. Thus, the merging procedure for two connected busbars has in principle to execute the following tasks:

- If a line or a switch is connected to the first busbar, then it has to be connected virtually also to the second busbar.
5.2 The Condensing Algorithm

Figure 5.11: Merging of the two busbars Busbar A and Busbar B due to the closed busbar coupler denoted as Switch 2.

- If a line or a switch is connected to the second busbar, then it has to be connected virtually also to the first busbar.

Using the derived transparency matrix, this merging procedure can easily be done by copying all connectivity integers appearing in the row of the first busbar to the corresponding matrix elements in the row of the second busbar and vice versa. Thus from a mathematical point of view, the merging of two connected busbars can be done by a special kind of disjunction function. Unlike the standard logic disjunction, which only uses two logical values, namely true respectively 1 and false respectively 0, the special disjunction function for merging busbars has to operate with the 0 integer and all connectivity integers -2, -1, 1 and 2. More precisely, the connectivity integers act like the logical value true and the 0 integer acts like the logical value false.

A simple busbar configuration with two busbars, one busbar coupler and two lines connected over single switches is taken as concrete example to illustrate how the special disjunction function works in Figure 5.11. Assuming that the two single switches, denoted as Switch 1 and Switch 3, are closed, the first subprocedure of the condensing algorithm has for each of them virtually linked its adjacent line to its adjacent busbar. Since the switches are not yet filtered out from the derived
transparency matrix, Switch 1 constitutes a redundant connection from Line 1 to Busbar A, and Switch 3 constitutes a redundant connection from Line 2 to Busbar B. The closed busbar coupler, denoted as Switch 2, has in Step 9 of the first subprocedure caused that its column number is stored together with the row number of Busbar A and the row number of Busbar B in the designated double busbar list. These stored column and row numbers indicate the third subprocedure that it has to merge Busbar A and Busbar B due to the closed Switch 2. In Figure 5.11a), this situation is pictured together with the part of interest of the derived transparency matrix. Highlighted with red arrows, it is shown how the merging subprocedure has to copy each connectivity integer appearing in the row of one busbar to the corresponding matrix element in the row of the other busbar. Figure 5.11b) schematically sketches the two busbars and the part of interest of the derived transparency matrix after the special disjunction function has established all required virtual connections. In the graphical scheme of the two busbars, the established virtual connections are sketched as dashed lines. In addition, the derived transparency matrix shows that now all switches and lines are connected to both Busbar A and to Busbar B. Hence, the derived transparency matrix row storing Busbar A is identical with the one storing Busbar B. To finally merge these two busbars into only one, the subprocedure has to filter out one busbar and to keep the other one in the derived transparency matrix. For the decision which busbar has to be kept and which has to be deleted, the identifiers of the busbars are taken: The busbar that due to its identifier appears first in the derived transparency matrix is taken as the condensed busbar and the row of the other busbar is marked as To be deleted. This final stage of the third subprocedure is visualized in Figure 5.11c).

For substation configurations with more than two busbars that are coupled by busbar couplers or busbar sectionalizers, the merging subprocedure is more complex. The reason is that in such a substation configuration busbars can be indirectly connected to each other. Indirectly, in this context means that the shortest connecting path between two busbars contains at least two switches and at least one busbar. A concrete example of such a substation configuration composed of three busbars is pictured in Figure 5.12. Since Busbar A is connected to Busbar B over a busbar coupler, denoted as Switch 2, and since Busbar B is connected to Busbar C over another busbar coupler, denoted as Switch 4, Busbar A and Busbar C are indirectly connected to each other. The closed busbar couplers Switch 2 and Switch 4 caused in Step 9 of the first subprocedure that their column numbers are stored together with the row numbers of their adjacent busbars in the double busbar list.
5.2 The Condensing Algorithm

Figure 5.12: Merging of the three busbars Busbar A, Busbar B and Busbar C due to the closed busbar couplers Switch 2 and Switch 4.
Since the column number of Switch 2 is less than the column number of Switch 4, Switch 2 appears first in this list. Consequently, the merging subprocedure starts with Busbar A and Busbar B as pictured in Figure 5.12a). After the incidence matrix elements in the row of Busbar A and in the row of Busbar B are updated according to the special disjunction function, the merging subprocedure checks the row numbers noted together with the stored column number of Switch 4. It ascertains then that Busbar B and Busbar C have to be merged together, what is visualized in Figure 5.12b). Since Busbar A is also connected to Busbar B, the update of the incidence matrix elements in the row of Busbar B has to have also an implication on the incidence matrix elements in the row of Busbar A. Hence, the merging subprocedure has to pass again through the double busbar list and to check for all switches listed by column number if their adjacent busbars have to be updated by the special disjunction function. Therewith as shown in Figure 5.12c), Busbar A gets correctly updated, and after two passes through the double busbar list, the incidence matrix elements of the row of Busbar A are identical with the ones of the row of Busbar B and also with the ones of the row of Busbar C. By filtering out the row of Busbar B and the row of Busbar C, the three busbars are condensed together on Busbar A, which becomes the required condensed busbar. The row of Busbar B and the row of Busbar C are hence marked as To be deleted. This final stage of the merging subprocedure is shown in Figure 5.12d).

The concrete example of Figure 5.12 shows that it is possible to correctly merge three busbars by passing two times through the double busbar list and by doing the according updates in the rows of the relevant busbars. Hence, it is plausible that also more than three busbars can be merged. What matters is that the number of needed passes through the double busbar list can be determined. But primarily, it has to be figured out if the number of needed updates of the relevant busbar rows can be determined. All changes in a row of one busbar, which have to be done due to a computed special disjunction function, are counted as one update. Consequently, if only two busbars have to be merged by one special disjunction function, two updates have to be done: One update changes the incidence matrix elements in the row of the first busbar and one update changes the incidence matrix elements in the row of the second busbar. Besides, it has also to be considered, if the number of coupling switches has an influence on the number of needed updates. It is obvious that for connecting $n$ busbars at least $(n - 1)$ switches are necessary. From a mathematical point of view, for the disjunction of $n$ busbars exactly $(n - 1)$ disjunction operators are needed. Since each coupling switch acts like a disjunction operator, an additional $n$–th switch would only cause a redundant disjunction.
Figure 5.13: Determination of the needed updates for merging the five busbars Busbar A, Busbar B, Busbar C, Busbar D and Busbar E due to the closed busbar couplers Switch 1, Switch 2, Switch 3 and Switch 4.

function. Hence, the number of needed updates only depends on the number of connected busbars.

The mathematical function for the number of needed updates can be derived with the help of a concrete example: As pictured in Figure 5.13, this exemplary substation configuration contains five busbars that are connected over four closed switches. To derive the searched mathematical function, instead of using the merging subprocedure described in Figure 5.11 and Figure 5.12 above, the busbars are merged together one by one. By doing so, the merging algorithm has also to update all rows of indirectly connected busbars that are already merged with another busbar but are affected by the merging of the actual busbar with its subsequent busbar. Starting with Busbar A, this merging algorithm proceeds as follows:

**Busbar A is the actual busbar:**

Merging Busbar A and Busbar B:
- 1 update for the row of Busbar A
- 1 update for the row of Busbar B

⇒ 2 updates are needed.
Busbar B is the actual busbar:

Merging Busbar B and Busbar C:
- 1 update for the row of Busbar B
- 1 update for the row of Busbar C

Merging has an implication on Busbar A:
- 1 update for the row of Busbar A

⇒ 3 updates are needed

Busbar C is the actual busbar:

Merging Busbar C and Busbar D:
- 1 update for the row of Busbar C
- 1 update for the row of Busbar D

Merging has an implication on Busbar A and Busbar B:
- 1 update for the row of Busbar A
- 1 update for the row of Busbar B

⇒ 4 updates are needed

Busbar D is the actual busbar:

Merging Busbar D and Busbar E:
- 1 update for the row of Busbar D
- 1 update for the row of Busbar E

Merging has an implication on Busbar A, Busbar B and Busbar C:
- 1 update for the row of Busbar A
- 1 update for the row of Busbar B
- 1 update for the row of Busbar C

⇒ 5 updates are needed

Therefore, the total number $N_U$ of needed updates for this substation configuration is calculated as

$$N_U = 2 + 3 + 4 + 5 = 14. \quad (5.1)$$

The proceeding of this merging algorithm is clearly visualized in Figure 5.13. A further sixth busbar connected to Busbar E would cause six additional updates, since the merging with this sixth busbar would influence the rows of all other busbars. Hence, it is obvious how the number of needed updates would grow with each further busbar. The required mathematical function for the number of needed
updates for a substation configuration with \( n \) connected busbars is thus given by the equation

\[
N_U = 2 + 3 + \ldots + (n-1) + n = \sum_{k=1}^{n} k - 1 = \frac{n \cdot (n+1)}{2} - 1, \tag{5.2}
\]

which contains the sum of the first \( n \) natural numbers (see for instance [RaWe00]) but is reduced by 1. The reason for this subtraction by 1 becomes evident when the proceeding of the merging algorithm in Figure 5.13 is examined: Since Busbar A has not to be merged with itself, this update is missing and reduces the number of needed updates by 1.

Extensive studies at test examples showed that the number of needed passes through the double busbar list not only depends on the number of connected busbars but also on the coupling topology of the busbars and on how the coupling switches are sorted in this double busbar list. Thus for a general substation configuration, it is not possible to determine the exact number of needed passes just as a function of the number of connected busbars. But at least the maximal number of needed passes, which the merging subprocedure has to make when the switches in the double busbar list are sorted in the most unfavorable way, can be determined. For the derivation of this required function, the merging algorithm presented in Figure 5.13 is examined again. During the proceeding of this merging algorithm, each of the first \((n-1)\) busbars becomes once the actual busbar causing the update of the row of its subsequent busbar and the updates of the rows of the already merged busbars. The update of the actual busbar and the update of the subsequent busbar might as well be caused by searching in the double busbar list for the switch connecting these two busbars. Similarly, the further needed updates of the rows of the already merged busbars might as well be caused by completing one pass through the double busbar list searching for all switches connecting the actual busbar with the already merged busbars. Hence, doing all needed updates for a specific actual busbar corresponds to one complete pass through the double busbar list. Consequently, even in the most unfavorable case, all \( n \) busbars can be correctly merged together by passing \((n-1)\)-times through the double busbar list. The maximal number of needed passes is hence simply given by

\[
N_{MaxPass} = n - 1. \tag{5.3}
\]
This derived formula can be easily verified by a mathematical induction (see for instance [RåWe00]): Assuming that one additional busbar is connected to a substation configuration where all busbars are already merged together in the derived transparency matrix. To merge this additional busbar with all other busbars, one additional pass through the double busbar list is needed to update each busbar row accordingly.

Since the maximal number \( N_{\text{MaxPass}} \) of needed passes has the function of a conservative stopping criterion for the merging subprocedure, it is important that also a non-conservative stopping criterion can be found for the final implementation of the condensing algorithm. The reason is the above mentioned fact that depending on the coupling topology of the busbars and on how the coupling switches are sorted in the double busbar list, all needed updates can be completed within less than \((n-1)\) passes. In such a case, the merging subprocedure can or rather should terminate already before the conservative stopping criterion is fulfilled, what has to be assured by a non-conservative stopping criterion. A non-conservative stopping criterion can be done by counting all accomplished updates of the derived transparency matrix rows or by using a flag that signifies when in a complete pass no update has been done anymore. The concrete implementation of the condensing algorithm will show which non-conservative stopping criterion is most qualified.

4. Deleting of Switch Columns and Marked Partition Node Rows

The fourth and last subprocedure has as first task to delete or rather to filter out all derived transparency matrix columns that contain a switch, a meta data object or a grid branch that is marked as \textit{Disconnected}. The second task of this fourth subprocedure is to delete all partition and grid node rows of the derived transparency matrix that are marked as \textit{To be deleted}. In Figure 5.14, this development stage of the derived transparency matrix is visualized, whereby the rows and columns that are going to be deleted are black colored. Finally, the derived transparency matrix is renamed \textit{condensed transparency matrix}. The special incidence matrix of this condensed transparency matrix is denoted as \textit{condensed incidence matrix}.

It has to be mentioned that the data objects of the busbars are still appearing in the condensed transparency matrix. The reason is that the attributes of these busbar data objects are needed for the advanced applications, for instance the measurements of the complex busbar voltages. Thus, the busbar data objects must not be filtered out. On the other hand, the meta data objects of the substations
5.2 The Condensing Algorithm

The Condensing Algorithm

The above described condensing algorithm has as outcome the condensed transparency matrix representing the desired condensed grid model containing only grid components, which as described in Subsection A.1.2 comprise grid nodes, namely the busbars of the substations, and grid branches, which are either lines or infeeds. The partition nodes that are not deleted during the condensing algorithm and consequently are appearing in the condensed grid model are counted as grid nodes too. Consequently, the number of columns of the condensed incidence matrix corresponds to the sum of busbars, lines and infeeds in operation. Similarly, the number of rows of the condensed incidence matrix corresponds to the number of busbars and not deleted partition nodes in operation. Using Test Distribution Grid 1 introduced in Subsection 4.2.1, the condensing algorithm is verified for

Figure 5.14: The black colored rows and columns are going to be filtered out of the derived transparency matrix so that it contains only busbars, partition nodes not to be deleted, lines and infeeds.

do not necessarily have to appear in the condensed transparency matrix, since the attributes of these meta data objects will not be needed, at least not for power flow calculation. Consequently, these substation meta data objects are filtered out in this last subprocedure. With the finishing of the fourth subprocedure, the entire condensing algorithm is terminated.

5.2.3 The Resulting Condensed Transparency Matrix

The above described condensing algorithm has as outcome the condensed transparency matrix representing the desired condensed grid model containing only grid components, which as described in Subsection A.1.2 comprise grid nodes, namely the busbars of the substations, and grid branches, which are either lines or infeeds. The partition nodes that are not deleted during the condensing algorithm and consequently are appearing in the condensed grid model are counted as grid nodes too. Consequently, the number of columns of the condensed incidence matrix corresponds to the sum of busbars, lines and infeeds in operation. Similarly, the number of rows of the condensed incidence matrix corresponds to the number of busbars and not deleted partition nodes in operation. Using Test Distribution Grid 1 introduced in Subsection 4.2.1, the condensing algorithm is verified for
two cases that have basically different actual grid topologies:

1. **Decoupled double busbar**
   For an actual grid topology where the busbars Busbar 1 and Busbar 2 are decoupled, the condensed actual topology and the condensed transparency matrix are pictured together in Figure 5.15.

2. **Closed pair switches**
   For an actual grid topology where due to the closed busbar coupler Circuit Breaker 1 the two busbars Busbar 1 and Busbar 2 are coupled together, the condensed actual topology and the condensed transparency matrix are pictured together in Figure 5.16. Due to the busbar coupling, Busbar 2 is eliminated and does not appear in the condensed grid model.

The condensed transparency matrices of the two cases show that the condensing algorithm reduces considerably the amount of data objects. Hence, the advantage of the condensed transparency matrix is not only that it represents the desired node-branch-grid model but also that it has a data size, which is better manageable for advanced application functions.

### 5.2.4 The Condensing Algorithm as Topology Processor

The virtue of the condensing algorithm is that it establishes a direct and efficient link between SCADA functions using the complete grid model and application-oriented algorithms requiring a node-branch-grid model. Because of this link between the data model appropriate for SCADA functions and the advanced data models, the condensing algorithm functions like a *topology processor*. According to [AbGó04], a topology processor (TP) as part of a state estimator (see for instance also [AbGó04]) for an EMS converts a detailed model with busbars and switches into a compact and more suitable busbar-branch-model in consideration of gathered status data. From this perspective, the condensing algorithm could be referred to as topology processor. However, since the condensing algorithm uses as source the transparency matrix to produce as an outcome a matrix that in comparison with its original matrix is much more compact or rather condensed, the name condensing algorithm will be retained. The intention is also to emphasize with the name that the developed algorithm bases on a new conceptual model to store all data objects of the control system in one unique matrix. Hence, the name
Figure 5.15: Case of decoupled double busbar in Station A: The condensed transparency matrix on the left side represents the condensed grid model referring to the condensed actual topology, which is pictured on the right side.
Figure 5.16: Case of *closed pair switches* in Station A: The condensed transparency matrix on the left side represents the condensed grid model referring to the condensed actual topology, which is pictured on the right side.
The condensing algorithm accentuates that the needed information is gained by simply condensing the information contained in the transparency matrix.

5.2.5 The Transparency Level of the Condensing Algorithm

The derived condensing algorithm finally allows to evaluate if the third level of transparency can be achieved. According to the definition of the transparency levels stated in Subsection 4.1.2, data consistency and data correctness of derived data models have to be guaranteed at all times to achieve the third level of transparency. Moreover, the definition requires that the so called derivation algorithms generate such data models in an efficient way.

The condensing algorithm uses mainly the filterability of the transparency matrix to generate the condensed transparency matrix: The task of the first three subprocedures of the condensing algorithm is mainly to determine and mark which rows and columns have to be deleted. Then, using the filterability of the matrix structure, the final fourth subprocedure deletes the marked columns and rows. Since the filtering does not alter any data object information stored in the derived transparency matrix, no data inconsistency or data incorrectness can be created by the condensing algorithm. Consequently, as data consistency and data correctness are guaranteed for the transparency matrix being the data source of the condensing algorithm, these properties are also guaranteed for the condensed transparency matrix. And since the condensed transparency matrix represents a derived data model, it can be logically stated that concerning data consistency and data correctness the third level of transparency is achieved with the help of the condensing algorithm. But the remaining question is if the condensing algorithm generates this derived data model in an efficient and reliable way. By briefly recapitulating the tasks of each subprocedure, the developed condensing algorithm can be evaluated with regard to its efficiency:

1. Construction of the Derived Transparency Matrix
   For every closed switch, the establishing of the virtual link between its adjacent line and its adjacent busbar can be reliably and easily done: In the derived transparency matrix just one connectivity integer has to be added so to have a connectivity square.

2. DetectingDisconnected Branches and Not to Delete Partition Nodes
   By summing up and counting the appearing connectivity integers in the rele-
vant column of the derived transparency matrix, it is possible to distinguish by use of simple inequality and equality conditions if a line is completely disconnected or if the adjacent partition nodes have to be deleted or not.

3. Merging of Multiple Busbars
   The merging of multiple connected busbars can easily be done with the derived transparency matrix or rather its incidence matrix. Since for a concrete implementation of the condensing algorithm, a simple non-conservative stopping criterion can be determined, no unnecessary passes through the double busbar list are executed and the merging subprocedure will terminate as soon as possible.

4. Deleting of Switch Columns and Marked Partition Node Rows
   Using the filterability of the derived transparency matrix, the columns and rows to be deleted can be easily filtered out.

This brief recapitulation of all subprocedures shows that they all have some precisely defined tasks to do. Since these tasks are easy and straightforward to perform, it can be concluded that the condensing algorithm works effectively and thus fulfills the requirements for the third level of transparency. Hence, it is finally proved that the conceptual transparency model discussed in Chapter 4 allows to develop useful derivation algorithms so that the third level of transparency can be reliably achieved.

5.3 The Condensed Grid Topology Detection Algorithm

As described in Subsection 5.1, possible existing rings have to be detected before a power flow calculation for distribution grids or any other advanced application function can be started. That is why an algorithm has to be developed that uses the condensed grid model to identify grid branches which are within a ring structure of the actual topology. Since the condensed grid model is represented by the condensed transparency matrix, the algorithm has to use the latter as input source. By doing so, it should not only detect possible existing rings but also investigate the actual grid topology more extensively. As the condensed transparency matrix contains the condensed incidence matrix, the algorithm has to examine the incidence matrix elements in a particular manner: The aim is that the way how
5.3 The Condensed Grid Topology Detection Algorithm

the algorithm examines the condensed incidence matrix will allow to gain more information about the condensed grid topology than just to detect if rings are present. For instance, junction nodes and end nodes should be detected too and the distance of each node to the infeeding node should be determined as well (see Subsection 5.3.2). Consequently, the algorithm should detect the condensed grid topology and is therefore named condensed grid topology detection algorithm. As the condensing algorithm is executed after every switching action, it is clear that the condensed topology represented by the condensed transparency matrix is the actual one. Thus, it is not necessary to refer to this fact by the name of the algorithm.

5.3.1 Well-Known Published Algorithms for Traversing and Searching of Radial or Meshed Network Structures

In mathematics and computer science, especially in the field of graph theory, a wide range of different algorithms for network structures exist. Many of them have the task to search a path or rather the shortest path between two nodes of the given network. A well known algorithm that solves this problem of finding the shortest-path through a network structure is the Dijkstra’s algorithm, which for instance is explained in [Weis99]. Special attention is also attracted by so called tree structures, which are defined as being connected graphs respectively connected networks containing no cycles, as for instance stated in [Bigg99]. Since trees are fundamental data structures, many different algorithms for traversing, searching and sorting of trees exist. There are also popular traversing and searching algorithms that work on tree structures as well as on meshed network structures, such as the depth-first search algorithm (DFS) or the breadth-first search algorithm (BFS). These two algorithms, which are often denominated as traversing algorithms, are alternatives to each other and are for instance described in [Bigg99]. As the algorithm to be developed here has to discover the condensed grid topology, which can have a tree structure – in the context of distribution grids more often denoted as radial structure – or a meshed structure containing rings, it can be expected that the algorithm has some strong similarities with the depth-first search algorithm or the breadth-first search algorithm. However, the aim is not simply to choose one of these well-known published algorithms and to modify the chosen one so that the condensed grid topology can be discovered in the desired way. The objective is rather to obtain first a principal idea of how the algorithm has to travel or rather to discover the grid structure so that not only possible existing rings can be detected
but also that the unknown quantities of a power flow problem can be calculated. Then, it can be determined if the proposed traversing method is similar to one of these well-known principles.

5.3.2 Detecting of the Condensed Grid Structure

As stated above, the main task of the algorithm to be developed is the detection of existing rings. For this task, a suitable strategy for traversing through the condensed grid topology has to be determined. Since it is the intention to use, if possible, the same traversing strategy also for the calculation of unknown quantities when solving power flow problems, it has to be considered how the unknown quantities like complex voltages and currents are associated to grid nodes and grid branches.

For this reason, a radial grid structure with ten grid nodes and ten grid branches, whereof one is an infeed, serves as example to study the sequence in which the complex currents have to be calculated. In Figure 5.17, this exemplary grid structure is pictured together with the branch currents and the complex power values at the nodes. The radial grid structure is energized by Infeed 1 being connected to Node A, which is at the “top” of the tree structure. In mathematics and computer science, this node at the top that does not have any predecessor node, is denoted as the root or the root node, as it is the case in [Bigg99] or in [Weis99]. In the context of power systems, it is more common to denominate these nodes adjacent to an infeed as source nodes, like it is done in [Kers02]. Consequently, Node A can be referred to as the root node or better as the source node. Each of the two outgoing branches leading away from Node A constitutes together with its successor branches and successor nodes a so called distribution feeder (see for instance [Kers02]). The branches belonging to each of these two distribution feeders are highlighted in blue respectively in green color in Figure 5.17. In graph theory, nodes that are adjacent to only one branch are denoted as leaves or leaf nodes. According to this definition, Node E, Node F, Node I and Node J of the exemplary grid structure can be referred to as leaf nodes. However, in [Kers02], considering distribution grids, these nodes are denoted as end nodes. The reason for this is that in each of these leaf nodes one arm of a distribution feeder is ending. So in the following, Node E, Node F, Node I and Node J are referred to as end nodes.

The source node and the end nodes have an important meaning during power flow calculations: Depending on which power flow variables are known and which ones
have to be calculated or rather updated (see Section 6.4), the determination of a specific power flow variable starts or ends at one of these nodes. For instance, assuming that for each node feeding a load, its complex load current is given, the complex line currents can be calculated by starting at the end nodes using the given complex load currents and proceeding towards the source node. On the path up towards the source node, at each node the complex line currents and its complex load current have to be calculated according to Kirchhoff’s current law (KCL). At Node C, for example, the following equation holds:

\[ I_1 = I_6 + I_8 + I_C, \]  

whereas \( I_C \) is the complex load current supplying the load of Node C and the complex line currents \( I_6 \) and \( I_8 \) have been calculated by using the Kirchhoff’s current law at Node B respectively Node F. This simple consideration of how to calculate the complex line currents reveals how the algorithm has to travel through the radial tree structure: It has to start at the source node and then to pass in
the direction of the power flow until it finds an end node. There, the algorithm has to turn and to pass back towards the source node. On its way back, it checks for each node, if there is a branch not yet being explored. If this is the case, the algorithm has to pass on this branch following the direction of the power flow until it again finds an end node. In Figure 5.18, this described traversing method is visualized by means of red, green and blue arrows. Steps towards an end node are visualized with red arrows. Since in radial grid structures, such steps are always in the direction of the power flow (as long as there is no distributed generation), they can be denoted as *downstream steps*. Likewise, steps back towards the source node and hence in the opposite direction of the power flow can be denoted as *upstream steps*. These upstream steps are visualized with green arrows. The blue arrows at the end nodes indicate that the algorithm changes its traversing direction from *downstream* to *upstream*. At the head of each arrow, the associated step number is annotated so that the traversing path of the algorithm can easily be traced when observing Figure 5.18. This observation of the traversing path finally reveals that the proposed method has the same principle as the depth-first search method. Since the algorithm to be developed uses the condensed incidence matrix as basis, the resulting depth-first search path is essentially also determined by the identifiers of the data objects, especially by the identifiers of the lines. To make this comprehensible, all downstream steps of the traversing method are visualized with red arrows together with the associated step number in the condensed transparency matrix, which for this reason is also presented in Figure 5.18.

The gained knowledge that the traversing method uses the same principle as the depth-first search algorithm is however not of importance: The determination of which operations have to be done at each node or along each branch is more important. For instance, it has to be determined how the Kirchhoff’s current law or voltage law have to be used at each node. From this point of view, the depth-first search strategy actually only provides the *sequence* in which the relevant calculations have to be performed. As this sequence mainly depends on how many neighbors each node has, it is worthwhile to classify the nodes into different types:

1. Source node
2. Normal node
3. Junction node
4. End node
Figure 5.18: Illustration of the traversing method together with the condensed transparency matrix and the four node types highlighted with different colors.

In Figure 5.18, these four different types of nodes are indicated with different colors. The red colored source node and the green colored end nodes are already discussed above. In contrast to these exterior nodes, the normal nodes and the junction nodes are highlighted with different colors in the transparency matrix.
nodes, mentioned here for the first time, are appearing inside of the tree structure. Tree topologies assumed, the gray colored normal nodes are adjacent to exactly two branches, whereas the blue colored junction nodes are adjacent to at least three branches\(^2\). This classification of the nodes can be used to briefly describe the working principle of the depth-first search strategy depending on the type of node. Actually, the depth-first search algorithm as one of the most used traversing algorithms is well documented. But since the algorithm to be developed will in addition perform special calculation tasks at each node on the way downstream and upstream, the basic working principle for each type of node is reported in the interest of clarity:

1. **Source node:**
   - If the source node is reached via the infeed branch, mark this infeed branch as *Explored downstream*. Then, chose the adjacent branch that appears in the condensed transparency matrix first from left and is not yet explored. Follow this branch downstream to the next node. Finally, mark both the source node and the chosen branch as *Explored downstream*.
   - If the source node is reached via a branch on the way upstream, chose the adjacent branch that appears in the condensed transparency matrix first from left and is not yet explored. Then, follow this branch downstream to the next node. Finally, mark the chosen branch as *Explored downstream* and change the traversing direction marker to *Downstream*.
   - If the source node is reached via a branch on the way upstream and if there is no branch that is not yet explored, then mark both the source node and the infeed branch as *Explored upstream*. Herewith, the algorithm terminates.

2. **Normal node:**
   - If a normal node is reached via a branch on the way downstream, follow the other adjacent branch further downstream to the next node. Finally, mark both the normal node and this branch as *Explored downstream*.
   - If a normal node is reached via a branch on the way upstream, follow the other adjacent branch further upstream to the next node. Finally, mark both the normal node and this branch as *Explored upstream*.

\(^2\) Junction nodes and normal nodes can change their type in ring topologies, as written below.
3. Junction node:

- If a junction node is reached via a branch on the way downstream, chose the adjacent branch that appears in the condensed transparency matrix first from left and is not yet explored. Then, follow this branch downstream to the next node. Finally, mark both the junction node and the chosen branch as *Explored downstream.*

- If a junction node is reached via a branch on the way upstream, chose the adjacent branch that appears in the condensed transparency matrix first from left and is not yet explored. Then, follow this branch downstream to the next node. Finally, mark the chosen branch as *Explored downstream* and change the traversing direction marker to *Downstream.*

- If a junction node is reached via a branch on the way upstream and if there is no branch that is not yet explored, then follow the branch that leads further upstream to the next node. Finally, mark both the junction node and this branch as *Explored upstream.*

4. End node:

- If an end node is reached via a branch on the way downstream, return back on this branch to its previously considered node being located upstream. Finally, mark both the end node and this branch as *Explored upstream* and change the traversing direction marker to *Upstream.*

With these instructions listed for each type of node, the complete depth-first search algorithm for a radial grid structure is described in a formal way. For instance, by examining Figure 5.18, the listed instructions can be easily verified. It has to be mentioned that actually, for a concrete implementation of such a depth-first search strategy, it is not explicitly necessary to distinguish between normal nodes and junction nodes. The reason is that the instructions to perform for normal nodes are covered within the instructions for junction nodes: A normal node appears like a junction node from which all adjacent downstream branches but one are already explored.

Once the basic working principle of the depth-first search strategy is explained for radial grid structures, it can be enhanced for grid topologies containing rings. For this reason, the exemplary radial grid structure is expanded to a weakly meshed grid structure containing two rings: As pictured in Figure 5.19, by inserting Line 10 and Line 11, two rings are formed. If in such a ring topology situation, a branch can be found that is in more than one ring, the concerning rings are referred to as...
Figure 5.19: Illustration of the traversing method for the weakly meshed grid structure together with the condensed transparency matrix and the five node types.

meshed rings. A close inspection of Figure 5.19 shows that the two rings are meshed ones. It is important that the exemplary grid structure used for the development of the ring detection algorithm contains meshed and not only separated rings since
5.3 The Condensed Grid Topology Detection Algorithm

in general case, rings are meshed. By comparing Figure 5.18 and Figure 5.19, it can be seen that some of the nodes changed their type due to the inserted lines or rather due to the formed rings.

Using the depth-first search strategy for the grid topology pictured in Figure 5.19, the algorithm on its alleged way downstream will meet again nodes that it has already passed by a downstream step. Consequently, such a node is a special junction node with at least three adjacent branches. Two of these adjacent branches form with other branches together the ring structure, which is the reason that the same node is reached again on the “way downstream”. The junction node itself is like an entrance or an exit node to this ring structure. Hence, such a node is denominated as ring junction node. Due to the two meshed rings in the exemplary grid structure, this is the case for Node C and Node A. Thus, Node A is both a source node and a ring junction node. In Figure 5.19, the ring junction nodes are highlighted with magenta color. Arrived again at such a ring junction node, the algorithm has to turn and to pass back towards the source node, like it is also the case for end nodes. Accordingly, the basic working principle of the depth-first search algorithm for a ring junction node type combines the instructions for the end node type and the instructions for the junction node type:

5. Ring junction node:

- If a ring junction node is reached via a branch on the way downstream and is not yet marked as Explored downstream, chose the adjacent branch that appears in the condensed transparency matrix first from left and is not yet explored. Then, follow this branch downstream to the next node. Finally, mark both the considered branch and the ring junction node as Explored downstream.

- If a ring junction node is reached via a branch on the way downstream and is already marked as Explored downstream, return back on this branch to its previously considered node being located upstream. If the ring detection flag is not yet set to True, this has to be conducted. Finally, mark the considered branch as Explored upstream and change the traversing direction marker to Upstream.

- If a ring junction node is reached via a branch on the way upstream, chose the adjacent branch that appears in the condensed transparency matrix first from left and is not yet explored. Then, follow this branch downstream to the next node. Finally, mark the considered branch as
Explored upstream and change the traversing direction marker to Downstream.

– If a ring junction node is reached via a branch on the way upstream and if there is no branch that is not yet explored, then follow the branch that leads further upstream to the next node. Finally, mark both this branch and the ring junction node as Explored upstream.

The comparison of the instructions for ring junction nodes and the instructions for junction nodes also shows that ring junction nodes are special junction nodes: The instructions to perform for junction nodes are covered within the instructions for ring junction nodes.

Although the instructions for the ring junction node type are described in a formal way, they are rather a broad outline, since so far no details are given what exactly has to be done when a ring junction node is detected. The reason for this is that first it has to be decided if rings have only to be detected and marked or if there should be a possibility to treat them in a more sophisticated way: For instance, the aim can be to determine the optimal cut-off points where the rings can be broken, whereas it has yet to be defined what is to be understood as optimal. The power flow method that will be developed in Chapter 6 requires that in the case of slightly meshed grid topologies all rings have to be “virtually” broken so that the unknown quantities can be calculated. Hence, for each detected ring a cut-off point has to be anyway determined if this power flow calculation method should be executed.

Actually, it would be possible to break open the rings exactly at the detected ring junction nodes or, more precisely, to switch off the grid branch over which the ring junction node is met again. Then, the ring detection, the identification of grid branches that are within a ring structure and the “virtually” breaking open of the rings would be done with only one depth-first search cycle. Nevertheless, the cut-off points determined in this way would most likely not be optimal ones. The reason is that in the resulting tree structure not all nodes have the smallest possible node distance to the source node. Thereby, the number of branches that have to be passed to get from one node to another node is referred to as their node distance. Assuming that the impedances of the lines are about the same and that all nodes have loads with more or less the same complex power value, then such a radial grid topology where not all nodes have the smallest possible node distance to the source node is in terms of power consumption not an optimal one. Since not all loads are supplied by feeders having the smallest possible node distance,
power losses that could be avoided appear in the relevant lines. Hence, it can be
defined that cut-off points of rings are really *optimal* only when the power losses
in the feeder lines of the resulting radial grid topology are minimized. But since
the determination of such optimal cut-off points for rings would need already a
special algorithm of its own (see Section 8.2), a less sophisticated criterion has
to be chosen for how existing rings can be broken in a *distance-optimal* way: As
distance-optimal can be denoted those cut-off points that break existing rings such
that in the resulting radial grid structure each node has the smallest possible node
distance to the source node. Figure 5.20 presents the radial grid structure that
results when the rings of the weakly meshed grid structure of Figure 5.19 are
broken at the detected ring junction nodes during one depth-first search cycle. In
comparison, Figure 5.21 presents the resulting radial grid structure that would be
the outcome of a method breaking the rings distance-optimally. In Figure 5.20 and
Figure 5.21, the determined node distance to the source node, hereinafter referred
to simply as *source node distance*, is annotated for each node. In addition, the nodes
are colored according to their source node distance number. As can be seen by
comparing Figure 5.20 and Figure 5.21, the method of breaking the rings distance-
optimally is clearly the better one concerning the resulting node distances of the
end nodes: The method of breaking open the rings distance-optimally produces
as result a radial grid topology where only three end nodes have a source node
distance of 3 and all other nodes have source node distances of less than 3. In
contrast, in the radial grid topology that results from breaking open the rings at
the detected ring junction nodes, two end nodes have a source node distance of 6,
two nodes have a source node distance of 5 and two other nodes have a source node
distance of 4. Consequently, if the loads have more or less the same complex power
values and if all lines have approximately the same impedance values, the radial
grid topology with the distance-optimal cut-off points will be in terms of power
consumption almost optimal or even completely optimal.

5.3.3 The Procedure of the Condensed Grid Topology Detection
Algorithm

The insight gained in Subsection 5.3.2 leads to that the condensed grid topology
detection algorithm has not only to detect existing rings but also to determine their
distance-optimal cut-off points. At these distance-optimal cut-off points, the rings
can be “virtually” broken so that the unknown quantities can be determined by a
power flow analysis. Although, as mentioned in Subsection 5.3.2, it is possible to
Figure 5.20: The radial grid structure that results when the rings are broken at the detected ring junction nodes: Node B and Node J have a source node distance of 6.

Figure 5.21: The radial grid structure that results when the rings are broken in a distance-optimal way: The maximal appearing source node distance is 3.
detect all existing rings and to identify all grid branches that are within such a ring by only one depth-first search cycle, the distance-optimal cut-off points cannot be determined in the same search cycle. Hence, for the determination of the distance-optimal cut-off points, additional search cycles are needed or even another search method is required. The condensed grid topology detection algorithm can thus be structured into two subprocedures, each with an assigned task:

1. Ring detection and determination of the distance-optimal cut-off points
2. Identification of the grid branches within the “virtually” broken rings

In the following, the two subprocedures of the condensed grid topology algorithm are described in detail.

1. Ring Detection and Determination of the Distance-Optimal Cut-Off Points

To find out if the determination of the distance-optimal cut-off points needs several depth-first search cycles or if another search method is required, various test examples have been studied and careful considerations have been made. The findings revealed that even though the depth-first search strategy is suitable for ring detection or power flow calculation, it is not an appropriate search method to determine for each node of the power grid its smallest possible source node distance, hereinafter denoted as minimal source node distance. The reason is that the depth-first search strategy, whenever possible, moves downstream over an adjacent grid branch to the next node instead of checking first all neighboring nodes of the considered node. But for the determination of the minimal source node distances exactly this search method is needed: The search algorithm has to check first all neighboring nodes of the source node, which consequently all have 1 as the minimal source node distance, before moving downstream to one of these neighboring nodes. Then, one after another, these neighboring nodes are checked for their own neighboring nodes not yet explored, which accordingly have a minimal source node distance of 2. In this way, the search algorithm proceeds till it has explored all grid nodes. Hence, this search principle of spreading out radially from the source node to explore first the nearer nodes in the breadth of the power grid is the breadth-first search method. The condensed grid topology detection algorithm therefore needs both the depth-first search principle as well as the breadth-first search principle. That for the determination of the minimal source node distances the breadth-first search method is more advantageous than the depth-first search method is no real
surprise according to [Bigg99]. As stated there, breadth-first search is better suited to problems where some kind of optimization is required, such as finding a path with the smallest possible number of branches.

In Figure 5.22, the traversing principle of the breadth-first search method is visualized by means of arrows with different colors: Steps towards grid nodes with a minimal source node distance of 1 are visualized with blue arrows. Steps from these nodes to those with a minimal node distance of 2 are drawn with cyan colored arrows. Similarly, steps to nodes with a minimal node distance of 3 are visualized with green arrows and steps to nodes with a minimal node distance of 4 have arrows with dark green color. Since this search principle of spreading out radially explores first the nearer nodes with the smaller minimal node distances, the grid nodes are processed like in levels. For each level, its level number is given by the minimal source node distance of the grid nodes belonging to it. For the exemplary grid structure, the levels 0 to 3 are visualized with different colored areas in Figure 5.22. In addition, at the head of each arrow, the associated step number is annotated so that the spreading out of the algorithm can easily be observed. For instance, Node B is reached by step 4 of the breadth-first search algorithm coming from Node C. As Node C has a source node distance of 1, the minimal source node distance of Node B is consequently set to 2.

Due to the two existing rings, the spreading out of the breadth-first search algorithm reaches Node G and Node I over more than one grid branch, as can be seen in Figure 5.22. For this kind of grid nodes, their minimal source node distance has to be determined when they are reached by the spreading out of the algorithm for the first time. The reason is that the associated path over which such a node is reached the first time is the shortest possible one (or one of the shortest possible paths). The second time such a grid node is reached, the associated path has at least the same length or is even longer than the one over which the considered node was reached the first time. This can be verified for instance for Node G, which is reached the first time by step 6 of the algorithm. Since this step comes from Node D with the minimal source node distance of 1, the minimal source node distance of Node G is determined to 2. The second time Node G is reached from Node F over Line 10 by step 8 of the algorithm. As Node F already has a minimal source node distance of 2, the associated path over which Node G is reached the second time is longer than the one over which it is reached the first time. To visualize that the second time Node G respectively Node I is reached, its minimal source node distance has not anymore to be determined, step 8 and step 11 respectively are drawn with dashed arrows in Figure 5.22. The comparison of the minimal source
5.3 The Condensed Grid Topology Detection Algorithm

Figure 5.22: Visualization of the breadth-first search method for the weakly meshed grid structure together with the condensed transparency matrix.

node distances of Node G and Node I with their respective neighboring nodes shows that these two special nodes have the largest minimal source node distance (or one of the largest source node distances) of all nodes of the concerning ring.
Hence, given that the minimal source node distance is considered like a negative altitude difference, the two special nodes are lying like in the “valley bottom” of the concerning ring. Thus in the following, this kind of grid nodes are denominated as "valley nodes".

In Figure 5.22, the valley nodes are highlighted with yellow color. As a valley node per definition has the largest minimal source node distance (or one of the largest source node distances) of all nodes of the concerning ring, it constitutes the distance-optimal cut-off point at which this ring has to be broken. The appropriate grid branch that has to be “virtually” switched off is the one over which the valley node is reached the second time. Hence, in the case of the exemplary grid structure, Line 10 and Line 11 have to be switched off. The comparison of the thus resulting radial grid structure with the optimal radial grid structure pictured in Figure 5.21 proves that the breadth-first search method works correctly.

As the condensed grid topology detection algorithm uses the condensed incidence matrix as basis, the sequence in which the nodes of the same level are explored is essentially determined by the identifiers of the data objects of the condensed incidence matrix. To make this comprehensible, all steps of the spreading out of the breadth-first search strategy are visualized with arrows together with the associated step numbers in the condensed transparency matrix, which for this reason is also presented in Figure 5.22. Each of these arrows has the same color as the corresponding step arrow pictured together with the exemplary weakly meshed grid structure.

With the basic working principle of the breadth-first search method explained, the algorithm can be described in more detail. In particular, the setting of the minimal source node distances, the detection of the valley nodes and the identification of the grid branches to “virtually” switch off are explicitly described. These special tasks that have to be performed are the reason why at all the breadth-first search algorithm is explained in detail in the following.

To make the gained results of this first subprocedure of the condensed grid topology detection algorithm available for the second subprocedure and also for power flow calculations, they have to be stored in an appropriate form. Hence, the condensed transparency matrix has to be enhanced with the following two attribute rows:

- Minimal source node distance row
- Ring number stack row
As its name suggests, the first of these attribute rows stores the determined minimal source node distances of all nodes of the condensed grid topology. Consequently, the elements of this attribute row are only determined for data objects of grid nodes and are empty for data objects of grid branches. The second of these attribute rows contains for each data object representing a grid node or a grid branch a special stack. Such a stack stores the ring number of each ring that is attached to the concerning data object.

To temporarily store the minimal source node distances already determined during the execution of the breadth-first search algorithm, a column vector is needed, which is denominated minimal source node distance vector. This column vector has for each node of the actual grid topology one element as shown in Figure 5.22. In addition, to store the level number on which the algorithm is working at the moment or rather to store the actual source node distance, an integer variable is needed, which is called actual source node distance variable. For counting the detected rings, an integer variable has to be defined, which due to its purpose is denominated as number of detected rings and is initially set to 0. Similarly, a boolean variable is required to store if the whole grid structure is already explored. This boolean variable is called grid discovered boolean and is initialized with True. Before the breadth-first search algorithm is started, the actual source node distance variable and all elements of the minimal source node distance vector are set to -1, what means that they are not yet determined. With the data preparation done, the breadth-first search algorithm can be started:

**Step 1:** Search in the condensed transparency matrix for the infeed branch. Then, search the adjacent node of this infeed branch, which is the source node. Set the value in the corresponding element of the minimal source node distance vector to 0. Finally, mark the infeed branch as Explored and set the actual source node distance variable to 0.

**Step 2:** Search in the minimal source node distance vector for elements that have the same value as the actual source node distance variable. For each such vector element that is found, examine the corresponding row of the condensed transparency matrix by doing the following:

**Step A:** Chose the adjacent branch that appears in the condensed transparency matrix first from left and is not yet explored. Then, search for this branch its other adjacent node. For this other node, check if the value in the corresponding element of the minimal source node distance vector is set to -1:
If this is the case, then set the value in this element of the minimal source node distance vector to the value of the actual source node variable increased by 1. Set the value in the corresponding element of the minimal source node distance row of the condensed transparency matrix to the same value as well. If the grid discovered boolean is not yet set to False, this has to be done.

If this is not the case, then this other node has already been reached and consequently is detected as a valley node of a ring. Thus, the variable storing the number of detected rings has to be increased by 1. As the ring itself is to be identified by the actual number of detected rings, this number has to be stored in the ring number stack of the considered grid branch. In addition, the actual number of detected rings has also to be stored in the ring number stack of the valley node and in the ring number stack of the other adjacent node. Then, the considered branch has to be “virtually” switched off. For that, check all incidence elements of the corresponding column of the condensed transparency matrix:

- If the condensed incidence matrix element is 0, nothing has to be changed.
- If the condensed incidence matrix element has a negative connectivity integer, replace it with the connectivity integer \(-5\) expressing a “virtually” opened branch that is entering the concerning node.
- If the condensed incidence matrix element has a positive connectivity integer, replace it with the connectivity integer \(5\) expressing a “virtually” opened branch that is leaving the concerning node\(^3\).

Finally, mark the considered branch as Explored.

Step B: Repeat Step A until all adjacent branches of the considered node are explored.

Step 3: After all elements of the minimal source node distance vector are examined, check the value of the grid discovered boolean:

\(^3\)That the integers -5 and 5 are chosen as connectivity integers to express a “virtually” opened branch is due to the fact that the integers -4, -3, 3 and 4 are reserved for the possible further development of additional topology determination algorithms and their visualization.
5.3 The Condensed Grid Topology Detection Algorithm

Step A: If the grid discovered boolean is set to True, then in this just completed pass through the minimal source node distance vector, no nodes have been found that are not yet explored. Hence, all nodes of the grid are discovered and herewith the algorithm terminates.

Step B: Otherwise, if the grid discovered boolean is set to False, then in this just completed pass through the minimal source node distance vector, nodes have been found that are not yet explored. Hence, increase the actual source node distance variable by 1, set the grid discovered boolean to True and go back to Step 2.

Within these three steps of the algorithm, all instructions needed to determine the minimal source node distance of every node, to detect every valley node and to identify every grid branch that has to be “virtually” switched off are included. Thereby, Step 2 and Step 3 are repeated until the grid structure is completely discovered. The listed instructions can easily be verified by examining the exemplary grid structure together with the visualized step arrows and the depicted condensed transparency matrix in Figure 5.22.

In Figure 5.23, the result of this described algorithm is presented for the case of the exemplary grid structure: Line 10 and Line 11 are correctly identified as branches that have to be “virtually” switched off. In the corresponding columns of the condensed transparency matrix, which for this reason is pictured as well, it can be seen that their connectivity integers are set to -5 and 5 as it should be for “virtually” switched off branches. For each of these “virtually” switched off branches, the identifier of the attached ring is stored in the corresponding cell of the ring number stack row. Thus, it can be checked that Line 10 is contained in the ring with the number 1 and that Line 11 is contained in the ring with the number 2. In addition, the ring number stacks of these branches and the ring number stacks of their adjacent nodes are also annotated in the picture of the weakly meshed grid structure. For each node of the exemplary grid structure, the determined minimal source node distance is annotated as well.

In the following, the identification of the grid branches that are within a ring structure will show that each of the “virtually” switched off branches is belonging to just one ring. Hence, these special branches are denominated as ring branches. As the depth-first search algorithm will not pass over these ring branches, but turn at their adjacent nodes and pass back towards the source node, these adjacent nodes, namely the valley nodes and the nodes at the other end of the ring branches, get a new meaning: With regard to the ring branches, these nodes act like special end...
Figure 5.23: The exemplary grid structure after the first subprocedure of the condensed grid topology detection algorithm: The ring end nodes are highlighted with cyan color and the ring branches are highlighted with orange color.
nodes. Because of that, they are renamed to ring end nodes and are highlighted with cyan color in Figure 5.23.

2. Identification of the Grid Branches within the “Virtually” Broken Rings

If no rings are detected during the first subprocedure, there are consequently no grid branches within a ring structure that have to be identified and thus the condensed grid topology detection algorithm will terminate without performing this second subprocedure. Otherwise, in the case of existing rings in the actual topology, the first subprocedure identifies all ring end nodes. This makes it possible to identify all branches that are within a “virtually” broken ring by just one second depth-first search cycle through the condensed grid topology. Thereby, besides the basic instructions described in Subsection 5.3.2, additional tasks have to be executed for each type of node. For this purpose, the ring number stacks are used again: For each branch, its allocated ring number stack stores in which rings it is contained. For each node, its allocated ring number stack stores which rings leads through it. In the following, the additional instructions to perform in this depth-first search cycle are described in detail. For reasons of simplicity, the basic instructions are not repeated again.

1. Source node:

- If the source node is reached via the infeed branch, there is nothing to be done in addition to the basic instructions.

- If the source node is reached via a branch on the way upstream, in addition to the basic instructions, copy all ring numbers of the ring number stack of the previously checked node to the ring number stack of the considered branch. Then, compare the ring number stack of the source node with the ring number stack of the previously checked node:

  - If a ring number in the ring number stack of the previously checked node does not appear in the ring number stack of the source node, copy this ring number to the ring number stack of the source node.

  - If the same ring number is found in both stacks that implies that the source node has already been reached via a branch contained in the ring with this ring number. Hence, the considered branch “closes” the concerning ring. For that reason, delete this ring number from the ring number stack of the source node.
Note: When the algorithm terminates, the ring number stack of the source node is empty. The reason is that after a complete depth-first search cycle through the condensed grid structure, all branches that are within a ring are identified and hence all rings are “closed”.

2. Normal node:
   - If a normal node is reached via a branch on the way downstream, nothing has to be done in addition to the basic instructions.
   - If a normal node is reached via a branch on the way upstream, in addition to the basic instructions, copy all ring numbers of the ring number stack of the previously checked node to the ring number stack of the considered branch. Then, copy all ring numbers of the ring number stack of the previously checked node to the ring number stack of the normal node.

3. Junction node:
   - If a junction node is reached via a branch on the way downstream, nothing has to be done in addition to the basic instructions.
   - If a junction node is reached via a branch on the way upstream, in addition to the basic instructions, copy all ring numbers of the ring number stack of the previously checked node to the ring number stack of the considered branch. Then, compare the ring number stack of the junction node with the ring number stack of the previously checked node:
     - If a ring number in the ring number stack of the previously checked node does not appear in the ring number stack of the junction node, copy this ring number to the ring number stack of the junction node.
     - If the same ring number is found in both stacks that implies that the junction node has already been reached via a branch contained in the ring with this ring number. Hence, the considered branch “closes” the concerning ring. For that reason, delete this ring number from the ring number stack of the junction node.

4. End node:
   - If an end node is reached via a branch on the way downstream, there is nothing to be done in addition to the basic instructions.

5. Ring end node:
   - If a ring end node is reached via a branch on the way downstream, nothing has to be done in addition to the basic instructions.
5.3 The Condensed Grid Topology Detection Algorithm

- If a ring end node is reached via a branch on the way upstream, in addition to the basic instructions, copy all ring numbers of the ring number stack of the previously checked node to the ring number stack of the considered branch. Then, compare the ring number stack of the ring end node with the ring number stack of the previously checked node:
  - If a ring number in the ring number stack of the previously checked node does not appear in the ring number stack of the ring end node, copy this ring number to the ring number stack of the ring end node.
  - If the same ring number is found in both stacks that implies that the ring end node has already been reached via a branch contained in the ring with this ring number. Hence, the considered branch “closes” the concerning ring. For that reason, delete this ring number from the ring number stack of the ring end node.

These additional instructions are practically the same for all type of nodes: Actually, only the instructions of the end node type differ completely from the instructions of the other node types. The reason for these similarities between the additional instructions of the different node types is that the search cycle through the grid structure is guided by the basic instructions. The additional instructions only determine the elements in the ring number stacks of the branches and nodes. However, for clarity reasons, it is recommendable to list the additional instructions separately.

By examining Figure 5.24, in which the result of this second subprocedure is presented, the listed instructions can be verified. In the ring number stack row, all elements are determined after the second subprocedure is completed. In addition, the ring number stacks of these branches and the ring number stacks of their adjacent nodes are also annotated in the picture of the exemplary grid structure. For each node of the exemplary grid structure, the determined minimal source node distance is annotated as well. A better visualization of the result of the completed condensed grid topology detection algorithm is presented in Figure 5.25: Each branch within a ring structure is highlighted with the color appropriate for the concerning ring and its line width is thicker than the one of a normal branch. This kind of visualization can be used to present the results of the condensed grid topology detection algorithm in a graphical overview to the human operator.
Figure 5.24: The exemplary grid structure after the condensed grid topology detection algorithm is completed: For each branch the ring number stack is noted.
5.3 The Condensed Grid Topology Detection Algorithm

Figure 5.25: Result visualization of the completed condensed grid topology algorithm: Each branch that is within a ring structure is highlighted with the color appropriate for the concerning ring and has a thicker line width.

5.3.4 Concluding Remarks

The condensing algorithm presented in this chapter establishes a direct and efficient link between base system functions using the transparency matrix and application-oriented functions using a simplified grid model. In addition, the presented condensed grid topology detection algorithm is an important pre-routine for any grid analysis function, since it detects all existing rings and identifies all grid branches that are lying within such a ring. Thus, the condensing algorithm and the condensed grid topology detection algorithm both together guarantee that the information an advanced application function needs is only coming from the transparency matrix, which is the unique data source of the fully transparent DMS. Hence, the transparency matrix, the condensing algorithm and the condensed grid topology detection algorithm work closely together to provide a fully transparent data architecture for a DMS that has base system functions and also advanced application functions like for instance a power flow calculation tool.
Power Grid Models and Grid Topology Detection for a Fully Transparent DMS
6 Power Flow Calculation Method for a Fully Transparent DMS

6.1 Introduction

The goal of this chapter is to develop a power flow calculation method for distribution grids that can be used as an application-oriented calculation tool in a fully transparent DMS. Due to its intended purpose, this power flow calculation method has to have the capability to handle a large variety of distribution grids. Hence, it is not the aim to develop a power flow calculation method that solves the power flow equations for one given specific grid topology in the best possible way. The aim is rather to develop a power flow calculation method that can determine the unknown quantities for any possible actual topology of a typical distribution grid. As shown in Figure 2.3, the possible topology of a typical distribution grid contains not only radial feeders but also various rings, whereof some are nested. Even though distribution grids are normally operated as radial tree structures, there could be a grid situation where rings are existing and power flow analysis is required. Hence, a power flow calculation method to be developed for a DMS has to be able to handle ring situations as well.

6.2 Power Flow Analysis in a Fully Transparent DMS

With the condensing algorithm described in Section 5.2 and the condensed grid topology detection algorithm presented in Section 5.3, all needed preparatory procedures for power flow analysis are already developed: The condensing algorithm generates the desired node-branch-grid model of the actual topology using the transparency matrix as data base and thus guarantees data consistency and data correctness. This condensed grid model is then analyzed by the condensed grid
topology detection algorithm, which not only detects existing rings but also identifies grid branches that are within such a ring structure and in addition “virtually” switches off all determined ring branches so that power flow calculation can be done. Thus, with these two preparatory procedures, the complete process of power flow analysis can be structured as follows:

1. Condensing algorithm
2. Condensed grid topology detection algorithm
3. Power flow calculation method

The power flow analysis process is slightly different for radial grid topologies and for meshed grid topologies, as the second subprocedure of the condensed grid topology detection algorithm has only to be executed in operation situations when rings are existing. As mentioned above, the aim is to develop the power flow calculation method for distribution grids such that it can be used for both radial and ring situations. Even though it can be assumed that in the case of existing rings some more preparatory tasks are needed, this should cause no problems, since these additional preparations can be solved by one extra subprocedure. Thus, the power flow calculation method should be able to handle radial grid situations as well as ring situations, and the process of power flow analysis is similar for both these situations.

Power flows are due to complex power values described by non-linear equations that cannot be solved analytically. That is why iterative solutions have to be developed and implemented in form of computer programs, as for instance written in [Ande04a]. Many techniques for power flow computations have been developed over the years, whereof the Gauss-Seidel iterative method and the Newton-Raphson method are very popular and most commonly used (for details see for instance [Ande04a] or [GóCo09]). However, according to [Kers02] and [KePh92], iterative techniques commonly used for power flow analysis in transmission grids, which are strongly interconnected power grids, should not be used for power flow analysis in distribution grids due to poor convergence characteristics. Corresponding to [ArGa01], methods based on Newton-Raphson have for most distribution grids poor convergence performance because of their high ratios of series resistance to series reactance, which deteriorate the diagonal dominance of the Jacobian matrix. Hence, an iterative method based on Gauss-Seidel seems to be more adequate for power flow calculation in distribution grids. On the other hand, for weakly meshed distribution grids, a calculation method based on Newton-Raphson may have good
convergence performance. Thus, the aim is not simply to choose one of the well-known iterative solution methods and to modify the chosen one so that the power flow calculation converges very well for a couple of specific distribution grids. The objective is rather to figure out by means of test examples how the non-linear power flow equations can be solved effectively for any typical distribution grid. In doing so, it has to be considered that the aim mentioned in Subsection 5.3.1 is to calculate the unknown quantities of a given power flow problem, if possible, in the same sequence as the depth-first search strategy passes through the grid structure.

6.3 Sequence of Solving the Power Flow Equations

The aim of this subsection is to determine how the non-linear power flow equations can be solved in a very appropriate way, thereby using the sequence of the depth-first search strategy, if possible. For this purpose, the exemplary grid structure presented in Figure 6.1 is studied. Details about the grid modeling and the assumptions made are reported in Appendix D.1.

This radial exemplary grid consists of eight grid nodes and eight grid branches, whereof one is an infeed branch, namely Infeed A. Since it is assumed that there is no distributed generation, only Infeed A is energizing this grid. Consequently, Node A, to which Infeed A is connected, is the only source node and has thus to take the role of the slack bus, also called reference bus (see for instance [Ande04a] or [GõCo09]). All other nodes of this grid structure are load nodes supplying their connected loads with the energy needed. For power flow analysis, such load nodes have to be modeled as PQ-buses (for definition see also [Ande04a] or [GõCo09]), for which the active and reactive power values are specified or known by measurements. For example, the specified complex power value $S_{SpecC}$ consumed by the load of Node C is given by

$$S_{SpecC} = P_{SpecC} + j Q_{SpecC},$$

where $P_{SpecC}$ is the specified active power and $Q_{SpecC}$ is the specified reactive power. Assuming that the complex voltage $U_B$ of Node B is given or rather already determined, the complex voltage $U_C$ of Node C can be calculated by

$$U_C = U_B - Z_{BC} \cdot I_{BC},$$
Figure 6.1: Radial exemplary grid consisting of eight grid nodes and eight grid branches shows how the complex node voltages, the complex line currents and the complex power values are associated to their respective grid node or grid branch.

where $Z_{BC}$ is the series impedance of Line BC and $I_{BC}$ is the complex line current flowing over this line from Node B to Node C. Using the Kirchhoff’s current law (KCL), the complex line current $I_{BC}$ can be determined with

$$I_{BC} = I_{CE} + I_{CF} + I_C,$$  \hspace{1cm} (6.3)

where $I_{CE}$ and $I_{CF}$ are the complex line currents flowing from Node C to Node E respectively to Node F. As the current $I_C$ is the load current of Node C, it can be written as
Using equations (6.3) and (6.4), the expression for the complex node voltage $U_C$ of equation (6.2) can be rewritten as

$$U_C = U_B - Z_{BC} \cdot \left( I_{CE} + I_{CF} + \frac{S_{specc}}{U_C^*} \right),$$

where on the right side of the equation the complex conjugate of the node voltage $U_C$ appears. Thus, the complex node voltage $U_C$ is mapped to itself by equation (6.5). In mathematics, such a function is denominated as fixed-point problem, whose solution has to be iteratively obtained by starting from an initial value (see for instance [BuDo05]). As the Gauss-Seidel method is an iterative technique, which sequentially updates for each node its complex voltage, it is the appropriate method to solve the fixed-point problems of the complex node voltages. The sequence in which the Gauss-Seidel method updates the unknown quantities is thereby determined by the sequence of the depth-first search strategy: By a depth-first search cycle, the complex line currents are updated on the way upstream as explained in Subsection 5.3.2. Using this sequence to solve the power flow equations is very similar to the forward and backward sweep method proposed in [Kers02], which is based on the ladder network theory. The main difference between the method proposed by [Kers02] and the power flow calculation method to be developed in this thesis is that the former method updates in one backward sweep not only the complex current values but also the complex voltage values. This has as a consequence that the complex node voltage of a junction node is more than once updated during one sweep, what seems to be a shortcoming of this method. Hence, the method proposed here is based more on the original Gauss-Seidel iterative method. The idea is that the load currents are determined by the given complex power values and by the complex node voltages calculated in the previous depth-first search cycle or by initially set voltage values. In the next depth-first search cycle, the complex node voltages are updated on the way downstream using the before determined complex line currents. The power flow calculation method thus can be structured in the following three subprocedures:

1. Initializing the start values of the complex node voltages
2. Updating of the complex line currents by a depth-first search cycle
3. Updating of the complex node voltages by a depth-first search cycle

As the first subprocedure constitutes the initialization, it has to be done only once per power flow analysis. In contrast, the second and third subprocedure have to be repeated until the convergence criterium is fulfilled, which will be defined below.

With the three subprocedures of the power flow calculation method defined, all subprocedures for power flow analysis are known. The complete process of such a power flow analysis is schematically pictured in Figure 6.2 using different colored arrows to illustrate which subprocedures have to be executed in which situation. As mentioned in Section 6.2 and as can be seen in Figure 6.2, the process is slightly different for grid situations when rings are existing. In ring situations, besides the second subprocedure of the condensed grid topology detection algorithm an additional subprocedure has to execute a particular data preparation so that the power flow calculation method can be started also for this case. Before this special subprocedure will be explained and developed, the three subprocedures of the power flow calculation method are described in detail for radial grid situations.

6.4 The Procedure of the Power Flow Calculation Method

To perform the requested Gauss-Seidel iterations, the power flow calculation method consists of three subprocedures, as elaborated in Section 6.3. Each of these subprocedures has to execute an assigned task for which it needs to access and to store variables like the complex line currents or the complex node voltages. Thus, these at the beginning unknown quantities have to be stored in an appropriate form, namely in the condensed transparency matrix, which for this purpose has to be enhanced with some additional attribute rows. These additional attribute rows are briefly explained in Appendix D.2.

6.4.1 Initializing the Start Values of the Complex Node Voltages

To start the Gauss-Seidel iterative method, a first guess or rather an appropriate start value has to be determined for each nonlinear power flow equation. One possibility to select the start values for all unknown node voltages is to set them on the same value as the reference bus has. Thus, for each load node $k$ of the
Figure 6.2: The schematically pictured complete process of a power flow analysis illustrates which subprocedures have to be executed for radial grid situations and for ring situations and which ones have to be executed only for ring situations.
regional distribution grid to be analyzed, its voltage magnitude \( U_k \) is set to

\[
U_k^0 = U_0 ,
\]

where \( U_0 \) is the voltage magnitude of the reference bus, which is equal to the nominal voltage magnitude of the concerned regional distribution grid. The superscript of the voltage magnitude on the left side indicates the iteration number, so 0 stands for the initialization. As the voltage angle of the reference bus is set to 0 by definition, the voltage angles \( \theta_k \) of all load nodes \( k \) are set to 0 too:

\[
\theta_k^0 = 0
\]

This kind of initialization, which is denominated as *flat start initialization*, is often chosen when no a priori knowledge of the voltage magnitudes of the load nodes are available. However, as the condensed grid topology detection algorithm determines for each node its source node distance, a certain knowledge of the voltage magnitudes is already available: In a radial grid topology, where besides the source node only load nodes exist, a load node that has a higher source node distance than another load node consequently has a lower voltage magnitude than this other node. Thus, the start voltage magnitude of each load node can be set as a linear function of its source node distance \( d_{SN_k} \) and of a voltage drop \( U_{drop} \):

\[
U_k^0 = U_0 - d_{SN_k} \cdot U_{drop}
\]

The value of the voltage drop \( U_{drop} \) has yet to be chosen. Power flow simulations on Test Distribution Grid 2 presented in Appendix B.2 have shown that a good range for the voltage drop \( U_{drop} \) is about 0.8 % to 1.2 % of the nominal voltage magnitude:

\[
0.008 \cdot U_0 \leq U_{drop} \leq 0.012 \cdot U_0
\]

The conducted power flow simulations have proven that an adequately set voltage drop can reduce the amount of needed iterations. Even if the voltage drop is set too high or too low, the number of needed iterations does not get higher than the number of iterations needed for a flat start initialization. Hence, it is at least as
good or even better to set for each node its initial voltage magnitude as a function of the source node distance than to commence with a flat start initialization.

### 6.4.2 Updating of the Complex Line Currents by a Depth-First Search Cycle

The second subprocedure of the method has the task to update all complex current values by a depth-first search cycle through the condensed grid topology. The determined complex current values, namely the complex load currents $I_k$ and the complex line currents $I_{kl}$, are stored in the designated rows of the condensed transparency matrix (see Appendix D.2 for more details). In addition, each node needs a variable to store the added up complex line currents of the already examined branches. As the currents of these already examined branches are flowing downstream to the corresponding adjacent nodes, they are like outflows from the considered node. Hence, the variable to store of each node $k$ its “outflowing” currents is denominated as node outflow current $I_{Oflw_k}$. In Figure 6.3, this principle of the node outflow current $I_{Oflw_k}$ is schematically visualized.

Since this second subprocedure is executed in alternation with the third subprocedure until the convergence criterion is fulfilled, two integers are designated for each cycle instead of using literal expressions for the node and branch markers. For the depth-first cycle of the second subprocedure, the following integers are used:

- Instead of *Explored downstream* the integer $[1]$ is used
- Instead of *Explored upstream* the integer $[2]$ is used

In the following, this depth-first search cycle to update all complex load currents and all complex line currents is described in detail for the $\nu$-th iteration run. Thereby, the relevant power flow quantities of the lines and the nodes are written with superscripts to indicate the actual iteration number $\nu$, which has to be increased by 1 every time before a new iteration run doing a depth-first search cycle is started. In addition, the node outflow current variable $I_{Oflw_k}$ of each node $k$ has to be set to 0. All calculation instructions for updating and storing the complex currents have to be performed on the way upstream of the depth-first search cycle. Consequently, if a node is reached on the way downstream, only the basic instructions have to be performed (These basic instructions are already reported in Subsection 5.3.2 and are not repeated here). In contrast, on the way upstream, for
each type of node, the calculation instructions for updating the complex currents and the basic instructions for the cycling through the condensed grid topology are reported. The reason is that on the way upstream the basic instructions and the additional calculation instructions are so closely integrated that the latter cannot be well described without the former.

1. **Source node:**

   - If the source node \( k \) is reached via the infeed branch \( jk \), there is nothing to be done in addition to the basic instructions.
   
   - If the source node \( k \) is reached via a branch \( kl \) on the way upstream, increase the complex value of its node outflow current with the complex
current value of the branch $kl$:

$$I_{Oflw_k} = I_{Oflw_k} + I'_v$$  \hspace{1cm} (6.10)

Then, chose the adjacent branch $km$ that appears in the condensed transparency matrix first from left and is not yet explored. Follow this branch $km$ downstream to the next node $m$. Finally, mark the chosen branch $km$ with [1] and change the traversing direction marker to Downstream.

- If the source node $k$ is reached via a branch $kl$ on the way upstream and if there is no branch that is not yet explored, copy the complex value of its node outflow current to the complex current value of the infeed branch $jk$:

$$I'_{jk} = I_{Oflw_k}$$  \hspace{1cm} (6.11)

Mark both the source node $k$ and the infeed branch $jk$ with [2]. Here-with, the algorithm terminates.

2. **Normal node:**

- If a normal node $k$ is reached via a branch $jk$ on the way downstream, nothing has to be done in addition to the basic instructions.

- If a normal node $k$ is reached via a branch $kl$ on the way upstream, increase the complex value of its node outflow current with the complex current value of the branch $kl$:

$$I_{Oflw_k} = I_{Oflw_k} + I'_v$$  \hspace{1cm} (6.12)

Then, determine for the normal node $k$ the value of its complex load current:

$$I'_k = \frac{S_{Spec_k}}{(U'_k - 1)^*}$$  \hspace{1cm} (6.13)

Store the value of this complex load current increased with the complex node outflow current of the normal node $k$ in the complex current value of the other adjacent branch $jk$:

$$I'_{jk} = I'_k + I_{Oflw_k}$$  \hspace{1cm} (6.14)

Follow this other adjacent branch $jk$ further upstream to the next node $j$. Finally, mark both the normal node $k$ and this branch $jk$ with [2].

**Note:** Actually, the value of the complex node outflow current on the right side of equation (6.12) is still 0 and could thus be omitted. But to emphasize the principle of the node outflow current, this is not done here.
3. Junction node:

- If a junction node \( k \) is reached via a branch \( jk \) on the way downstream, nothing has to be done in addition to the basic instructions.

- If a junction node \( k \) is reached via a branch \( kl \) on the way upstream, increase the complex value of its node outflow current with the complex current value of the branch \( kl \):

\[
I_{Oflw_k} = I_{Oflw_k} + I_{\nu}^{kl}
\]  

(6.15)

Then, chose the adjacent branch \( km \) that appears in the condensed transparency matrix first from left and is not yet explored. Follow this branch \( km \) downstream to the next node \( m \). Finally, mark the chosen branch \( km \) with \( 1 \) and change the traversing direction marker to \( Downstream \).

- If a junction node \( k \) is reached via a branch \( kl \) on the way upstream and if there is no branch that is not yet explored, increase the complex value of its node outflow current with the complex current value of the branch \( kl \):

\[
I_{Oflw_k} = I_{Oflw_k} + I_{\nu}^{kl}
\]  

(6.16)

Then, determine for the junction node \( k \) the value of its complex load current:

\[
I_{\nu}^{k} = \frac{S_{Speck}^*}{(U_{\nu}^{k} - 1)^*}
\]  

(6.17)

Store the value of this complex load current increased with the complex node outflow current of the junction node \( k \) in the complex current value of the branch \( jk \) that leads further upstream:

\[
I_{\nu}^{jk} = I_{\nu}^{k} + I_{Oflw_k}
\]  

(6.18)

Follow this branch \( jk \) further upstream to the next node \( j \). Finally, mark both the junction node \( k \) and this branch \( jk \) with \( 2 \).

4. End node:

- If an end node \( k \) is reached via a branch \( jk \) on the way downstream, first determine the value of the complex load current:

\[
I_{\nu}^{k} = \frac{S_{Speck}^*}{(U_{\nu}^{k} - 1)^*}
\]  

(6.19)
Then, store the value of this complex load current increased with the complex node outflow current of the end node $k$ in the complex current value of this branch $jk$:

$$ I_{νjk} = I_{νk} + I_{Outw_k} \quad (6.20) $$

Return back on this branch $jk$ to the previously considered node $j$ being located upstream. Finally, mark both the end node $k$ and this branch $jk$ with $[2]$ and change the traversing direction marker to $\textit{Upstream}$.

**Note:** Actually, the value of the complex node outflow current on the right side of equation (6.20) is 0 as initialized and could thus be omitted. But to emphasize the principle of the node outflow current, this is not done here.

These additional instructions to perform on the way upstream are very similar for the different type of nodes, because the search cycle through the condensed grid structure is guided by the basic instructions presented in Subsection 5.3.2. The additional instructions concern only the power flow calculation, namely the updating of the complex line currents and the updating of the complex load currents. Hence, these calculations have to be carried out according to the Kirchhoff’s current law (KCL) and consequently are similar for all type of nodes. These calculation instructions can be easily verified for example by examining Figure 6.1, Figure 6.3 or also Figure 5.17.

### 6.4.3 Updating of the Complex Node Voltages by a Depth-First Search Cycle

The task of the third subprocedure of the distribution power flow method is to determine the $ν$-th updates of all complex node voltages $U_k$ of the condensed grid topology. Together with the $ν$-th updates of the complex current values, which are already determined in the second subprocedure and are stored in the designated rows of the condensed transparency matrix (see Appendix D.2 for details), the complex power values can be updated using the $ν$-the updates of the concerning complex node voltage $U_k$ and of the concerning complex load current $I_k$. Since for each PQ-node its complex power value $S_{Spec_k}$ is specified, its calculated update of the complex power value $S_{Calc_k}$ can be compared with it. The difference between the specified complex power value and the calculated complex power value provides for each load node the input to the required convergence criterion: When for
all load nodes of the condensed grid topology the respective difference is smaller than a given value, the entire power flow calculation has converged, and thus, the distribution power flow calculation method can be terminated, like schematically depicted in Figure 6.2. To store if an additional iteration run is necessary, a boolean variable is required, which is denominated *Iteration needed*.

As it is the case for the second subprocedure, two integers are designated also for the third subprocedure, namely the following ones:

- Instead of *Explored downstream* the integer 3 is used
- Instead of *Explored upstream* the integer 4 is used

In the following, the depth-first search cycle to update all complex node voltages and all complex power values is described in detail for the $\nu$-th iteration run. For this purpose, the relevant power flow quantities are written with superscripts to indicate the actual iteration number. All calculation instructions for updating and storing of the complex node voltages and the calculated complex power values have to be performed on the way downstream of the depth-first search cycle. As the determination of the complex node voltage of each node only depends on the power flow variables and parameters of its adjacent grid branch and its adjacent grid node that are located upstream, the calculation instructions are identical for all load nodes. Due to the fact that load nodes are either normal nodes, junction nodes or end nodes, the calculation instructions of these different node types can be explained together, especially since the basic instructions, which are already described in Subsection 5.3.2, are not repeated here. Only the source node type is different, since it constitutes the reference node energizing the considered distribution grid. Thus, the additional calculation instructions to perform on the way downstream are explained separately for source nodes and load nodes.

1. **Source node:**

   - If the source node $k$ is reached via the infeed branch $jk$, in addition to the basic instructions, calculate the complex power that is provided by this infeed:

   $$ S_k^\nu = - (U_k^\nu \cdot I_{jk}^\nu)^* $$

   (6.21)

   **Note 1:** The complex node voltage $U_k$ does not need a superscript to indicate the actual iteration number. The reason is that the source node $k$ is the reference node, whose voltage value is given by the nominal voltage value of the considered regional distribution grid.
Note 2: The minus sign in the calculation of the complex power expresses that the complex power of the source node energizes the connected distribution grid; it is not consumed by the source node itself.

- If the source node \( k \) is reached via a branch \( kl \) on the way upstream, nothing has to be done in addition to the basic instructions.

2. Load node (normal node, junction node or end node):

- If a load node \( k \) is reached via a branch \( jk \) on the way downstream, in addition to the basic instructions, determine the complex voltage of this load node \( k \) by using the complex voltage of the previously checked node \( j \) being located upstream:

\[
U_k^\nu = U_j^\nu - Z_{jk} \cdot I_{jk}^\nu
\]  
(6.22)

Then, calculate the complex power that is consumed by the load of this node \( k \):

\[
S_{Calc}^\nu_k = U_k^\nu \cdot (I_k^\nu)^* 
\]  
(6.23)

Compute the difference between the specified complex power value and the calculated complex power value:

\[
S_{Diff}^\nu_k = S_{Spec}^\nu_k - S_{Calc}^\nu_k
\]  
(6.24)

Finally, check if this power value difference is within the allowed tolerance constraints given by the equations (D.2) and (D.3). If the constraints are fulfilled, then the complex node voltage \( U_k \) and the complex load current \( I_k \) of the node \( k \) have converged. Otherwise, if at least one more iteration is needed, the iteration needed boolean has to be set to \( True \), in case this is not yet done.

Note: The calculation of the complex node voltage of node \( k \) using the already updated complex node voltage of the previously checked node \( j \) means that the here used iteration scheme is the one of Gauss-Seidel and not the Gauss iteration scheme (see for instance [Ande04a] for further explanations).

- If a load node \( k \) is reached via a branch \( kl \) on the way upstream, nothing has to be done in addition to the basic instructions.

Note: Actually, only normal nodes and junction nodes can be reached on the way upstream. By definition, an end node cannot be reached on the way upstream. However, as the focus is not on the basic instructions, this difference between end nodes and the other load nodes is ignored.
After the depth-first search cycle of this third subprocedure is completed, the algorithm checks if the iteration needed boolean is set to \textit{True}. If this is the case, the iteration procedure is restarted with the second subprocedure updating the complex current values again. Otherwise, the power flow calculation has converged and the distribution power flow method can be terminated by exporting or storing the obtained results. These calculation instructions can easily be verified for example by examining Figure 6.1.

\section{Power Flow Calculation for Ring Situations}

As already stated in Subsection 6.3, in cases when rings are detected by the condensed grid topology detection algorithm, particular data preparation is required before the distribution power flow calculation method can be started. This required data preparation is carried out by an additional subprocedure, which is only executed for detected ring situations as pictured in Figure 6.2, where the process chart of the complete power flow analysis is sketched. Before describing the required data preparation and the needed additional instruction steps, the basic idea of how the power flow quantities can be determined in ring situations will be elaborated in the following subsection.

\subsection{Principal Approach to Power Flow Calculation for Ring Situations}

The principal approach to power flow calculation for ring situations is elaborated by enhancing the grid structure of Figure 6.1 with two additional lines, namely Line FJ and Line HJ, such that the resulting grid structure contains a ring. Figure 6.4 presents the resulting grid structure, whereby those grid nodes and grid branches that are contained in the ring are marked by blue color.

To determine all power flow quantities in cases of weakly meshed grid topologies, all rings have to be “virtually” broken as already stated in Subsection 5.3.3. The current which actually would flow over such a “virtually” broken grid branch can be substituted by two equivalent currents at both ends of this grid branch, whereby one of these equivalent currents flows into one ring end node and the other equivalent current flows out of the other ring end node (see Subsection 5.3.3 for the definition of ring end nodes). This idea of partitioning the current into two so
Figure 6.4: The enhanced exemplary grid structure contains one ring: Grid nodes and grid branches that are in this ring are emphasized with medium blue color.

called equivalent injection currents is visualized for the enhanced exemplary grid structure in Figure 6.5, where the ring end nodes Node H and Node J are highlighted with cyan color and the equivalent injection currents $I_{HJ_R}$ are drawn in pink color. Thereby, the second subscript $R$ denotes that the considered current is equivalent to the ring branch current $I_{HJ}$ of the “virtually” broken ring branch (see Subsection 5.3.3 for the definition of ring branches). The idea to use equivalent injection currents came like naturally when the starting point of the power flow calculation for ring situations was looked for. However, this concept of equivalent injection currents is a quite often used approach to determine the power flow quantities in a weakly meshed distribution grid, so for instance in [LuSe90], [LiTe96], [ChLi00] and [ChTa01]. The fact that this concept of equivalent injection currents is fairly frequently used for power flow calculation in weakly meshed distribution
grids can be evaluated as proof for its appropriateness.

Depending if the equivalent injection current is flowing into or out of the ring end node, it has either to be added or to be subtracted from the complex load flow current of the relevant node. Using the Kirchhoff’s current law (KCL) for the ring end nodes Node H and Node J of the enhanced exemplary grid structure the following equations hold:

\[
I_{EH} = I_H + I_{HJ} = I_H + I_{HJ} \quad (6.25a)
\]
\[
I_{FJ} = I_J - I_{HJ} = I_J - I_{HJ} \quad (6.25b)
\]

The values of the complex load currents \( I_H \) and \( I_J \) are given by equation (6.4), whereby it does not matter whether the complex node voltages \( U_H \) and \( U_J \) are
already converged to their final values or not. The searched complex current values $I_{EH}$ and $I_{FJ}$ of the grid branches being located adjacently upstream of the ring end nodes can thus easily be determined given that the value of the injection current $I_{HJR}$ or rather the value of the ring branch current $I_{HJ}$ is known. Using the complex node voltages $U_H$ and $U_J$ of the ring end nodes, the value of the ring branch current $I_{HJ}$ is given by

$$I_{HJ} = \frac{U_H - U_J}{Z_{HJ}}.$$  \hspace{1cm} (6.26)

However, the critical point of this injection current method is that at the beginning of the Gauss-Seidel iterations the complex node voltages of the adjacent ring end nodes are not yet determined or rather have not yet converged to their final values. Thus, using equation (6.26) would result in an incorrect value for the ring branch current $I_{HJ}$ respectively for the injection current $I_{HJR}$, which in turn would lead first to wrong results for the complex line currents of grid branches within the relevant ring structure and then to wrong results for the complex node voltages of the adjacent nodes. Consequently, the Gauss-Seidel iterations cannot converge to the right solution.

In view of the above, the calculation of the complex ring branch current $I_{HJ}$ has to be done in a different way. Since each such complex ring current per definition belongs to a ring structure, the usage of the mesh current method, which is besides the branch current method one of the two general methods mostly used for circuit analysis (see for instance [JoJo89]), is almost predestined. Thus, the ring branch current to be determined is equal to the mesh current $I_m$ and in the case of the enhanced exemplary grid structure is given by

$$I_{HJ} = I_m.$$  \hspace{1cm} (6.27)

The associated mesh passes through all grid branches of the considered ring structure, as can be seen in Figure 6.6. Applying Kirchhoff’s voltage law (KVL) around this mesh of the enhanced exemplary grid structure gives

$$U_{CE} + U_{EH} + U_{HJ} - U_{FJ} - U_{CF} = 0.$$  \hspace{1cm} (6.28)

Rearranging equation (6.28) such that only the ring branch voltage $U_{HJ}$ is on the left side of the equation leads to the following expression:
Figure 6.6: The mesh associated to the ring current $I_{HJ}$ or rather to the mesh current $I_M$ passes through all grid branches that are part of the considered ring.

\[
U_{HJ} = (U_{CF} + U_{FJ}) - (U_{CE} + U_{EH})
\]  

(6.29)

(6.30)

To visualize these so called left fork arm and right fork arm of the “virtually” broken ring structure, the grid branches belonging to them are highlighted in blue respectively in green color in Figure 6.6. There, it can be seen that the left fork arm contains the grid branches Line CE and Line EH. Line CF and Line FJ are contained in the right fork arm. The values of the complex line voltages of equation (6.29) can be expressed by their associated line currents and series impedances. Thus, the following equation holds:

\[
U_{HJ} = \left( \frac{Z_{CF}}{U_{CF}} \cdot I_{CF} + \frac{Z_{FJ}}{U_{FJ}} \cdot I_{FJ} \right) - \left( \frac{Z_{CE}}{U_{CE}} \cdot I_{CE} + \frac{Z_{EH}}{U_{EH}} \cdot I_{EH} \right)
\]  

Applying Kirchhoff’s current law (KCL) at Node E and at Node F and using
6.5 Power Flow Calculation for Ring Situations

Equations (6.25a) and (6.25b), all line currents can be expressed in dependence of the load currents and the ring branch current \( I_{HJ} \). Thus, the ring branch voltage can be written only in dependence of the load currents and the ring branch current:

\[
U_{HJ} = \left( \frac{Z_{CF} \cdot (I_F + I_J - I_{HJ})}{U_{CF}} + \frac{Z_{FJ} \cdot (I_J - I_{HJ})}{U_{FJ}} \right) - \left( \frac{Z_{CE} \cdot (I_E + I_H + I_{HJ})}{U_{CE}} + \frac{Z_{EH} \cdot (I_H + I_{HJ})}{U_{EH}} \right) \tag{6.31}
\]

Rearranging equation (6.31) by grouping together all terms that have as common factor the ring branch current \( I_{HJ} \) leads to the following equation:

\[
U_{HJ} = - (Z_{CE} + Z_{EH} + Z_{FJ} + Z_{CF}) \cdot I_{HJ} + \left( \frac{Z_{CF} \cdot (I_F + I_J) + Z_{FJ} \cdot I_J}{U_{CF}} \right) - \left( \frac{Z_{CE} \cdot (I_E + I_H) + Z_{EH} \cdot I_H}{U_{CE}} \right) \tag{6.32}
\]

The ring branch voltage \( U_{HJ} \) can be expressed as a function of the ring branch current \( I_{HJ} \) and the ring branch impedance \( Z_{HJ} \), which leads to

\[
Z_{HJ} \cdot I_{HJ} = - (Z_{CE} + Z_{EH} + Z_{FJ} + Z_{CF}) \cdot I_{HJ} + \left( \frac{Z_{CF} \cdot (I_F + I_J) + Z_{FJ} \cdot I_J}{U_{CF}} \right) - \left( \frac{Z_{CE} \cdot (I_E + I_H) + Z_{EH} \cdot I_H}{U_{CE}} \right) \tag{6.33}
\]

By rearranging equation (6.33), the searched algebraic expression for the ring branch current \( I_{HJ} \) results in

\[
I_{HJ} = \frac{\left( \frac{Z_{CF} \cdot (I_F + I_J) + Z_{FJ} \cdot I_J}{U_{CF}} \right) - \left( \frac{Z_{CE} \cdot (I_E + I_H) + Z_{EH} \cdot I_H}{U_{CE}} \right)}{Z_{CE} + Z_{EH} + Z_{HJ} + Z_{FJ} + Z_{CF}} \tag{6.34}
\]

Equation (6.34) can be rewritten in a more compact form by introducing and using the following complex voltage expressions:

\[
\begin{align*}
U'_{CE} & = Z_{CE} \cdot (I_E + I_H) \tag{6.35a} \\
U'_{EH} & = Z_{EH} \cdot I_H \tag{6.35b}
\end{align*}
\]
\[ U'_{CF} = Z_{CF} \cdot (I_F + I_J) \]  
(6.35c)

\[ U'_{FJ} = Z_{FJ} \cdot I_J \]  
(6.35d)

Thereby, the ‘-sign indicates that these voltages would appear over the concerning branches in the case when the ring branch Line HJ does not exist respectively when the ring branch current is exactly equal to 0. Thus, using equations (6.35a) to (6.35d), the ring branch current \( I_{HJ} \) can be rewritten as follows:

\[ I_{HJ} = \frac{(U'_{CF} + U'_{FJ}) - (U'_{CE} + U'_{EH})}{Z_{CE} + Z_{EH} + Z_{HJ} + Z_{FJ} + Z_{CF}} \]  
(6.36)

On closer consideration of equation (6.36), the dependence of the ring branch current on the line voltages and series impedances within the ring structure becomes clear for general ring situations: In the numerator stands the difference between two voltage drops, namely the voltage drop over the right fork arm and the voltage drop over the left fork arm of the “virtually” broken ring structure. Thereby, the voltage drops are determined for the case when the ring branch current is equal to 0. In the denominator, the sum of all series impedances within the considered ring structure – including the series impedance of the ring branch as well – are summed up. Thus, equation (6.36) can be written in the following general and more compact form:

\[ I_{R} = \frac{\sum_{k \in \Omega_{Ri}} U'_{k} - \sum_{j \in \Omega_{Le}} U'_{j}}{\sum_{j \in \Omega_{Le}} Z_{j} + Z_{R} + \sum_{k \in \Omega_{Ri}} Z_{k}} \]  
(6.37)

Thereby, \( j \) indicates the grid branches that are part of the left fork arm, \( k \) indicates the branches of the right fork arm, and \( R \) denotes the ring branch. The set of all grid branches of the left fork arm and the set of all grid branches of the right fork arm are denoted with \( \Omega_{Le} \) and \( \Omega_{Ri} \) respectively. The positive counting direction of the ring branch current \( I_{R} \) is from the ring end node of the left fork arm to the ring end node of the right fork arm, what in the following will be the conventional counting direction. For general ring situations, it may be difficult to determine which fork arm of the considered ring lies left and which lies right. Thus, the rule chosen here is to refer to that fork arm which due to the identifiers of its grid branches is examined first during the depth-first search cycle as the left one.
Accordingly, the fork arm that is examined as the second one is referred to as the right one.

As the value of the ring branch current has to be determined during the process of the Gauss-Seidel iterations, it has to be calculated in each depth-first search cycle when the complex line currents are updated. For the $\nu$-update of the ring branch current the following equation holds:

$$I_R^\nu = \frac{\sum_{k \in \Omega_{Ri}} U_k^{\nu-1} - \sum_{j \in \Omega_{Le}} U_j^{\nu-1}}{\sum_{j \in \Omega_{Le}} Z_j + Z_R + \sum_{k \in \Omega_{Ri}} Z_k} \quad (6.38)$$

The question is now how the difference of the voltage drops in the numerator can be calculated when updating the ring branch current. Since these voltage drops are not really appearing over the fork arms of the considered ring structure, they can not be calculated only as functions of the complex line currents. But comparing equation (6.31) with equations (6.35a) to (6.35d) shows that each complex line voltage of a grid branch belonging to one of the fork arms can be expressed as function of the complex line voltage that would appear without the ring branch current and as function of the ring branch current:

$$U_j^{\nu-1} = U_j^{\nu-1} + Z_j \cdot I_R^{\nu-1} \quad \text{for } j \in \Omega_{Le} \quad (6.39a)$$
$$U_k^{\nu-1} = U_k^{\nu-1} - Z_k \cdot I_R^{\nu-1} \quad \text{for } k \in \Omega_{Ri} \quad (6.39b)$$

Bringing the complex line voltages $U_j^{\nu-1}$ and $U_k^{\nu-1}$ on the left side of their respective equation and expressing the really appearing complex line voltages $U_j^{\nu-1}$ and $U_k^{\nu-1}$ as function of their respective complex line current leads to the following equations:

$$U_j^{\nu-1} = Z_j \cdot I_j^{\nu-1} - Z_j \cdot I_R^{\nu-1} \quad \text{for } j \in \Omega_{Le} \quad (6.40a)$$
$$U_k^{\nu-1} = Z_k \cdot I_k^{\nu-1} + Z_k \cdot I_R^{\nu-1} \quad \text{for } k \in \Omega_{Ri} \quad (6.40b)$$

The sums of the line voltages of all grid branches belonging to the left fork arm and to the right fork arm respectively – so the voltage drops over the complete left fork arm and the complete right fork arm respectively – are given by
\[
\sum_{j \in \Omega_{Le}} U_{j}^{\nu-1} = \sum_{j \in \Omega_{Le}} Z_j \cdot I_{j}^{\nu-1} - I_{R}^{\nu-1} \cdot \sum_{j \in \Omega_{Le}} Z_j \quad \text{for} \ j \in \Omega_{Le}; \quad (6.41a)
\]
\[
\sum_{k \in \Omega_{Ri}} U_{k}^{\nu-1} = \sum_{k \in \Omega_{Ri}} Z_k \cdot I_{k}^{\nu-1} + I_{R}^{\nu-1} \cdot \sum_{k \in \Omega_{Ri}} Z_k \quad \text{for} \ k \in \Omega_{Ri}. \quad (6.41b)
\]

Finally, the insertion of equations (6.41a) and (6.41b) into equation (6.38) results in the searched function for the ring branch current:

\[
I_{R}^{\nu} = \frac{\sum_{k \in \Omega_{Ri}} Z_k \cdot I_{k}^{\nu-1} - \sum_{j \in \Omega_{Le}} Z_j \cdot I_{j}^{\nu-1} + I_{R}^{\nu-1} \cdot \left( \sum_{j \in \Omega_{Le}} Z_j + \sum_{k \in \Omega_{Ri}} Z_k \right)}{\sum_{j \in \Omega_{Le}} Z_j + Z_R + \sum_{k \in \Omega_{Ri}} Z_k} \quad (6.42)
\]

Thus, the new update of the ring branch current depends on its previous value calculated during the \((\nu - 1)\)-th iteration run of the Gauss-Seidel method. In that algebraic form it can be used for the depth-first search cycle.

In the following subsections, the data preparation subprocedure and the additional calculation steps to determine the ring branch currents are explained in more detail.

### 6.5.2 Data Preparation for Power Flow Calculation with Rings

The task of this data preparation subprocedure is to provide for all ring end nodes the needed power flow variables in such a form that the ring branch current \(I_{R}\) can be updated according to equation (6.42). As in the nominator of this equation the sum of all series impedances of the considered ring appears and as in the numerator the sum of all series impedances of the left fork arm and the sum of all series impedances of the right fork arm are included, the corresponding sums of series impedances have to be made available for all calculation instructions performed at a ring end node. Hence, the task of this data preparation is to sum up the relevant series impedances such that equation (6.42) can be solved for each ring end node.

Since, as described in Subsection 5.3.3, all existing rings are already marked after the condensed grid topology detection algorithm is run, the summing up of the series impedances that are within a ring structure can easily be done by an additional depth-first search cycle through the condensed grid topology. Thereby the corresponding impedance sums are not only determined for the ring end nodes but
for all grid nodes that are part of a ring structure. The idea is the following: A special so-called ring impedances matrix is prepared that has for every grid node a row and for every ring detected during the condensed grid topology detection algorithm a column. At the beginning, the values in each element of this ring impedances matrix are set to 0 by default. Then, during the depth-first search cycle, it is checked for each grid node if it is part of a ring structure. For each ring in which the considered grid node is contained, the corresponding element of the ring impedances matrix has then to be determined: The series impedances of all grid branches that are lying in the relevant ring and are located upstream of the considered grid node have to be added together and to be stored in this corresponding matrix element. So, the difference between the impedance sum of a grid node and its adjacent grid node located downstream is the series impedance of the connecting grid branch. Hence, for each grid node contained within a ring structure, the searched impedance sum can easily be determined when using the impedance sum of the grid node being located adjacent upstream and the value of the series impedance of the connecting grid branch leading upstream. This means that the value of the impedance sum for a specific grid node can only be calculated when the value of the impedance sum of its neighboring grid node lying upstream in the same ring structure is already determined.

As an immediate consequence of this, it follows that the summing up of the series impedances has to be done on the way downstream of the depth-first search cycle. This in turn has as a consequence that the instructions for the determination of the sum impedances are identical for almost all type of nodes, since this determination of the impedance sum of a grid node within a ring structure only depends on the series impedance of its adjacent grid branch and the impedance sum of its neighboring node being located upstream. Consequently, the calculation instructions for most of the different node types can be explained together, especially since the basic instructions, which are already described in Subsection 5.3.2, have not to be repeated. Only the source node type and the end node type are different since these node types do not need additional calculation instructions:

- For the source node type, this is due to the fact that this node type has no neighboring nodes being located upstream.

- For end nodes, this is due to the fact that they are at the end of their respective distribution feeders (see Subsection 5.3.2), as their name indicates. Thus, end nodes can never be part of a ring structure.

Therefore, no additional calculation instructions have to be reported for the source
node type and the end node type. For the other node types, namely the normal node type, the junction node type and the ring end node type, the additional calculation instructions are identical as mentioned above and are thus explained for all those node types together:

- **Normal node, junction node and ring end node:**

  - If a node \( k \) is reached via a branch \( jk \) on the way downstream, in addition to the basic instructions, check its ring number stack. For each ring \( r \) in which this node \( k \) is contained, the following has to be done:
    
    - Add to the impedance sum \( Z_{j\Sigma} \) of the previously checked node \( j \) being located upstream the series impedance \( Z_{jk} \) of the connecting branch \( jk \) to determine the impedance sum \( Z_{k\Sigma} \) of the considered node \( k \):
      
      \[
      Z_{k\Sigma} = Z_{j\Sigma} + Z_{jk}
      \]
      
      (6.43)

    Then, store the determined impedance sum \( Z_{k\Sigma} \) in the element \([k, r]\) of the ring impedances matrix.

  - If a node \( k \) is reached via a branch \( kl \) on the way upstream, nothing has to be done in addition to the basic instructions.

By checking Figure 6.7, in which the weakly meshed grid structure of Section 5.3 is shown together with the condensed transparency matrix, the annotated ring number stacks and the ring impedances matrix containing the determined sum impedances, the described calculation instructions can easily be verified. It has to be emphasized that the summing up of the series impedances for a specific ring is stopped at its ring end nodes so that for each ring end node the impedance sum of the respective fork arm is correctly determined. The ring branches are not examined in this data preparation subprocedure as their series impedances do not have to be added to any impedance sum. Since ring branches have special connectivity integers with the integers -5 or 5, those can just be ignored when the depth-first search cycle passes through the condensed grid topology.

After the depth-first search cycle of this data preparation subprocedure is completed, all elements of the ring impedances matrix are set correctly. Thus, for each ring end node \( k \) of a ring \( l \), the needed impedances sum can be found in the corresponding element \([k, l]\) of the ring impedances matrix.
Figure 6.7: The exemplary grid structure with the condensed transparency matrix, the annotated ring number stacks and the ring impedances matrix after the data preparation subprocedure is terminated.
6.5.3 Updating of the Complex Line Voltages

For power flow calculation with rings, in addition to the data preparation subprocedure described in Subsection 6.5.2 and the additional determination of the ring branch currents in the depth-first search cycle updating all complex line currents, the depth-first search cycle, which calculates the complex node voltages, has to be enhanced too. The reason is that in equation (6.42), which is used to update the ring branch currents, the voltage drops of the grid branches within the considered ring structure appear in the numerator. Thus, for each ring end node, the relevant voltage drops have to be made available so that its ring branch current can be updated according to equation (6.42).

A very similar idea as elaborated in Subsection 6.5.2 is used here: A voltage drop matrix is introduced that, like the ring impedances matrix, has for every grid node a row and for every detected ring a column. At the beginning of each depth-first search cycle, the values in the elements of the voltage drop matrix are set to 0. During the depth-first search cycle when the complex node voltages of all grid nodes are updated, the voltage drops of the grid branches that are lying within a ring structure are updated as well. These calculations of the voltage drops consequently have to be done on the way downstream. As it is the case with the ring impedances matrix, only the normal node type, the junction node type and the ring end node type need to be considered for the determination of their respective voltage drops. The additional calculations for these node types are the followings:

- Normal node, junction node and ring end node:
  - If a node $k$ is reached via a branch $jk$ on the way downstream, in addition to the basic instructions and the calculation instructions already reported in Subsection 6.4.3, check its ring number stack. For each ring $r$ in which the considered node $k$ is contained, the following has to be done:
    - Determine the voltage drop of this node $k$ by using the voltage drop $U_{j_{\Sigma}r}$ of the previously checked node $j$ being located upstream and by using the complex line current $I_{jk}$ of the connecting branch $jk$ as well as its series impedance $Z_{jk}$:
      \[
      U_{k_{\Sigma}r} = U_{j_{\Sigma}r} + Z_{jk} \cdot I_{jk}
      \]  \hfill (6.44)
      Then, store this voltage drop $U_{k_{\Sigma}r}$ in the element $[k, r]$ of the voltage drop matrix.
If a node $k$ is reached via a branch $kl$ on the way upstream, no additional calculation instructions have to be done.

After the depth-first search cycle for updating the complex node voltages is completed, all elements of the voltage drop matrix are set correctly. During the depth-first search cycle for updating the complex line currents, the values of the voltage drop matrix will be used, namely for the determination of all ring branch currents.

It has to be mentioned that since ring end nodes are load nodes, the calculation instructions to update their complex node voltages are identical with the ones of the other load nodes described in Subsection 6.4.3. Hence, this procedure need not be repeated here.

### 6.5.4 Updating of the Complex Line Currents for Ring End Nodes

With the introduced ring impedances matrix and the introduced voltage drop matrix, the additional determination of the ring branch currents in the depth-first search cycle updating all complex line currents can easily be done. Since the determination of the ring branch currents only concerns the ring end nodes, the calculation instructions for source nodes, normal nodes, junction nodes and end nodes described in Subsection 6.4.2 remain unchanged. Only the calculation instructions for ring end nodes have to be changed. This node type has not been considered in Subsection 6.4.2, since it is only in Subsection 6.5.1 that the principal approach for determining ring branch currents has been elaborated.

During each depth-first search cycle updating all complex line currents, all ring branch currents have to be updated as well by using equation (6.42). Since each ring branch is adjacent to two ring end nodes, it is found twice during the depth-first search cycle: Once over one ring end node and once over the other ring end node. Thus, a flag is needed that has to be set when the ring branch current has been updated. When during the depth-first search cycle a specific ring branch is found for the first time, the ring branch current has to be updated and the so called \textit{Current updated} flag has to be set to \textit{True}. Afterwards, when the other ring end node of the considered ring branch is found, the set flag indicates that the ring branch current has already been updated in this depth-first search cycle.

Concerning the determination of the complex load current and the determination of the complex node outflow current, a ring end node appears like a common node
Figure 6.8: Visualization of node injection current principle: The sum of all injection currents of a ring end node is denominated as its node injection current.

type with the only difference that each ring branch current is like an additional inflow or outflow from the considered ring end node. Since the determination of the ring branch currents is based on the concept of equivalent injection currents, the values of these additional “inflowing” and “outflowing” currents are combined in a variable called node injection current $I_{Inj}$. Because of its name, the counting direction is positive for “inflowing” currents and negative for “outflowing” currents. In Figure 6.8, this principle of the node injection current $I_{Inj}$ and the node outflow current $I_{Out}$ is schematically sketched. Due to this introduced node injection current, the value of the complex node outflow current for a ring end node can be determined as for a common node type. Then, the ring branches are examined to determine the values of their complex node injection currents. In the following, the necessary calculation instructions for the determination of the complex node outflow current and the complex node injection current are described in detail.
• Ring end node:
  
  – If the ring end node $k$ has besides the ring branches just two adjacent grid branches, it appears like a normal node concerning its complex node outflow current. Hence, first determine the value of its complex node outflow current and the value of the complex load current by using the calculation instructions of the normal node type described in Subsection 6.4.2.

  – If the ring end node $k$ has besides the ring branches at least three adjacent grid branches, it appears like a junction node concerning its complex node outflow current. So, first determine the value of its complex node outflow current and the value of the complex load current by using the calculation instructions of the junction node type described in Subsection 6.4.2.

  – If the ring end node $k$ has besides the ring branches only one adjacent grid branch, it appears like an end node concerning its complex outflow current. Consequently, the complex outflow current has to be set to 0. Determine the value of the complex load current by using the calculation instructions of the end node type described in Subsection 6.4.2.

  – Then, when the complex outflow current is determined, consider all ring branches. Since all ring branches have as connectivity integers either a -5 or a 5 stored in the corresponding row of the transparency matrix, they can easily be found. Check for each ring branch $kl$ found, if its flag Current updated is still set to False.

  – If this is the case, the ring branch current has not yet been updated in this depth-first search cycle. Thus, follow the considered ring branch $kl$ to the other ring end node $l$. By doing so, access the series impedance $Z_{kl}$ of the ring branch, the impedance sum $Z_{l\Sigma r}$ and the voltage drop $U_{l\Sigma r}^{\nu-1}$ of the other ring end node $l$. Then, use the value of the ring branch current $I_{r}^{\nu-1}$ from the previous $(\nu-1)$-th iteration run to determine the new update for the ring branch current:

$$ I_{r}^{\nu} = \frac{U_{l\Sigma r}^{\nu-1} - U_{k\Sigma r}^{\nu-1} + I_{r}^{\nu-1} \cdot (Z_{k\Sigma r} + Z_{l\Sigma r})}{Z_{k\Sigma r} + Z_{lk} + Z_{l\Sigma r}} \quad (6.45) $$

Now, set the Current updated flag to True. Since this ring end node $k$ is examined as the first one of the ring, according to the convention introduced in Subsection 6.5.1, it belongs to the left fork arm and its ring branch current is leaving it. Consequently, subtract this ring branch current from the complex node injection current:

$$ I_{lnjk} = I_{lnjk} - I_{r}^{\nu} \quad (6.46) $$
– Otherwise, if the Current updated flag is already set to True, the ring branch current has already been updated in this depth-first search cycle. Consequently, this ring end node $k$ is examined as the second one of the ring. According to the convention introduced in Subsection 6.5.1, the ring end node $k$ thus belongs to the right fork arm and the ring branch current is entering this ring end node. Hence, take the already updated ring branch current $I^\nu_r$ and add this to the complex node injection current:

$$I_{Inj_k} = I_{Inj_k} + I^\nu_r$$  (6.47)

Finally, reset the value of the Current updated flag to False again so it is prepared for the next iteration run.

– Then, when all adjacent ring branches of this ring end node $k$ are examined, the complex injection current is determined. Use the values of the complex load current, the complex node outflow current and the complex node injection current to calculate the complex current value of the branch $jk$ that leads further upstream:

$$I^\nu_{jk} = I^\nu_k + I_{Oflw_k} - I_{Inj}$$  (6.48)

Follow this branch $jk$ further upstream to the next node $j$. Finally, mark both the ring end node $k$ and this branch $jk$ with 2.

By examining Figure 6.5 or Figure 6.8, the described calculation instructions can be verified. It has to be mentioned that actually, for a concrete implementation of this depth-first search cycle updating all complex current values, it is not necessary to provide for each ring end node an additional variable $I_{Inj}$ for storing the injection currents. Instead, the updated ring branch currents can be directly added to or subtracted from the complex node outflow current, whereby in comparison of equation (6.48) and (6.47) respectively with equation (6.48), the sign of the ring branch current has to be changed. Nevertheless, this variable for the node injection current variable is introduced here to emphasize the idea of summing up all ring branch currents of the ring end node in one injection current.

A closer inspection of equation (6.45) shows that all ring branch current updates are still 0 after the first iteration run: Since in the equation of the updated ring branch current the previous updates of the voltage drops and the previous update of the ring branch current appear, which all are initialized to 0, the expression in the numerator becomes 0 for the first iteration step and thus the first update
of each ring branch current becomes 0. Consequently, the values of the currents and voltages resulting from the first iteration run are determined for that radial grid situation which would emerge if the ring branches are not only “virtually” but really opened. Only in the second iteration run, the ring branch currents are updated by using the appropriate voltage drops determined in the first iteration run. So after the second iteration run, the ring branch currents are in general not equal to 0 anymore. It is important to be aware of this fact, since even if the power flow calculation for a ring situation converges fast, it always needs at least one iteration step more than for a power flow calculation of an almost identical but radial grid.

6.6 Concluding Remarks

As mentioned at the beginning of this chapter, the power flow calculation method developed here is intended to be used as an application-oriented calculation tool in a fully transparent DMS. Because of that, the main objective of this calculation method is that it has the capability to handle all realistic possible actual grid topologies that can appear in a typical distribution grid. With the condensed grid topology detection algorithm presented in Subsection 5.3, the principal approach of using injection currents to determine the ring branch currents in Subsection 6.5.1 and the special data preparation subprocedure developed in Subsection 6.5.2, the power flow calculation method developed in this chapter can perform power flow analysis for both radial grid situations and ring situations. The main new contributions of this developed power flow calculation method are that for actual topologies containing rings the same three subprocedures are used as for normal radial grid structures and that the additional calculation steps needed for the determination of the ring branch currents are automatically done.

All information the power flow calculation method needs is either provided by the condensed transparency matrix, which represents the grid model in the desired node-branch-form, or by the condensed grid topology detection algorithm, which detects the grid branches that are within a ring structure. Hence, the power flow calculation method seamlessly fits in the data flow chain: Starting from the transparency matrix, which represents the grid topology such that it is appropriate for SCADA functions, the needed data is provided to the condensing algorithm generating the condensed transparency matrix, which in turn is used first by the condensed grid topology detection algorithm and then by this power flow calcula-
tion method. Since this data flow is neither interrupted, changed nor enhanced by additional data, all data the power flow analysis depends on are coming from the original, unique data source, namely the transparency matrix. As a consequence, data inconsistency or data incorrectness are eliminated. As long as the data which is stored in the transparency matrix is correct and the condensing algorithm, the condensed grid topology detection algorithm as well as the power flow calculation method work correctly, the results of the power flow analysis are correct.

Finally, it can be concluded that the power flow calculation method is a concrete example that demonstrates how a fully transparent DMS provides the whole range from SCADA functions to application-oriented algorithms, thereby assuring an uninterrupted, smooth and consistent data flow from the transparency matrix as original data base to the application-oriented data models.

The developed power flow calculation method has been implemented in the RITOP® process control system (see Chapter 7 for general information about the software implementation). Thus, the correctness of the method has been tested with numerous simulations, which proved that not only the power flow calculation method works correctly but also showed that the implemented transparency matrix, the condensing algorithm and the condensed grid topology detection algorithm operate correctly together. For 10 special grid topologies, the results of the executed power flow calculations are documented in detail in [Ruh09].
7 Software Implementation of the Developed Concepts and Algorithms

7.1 Introduction

The transparency matrix and the condensed transparency matrix derived from it provide optimal bases for applications like the condensing algorithm, the condensed grid topology detection algorithm or the power flow calculation method. But as the transparency matrix is only a conceptual model for the data architecture of a fully transparent DMS, it cannot be implemented one-to-one in a real software solution. Yet the aim is to find a method how the data base of an existing, not fully transparent DMS can be made fully transparent using the conceptual idea of the transparency matrix. More precisely, this method to develop should enable that the transparency matrix can be mapped on the data base of an existing DMS. Thereby, the transparency matrix would still be available: It can be generated out of the data base into the random access memory (RAM) of the DMS, as already mentioned in Subsection 4.2.3 (For further information about RAM technology, the interested reader is referred to [SiGa00]). In the RAM, it provides the basis for the condensing algorithm deriving the condensed transparency matrix, which in turn is the basis for the condensed grid discovery algorithm, the power flow calculation method and all other application-oriented algorithms using a node-branch-grid model.

The software realization of this method will finally give the proof that the presented transparency concept is not only a theoretical idea but can really be turned into an existing DMS to make the latter fully transparent. That the developed algorithms can be implemented into the process control software is then not necessary to prove, as long as the implemented transparency matrix contains the same information as
the conceptual transparency matrix. Since these algorithms use the transparency matrix as the single information source, only the software feasibility of the transparency concept in itself has to be proved. It has already been described in detail in Section 5.2, in Section 5.3 and in Chapter 6 that the developed algorithms and methods, namely the condensing algorithm, the condensed grid topology detection algorithm and the power flow calculation method as such work correctly. Anyhow, it is not the intention of this chapter to present the implemented concepts and algorithms on code level or to discuss an existing DMS in detail. The aim is rather to demonstrate that the concept of the transparency matrix can be implemented into a real distribution control system.

7.2 Mapping the Transparency Matrix on the Data Base of an Existing DMS

Part of the data base of an exemplary state-of-the-art DMS with an object-oriented data architecture is pictured in the upper part of Figure 7.1 (For background knowledge about object-oriented data bases, the interested reader is referred to [Harr00]). The most important data object types of such a DMS are the busbar type, the cable type, the infeed type and the switch type as well as the substation type, which is a meta data object type. Seven data objects of these types are shown, namely two substation types, two busbar types, one cable type and two switch types. They represent two substations each with a single busbar, namely Substation Y with Busbar 1 and Substation Z with Busbar 1, which are connected over the cable Line 1 and the switches Load Switch 1 and Load Switch 2, like shown in the lower part of Figure 7.1.

To store the grid topology, some data types need special attributes expressing connectivity between grid devices or rather indicating the identifiers of the adjacent grid devices. In addition, some of these special attributes have to express, which grid devices are part of the regular grid topology and which ones are just possible connections or junctions. For this exemplary state-of-the-art DMS, it is assumed that the switch data type has three such special attribute data fields, denominated as REG_TOPOLOGY, ADJACENCY_1 and ADJACENCY_2 all being part of the TOPOLOGY data field. The REG_TOPOLOGY data field stores whether the concerning switch belongs to the regular topology or not. Each of the other two attribute data fields contains one entity storing the name or the identifier of one of the two adjacent grid devices. For substation configurations appearing in
Figure 7.1: Of the data base of an exemplary, object-oriented DMS, a portion with seven data objects is illustrated in the upper part. The substation-line-substation configuration to which these seven data objects refer is shown in the lower part.

Test Distribution Grid 1 and in Test Distribution Grid 2, which are presented in Appendix B.1 and in Appendix B.2 respectively, a switch type has either a cable and a busbar as adjacent grid devices or else two busbars. For example, the adjacent devices of Load Switch 1 are Line YZ and Busbar Z, whose identifiers are accordingly stored in the adjacency data fields of Load Switch 1 as shown in Figure 7.1. The challenge, however, is that from a mathematical point of view, a switch type is like an active branch or rather a partition branch (see Appendix A.2). Thus, it is actually presumed to be connected on both sides to a node. But due to the fact that the adjacency data fields store just the adjacent grid devices, this is
not the case for line-busbar-configurations: Instead of a second node, a cable type is stored in one of the adjacency data fields. For instance, in the adjacency data fields of Load Switch 2, the identifier of Line YZ appears besides the identifier of Busbar Y. Still, for SCADA functions like topological coloring or topological interlocking, this way of representing the grid topology by indicating the identifiers of the adjacent grid devices in the designated data fields is efficient as well as sufficient. But for application-oriented algorithms needing a node-branch-grid model another method to represent connectivity between different grid devices is required.

The usage of the partition nodes, which are part of the transparency concept, provides such a method to represent the grid connectivity. Hence, the procedure to generate the transparency matrix out of the data base from an existing DMS has first to create all partition nodes, why it has to be structured in two subprocedures:

1. Generation of the partition nodes
2. Generation of the transparency matrix

This generation of the partition nodes has to be conducted after each configuration change of the DMS when new data objects have been inserted or existing ones have been altered. With the partition nodes or rather the identifiers of them stored in the data base, which for this purpose has to be enhanced with additional attribute data fields, the transparency matrix can be generated into the RAM.

7.2.1 Generation of the Partition Nodes

Partition nodes are mathematical objects that constitute the junctions between adjacent switches and lines. Since they are not referring to an existing grid device, they should not be stored as real data objects into the data base. As they are nodes linking branches, they can be expressed through the connectivity attributes of the concerning adjacent switches and lines. For this purpose, the data objects of the cable data type and the switch data type are enhanced with additional attribute data fields so that the identifiers of the partition nodes can be stored.

The same portion of the data base of the exemplary DMS as shown in Figure 7.1 is sketched in Figure 7.2 again, this time enhanced with the new attribute data fields. Since these new attribute data fields are part of the transparency concept or rather are required to implement the transparency concept into a real DMS, they are denominated as TRSP_TOPOLOGY data fields, whereby the word transparency
appears in abbreviated form. Besides the new attributes, the topology attributes of the original data base are still there, for the idea is not to alter the data base of the existing DMS significantly. The aim is rather to enhance the data base with appropriate additional attributes that are forcing transparency. Consequently, the identifiers of the grid nodes appear in the original adjacency data fields and also in the new transparency topology data fields, as can be seen in Figure 7.2. Nevertheless, this does not cause redundancy problems, as only the subprocedure generating the partition nodes has write access to these new attribute data fields. Anyway, with one exception, the transparency topology data fields are empty before the sub-
procedure that generates the partition nodes is executed. The only exception are constituted by the \textit{LEAVE\_FROM} data fields, which are part of the transparency data fields of the cable data objects. These data fields are reserved to store the counting direction of the cables, what has been denominated as incidence orientation in Subsection 4.2.1. These counting directions can either be chosen freely or they are specified by data sheets of the considered distribution grid; however, they can be manually set by the operator when inserting the data objects of the concerning cables or they are set by a simple manager taking the needed direction information out of the designated data file. As the cable direction provides the information from which busbar to which busbar the considered cable leads, it can be stored by indicating the identifier of one of these two busbars. In this exemplary DMS, the identifier of the busbar from which the considered cable leaves is stored. According to this convention, the concerned transparency data field is denominated as \textit{LEAVE\_FROM} data field. Hereafter, it is assumed that these cable directions are already set. It is also important to note that Figure 7.2 shows the state of the DMS data base \textit{after} the generation of the partition nodes is done.

In the following, the principle of the subprocedure generating the partition nodes is described in a general way, whereby coding details are deliberately omitted for reasons of clarity. This subprocedure to generate the partition nodes – or better to construct their identifiers and to store them in the appropriate attribute data fields – has to execute for every switch data object the following instructions:

\textbf{Step ①}: Check the identifier \textit{ADJ\_IDENT\_1} that is stored in the adjacency data field \textit{ADJACENCY\_1}:

- If this identifier \textit{ADJ\_IDENT\_1} is the identifier of a real grid node – consequently of a busbar data object – make a copy of this identifier and denominate this copy as \textit{ND\_IDENT\_1}. Thereby, the characters \textit{ND} indicate that this new identifier belongs either to a grid node or to a partition node. Then, store this identifier \textit{ND\_IDENT\_1} into the transparency topology data field \textit{NODE\_1}.

- Otherwise, if the identifier \textit{ADJ\_IDENT\_1} does not belong to a real grid node, it is necessarily the identifier of a cable data object. In this case, the associated partition node has to be generated by constructing its identifier \textit{ND\_IDENT\_1} with the following character string operations:

  - Copy from the considered switch the first part of its identifier where the substation name is given to the new identi-
7.2 Mapping the Transparency Matrix on the Data Base of an Existing DMS

The identifier \textit{ND\_IDENT\_1} (e.g. from the identifier “StationZ\_LS1” of Load Switch 1 take the first part “StationZ”).

- Enhance the identifier \textit{ND\_IDENT\_1} with the character string “.PN.” indicating that this will be the identifier of a partition node (e.g. in case of Load Switch 1 this results in “StationZ\_PN.” for the identifier \textit{ND\_IDENT\_1}).

- Then, enhance the identifier \textit{ND\_IDENT\_1} with the second part of the cable identifier \textit{ADJ\_IDENT\_1} where the cable name is given (e.g. from the identifier “LineYZ” of Line YZ copy the second part “YZ”, what leads to “StationZ\_PN\_YZ” for the identifier \textit{ND\_IDENT\_1}).

After these character string operations are completed, copy the generated identifier \textit{ND\_IDENT\_1} to the transparency data field \textit{NODE\_2}. Then, search for the cable data object to which the identifier \textit{ADJ\_IDENT\_1} belongs. For this cable, check if the identifier stored in the transparency data field \textit{LEAVE\_FROM} belongs to a busbar lying in the same substation as the generated partition node is part of (e.g. for “StationZ\_PN\_YZ” as the generated identifier \textit{ND\_IDENT\_1} of the partition node, this is not the case, since “StationY\_BB1” is the identifier stored in this \textit{LEAVE\_FROM} data field):

- If this is the case, the considered cable is leaving the generated partition node and hence the identifier \textit{ND\_IDENT\_1} has to be copied to the first transparency data field \textit{NODE\_1} of this cable, if this is not yet filled (This data field can already be filled when the considered switch has a pair switch that has already been checked.)

- Otherwise, the considered cable is entering the generated partition node. Thus, the identifier \textit{ND\_IDENT\_1} has to be stored in the second transparency data field \textit{NODE\_2} of this cable, if this is not yet filled (This data field can already be filled when the considered switch has a pair switch that has already been checked.)

\textbf{Step 2:} Check the identifier \textit{ADJ\_IDENT\_2} that is stored in the adjacency data field \textit{ADJACENCY\_2}:

- If this identifier \textit{ADJ\_IDENT\_2} is the identifier of a real grid node
– consequently of a busbar data object – make a copy of this identifier and denominate this as ND_IDENT_2. Thereby, the characters ND indicate that this new identifier belongs either to a grid node or to a partition node. Then, store this identifier ND_IDENT_2 into the transparency topology data field NODE_1, if this is not yet filled. Namely, in the case of a busbar coupler switch or a busbar sectionalizer switch, this first transparency topology data field is already filled. Thus, the identifier has to be stored into the transparency topology data field NODE_2.

– Otherwise, if the identifier ADJ_IDENT_2 does not belong to a real grid node, it is the identifier of a cable data object. In this case, the associated partition node has to be generated by constructing its identifier ND_IDENT_2 with the following character string operations:

– Copy from the considered switch the first part of its identifier where the substation name is given to the new identifier ND_IDENT_2 (e.g. from the identifier “StationY_LS2” of Load Switch 2 take the first part “StationY”).

– Enhance the identifier ND_IDENT_2 with the character string “_PN_” indicating that this will be the identifier of a partition node (e.g. in case of Load Switch 2 this results in “StationY_PN_” for the identifier ND_IDENT_2).

– Then, enhance the identifier ND_IDENT_2 with the second part of the cable identifier ADJ_IDENT_2 where the cable name is given (e.g. from the identifier “LineYZ” of Line YZ copy the second part “YZ”, what leads to “StationY_PN_YZ” for the identifier ND_IDENT_2).

After these character string operations are completed, copy the generated identifier ND_IDENT_2 to the transparency data field NODE_2. Then, search for the cable data object to which the identifier ADJ_IDENT_2 belongs. For this cable, check if the identifier stored in the transparency data field LEAVE_FROM belongs to a busbar lying in the same substation as the generated partition node is part of (e.g. for “StationY_PN_YZ” as the generated identifier ND_IDENT_2 of the partition node this is the case, since “StationY_BB1” is the identifier stored in this LEAVE_FROM data field):
If this is the case, the considered cable is leaving the generated partition node and hence the identifier $ND\_IDENT\_2$ has to be copied to the first transparency data field $NODE\_1$ of this cable, if this is not yet filled (This data field can already be filled when the considered switch has a pair switch that has been previously checked.)

Otherwise, the considered cable is entering the generated partition node. Thus, the identifier $ND\_IDENT\_2$ has to be stored in the second transparency data field $NODE\_2$ of this cable, if this is not yet filled (This data field can already be filled when the considered switch has a pair switch that has been previously checked.)

By execution of these two described steps for each switch data object, all partition nodes can be generated. The only precondition for this generation method is that all identifiers of busbar, cable and switch data objects belonging to the original data base are constructed according to exact character string rules. These rules define the number of characters that are used for the identification of the data type (e.g. “BB” for a busbar data type or “Line” for a cable data type), the number of characters that are used for the distinction between different data objects of the same type (e.g. YZ for the above considered cable) and also the positions of these substrings within the whole identifier string. Are these rules fulfilled for all data objects of the DMS, it is possible to construct these identifiers for the partition nodes according to appropriate character string operations.

If there is a busbar coupler line, then the names of the associated partition nodes generated by the above procedure have to be enhanced. As this is just a special case and not necessary for the understanding of the general idea, this is not elaborated in this thesis.

### 7.2.2 Generation of the Transparency Matrix

With the partition nodes created, the transparency matrix can be generated. As already mentioned, the transparency matrix is generated out of the data base into the RAM of the DMS. Consequently, it will be deleted when the computer on that the DMS is running is turned off or when the DMS itself is halted. Thus, the transparency matrix has to be generated after each restart of the DMS or after each configuration change. On the other side, the partition nodes have just to be
generated *once* after a configuration change. As they are part of the enhanced data base, they do not get lost when the computer is turned off or when the DMS software itself is halted.

In the following, the subprocedure generating the transparency matrix is explained in a general way. Programming details are not reported in order to focus on the essential ideas.

**Step ①:** Determine the number $N_{TrspCol}$ of needed transparency matrix columns by counting all data objects of the DMS and adding 3:

$$N_{TrspCol} = N_{Busbar} + N_{Cable} + N_{Infeed} + N_{Switch} + N_{Substation} + 3 \quad (7.1)$$

Thereby, $N_{Busbar}$ stands for the number of busbar data objects, $N_{Cable}$ signifies the number of cable data objects, $N_{Infeed}$ represents the number of infeed data objects, $N_{Switch}$ is the number of switch data objects and $N_{Substation}$ stands for the number of substation meta data objects. Of course, if the DMS contains other types of data objects, they have to be added as well. The reasons that the sum of all data objects has to be increased by 3 are the followings:

- The first column of the transparency matrix has to store the names of the attributes or the identifiers of the grid nodes respectively the identifiers of the partition nodes (see Section 4.2).
- The second last column has to contain the special symbol # to express that this column is separating the data object columns from the last column.
- The last column has to store the name of the node type stored in the corresponding incidence row (see Section 4.2 and Section 5.2).

**Step ②:** Start the generation of the transparency matrix by initializing a dynamic, two-dimensional array of character strings consisting of $N_{TrspCol}$ columns (see for instance [Prat04] for explanations about dynamic, two-dimensional arrays). Then, fill this first row of the transparency matrix by storing in each of its elements the identifier of one data object of the DMS, whereby the matrix element in the first column has to remain empty. Put into the matrix element in the second last column the string “EOD”, which is an abbreviation for “end of data objects”. Then, store into the matrix element in the last column the name “Node Type”.
In principle, the data objects can be sorted in any order in the transparency matrix, but it is advantageous for the rest of the generation subprocedure to have them sorted by type. Thus, first fill the identifiers of the busbar data objects from left to right into the first transparency matrix row. Then, follow with the identifiers of the cable data objects, and subsequently put the identifiers of the infeed objects in the first transparency matrix row. Following the infeed objects, store the switch objects from left to right. Finally, fill the substation meta data objects in the first transparency matrix row.

For the concrete example of Test Distribution Grid 1, the first row of the transparency matrix is illustrated in Figure 7.3 after the completion of Step 2, whereby only the data objects of Station A are shown explicitly.

**Step 3:** In this next step, create the part of the transparency matrix that contains the incidence matrix rows with the grid nodes – these are consequently the busbars. For this purpose, check all elements of the first transparency matrix row storing an identifier of a busbar object. For each such busbar identifier found, add a new row to the already existing transparency matrix. In the first matrix element of such a new row, store a copy of the considered busbar identifier. Then, with exception of the last and the second last matrix element, fill all remaining matrix elements of this new row with default values as these ones are all laying in the incidence matrix. The chosen default values are 0 integers expressing that – at least so far – no connectivity has been established between the data objects stored in the transparency matrix columns and the considered busbar data object stored as a grid node in the actual new row. Of course, if in the further course of the subprocedure, a connectivity between a certain data object and a certain busbar is determined, the concerning default values are replaced accordingly (see Step 6).

Then, as the considered busbar data object has to be linked with itself in the incidence matrix (see Subsection 4.2.1), replace its relevant default value as follows: Put a connectivity integer 2, which expresses regular topology, into the matrix element laying in the column in that the considered busbar data object is stored. Finally, put into the second last matrix element the symbol # and store into the last matrix element the string “Grid Node”.
In Figure 7.3, the already generated part of the transparency matrix is shown again as Step 3 is terminated.

**Step 4:** Arrived at this step of the subprocedure, each busbar data object is stored as a grid node in one incidence matrix row of the transparency matrix part so far generated. To complete the incidence matrix part, enhance the transparency matrix with the rows storing the partition nodes. To create all these rows with partition nodes, check each cable data object and each infeed data object, whereby – using the order in that these objects are sorted in the first transparency matrix row – begin with the cable data objects.

For each cable data object, check the identifier of the first partition node stored in the transparency data field \textit{NODE}_1. Then, add a new row to the existing part of the transparency matrix, and store in the first matrix element of this new row the identifier of the first partition node. With exception of one matrix element, fill all other matrix elements of this new transparency matrix row up until the second last matrix element with default values, namely 0 integers. The only exception is the matrix element lying in the column in that the considered cable data object is stored. The entry there has to be a connectivity integer expressing that the considered cable and the first partition node are adjacent. Thus, determine the value of this connectivity integer by checking the transparency data field \textit{LEAVE}_FROM of the considered cable data object. If the identifier stored there refers to a busbar that is part of the same substation as the first partition node, the considered cable is leaving from this busbar. Therefore, store into the concern matrix element the connectivity integer 2 expressing regular topology for leaving branches. Otherwise, the considered cable is entering this busbar. In this case, put into the concerning matrix element the connectivity integer -2 representing regular topology for entering branches. Finally, put into the second last matrix element the special symbol #, and store the string “Partition Node” into the last matrix element.

Subsequently, check the identifier of the second partition node stored in the transparency data field \textit{NODE}_2. For this identifier of the second partition node, carry out the same executions as have been made for the identifier of the first partition node. Thus, the considered cable data object gets linked to the incidence matrix row containing the second partition node. Consequently, in the column in that this cable data
object is stored, two connectivity integers appear, namely once the integer -2 and once the integer 2.

Figure 7.3 illustrates the transparency matrix with the added incidence rows containing the partition nodes of the cables. As can be seen there, so far only columns storing cable data objects contain connectivity integers.

Step 5: Examine now all infeed data objects in the same way as the cable data objects have been checked. But since infeeds are half branches (see Subsection 4.2.1), they are adjacent to just one partition node. That is why an infeed data object has but one transparency data field for storing the partition node identifier, namely the data field NODE.

For each infeed data object, check this transparency data field NODE. For the partition node identifier stored there, execute the same instructions as have to be executed for the two partition node identifiers of a cable data object. The only difference is that the connectivity integer for such an infeed data object is always the integer -2. The reason for this is that by definition each infeed is entering its partition node.

After the last infeed data object has been completely examined, all incidence matrix rows of the transparency matrix are generated. Only the entries in the columns in which the switch data objects or the substation meta data objects are stored have still to be adjusted in the following.

The transparency matrix with the newly added incidence rows is again pictured in Figure 7.3, where it can be seen that only columns with switch data objects or columns with substation meta data objects not yet contain connectivity integers.

Step 6: In this step, all switch data objects have to be examined in order to determine their connectivity values in the incidence matrix rows.

For each switch data object, check the identifier of the first node stored in the transparency data field NODE_1, which is either the identifier of a busbar or the identifier of a partition node. Then, search in the first column of the transparency matrix in which matrix element this same node identifier appears.

Having found the searched node identifier, change in the incidence matrix row of this first node the value of the matrix element lying in the
Figure 7.3: Visualization of how the transparency matrix is developed during the first six steps of the generation subprocedure.
column in that the considered switch data object is stored. The reason for this is that the entry in this matrix element has to be a connectivity integer expressing if the considered switch is closed in regular grid situations or not. Thus, determine the value of this connectivity integer by checking the REG_TOPOLOGY data field of the considered switch data object (see Figure 7.1). If the entry of this field is set to TRUE, then the switch is part of the regular topology, that means it is closed in all regular grid situations. Thus, put the connectivity integer 2 into the concerning matrix element expressing regular topology. Otherwise, the switch is open in all regular grid situations providing only a possible connection between its adjacent cable and its adjacent busbar. In this case, store the connectivity integer 1 into the concerning matrix element to express possible topology.

Subsequently, check the identifier of the second node stored in the transparency data field NODE_2. For this node identifier execute the same instructions as have been executed for the node identifier of the first node. Only the REG_TOPOLOGY data field has not to be examined a second time. Instead, store the same connectivity integer that has been stored into the incidence matrix row of the first node into the incidence matrix row of this second node, namely into the matrix element lying in the column of the considered switch data object.

In Figure 7.3, the transparency matrix is presented updated with the connectivity integers for the switch data objects, where it can be observed that the switch data objects contain as connectivity integers either the integer 1 or the integer 2.

**Step 7:** Finally, the substation meta data objects have to be checked. For these data objects perform similar instructions as have been executed for the switch data objects. The only difference is that each virtual link between a substation and one of its busbars is considered as being part of the regular topology. This is the reason why the substation meta data object does not have a REG_TOPOLOGY data field but only a BUSBARS data field, in which all busbars being part of it, are listed. Thus, find all incidence matrix rows of the relevant busbars. For each such incidence matrix row found, put into its matrix element lying in the column in that the considered substation is stored, the integer 2 to express regular topology.

The state of the transparency matrix after the completion of **Step 7**
### Step 7:

<table>
<thead>
<tr>
<th>Node</th>
<th>EON</th>
<th>Node Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Station_A_BB1</td>
<td>2</td>
<td>Grid Node</td>
</tr>
<tr>
<td>Station_A_BB2</td>
<td>0</td>
<td>Grid Node</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

Incidence columns of the substations filled with the connectivity integers.

### Step 8:

<table>
<thead>
<tr>
<th>Node</th>
<th>EON</th>
<th>Node Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Station_A_BB1</td>
<td>2</td>
<td>Grid Node</td>
</tr>
<tr>
<td>Station_A_BB2</td>
<td>0</td>
<td>Grid Node</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

Graphical presentation attributes

- Overview Edge Vector
  - 85.2,199.0;922.0,165.3;
  - 922.0,180.0;185.3,900.0,139.5
  - 71.5,196.5,196.5,900.0,139.5

Type description attributes

- Object Type
  - BUSBAR
  - CABLE
  - INFED

SCADA attributes

- Object State
  - Available
  - On

Advanced application attributes

- Ring
  - 11

Figure 7.4: Visualization of how the transparency matrix is developed during the last two steps of the generation subprocedure.
is pictured in Figure 7.4. As shown there, the incidence matrix part is now completed.

**Step 8:** Having checked all substation meta data objects, the incidence matrix of the transparency matrix is complete. Now, the attribute rows can be added.

For this reason, first add a new row to the transparency matrix. Put into the first element of this new row the string “EON”, which is an abbreviations for “end of node objects” and fill in all other matrix elements the special symbol # to express that this row is separating the incidence matrix rows from the attribute rows.

Then, generate for each attribute a new row. Store in the first matrix element of this new row the attribute name and in all other matrix elements up until the second last one the attribute value of the concerning data object. Put in the second last matrix element the special symbol # and in the last one the symbol – as sign for a no value entry.

Figure 7.4 shows the transparency matrix added with attribute rows representing four typical attribute groups, such as graphical presentation attributes, type description attributes, SCADA attributes and advanced application attributes. For each of these groups, one attribute row is explicitly pictured, namely the overview coordinates row, the object type row, the object state row and the ring number stack row.

The entire transparency matrix can thus be automatically generated by just executing the instructions of these eight steps. The only precondition for this described subprocedure is that the partition nodes are correctly generated and that the attribute values in the transparency data fields are accurately set. As can be observed in Figure 7.4, the final form of the implemented transparency matrix is identical with the form of the conceptual transparency matrix presented in detail in Section 4.2. Consequently, the earlier developed algorithms, namely the condensing algorithm, the condensed grid topology detection algorithm and the distribution power flow calculation algorithm, can be used without significant modifications or adjustments. Thus, they will work correctly when using the implemented transparency matrix as data source.
7.3 Implementation of the Transparency Matrix into RITOP®

As described in Subsection 7.2.1 and in Subsection 7.2.2, the developed subprocedures for generating the partition nodes and for generating the transparency matrix provide an efficient solution how an original, object-oriented data base of a DMS can be made fully transparent. However, as long as these subprocedures are not tested in a real process control system, the desired proof that the transparency concept is more than a theoretical idea is not satisfyingly furnished.

To produce the evidence that the implementation of the transparency matrix is feasible, the process control system RITOP® of the project partner Rittmeyer AG was used. The data base of one RITOP® reference installation representing a distribution grid of a typical small Swiss town was taken as a concrete example. Though to focus on the new algorithms and methods developed, this reference installation has been somewhat reduced and modified. The resulting test installation refers exactly to Test Distribution Grid 2, which is presented in Appendix B.2.

The original data object types of the RITOP® process control system provide no possibility to store the partition nodes, or more precisely their identifiers, which are needed for the generation of the transparency matrix. Therefore, additional attribute data fields have to be appended to the existing data object types as described in Subsection 7.2.1. This adding of new attribute data fields can easily be done since the RITOP® data type editor allows the modification of an existing data object type even though particular data objects of this data object type already exist. Hence, the subprocedure to generate the partition nodes can be developed and implemented into the RITOP® process control system without any problems or restrictions so that this subprocedure is very similar to the described partition node generation subprocedure of Section 7.2.1. Only some few modifications have to be done due to some special features of the existing RITOP® data structure. However, these modifications are insignificant.

The implementation of the subprocedure generating the transparency matrix can be done straightforward once the partition node generation subprocedure exists. In contrast to the conceptual transparency matrix, which has to be mainly the unique data source containing all data objects of the control system according to the criteria definitions of Subsection 4.1.1, the transparency matrix generated into the RAM has another primary objective to fulfill. As explained in Subsection 4.2.3,
the designated use of this implemented transparency matrix is to constitute a framework for transparency checks after configuration changes and to provide a basis for all application-oriented algorithms. As all data objects of the control system are stored in the RITOP® data base, it is not strictly required to copy all their attributes into the implemented transparency matrix, or more precisely, it is not absolutely necessary to construct for each attribute type a new transparency matrix row. Specific attributes that are not needed for transparency checks and are only rarely used in the DMS are thus not put into attribute rows appended to the transparency matrix. Nevertheless, these attributes can get indirectly accessed over the concerning data object identifier stored in the first transparency matrix row. Using this data object identifier, the searched attribute value can be accessed in the RITOP® data base.

By contrast, important attributes being often used have to be placed into the transparency matrix to provide an efficient and fast access to their values. Especially attributes needed for transparency checks or for advanced applications like the condensing algorithm or the distribution power flow calculation method have to be represented in the relevant attribute rows of the implemented transparency matrix. However, the most important part of the implemented transparency matrix are not the needed attribute rows but the incidence matrix rows, which represent the grid model with the highest device resolution. Already in Subsection 4.2.1, it has been stated that this incidence matrix is the most important part of the (conceptual) transparency matrix. Now, this is even more true for the implemented transparency matrix as the incidence matrix here is like a mesh structure connecting all data objects of the RITOP® data base. In fact, the implementation of the transparency matrix into the RITOP® process control system showed that as long as the incidence matrix part is correctly generated, it does not matter so much which attributes are stored in attribute rows of the implemented transparency matrix and which ones have to be indirectly accessed over their identifiers stored in the first transparency matrix row. Incidentally, the fact that not all attributes have really to be put into transparency matrix rows is not a sign of weakness of the transparency concept. Quite the contrary is true: This shows that the transparency matrix in its implemented form is even more flexible than as a theoretical concept.

With the transparency matrix generated into the RAM of the RITOP® process control system, all prerequisites for the advanced application functions, namely the condensing algorithm, the condensed grid topology detection algorithm and the distribution power flow calculation algorithm, are fulfilled. They are integrated into
the RITOP® process control system almost exactly as described since all attributes needed for them are put into attribute rows of the implemented transparency matrix. Thus, the RITOP® process control system is made completely transparent providing not only SCADA functions but also supplying application functions.

7.4 Concluding Remarks

This chapter showed that the developed conceptual model of the transparency matrix can be turned into a real software solution. Using the data base of an existing DMS with an object-oriented data architecture, the desired transparency matrix can be completely automatically generated by first creating the needed partition nodes. Only a few modifications of the original data base, namely the appending of the transparency data fields, are needed to create the preconditions for the creation of the partition nodes and the subsequent generation of the transparency matrix. In fact, that with only slight changes of some data objects and with only the execution of two straightforward subprocedures an existing DMS can be made completely transparent, is remarkable and has to be evaluated as strength of the transparency matrix concept.

With the implementation of the transparency matrix into the process control system RITOP® of the project partner Rittmeyer AG, a proof that the transparency matrix can be used to make an existing DMS fully transparent is satisfyingly furnished. The realization of the developed concepts and algorithms in the RITOP® process control software even showed that the implemented transparency matrix is more flexible than the conceptual one, mainly what concerns the access to the attributes of the data objects (see Section 7.3). Thus, it can be concluded that not only the software feasibility of the transparency matrix has been proved. Moreover, this chapter has also shown that the implementation of the transparency matrix can be done in a very effective manner.
8 Discussion and Outlook

8.1 Summary and Discussion

Starting point for this thesis was the state-of-the-art survey that has been carried out to explore innovation potential of existing DMSs. The major conclusion of this state-of-the-art survey was that there is a need for a fully integrated DMS with a transparent data architecture. As this need is considered to be of most importance, the first and most important goal of this thesis is the development of a new conceptual model for a fully transparent DMS.

Due to the fact that grid models are very important in a DMS providing not only SCADA functions but also application-oriented calculation tools, this has to be reflected in the conceptual model to develop. Thus, the transparency matrix is introduced with the idea that the grid model is represented in matrix form. For this purpose, the upper part of the transparency matrix contains a block matrix, which is a special incidence matrix representing the grid model. Besides this special incidence matrix, the transparency matrix contains attribute rows, in which the attributes of the data objects of the distribution grid control system are placed. Since the transparency matrix has to be the unique data source, it has to store all data objects refering to the distribution grid. As a consequence, the grid model represented by this special incidence matrix has the most detailed degree of description, which in [NoWi07] is denominated as highest device resolution. This complete grid model with the highest device resolution is particularly appropriate for SCADA functions but it is too detailed for application-oriented calculation tools, which prefer simple node-branch-grid models.

Since many of the advanced application functions need only a simplified equivalent of the grid model represented in the transparency matrix, the second aim of this
thesis is to develop an algorithm that establishes a direct and efficient link between the transparency matrix and the desired simplified grid model. The filterability of the transparency matrix allows to establish such an efficient link by filtering – or rather condensing – the transparency matrix into the desired matrix form for application-oriented algorithms. Hence, this algorithm is denominated as condensing algorithm, and the matrix it derives is referred to as condensed transparency matrix.

In order to evaluate the transparency concept to develop, six criteria for transparency have been formulated at the beginning of this PhD-thesis. In addition, three levels of transparency that a fully transparent DMS must achieve have been defined. Thus, the conceptual model for a fully transparent DMS is assessed regarding these six transparency criteria and the three levels of transparency. This evaluation has proved that all transparency criteria and hence the first level of transparency can be fulfilled by the transparency matrix. With an additional transparency configuration manager, which prescribes how data is written into the transparency matrix, the second level of transparency can be achieved as well. Therefore, data consistency and data correctness can be guaranteed at all times. Finally, the evaluation of the condensing algorithm, which derives the condensed transparency matrix in an effective and reliable way, has shown that also the third level can be fulfilled.

With the transparency matrix, the transparency configuration manager and the condensing algorithm deriving the condensed transparency matrix, very important concepts and algorithms are developed, tested and evaluated. With them, the goal of a fully transparent DMS providing also application-oriented calculation tools can be achieved. These calculation tools need information about the actual topology of the distribution grid to analyze; for instance, a power flow calculation method needs to have the information if rings are currently existing. Since power flow calculation is one of the most important grid analysis functions provided in a modern DMS, the aim is to develop a power flow calculation method for a fully transparent DMS. So to guarantee that the needed information is provided without the risk of data inconsistency or data incorrectness, a linking procedure between the condensed transparency matrix and this power flow calculation method is needed.

For this reason, a condensed grid topology detection algorithm is developed that acts like a linking procedure between the condensed transparency matrix and the power flow calculation method. More precisely, the condensed grid topology detection algorithm is an important pre-routine of the power flow calculation method,
since it detects all existing rings by executing special search cycles through the condensed grid structure. In the first search cycle the algorithm spreads out radially to determine for each grid node its shortest distance to the source node, denominated as minimal source node distance. By doing so, it can simultaneously detect the so called ring end nodes. These ring end nodes define the cut-off points of each ring, where the ring can be “virtually” broken. Since always two ring end nodes belong to one ring structure, the algorithm knows the number of existing rings when terminating this first search cycle.

If the condensed grid topology detection algorithm detects ring end nodes during the first search cycle, it executes one additional search cycle, in which, starting from each ring end node, the grid branches within the ring structures are identified on the way upstream. Thus, with only two search cycles, not only all rings are detected but also all grid branches belonging to one of these rings are identified. This identification of grid branches within ring structures can then be used for the power flow calculation method.

The aim for the power flow calculation method was to have a calculation tool that not only can be used for a large variety of radial distribution grids but that can also be used for weakly meshed distribution grids. As it was the intention to calculate the unknown quantities of a given power flow problem, if possible, in the same sequence as the depth-first search strategy passes through the grid structure, the feasibility of this approach was examined by means of test examples.

These test examples showed that it is possible to use the sequence of the depth-first search strategy to update the power flow quantities in each iteration run until they are converged. The method proposed is thus based on the Gauss-Seidel method: It starts with an initialization of the complex node voltages, whereby the minimal source node distances determined by the condensed grid topology detection algorithm are used to set for each node its initial voltage value as a function of its minimal source node distance and a function of an adequately set voltage drop. The second and the third subprocedure of the power flow calculation method are then repeated until the convergence criterion is fulfilled. The second subprocedure has the task to update the complex line currents and the complex load currents by using the values of the complex node voltages determined in the previous iteration run. The third subprocedure uses the updated complex line values to determine the new updates of the complex node voltages and calculates for all nodes the calculated complex power values to check if the convergence criterion is fulfilled or if an additional iteration run is needed.
The elaborated concept of equivalent injection currents provides a good method to determine the so called ring branch currents such that the process of power flow calculation for ring situations is almost identical to the process of power flow calculation for normal radial grid situations. The process of power flow calculation for ring situations needs one additional data preparation subprocedure to determine the impedance sums of grid branches located within a ring structure. The calculation steps of the iteratively repeated search cycles, namely the search cycle updating the complex line currents and the search cycle updating the complex node voltages, are almost the same for radial grid situations and for ring situations. One difference is that for ring end nodes additional calculations are needed to determine the values of the ring branch currents. Similarly, the other difference is that for all grid branches within a ring structure additional calculations are needed to determine their voltage drops.

That the developed power flow calculation method can be used for both radial and ring situations is very advantageous, not just because only a few additional calculation steps are needed for ring situations but mainly because all additional steps are executed completely automatically: The process of the power flow analysis detects by itself if rings are existing. It uses the knowledge about the actual grid topology, which is provided by the condensed grid topology detection algorithm. This algorithm in turn needs the condensed transparency matrix, which is derived by the condensing algorithm using the transparency matrix as single data source. Due to this data flow chain, no additional knowledge has to be provided to the power flow calculation method. Consequently, as long as the data stored in the transparency matrix is correct, the output of the power flow analysis is correct too.

8.2 Outlook

In this section, an outlook is given on possible development objectives for future DMSs. In Subsection 3.3, three development objectives to remedy the weak points of state-of-the-art systems have been derived, whereof the first objective of developing a future DMS with a fully transparent data architecture has been selected to be the main objective of this thesis. With the developed transparency matrix storing all data objects of the DMS, the condensing algorithm generating the condensed transparency matrix and the condensed grid topology detection algorithm identifying all branches lying within rings, this goal of a fully transparent DMS is completely achieved as described in detail in the previous chapters.
For the second and the third main objectives derived in Subsection 3.3, it has been stated that they depend directly or indirectly on the chosen data architecture of the DMS. Because of that, the intention was to develop first a DMS with a fully transparent data architecture before doing research in the areas of these other objectives. Now that methods and algorithms for a fully transparent DMS are developed, the focus of the further development has to be on these two objectives. In fact, it is very likely that the transparency matrix, the condensing algorithm and the condensed grid topology detection algorithm provide possibilities to now attain these two goals in an efficient way.

In addition, one new development objective has arisen during this thesis, namely to enhance the power flow calculation method such that it can also be used as a kind of simulation, optimization and planning tool. The motivation for this goal is that it would be very advantageous for the distribution system operator when grid planning and optimization work can also be done within the DMS. First, there is no need for an additional (mostly expensive) grid analysis software, and second, since no data exchange has to be done between the DMS and the grid analysis software, there is no risk for data inconsistency and data incorrectness. Thus, the aim of a DMS providing also a simulation, optimization and planning tool is added to the list of possible development objectives for future DMSs:

1. User-friendly and situation appropriate visualization
2. A capable of learning DMS giving the best possible support to the operator
3. DMS with a simulation, optimization and planning tool

In the three following subsections, the first elaborated ideas of how these development objectives can be achieved are briefly discussed.

8.2.1 User-Friendly and Situation Appropriate Visualization

As discussed in Section 3.2.2, an unmistakable and absolutely clear visualization is of utmost importance - particularly in critical operation situations when the human operator has to react quickly and target-oriented. Hence, a future DMS should visualize information more user-friendly and more appropriately to the actual grid situation. One objective of such a user-friendly and situation appropriate visualization is the prevention of operating errors by using an unmistakable and absolutely unambiguous presentation of the system states. Such an unambiguous
visualization has to be done by strictly ergonomic principles. A second objective is to provide an additional visual support to human operators by graphically highlighting the position where the DMS concludes the main cause of the disturbance to be. Another objective is the prevention of message showers, which cannot be overlooked by the human operators, by using intelligent suppression of consequential alarms and by formatting of collective alarms.

**Example for a Better Visualization of the Actual Operating State**

Figure 8.1 and Figure 8.2 show the difference between the overview screen of an exemplary DMS with a state-of-the-art visualization and a future exemplary DMS with a more user-friendly and situation appropriate visualization.

For example, if a disturbance occurs in the circuit breaker of the infeed of power plant Beta, the outgoing feeders connected to the districts Delta and Epsilon are also disturbed and the energy cannot be supplied as normally to these districts. On an overview screen of an exemplary state-of-the-art DMS as shown in Figure 8.1, all concerned connections are colored with orange to indicate that there is a disturbance. Since the origin of this disturbance is not specially highlighted, it takes the responsible operator some seconds to recognize where the disturbance originates. By contrast, a future DMS with a situation appropriate visualization indicates the origin of the above described disturbance by emphasizing the origin of the disturbance like shown in Figure 8.2: By highlighting the origin with a light orange glowing area and showing the unimportant information semi-transparent, this situation appropriate visualization helps the operator to look for the important information. Thus, this situation appropriate visualization achieves that the operator can react faster and more target-oriented.

The presented state-of-the-art visualization does not claim to appear exactly in this way in an existing DMS. Also, it has to be mentioned that the aim is not to denigrate on the market available DMSs but rather to show that concerning user-friendly visualization and ergonomic system design for DMSs considerable technical innovation potential still exists.
Figure 8.1: The actual topology with a disturbance without user-friendly visualization.

Figure 8.2: The actual topology with a disturbance and a more user-friendly visualization.
8.2.2 A Capable of Learning DMS Giving the Best Possible Support to the Operator

As described in Subsection 3.2.3, many of the grid analysis functions delivered with a state-of-the-art grid control system are only activated when the human operator explicitly requests their support. It can thus be assumed that they are not active during disturbances when quick decisions have to be made.

A future DMS should therefore be able to recognize on its own when the human operator needs more support. It should provide the required assistance to the operator in the control center during grid disturbances by suggesting him appropriate control actions. Moreover, the DMS should have the ability to recognize the human operators pattern of behavior and to know how to support him accordingly. This includes the identification and the prevention of mistakes before they really become operative. The DMS should further be capable of learning from former events and act accordingly to the knowledge gained from them.

Through foreseeing of dangerous situations and reacting at an early stage, problems should be avoided before they really appear. The future DMS should support the operator and assist him in his continuous learning process. By the use of an adaptive DMS, which gives the best possible support to the operator and can learn from former events, fluctuations in the performances of the operators resulting from shift changes, departures of experienced operators, employments of new inexperienced operators, overwork or tiredness can be smoothed out. The quality of the supervisory control done by the human operator should then not any more depend on fluctuations of his performance, on his daily form nor on his level of training and experience.

8.2.3 DMS with a Simulation, Optimization and Planning Tool

The development of a simulation, optimization and planning tool for a future DMS should be based on the power flow calculation method developed in this thesis. Hence, first the developed power flow calculation method has to be enhanced. Namely, the following enhancements should be done:

1. Power flow calculation for *multi-infeed* situations

2. Power flow calculation for *stronger meshed* distribution grids
The expression *multi-infeed* refers to situations where in a distribution grid containing several MV substations some local substations are fed from more than one MV substation due to the actual topology. This situation is very similar to a ring situation but the difference is that there is more than one source node. Assuming that the source nodes have identical complex voltage values, they could be condensed to one slack node for the power flow calculation. Thus, the condensed slack node would also become the (condensed) ring junction node (see Subsection 5.3.2), and the unknown quantities can be determined like for a normal ring situation. In general, the made assumption may not be valid as the voltage angles of different source nodes are normally not identical. Thus, these source nodes cannot be condensed to one single source node for the power flow calculation. One possibility to solve the power flow equations for multi-infeed situations is to model each source node as a separate slack node, more precisely as a separate $U\theta$-bus (for definition see also [Ande04a] or [GőCo09]). Consequently, the voltage angles of these source nodes have to be known. The achievement of this approach will be how these voltage angles of the source nodes can be determined or accurately set. A closer examination of the power flow equations for multi-infeed situations may reveal that for these grid situations a power flow calculation method is needed that is based more on the Newton-Raphson method than on the Gauss-Seidel iteration scheme.

For the second enhancement concerning power flow calculation for stronger meshed distribution grids, similar considerations can be done. The expression *stronger meshed* means that the topology of the distribution grid is rather meshed than radial since a large number of meshed rings are existing. This can be the case for simulations based on the possible topology where all circuit breakers at the cut-off points are closed (see Subsection 2.1.3). For these stronger meshed distribution grids, a power flow calculation method based on the Newton-Raphson method is probably more adequate. To develop such another power flow calculation method using Newton-Raphson iterations is no problem. But most important is the aim that the fully transparent DMS itself knows in which grid situation which power flow calculation method has to be started. Thus, tests, simulations and careful considerations are needed to elaborate a general rule that determines at which degree of meshes which power flow calculation method has to be started.

These reported enhancements for multiple-infeed situations and for stronger-meshed distribution grids are then the basis for the further development of a simulation, optimization and planning tool. For instance, one possible optimization method is the determination of the optimal break-off points as mentioned in Subsection 5.3.2. There it is defined that optimal cut-off points break rings such that
the power losses in the feeder lines of the resulting radial grid topology are minimized. This method is thus an enhancement of the first subprocedure of the condensed grid topology detection algorithm. Instead of using the minimal source node distances, the method determines for each node a special complex power value. This special complex power value is a sort of weighting function of the complex power values of all load nodes that are located downstream of the considered node. The breadth-first search cycle then uses these special complex power values to determine the optimal-cut off points, namely the ring end nodes. The main achievement of this method will be to find out of which components the weighting function determining the special complex power flow values consists.

### 8.3 Concluding Remarks

The possible development objectives discussed in Section 8.2 show that there is a wide range of further development for DMSs. The basis for the further development is thereby done by the concepts and algorithms presented in this thesis. All this algorithms and methods provide now an ideal basis for future work. With them, it is possible to further develop the DMS so that really advanced application tools like simulation tools and power flow optimization methods can be developed. Actually, the fact that so many possibilities for further development for DMSs exist now, is another proof for the effectiveness of the transparency matrix concept, the condensing algorithm and the condensed grid topology detection algorithm.
A Composition Models for Interconnected Power Grids

A.1 Mathematical Description of the Interconnected Power Grid

For power system control and thus for power grid control systems, a mathematical description of the structure of the power grid to be monitored is required. For this purpose, concepts of mathematical graph theory, especially concepts of topological graph theory, can be used. Topological graph theory is the branch of mathematical graph theory that primarily deals with drawing of graphs on surfaces.

In graph theory, a graph is normally a two-dimensional structure existing of nodes and edges joining in the nodes. Therefore, a galvanic coupled electric power grid can be described as a two-dimensional graph since it consists of only one voltage level: The busbars are the nodes of the graph and the couplers and lines are the edges or rather the branches of the graph. By addition of transformers forming the vertical edges between nodes of different voltage levels, the interconnected power grid becomes a three-dimensional graph.

Furthermore, it is necessary to partition the interconnected power grid into so called grid components, partitions and grid elements, as described in detail in [Tiet06a]. In the following, the mathematical descriptions needed in this thesis to model the topological structure of the power grid are briefly described.

A.1.1 Potential, Actual and Regular Topology

In Subsection 4.2.2, the different types of topology, namely the potential topology, the actual topology and the regular topology are briefly explained. To visualize the
relations between these three types of topology, they are sketched in Figure A.1. Since a grid control system like a EMS or a DMS has to store and to manage all three topologies, it is important to distinguish between them.

A.1.2 Grid Components

As mentioned above, the topological structure of a power grid can be described as a three-dimensional graph consisting of nodes and branches. Since the collectivity of all nodes and branches forms the potential topology of the interconnected power grid, nodes and branches are designated as grid components. Hence, grid components can be the following devices:

- Branches in the form of couplers within substations
- Branches in the form of lines between different substations of the same voltage
- Branches in the form of transformers between substations of different voltages
- Branches in the form of infeeds supplying substations with electric energy
– Nodes in the form of *busbars* within substations

By means of a substation with a double busbar configuration, some examples of different grid components are illustrated in Figure A.2. For the flawless guidance of an interconnected power grid, a structuring into different grid components is essential, since

– grid components are the smallest units represented on general overview screens,
– grid components constitute the biggest units for which collective states can be defined,
– grid components act as communication objects in case of switching requests,
– grid components set off as a whole in case of disturbances, have to be discharged as a whole in case of overload situations and are unavailable altogether during maintenance works,
– in connection with power grid expansions entire grid components are installed.

In this thesis, grid components that are branches are often referred to as *grid branches*. Similarly, grid components that are nodes are often called *grid nodes*.

### A.1.3 Partitions

Grid components can be divided in location and voltage dependent parts, so called *partitions*. For example, a transformer as a grid component consists of three partitions, namely the upper voltage partition of the transformer, the transformer partition and the lower voltage partition of the transformer. Figure A.3 shows some examples of different partitions in a substation with a double busbar configuration.

As a basic principle, partitions are applied to be located in just one substation and to belong to just one voltage level. According to [Tiet06a], for example the following devices are partitions:

– Switchbays like infeeds or outgoing feeders
– Equipments as transformers, generators or coils
– Medium voltage installations of local substations
Figure A.2: Examples of different grid components in a substation with a double busbar configuration.

Figure A.3: Examples of different partitions in a substation with a double busbar configuration.
A.1 Mathematical Description of the Interconnected Power Grid

In an interconnected power grid, there are a lot of identically constructed partitions, which therefore can or rather have to be categorized for the SCADA functions.

A.1.4 Grid Elements

The smallest physical devices of an interconnected power grid are denominated as *grid elements*. Grid elements can be switching devices, converters, transformer coils or not dividable line parts. They constitute the linking elements in the grid components and between them. Figure A.4 shows some examples of different grid elements in a substation with double busbar configuration.

A.1.5 Availability of Grid Components and Partitions

Grid components and – particularly in regional and local distribution grids – some removable partitions can have the following operating conditions:
– *Available, in operation, connected on both sides:*

The grid component or the partition can be used at any time as it is energized on both of its sides.

– *Available, out of operation (or in operation, but not connected on both sides):*

The grid component or the partition can be used at any time but is energized on just one of its sides.

– *Not available:*

The grid component or the partition is faulted or connected to the ground due to maintenance work.

In addition to the operating conditions, each grid component or each partition has a charge state derived by the relevant measured values. This state indicates how much the concerning grid component or partition is loaded:

– *Lightly loaded*

– *Medium loaded*

– *Heavily loaded*

Furthermore, besides the operating condition and the charge state, there may be the following failure states:

– *Earth fault*

– *Available to only a limited extent*

If the grid component or the partition is in operation, further states can be displayed. For instance, the grid group membership, which is a collective of subgrids that form a unity in the way they are operated, can be visualized.

### A.2 Partitions Branches and Partition Nodes

Grid branches and grid nodes are almost never fix connected to each other: As shown in Figure A.5, grid nodes like busbars and grid branches like lines or infeeds are connected or disconnected over switchable devices like circuit breakers or load
switches. The actual state of such a switchable device defines which grid node is connected to which grid branch. Hence, there is a need for introducing a sort of branches and a sort of nodes between the grid nodes and the grid branches so that all possible switch states, that means all states in which the power grid can be found, can be mapped on the resulting mathematical graph structure.

From a mathematical point of view, switching devices are like active branches. That is why for every switching device of the power grid a so called partition branch has to be introduced. Since branches are always connected by nodes, so called partition nodes have to be introduced between partition branches adjacent to a grid branch or between two adjacent partition branches. As an illustrative example, the mathematical graph structure of the entire Test Distribution Grid 1 resulting when the partition branches and the partition nodes are introduced is shown in Figure B.2 of Appendix B.1. This view of the power grid is called the mathematical view.

Summarized, the following properties are valid for partition branches respectively partition nodes:

**Partition Branches**

- Are connectable branches (from a mathematical point of view)
- Are real “active” grid elements or rather switching devices like
  - Circuit breakers
  - Load switches
  - Earth switches
- Belong to two different partitions
- Can have a voltage across themselves

**Partitions Nodes:**

- Are nodes connecting partition branches (from a mathematical point of view)
- Are real junctions between switching devices and grid branches
- Are neither grid components nor grid elements
- Belong to just one partition
- Have one electric potential
As can be seen for instance in Figure 4.3 or in Figure 4.4, the introduced partition nodes appear in the incidence matrix like normal grid nodes. The only difference is that a partition node just appears in a row and does not appear in a column of the transparency matrix. The reason therefore is that a partition node is just a mathematical object and not a real existing data object.

The reason why these introduced branches and nodes are named partition branches and partition nodes respectively becomes obvious when studying Figure A.5: For a single busbar with an infeed and for a double busbar with an infeed, the different partitions are highlighted with colors. Partitions are location and voltage dependent parts of grid components, as explained in [Tiet06a]. Together with the depicted partition nodes and partition branches, it becomes evident that different partitions are separated by partition branches and that every partition contains one node, which is either a normal grid node or a partition node. Therefore, the given names partition branches and partition nodes are reasonable.
B Test Distribution Grids

B.1 Structure of the Test Distribution Grid 1

Test Distribution Grid 1 is used to evaluate the general feasibility of the transparency matrix introduced in Section 4.2. This smaller of the two test distribution grids contains 1 MV substation, 6 local substations, 7 lines and 2 infeeds:

MV substation:
1. Station A

Local substations:
1. Station B  3. Station D  5. Station F
2. Station C  4. Station E  6. Station G

Lines:

Infeeds:
Infeed A1      Infeed A2

This Test Distribution Grid 1 is depicted in Figure B.1. Besides, Figure B.2 shows the mathematical view of Test Distribution Grid 1 with all grid nodes, grid branches, partition nodes and partition branches. Finally, Figure B.3 presents the transparency matrix of the entire Test Distribution Grid 1.
Figure B.1: Test Distribution Grid 1 contains 1 MV substation, 6 local substations, 7 lines and 2 infeeds.
Figure B.2: Mathematical view of Test Distribution Grid 1: Besides grid nodes and grid branches, this mathematical view contains partition nodes and partition branches.
Figure B.3: The transparency matrix containing all data objects of the entire Test Distribution Grid.
B.2 Structure of the Test Distribution Grid 2

Test Distribution Grid 2 is applied for more comprehensive tests to prove the effectiveness of the developed new concepts and algorithms for a fully transparent DMS. Especially the condensing algorithm, the topology detection algorithm and the power flow calculation method have been tested on Test Distribution Grid 2. This bigger of the two test distribution grids contains 2 MV substations, 22 local substations, 27 lines and 4 infeeds:

**MV substations:**

1. Station A  
2. Station M

**Local substations:**

1. Station B  
7. Station H  
13. Station O  
19. Station U
2. Station C  
8. Station I  
14. Station P  
20. Station V
3. Station D  
9. Station J  
15. Station Q  
21. Station W
4. Station E  
10. Station K  
16. Station R  
22. Station X
5. Station F  
11. Station L  
17. Station S
6. Station G  
12. Station N  
18. Station T

**Lines:**

1. Line AB  
8. Line AH  
15. Line NO  
22. Line UV
2. Line BC  
9. Line HI  
16. Line OP  
23. Line VL
3. Line CD  
10. Line IJ  
17. Line PQ  
24. Line VW
4. Line DE  
11. Line JK  
18. Line QR  
25. Line WJ
5. Line EF  
12. Line KL  
19. Line RS  
26. Line WX
6. Line FG  
13. Line LM  
20. Line ST  
27. Line XS
7. Line GA  
14. Line MN  
21. Line TU

**Infeeds:**

1. Infeed A1  
2. Infeed A2  
3. Infeed M1  
4. Infeed M2

Part A of this Test Distribution Grid 2 is depicted in Figure B.4, and Part B is shown in Figure B.5.
Figure B.4: Part A of Test Distribution Grid 2 contains 1 MV substation, 6 local substations, 7 lines, 2 infeeds and one connecting line to Part B.
Figure B.5: Part B of Test Distribution Grid 2 contains 1 MV substation, 16 local substations, 19 lines, 2 infeeds and one connecting line to Part A.
C Computation Functions of the Transparency Configuration Manager

C.1 Computation of the Dependent Graphical Coordinates

As described in Subsection 4.3.1, the transparency configuration manager has the task to invoke the relevant functions for the computation of the dependent values. So, for the computation of the dependent graphical coordinates, the manager has to invoke the geometric computation functions, as already mentioned in Subsection 4.3.2.

In the following, such a geometric computation function is explained with the help of a concrete example, namely Station B of Test Distribution Grid 1. As shown in Table 4.1, the substation symbols have been chosen to be the independent graphical coordinates. Hence, the graphical coordinates of Station B can be more or less arbitrarily set. Since the graphical symbol for a substation is a simple rectangle, the center coordinates of the rectangle can be taken to determine its position respectively the position of the substation on the graphical display. The graphical coordinates of Station B are thus given by

\[ C_{S_{B}} = [x_{C_{S_{B}}}, y_{C_{S_{B}}}] \]  \hspace{1cm} (C.1)

where \( C_{S_{B}} \) denotes the center point of the rectangle, and the letter \( S \) indicates that the concerning graphical symbol represents a substation. The letter \( B \) identifies Station B and the variables \( x \) and \( y \) stand for the \( x \)-coordinate and the \( y \)-coordinate respectively. With given width \( W_{S} \) and given height \( H_{S} \) of the substation symbol,
the following equations hold for coordinates \([x_{S_B}, y_{S_B}]\) that are covered by the graphical symbol of Station B:

\[
\begin{align*}
x_{C_{S_B}} - \frac{1}{2} \cdot W_S & \leq x_{S_B} \leq x_{C_{S_B}} + \frac{1}{2} \cdot W_S \quad (C.2) \\
y_{C_{S_B}} - \frac{1}{2} \cdot H_S & \leq y_{S_B} \leq y_{C_{S_B}} + \frac{1}{2} \cdot H_S \quad (C.3)
\end{align*}
\]

Equations (C.2) and (C.3) can be verified by inspection of Figure C.1a). Graphical symbols of grid elements belonging to Station B, like Load Switch LS1_B, Load Switch LS2_B or Busbar BB, have to be placed near the graphical symbol of Station B. Near means that their graphical symbols are only separated by a distance of some pixels, if not even touching or overlapping each other. By introducing for each graphical symbol an additional surrounding rectangle that has the same center point coordinates but larger dimensions, it is geometrically possible to determine if the concerning symbols are near enough or if they are too distant to each other.

The coordinates of points which lay in the surrounding rectangle of the graphical symbol of Station B satisfy the following equations:

\[
\begin{align*}
x_{C_{S_B}} - \frac{1}{2} \cdot W_S - \varepsilon_S & \leq x \leq x_{C_{S_B}} + \frac{1}{2} \cdot W_S + \varepsilon_S \quad (C.4) \\
y_{C_{S_B}} - \frac{1}{2} \cdot H_S - \varepsilon_S & \leq y \leq y_{C_{S_B}} + \frac{1}{2} \cdot H_S + \varepsilon_S \quad (C.5)
\end{align*}
\]

Thereby, \(\varepsilon_S\) denotes the distance margin between the edges of the substation symbol and the edges of its surrounding rectangle like depicted in Figure C.1b).

The coordinates of a graphical symbol that has to be near the graphical symbol of the Station B have to be determined so that its surrounding rectangle overlaps the surrounding rectangle of Station B as shown in Figure C.1c). Hence, the center point coordinates \([x_{LS1_B}, y_{LS1_B}]\) of the rectangular graphical symbol of Load Switch LS1_B have to keep the following conditions:

\[
\begin{align*}
x_{C_{S_B}} - \frac{1}{2} \cdot W_S - \varepsilon_S - \frac{1}{2} \cdot W_{LS} - \varepsilon_{LS} & < x_{C_{LS1_B}} < x_{C_{S_B}} + \frac{1}{2} \cdot W_S + \varepsilon_S + \frac{1}{2} \cdot W_{LS} + \varepsilon_{LS} \\
y_{C_{S_B}} - \frac{1}{2} \cdot H_S - \varepsilon_S - \frac{1}{2} \cdot H_{LS} - \varepsilon_{LS} & < y_{C_{LS1_B}} < y_{C_{S_B}} + \frac{1}{2} \cdot H_S + \varepsilon_S + \frac{1}{2} \cdot H_{LS} + \varepsilon_{LS} \quad (C.7)
\end{align*}
\]
Figure C.1: a) Graphical symbol of Station B. b) Surrounding rectangle of the graphical symbol of Station B. c) Overlapping surrounding rectangles of Station B and of Load Switch LS1_B. d) Rectangular area in that the center coordinates of Load Switch LS1_B are allowed to be.
Figure C.2: Surrounding rectangles of the graphical symbol of Station D and of Station E respectively.

Thereby, $W_{LS}$ denotes the width and $H_{LS}$ denotes the height of the graphical symbol of the load switch. $\varepsilon_{LS}$ stands for the distance margin between the edges of the load switch symbol and the edges of its surrounding rectangle. The equations (C.6) and (C.7) show that the area in which the center coordinates of Load Switch LS1_B are allowed to be is a rectangle, what can also be verified by inspection of Figure C.1d).

Similar conditions can be set up for all other graphical symbols which have to be placed near the graphical symbol of Station B. Equally, the geometric computation functions can be used for all other substations of Test Distribution Grid 1 or of Test Distribution Grid 2. In addition, it is also possible to examine if the distance between two substations is greater than a distance margin $\Delta_S$ both in x- and y-direction by using surrounding rectangles that must not overlap like shown in Figure C.2. By using these derived geometric computation functions, the transparency configuration manager then guarantees that all graphical symbols are placed within their allowed areas.
D Modeling of Distribution Grids

D.1 Modeling and Assumptions for the Power Grid

The three-phase distribution system is assumed to be balanced although in general distribution feeders are slightly unbalanced. But to outline the general procedure of the power flow calculation method to be developed, this assumption is justified. However, as explained in [Kers02], power flow calculation methods based on the ladder iterative technique can also be applied for three-phase unbalanced feeders, whereby the basic principle is the same as for three-phase balanced feeders.

The grid branches of the considered regional distribution grids are realized as cables. Since the shunt susceptance values of such cables are in comparison with the series impedance values negligible small, they are not taken into account for the line model. Thus, the lumped-circuit line model of each cable has just a series impedance composed of a series reactance and a series inductance.

As the considered distribution grids are passive systems, that means besides the feeding medium voltage substations there are no other sources, all grid nodes besides the grid nodes of the feeding MV substations are passive and thus have to be modeled as $PQ$-nodes (see for instance [Ande04a] or [GóCo09]).

D.2 Attribute Rows for Power Flow Calculation

In this section, the additional attribute rows that are needed to store the power flow quantities are briefly explained.
Grid nodes:
To store the measured or specified complex power value $S_{\text{Spec} k}$ of each node $k$, the condensed transparency matrix needs the following two attribute rows:

- Row for the storage of the active power value $P_{\text{Spec} k}$
- Row for the storage of the reactive power value $Q_{\text{Spec} k}$

Similarly, for the storage of the complex node voltage $U_k$ and the storage of the complex load current $I_k$ of each node $k$, four additional attribute rows are needed:

- Row for the storage of the node voltage magnitude $U_k$
- Row for the storage of the node voltage angle $\theta_k$
- Row for the storage of the load current magnitude $I_k$
- Row for the storage of the node current angle $\varphi_k$

Even though the complex load current $I_k$ depends on the complex power value $S_{\text{Spec} k}$ and on the complex node voltage $U_k$, it has to be explicitly stored in the condensed transparency matrix. As it is the case with the dependent graphical co-ordinates, which have to be stored in the graphical data row (see Subsection 4.2.6), these dependent power flow variables also have to be stored in the concerning row of the condensed transparency matrix. As only the subprocedures of the power flow calculation method have access to these variables or can change them, there is no risk for data incorrectness.

Grid branches:
To model each branch $kl$ as a lumped-circuit line model as explained above, the line parameters needed for this model have to be stored in the condensed transparency matrix. Thus, the condensed transparency matrix has to be enhanced with the following four attribute rows:

- Row for the storage of the series resistance per length $R'_{kl}$
- Row for the storage of the series reactance per length $X'_{kl}$
- Row for the storage of the shunt susceptance per length $B'_{kl}$
- Row for the storage of the cable length $l_{kl}$
D.3 Tolerance Constraints for Convergence Checks

With these stored parameters, the series impedance of the lumped-circuit line model can be calculated as follows:

\[ Z_{kl} = (R'_{kl} + j X'_{kl}) \cdot l_{kl} \]  \hfill (D.1)

As mentioned above, the shunt susceptance can be neglected for cables. Still, an attribute row is reserved for the value of the shunt susceptance per length. Namely, in case when a grid branch is realized as overhead line, the shunt susceptance cannot be neglected. So, it is necessary that this attribute can be stored in the condensed transparency matrix.

In addition, to store the calculated complex line current \( I_{kl} \) or the complex line power loss \( S_{kl}^{Loss} \) of each branch \( kl \), four additional attribute rows are needed:

- Row for the storage of the line current magnitude \( I_{kl} \)
- Row for the storage of the line current angle \( \varphi_{kl} \)
- Row for the storage of the active line power loss value \( P_{kl}^{Loss} \)
- Row for the storage of the reactive line power loss value \( Q_{kl}^{Loss} \)

Even though the complex line power loss \( S_{kl}^{Loss} \) is given by the complex line current \( I_{kl} \), it has to be explicitly stored in the condensed transparency matrix. There is no risk for data incorrectness as only the subprocedures of the power flow calculation method can change the value of the complex power loss or the value of the complex line current.

D.3 Tolerance Constraints for Convergence Checks

For each node \( k \), the convergence check verifies if the power value difference is within the allowed tolerance constraints. For the real part of the power value difference it is:

\[ |P_{Diff_k}^{\nu}| \leq \alpha \cdot P_{Spec_k} \]  \hfill (D.2)

And for the complex part it is:

\[ |Q_{Diff_k}^{\nu}| \leq \alpha \cdot Q_{Spec_k} \]  \hfill (D.3)
For the power flow simulations that have been executed in the process control system RITOP®, the convergence criterion factor $\alpha$ has been set to

$$\alpha = 1.0 \cdot 10^{-5}.$$  \hspace{1cm} (D.4)

Even with this very small factor, for most of the grid topologies tested, not more than 5 or 6 iterations have been needed till the computations have converged.
Bibliography

[AbG04] Ali Abur, Antonio Gómez Expósito:
*Power System State Estimation – Theory and Implementation*

[Ande03] Göran Andersson:
*Elektrische Energiesysteme*

[Ande04a] Göran Andersson:
*Modelling and Analysis of Electric Power Systems*
EEH - Power Systems Laboratory, ETH Zurich, Summer semester 2004.

[Ande04b] Göran Andersson:
*Dynamics and Control of Electric Power Systems*
EEH - Power Systems Laboratory, ETH Zurich, Summer semester 2004.

[ArGa01] P. Aravindhababu, S. Ganapathy, K. R. Nayar:
*A Novel Technique for the Analysis of Radial Distribution Systems*

[Bach07] Rainer Bacher:
*Arbeitsgruppe Leitungen und Versorgungssicherheit (AG LVS) - Protokolle und Anhänge*

[Bigg99] Norman L. Biggs:
*Discrete Mathematics*
[Boyé04] Stuart A. Boyer:  
*SCADA: Supervisory Control And Data Acquisition*  

[BuDo05] Richard L. Burden, J. Douglas Faires:  
*Numerical Analysis*  

[BrLo03] Klaus-Peter Brand, Volker Lohmann, Wolfgang Wimmer:  
*Substation Automation Handbook*  

[Cegr86] Torsten Cegrell:  
*Power System Control Technology*  

[ChLi00] G. J. Chen, K. K. Li, T. S. Chung, G. Q. Tang:  
*An Efficient Two-Stage Load Flow Method for Meshed Distribution Networks*  
Proceedings of the 5th International Conference on Advances in Power System Control, Operation and Management, Hong Kong, 2000.

[ChTa01] G. J. Chen, G. Q. Tang, K. K. Li, T. S. Chung:  
*A Novel Heuristic Method for Distribution System Planning*  

[Dorf97] Richard C. Dorf:  
*The Electrical Engineering Handbook*  

[FlHi05] René Flosdorff, Günther Hilgarth:  
*Elektrische Energieverteilung*  

[GiBo00] John K. Gilbert, Carolyn J. Boulter:  
*Developing Models in Science Education*  

[GóCo09] Antonio Gómez Expósito, Antonio J. Conejo, Claudio Cañizares:
Bibliography

*Electric Energy Systems: Analysis and Operation*

[Harr00] Jan L. Harrington:
*Object-Oriented Database Design Clearly Explained*

[HeDe02] Klaus Heuck, Klaus-Dieter Dettmann:
*Elektrische Energieversorgung*

[IEEE90] IEEE Computer Society:
The Institute of Electrical and Electronics Engineers, 1990.

[JoJo89] David E. Johnson, Johnny R. Johnson, John L. Hilburn:
*Electric Circuit Analysis*

[Kers02] William H. Kersting:
*Distribution System Modeling and Analysis*

[KePh92] William H. Kersting, W. Howard Phillips:
*Modeling and Analysis of Rural Electric Distribution Feeders*

[King08] Lee King:
*Common Information Model for Distribution: An Introduction to the CIM for Integrating Distribution Applications and Systems*

[Kund94] Prabha Kundur:
*Power System Stability and Control*

[Lali05] Thierry Lalive d’Epiney:
*Strompanne der SBB vom 22. Juni 2005*
[LiTe96] Wei-Min Lin, Jen-Hao Teng:
Phase-Decoupled Load Flow Method for Radial and Weakly-Meshed Distribution Networks

[LuSe90] Guang-Xiang Luo, Adam Semlyen:
Efficient Load Flow for Large Weakly Meshed Networks

[Maie04] Ronald Maier:
Knowledge Management Systems: Information and Communication Technologies for Knowledge Management

[NoWi07] James Northcote-Green, Robert Wilson:
Control and Automation of Electrical Power Distribution Systems

[Prat04] Stephen Prata:
C Primer Plus

[Rols88] David W. Rolston:
Principles of Artificial Intelligence and Expert Systems Development

[RuAn07] Monika Ruh, Göran Andersson, Andreas Borer:
A New Concept for a Fully Transparent Distribution Management System

[RuAn08] Monika Ruh, Göran Andersson, Andreas Borer:
Power System Modeling for a Fully Transparent Distribution Management System

[RuAn09] Monika Ruh, Göran Andersson, Andreas Borer:
A Configuration Manager for a Fully Transparent Distribution Management System

[Ruh06] Monika Ruh:
*Power System Control Technology*

[Ruh09] Monika Ruh:
*Verification and Documentation of Power Flow Calculation Results*

[RuSu89] Dieter Rumpel, Ji Rong Sun:
*Netzleittechnik*

[RaWe00] Lennart Råde, Bertil Westergren:
*Springers Mathematische Formeln*

[Sacc03] Fabio Saccomanno:
*Electric Power Systems - Analysis and Control*

[ScMe05] Jürgen Schlabbach, Dieter Metz:
*Netzsystemtechnik*

[Schw75] Gerhard Schwickardi:
*Elektro-Energietechnik 1*

[Schw79] Gerhard Schwickardi:
*Elektro-Energietechnik 2*

[Schw80] Gerhard Schwickardi:
*Elektro-Energietechnik 3*
Fachschriftenverlag Aargauer Tagblatt AG, Aarau, 1980.

[SiGa00] Abraham Silverschatz, Peter Galvin, Greg Gagne:
*Applied Operating System Concepts*
[SiSa00] M. Pedro Silva, João Tomé Saraiva, Alexandre V. Sousa: 
*A Web Browser Based DMS - Distribution Management System*

[SliI06] Daniel C. Slilaty: 
*Bias Matroids with Unique Graphical Representations*
Department of Mathematics and Statistics, Wright State University, Dayton, 2006.

[Swan96] Michael Swan: 
*Practical English Usage*

[Swiss08] *Swissgrid annual report 2008*
Swissgrid AG, Corporate Communications, Laufenburg, 2008.

[Tiet06a] Ernst-Günther Tietze: 
*Netzleittechnik Teil 1: Grundlagen*

[Tiet06b] Ernst-Günther Tietze: 
*Netzleittechnik Teil 2: Systemtechnik*

[WaWa08] Monika Walser, Fritz Wagner: 
*UCTE annual report 2008*
Secretariat of UCTE, Brussels, 2008.

[Weis99] Mark Allen Weiss: 
*Data Structures and Problem Solving Using Java*
Addison Wesley, Reading, 1999.

[Yang95] Jen Tsi Yang: 
*An Outline of Scientific Writing*
Curriculum Vitae

December 13, 1977  Born in Männedorf, Switzerland

1985 - 1991  Primary school, Rüti, Switzerland

1991 - 1993  Secondary school, Rüti, Switzerland

1993 - 1998  Gymnasium, Kantonsschule Zürcher Oberland, Wetzikon, Switzerland

January 1998  Matura Type C (Mathematics and Sciences)

1998  Military service, signal corps of Swiss Army; Recruit school, non-commissioned officer school and practical service as corporal

1998 - 2005  Studies in Electrical Engineering and Information Technology at ETH Zurich, Switzerland

2001 - 2002  Military service, signal corps of Swiss Army; Officer school and practical service as lieutenant

May 2005  Master in Electrical Engineering and Information Technology at ETH Zurich, Switzerland

2005 - 2010  Assistant at the Power Systems Laboratory, ETH Zurich, Switzerland; PhD thesis under the supervision of Prof. Dr. Göran Andersson