Delft University of Technology
Faculty of Electrical Engineering, Mathematics and Computer Science
Delft Institute of Applied Mathematics

Computational approaches for the MINLP waste water network model

A thesis submitted to the
Delft Institute of Applied Mathematics
in partial fulfillment of the requirements

for the degree

MASTER OF SCIENCE
in
APPLIED MATHEMATICS

by

ANNE CORNELIA VAN SWEEDEN

Delft, the Netherlands
June 2016

MSc THESIS APPLIED MATHEMATICS

"Computational approaches for the MINLP waste water network model"

ANNE CORNELIA VAN SWEEDEN

Delft University of Technology

Daily supervisor

Dr. J.A. El-Omari

Responsible professor

Prof.dr. E. de Klerk

Other thesis committee members

Prof.dr.ir. K.I. Aardal

Dr. D.J.P. Lahaye

June 3, 2016

Delft, The Netherlands

Acknowledgments

Starting as a student in Delft, I was looking for a broad field in which I could apply my technical interest to hands-on problems. That is why I started Civil Engineering. After finishing my bachelor's degree there, it was Martin van Gijzen who convinced me that Applied Mathematics would offer me what I was looking for: using mathematics to solve concrete problems in the application of logistics. And he was right. During my master's degree, I continued to look for the interface between optimization, transportation, logistics and management. Combining my master's thesis with an internship at ORTEC, concludes my education and development in Delft in an appropriate way.

I would like to thank several people for their contribution during the realization of this master's thesis. First of all, I would like to thank my colleagues at ORTEC, who made sure that graduating was not as tough as I thought it would be. The numerous suggestions, tips, and coffee breaks made it a memorable time. Special thanks to Jawad, for providing me with honest and useful feedback, for keeping me on track and for learning me about X-men. In addition, I would like to thank Rianne for the nice chats and for thinking along on topics not only related to this thesis.
I am very grateful for the assistance of my responsible professor, Etienne de Klerk. Even though we met on Fridays, our meetings were pleasant and very valuable. Without his assistance, this report would not have looked the same, hence many thanks for guiding me in the right direction.
Last but not least, I would like to thank my parents and partner for believing in me. They have offered me both the encouragement and distraction I needed most.

During the process of writing this thesis, I learned a lot; both professionally and personally. I am grateful for this experience and had a lot of fun. I hope this shows while reading this thesis, and I hope that you will enjoy it!

Delft, University of Technology — A.C. van Sweeden
June 3, 2016

Preface

This thesis is written in service of completing the master's degree Applied Mathematics at Delft University of Technology. It shows how the discipline of applied mathematics bridges theoretical mathematics and its application on problems arising from practice. In this particular case, methods solving complex optimization programs are used to design and operate industrial waste water usage and treatment.

Management summary

In industry, waste water is used, treated, and reused in vast amounts in a variety of applications. For instance, it is used in processes as manufacturing, washing and refinery operations, during which the water gets contaminated. The increasing scarcity of industrial water and binding environmental regulations require the most efficient use and treatment of water possible. One way to improve efficiency is by integrating the water usage of processing units and treatment operations in one single network. As a result, the intake of fresh water can be reduced, as well as the operation of treatment facilities. Hence, a reduction in total costs can be obtained. The design and operation of such an integrated network at minimal costs can be challenging. One reason being that an accurate mathematical model of such a problem has to be a Mixed Integer Nonlinear Program (MINLP), with which solvers struggle.

This thesis aims to detect suitable solution methods to solve the problem of optimizing an integrated industrial waste water network. The main research question is:

What solution methods are suitable for solving the MINLP total waste water network problem?

An analysis of the optimization model and on existing modeling approaches resulted in the selection of seven solution methods that might be suitable to solve this problem. The selected methods were tested on a set of cases from theory and practice. An analysis of the computation times and quality of the solutions showed that the combined information of two solution methods obtains high quality solutions within acceptable computation times. The recommendation is to solve the original MINLP model with a local nonlinear solver to obtain a feasible solution. Then, assess the quality of the feasible solution by solving a linearized version of the MINLP, which serves as a tight lower bound on the optimal solution value.

This research provided a solution on the total waste water network problem and is applicable to related network optimization problems. In addition, it gives an indication on possible enhancement of the selected methods.

Table of contents

List of figures

List of tables

Nomenclature

Table 1: Sets and indices

Description	Set	Index
Contaminants	C	c
Network units	U	u
Process units	$PU \in U$	p
Treatment units	$TU \in U$	t
Sources	$WI \in U$	w
Sinks	$WO \in U$	d
Mixers	M	m
Mixers before treatment unit	$MT \subseteq M$	m_t
Mixers before process unit	$MP \subseteq M$	m_p
Mixers before sink	$MO \subseteq M$	m_d
Splitters after unit u	S	s
Splitters after treatment unit	$ST \subseteq S$	s_t
Splitters after process unit	$SP \subseteq S$	s_p
Splitters after source	$SI \subseteq S$	s_w
Pipe connections	PC	(s, m)
Fixed pipe connections	$PC_{fixed} \subseteq PC$	(s, m)
Optional pipe connections	$PC_{free} \subseteq PC$	(s, m)

Table 2: Variables Total Flow model

Variable	Description	# Variables
Continuous non negative variables		
ϕ_w	water intake from source w (ton/h)	w
$\phi_{p,in}$	flow rate into process unit p (ton/h)	p
$\phi_{p,out}$	flow rate out of process unit p (ton/h)	p
$\phi_{t,in}$	flow rate into treatment unit t (ton/h)	t
$\phi_{t,out}$	flow rate out of treatment unit t (ton/h)	t
$\rho_{p,c,in}$	concentration of contaminant c in process unit p (ppm)	pc
$\rho_{p,c,out}$	concentration of contaminant c out of process unit p (ppm)	pc
$\rho_{t,c,in}$	concentration of contaminant c in treatment unit t (ppm)	tc
$\rho_{t,c,out}$	concentration of contaminant c out of treatment unit t (ppm)	tc
ϕ_d	flow rate in discharge stream of sink d (ton/h)	d
$\rho_{d,c}$	concentration of contaminant c in discharge stream of sink d (ppm)	cd
$\phi_{s,m}$	flow rate from splitter s to mixer m (ton/h)	$(w+p+t)(p+t+d)$
$\rho_{s,m,c}$	concentration of contaminant c from splitter s to mixer m (ppm)	$c(w+p+t)(p+t+d)$
Binary variables		
$\psi_{s,m}$	binary variable for pipe existence between splitter s and mixer m $(-)$	$(w+p+t)(p+t+d)$

Table 3: Network parameters Total Flow model

Parameter	Description
Water sources	
$\overline{\phi}_w$	maximum flow rate from source w (ton/h)
$\rho_{w,c}$	concentration of contaminant c from source w (ppm)
Process units	
$\overline{\phi}_p$	maximum flow rate through process unit p (ton/h)
$\underline{\phi}_p$	minimum flow rate through process unit p (ton/h)
$\overline{\rho}_{p,c,in}$	maximum concentration of contaminant c entering process unit p (ppm)
$\overline{\rho}_{p,c,out}$	maximum concentration of contaminant c exiting process unit p (ppm)
$\phi_{p,\Delta}$	water added or subtracted in process unit p from water flow (ton/h)
$\omega_{p,c}$	discharge load of contaminant c into water flow through process unit p (kg/h)
Treatment units	
$\overline{\phi}_t$	maximum flow rate through treatment unit t (ton/h)
$\underline{\phi}_t$	minimum flow rate through treatment unit t (ton/h)
$\overline{\rho}_{t,c,in}$	maximum concentration of contaminant c entering treatment unit t (ppm)
$\overline{\rho}_{t,c,out}$	maximum concentration of contaminant c exiting treatment unit t (ppm)
$\lambda_{t,c}$	percentage of contaminant c removed by treatment unit t (%)
Water sinks	
$\overline{\phi}_d$	maximum discharge flow rate in sink d (ton/h)
$\underline{\phi}_d$	minimum discharge flow rate in sink d (ton/h)
$\overline{\rho}_{d,c}$	maximum concentration of contaminant c in discharge flow in sink d (ppm)
$\underline{\rho}_{d,c}$	minimum concentration of contaminant c in discharge flow in sink d (ppm)
Pipe connections	
$\overline{\phi}_{s,m}$	maximum flow rate from splitter s to mixer m (ton/h)
κ_{min}	minimum number of pipes in the system $(-)$
κ_{max}	maximum number of pipes in the system $(-)$
Recycling parameters	
$\theta_{TU} \in \{0,1\}$	TU recycling allowed or not
$\theta_{PU} \in \{0,1\}$	PU recycling allowed or not

Table 4: Cost objective parameters Total Flow model

Parameter	Description
Water sources	
π_w	price per ton of water from source w ($/ton$)
Treatment units	
$\pi_{t,IC}$	investment cost coefficient of treatment unit t ($)
$\pi_{t,OC}$	operating cost coefficient of treatment unit t ($/ton$)
α	investment discount factor for treatment units $(-)$
Pipe connections	
$\pi_{s,m,fix}$	fixed pipe cost ($)
$\pi_{s,m,var}$	variable pipe cost ($)
$\pi_{s,m,op}$	pipe operation cost per ton ($/ton$)
γ	investment discount factor for pipes $(-)$
Other parameters	
H	number of hours of operation per year ($h/year$)
AR	annualized factor for investment $(-)$

Chapter 1

Introduction

In industry, waste water is used, treated, and reused in vast amounts. For instance, it is used in processes as manufacturing, washing and refinery operations, during which the water gets contaminated. The increasing scarcity of industrial water and binding environmental regulations require the most efficient use, recycling and treatment of water possible.

It was most common to dispose the contaminated water to a separate central treatment facility after usage. In this facility, the water would be purified to meet effluent restrictions before disposal in the environment. However, combining the treatment operations and water usage processes in a single network, would reduce the amount of fresh water demand and treatment needed.

Such an integrated network, that contains both water usage processes and treatment processes, has been used more often over the last decades and is called a Total Waste Water Network (TWWN). The TWWN consists of one or more of the following elements, schematically illustrated in Figure 1-1: (i) sources that take water, e.g. fresh drinking water or seawater, into the network, (ii) process units (PU's), such as cooling systems, that use the water and increase the contamination levels, (iii) treatment units (TU's), like Ultra Violet disinfection and sand filters, that purify the water with a certain removal ratio, (iv) sinks, in which water is disposed, whereafter the effluent is returned into the environment.

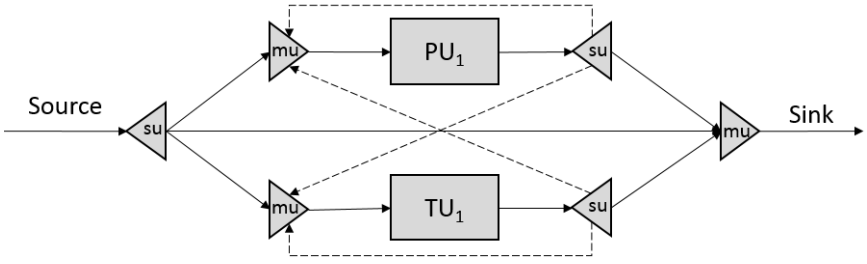


Figure 1-1: Schematic minimal example of a TWWN

Each sink, PU and TU is located after a mixer unit (mu) and each source, PU and TU is followed by a splitter unit (su), where different streams are combined and separated respectively. Between every splitter and mixer in the network, may exist a pipe connection.

1-1 Research motivation

For industries, it is important to operate the process units and to meet given effluent restrictions at minimal investment and operational costs. In order to meet these requirements, the question is which treatment facilities to include, which pipe connections to construct and how to operate the resulting network. This challenge is called the TWWN problem. This might be challenging, due to the rapid growth of possible unit arrangements within the network when the number of units increases. This is illustrated by Figure 1-2, which shows the *superstructure* of a TWWN with two PU's and two TU's, consisting of all possible network connections.

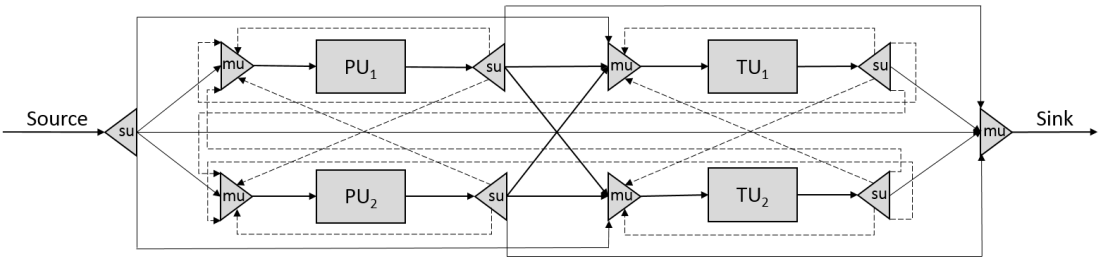


Figure 1-2: Superstructure of TWWN with two PU's and two TU's

The difficulty increases further when considering the specifications of flow rates and contaminant levels, not to mention the need to have a robust network design suitable for a wide range of input and output requirements. There are several variations within the domain of TWWN problems. They vary e.g. in: (1) the type of treatment facilities and (2) optimization objective. The type of TWWN studied here is characterized by the following:

1. The treatment units remove contamination with a fixed removal ratio.
2. The objective is to minimize costs with respect to source water intake, investment and operation of treatment units and investment and operation of pipe connections.

The design and operation problem of the TWWN can be modeled as a mathematical optimization program. It contains continuous variables and binary variables: the continuous variables represent the flow rates and contaminant levels in every part of the network, the binary variables represent the existence of a possible pipe connection in the network. Constraints are added to guarantee the network restrictions and water and contamination conservation within the network. The constraints that represent the balance conservation in a network are typically bilinear constraints. Accordingly, the result is a Mixed Integer Nonlinear Program (MINLP), which can be solved with a global

solver. Current global solvers have difficulty solving this type of problem to global optimality within reasonable time, if even at all. Finding a feasible solution is not even guaranteed. Since this type of problem appears in multiple applications [Misener and Floudas, 2010, Furman and Androulakis, 2008] and responds to current developments, alternative *suitable solution methods* are desired. Here, a solution method is defined as the combination of a mathematical model and a solver. A suitable method should provide a solution on the original design and operation TWWN problem with a certain accuracy, which is to be specified later, and within reasonable time. A common approach is to linearize the model, such that a Mixed Integer Program (MIP) remains to be solved. Since a wide variety of specialized linear solvers is available, the resulting problem can be solved more easily.

1-2 Research questions

This thesis aims to detect suitable solution methods for solving the MINLP TWWN problem and, moreover, to provide understanding about when to use which approach. The focus of this research is on modeling rather than search algorithms. This is the basis of the research question which is stated as follows:

What solution methods are suitable for solving the MINLP total waste water network problem?

The answers on the following three sub-questions yield a conclusion to the main question:

1. What are the characteristics of the MINLP formulation of the TWWN problem that affect the selection of a suitable solution method?
2. What modeling approaches exist in literature and how can they be implemented to solve the TWWN problem?
3. What is the performance of the proposed methods when applied to different problem instances?

1-3 Methodology

In order to answer the research questions, an iterative approach is used. This approach consists of five steps. The relation between these steps and the chapters of this report is illustrated in Figure 1-3.

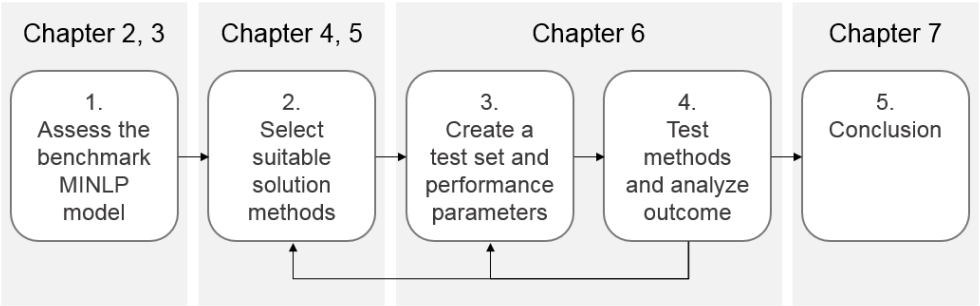


Figure 1-3: Reading guide

In the first step, the MINLP model of the TWWN problem is assessed in terms of difficulty and nonlinear characteristics. Based on this analysis, the second step is to find suitable solution methods. The process of obtaining such methods is illustrated in Figure 1-4, including the resulting methods and the chapters discussing them. Each method is obtained by first choosing between a linear or nonlinear model. If it is the latter, it is combined with a suitable nonlinear solver. If it is the former, further decisions on the linearization and formulation techniques have to be made, before selecting a suitable solver.

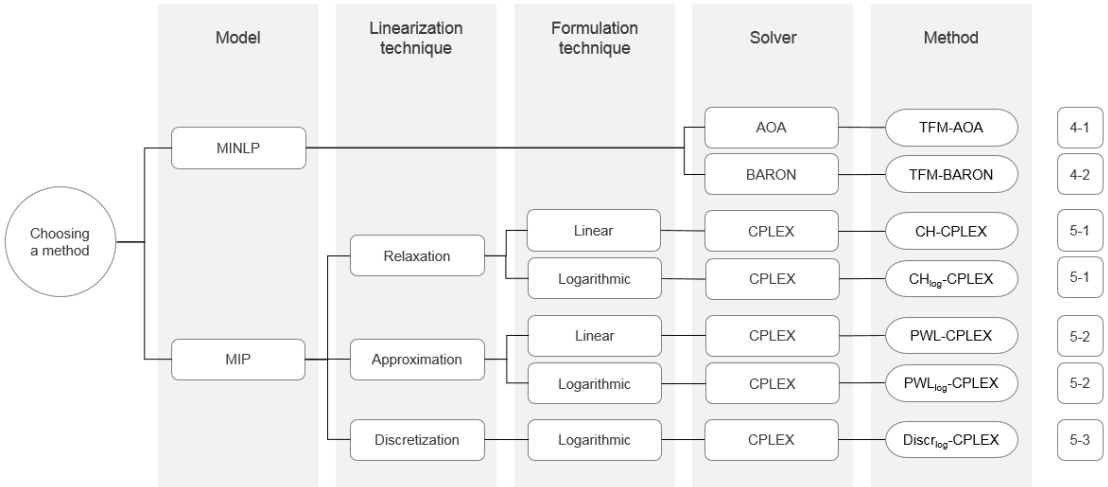


Figure 1-4: Solution methods

The resulting methods are tested on a set of TWWN instances taken from literature and real life application, and based on these experiments, the methods are assessed according to objective function value, running time and solution characteristics.

1-4 Main findings

An analysis on the performance of the seven methods from Figure 1-4 showed that it is difficult to obtain a high quality solution in reasonable time using TFM-BARON. This can be improved slightly if a feasible initial solution is provided. As an alternative, several approximation methods can be applied, based on different linearization techniques and model formulations. Two of these methods, obtained by a linear relaxation of the model, CH-CPLEX and CH_{log}-CPLEX, result in an accurate lower bound for the original model, where CH_{log}-CPLEX outperformed CH-CPLEX with respect to running time. Two other methods, based on a model obtained by piecewise linear approximation, PWL-CPLEX and PWL_{log}-CPLEX, obtain equally accurate approximations. However, these approximations are not guaranteed to be either an upper bound or lower bound on the global optimal solution.

A feasible upper bound on the global optimum can be found by using a local nonlinear solver, as in TFM-AOA. Another method, $Discr_{log}$-CPLEX, solves a linearized model, based on disretization of one of the variables constituting the bilinear terms. Both methods yield an equally accurate upper bound, but overall, TFM-AOA outperforms $Discr_{log}$-CPLEX with respect to running time. However, both methods have no guarantee on the solution quality.

In summary, combining the methods TFM-AOA and CH_{log}-CPLEX, is a promising approach to find a high quality feasible solution to the MINLP TWWN problem, with a known maximal gap to the global optimal solution.

The outline of this thesis is as follows. First, the MINLP model is formulated and analyzed in Chapter 2. Then, the computational complexity of the TWWN problem is specified in Chapter 3. Suitable solution methods are selected in the following two chapters: Chapter 4 assesses two methods that incorporate a nonlinear model formulation, and Chapter 5 discusses five solution methods based on linear model formulations. The different solution methods are tested in Chapter 6 on a set of instances, including the experimental results. Finally, Chapter 7 concludes this thesis and provides recommendations.

Chapter 2

Mathematical Model

In this chapter, the MINLP formulation of the TWWN model is assessed. The MINLP model is introduced in Section 2-1, formulated in Section 2-2, validated in Section 2-3 and analyzed on its nonlinear characteristics in Section 2-4. The analysis from this chapter will be input for choosing suitable solution methods.

2-1 MINLP models

The waste water network problem was firstly introduced in 1980 in [Takama et al., 1980], aiming to optimize a waste water treatment network. Since then, many mathematical formulations and configurations have been presented. In 1994, the integrated TWWN problem was raised in [Wang and Smith, 1994]. Not only mathematical optimization models, but also other approaches have been applied, such as the ones in [Wang and Smith, 1994]. The interested reader is referred to [Halim et al., 2015] for a recent survey, where it was shown that many approaches can obtain high quality solutions, but without guarantee on optimality. Solving the MINLP problem formulation with a global solver does guarantee convergence to the global optimal solution. Due to the computational complexity however, running time is an issue. The complexity is further specified in Chapter 3.

A general formulation of a MINLP problem for this type of network problem can be stated as follows, based on [Quesada and Grossmann, 1992]:

$$
\begin{aligned}
z = \min_{x,y} \quad & c^{\intercal} y + f(x), \\
\text{s.t.} \quad & g(x) + By \leq 0, \\
& x \in X = \{x \mid x \in \mathbb{R}^n, x^L \leq x \leq x^U\}, \\
& y \in Y = \{y \mid y \in \{0,1\}^m, By \leq b\}.
\end{aligned}
\tag{MINLP}
$$

The objective function z consists of investment and operational cost terms. The nonlinearities are contained both in the objective function in $f(x)$, and in the optimization

constraints in $g(x)$. The function $g(x)$ is defined as:

$$g(x) = \begin{bmatrix} g_1(x) & g_2(x) & \dots & g_p(x) \end{bmatrix}^\intercal .$$

The matrix B may contain zero row vectors, resulting in constraints which only involve continuous variables. Note that (MINLP) is linear with respect to its binary variables y. In case nonlinear terms including binary variables are present, the MINLP formulation can be reformulated into the general form of (MINLP) by introducing additional constraints and continuous variables.

Two ways exist to model the TWWN as a MINLP formulation, see [Quesada and Grossmann, 1995]. The main difference is the use of material balance versus mass balance equations for contaminant flows. The model that uses material balance equations formulation is hereafter called the *Total Flow Model (TFM)*. The model using mass balance equations will be called the *Component Based Model (CBM)*. The TFM contains variables representing total water flow rates and contaminant concentrations and was used in [Ahmetović, 2011]. The CBM contains variables for individual component flow rates in a stream, as applied in [Galan and Grossmann, 1998, Galan and Grossmann, 1999]. Using material balances, as in the TFM, results in a model with two types of variables: water flow rate variables ϕ (ton/h) and contaminant concentration variables ρ (ppm). Using mass balances, as in the CBM model, on the other hand, yields a model with the same water flow rate variables ϕ (ton/h), but with contamination flow rate variables f (kg/h) instead.

These formulations yield different models and it might accordingly affect the selection of suitable solution methods, the solving time and solution quality. Some statements about the differences and effects are made in [Karuppiah and Grossmann, 2006] without any further substantiation: for an equal number of splitter and mixer units in a system, the TFM would involve fewer bilinearities, which is an advantage with respect to solving time. Another advantage over the second model would be the uniform order of magnitude of bounds on the variables, which improves the model's numerical scalability. The impact and specification of both arguments is absent. Therefore, a more detailed research is conducted in order to make a clear distinction between these models and detect corresponding benefits and drawbacks, see Appendix A. After a detailed comparison, a conclusion on the best suited model formulation can be found in Appendix A-3. It is concluded that the TFM would indeed be the most promising formulation, due to the superior numerical solvability. For the sake of completeness, the CBM formulation is stated in Appendix A-1. The next section states the TFM, which is the benchmark MINLP model for this research.

2-2 TFM formulation

The TWWN problem will be formulated in the form of (MINLP). First, the sets, variables and parameters are introduced, after which the model constraints and objective value z are defined.

Sets and indices

The sets and corresponding indices are stated on page xiii in Table 1. These sets define the different types of network elements.

For the mixers and splitters it holds that $MT \cup MP \cup MO = M$ and $ST \cup SP \cup SI = S$.

Variables

Three types of model variables are distinguished:

1. Continuous non negative flow rate variables, defined by ϕ (ton/h). These variables are defined for each network unit and pipe connection.
2. The contaminant concentration variables are defined by ρ. This dimensionless variable represents the fraction of contaminant. Here, (ppm) is taken as unit of measurement, like is used in [Karuppiah and Grossmann, 2006]. The underlying assumption is that all molecules present in the flow are comparable in weight. These variables are defined for each network unit and pipe connection as well.
3. The last type of variables are (iii) binary variables ψ, define the topology of the network. These variables represent the existence or non-existence of every optional pipe connection from a splitter s to a mixer m in the network.

All variables are specified in Table 2 on page xiv. The network from Figure 1-1 for example, would yield nine variables $\psi_{s,m}$. The same holds for the continuous variables, which in addition are defined for each flow entering or leaving a network unit. One exception is the contaminant concentration of the source flow entering the network, $\rho_{w,c}$, which is fixed as input parameter. For the example from Figure 1-1, the total amount of both flow rate variables ϕ and contaminant concentration variables ρ is 15. Which is a result of adding the total splitter to mixer connections (9), flows entering a unit (3) and flows leaving a unit (3). A complete overview of the model variables is stated in Table 2, including a parameterized variable count in the last column.

Parameters

Together with the superstructure of the waste water network that has to be optimized, the TFM takes a variety of input parameters into account. These parameters define the following network characteristics:

- Bounds on maximum and minimum flow rates ϕ and contaminant concentrations ρ.
- Process unit properties such as the contamination rate ω.
- Treatment unit properties such as the contamination removal rate λ.
- Allowance of recycling around treatment and process units.
- Cost objective parameters.

A complete overview of all network parameters is stated in Table 3 on page xv. The cost objective parameters are listed on page xvi in Table 4.

Constraints

With the introduction of the variables and parameters in the previous paragraphs, the model constraints can be formulated. All constraints are either bound constraints or balance constraints and are defined both for water flow rates and contaminant concentrations. Bound constraints contain upper and lower limits and the balance constraints ensure flow and contaminant continuity in all network elements. The bound and balance constraints are grouped per network element type and are explained below. In addition one redundant constraint is added to the model formulation, representing the total network contaminant balance.

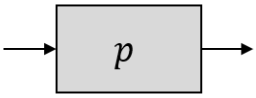
Water source

Source

Bounds on water intake :

$$\phi_w \leq \overline{\phi}_w \quad \forall w. \tag{2-1}$$

Process unit



Process unit

Bounds on water intake:

$$\underline{\phi}_p \leq \phi_{p,in} \leq \overline{\phi}_p \quad \forall p. \tag{2-2}$$

Bounds on contaminant concentration intake and output:

$$\rho_{p,c,in} \leq \overline{\rho}_{p,c,in} \quad \forall p, c, \tag{2-3}$$

$$\rho_{p,c,out} \leq \overline{\rho}_{p,c,out} \quad \forall p, c. \tag{2-4}$$

Water flow rate balance around unit:

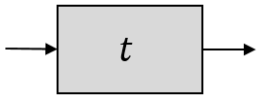
$$\phi_{p,out} = \phi_{p,in} + \phi_{p,\Delta} \quad \forall p. \tag{2-5}$$

Mass contaminant balance around unit:

$$\phi_{p,out} \cdot \rho_{p,c,out} = \phi_{p,in} \cdot \rho_{p,c,in} + 1000 \cdot \omega_{p,c} \quad \forall p, c. \tag{2-6}$$

In Equation (2-5), the water flow rate out, $\phi_{p,out}$ is obtained by adding the water produced by the process unit, $\phi_{p,\Delta}$ to the flow rate entering the unit $\phi_{p,in}$.
In Equation (2-6), the contaminant mass leaving the process unit is a result of adding the contaminant mass entering the unit and the production of contaminant in the process unit $\omega_{p,c}$ (kg/h).

Treatment unit



Treatment unit

Bounds on water intake:

$$\underline{\phi_t} \leq \phi_{t,in} \leq \overline{\phi}_t \quad \forall t. \tag{2-7}$$

Bounds on contaminant concentration intake and output:

$$\rho_{t,c,in} \leq \overline{\rho}_{t,c,in} \quad \forall t, c, \tag{2-8}$$

$$\rho_{t,c,out} \leq \overline{\rho}_{t,c,out} \quad \forall t, c. \tag{2-9}$$

Water flow rate balance around unit:

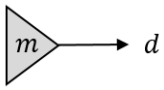
$$\phi_{t,in} = \phi_{t,out} \quad \forall t. \tag{2-10}$$

Contaminant concentration balance around unit:

$$\rho_{t,c,out} = (1 - \frac{\lambda_{t,c}}{100}) \cdot \rho_{t,c,in} \quad \forall t, c. \tag{2-11}$$

The contaminant concentration intake $\rho_{t,c,in}$ is reduced by the treatment unit with a percentage of $\lambda_{t,c}$, resulting in the remaining contaminant concentration $\rho_{t,c,out}$.

Water sink



Sink

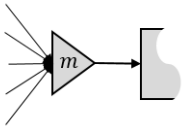
Bounds on water intake:

$$\underline{\phi_d} \leq \phi_d \leq \overline{\phi}_d \quad \forall d. \tag{2-12}$$

Bounds on contaminant concentration intake:

$$\underline{\rho_{d,c}} \leq \rho_{d,c} \leq \overline{\rho}_{d,c} \quad \forall d, c. \tag{2-13}$$

Unit mixer



Mixer

Water flow rate balance around mixer:

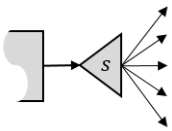
$$\phi_{u,in} = \sum_s \phi_{s,m_u} \quad \forall u. \tag{2-14}$$

Contaminant mass balance around mixer:

$$\phi_{u,in} \cdot \rho_{u,c,in} = \sum_s \phi_{s,m_u} \cdot \rho_{s,m_u,c} \quad \forall u, c. \tag{2-15}$$

The sum of contaminant masses entering the mixer unit equals in the contaminant mass leaving the mixer unit.

Unit splitter



Splitter

Water flow rate balance around splitter:

$$\phi_{u,out} = \sum_{m} \phi_{s_u,m} \quad \forall u. \tag{2-16}$$

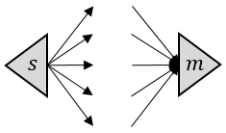
Contaminant concentration preservation around splitter:

$$\rho_{s_u,m,c} = \rho_{u,c,out} \cdot \psi_{s_u,m} \quad \forall u, c \text{ s.t. } (s_u, m) \in PC_{free}, \tag{2-17}$$

$$\rho_{s_u,m,c} = \rho_{u,c,out} \quad \forall u, c \text{ s.t. } (s_u, m) \in PC_{fixed}. \tag{2-18}$$

The contaminant concentration leaving the splitter is preserved in any connection for fixed connections. In case the connection s_u to m is optional and present, the binary variable $\psi_{s_u,m}$ has value one.

Pipe constraints



Pipe

Bounds on water flow rate:

$$\phi_{s,m} \leq \overline{\phi}_{s,m} \cdot \psi_{s,m} \quad \forall (s,m) \in PC_{free}, \tag{2-19}$$

$$\phi_{s,m} \leq \overline{\phi}_{s,m} \quad \forall (s,m) \in PC_{fixed}. \tag{2-20}$$

Bound on number of connections:

$$\kappa_{min} \leq |PC_{fixed}| + \sum_{\substack{(s,m) \\ \in PC_{free}}} \psi_{s,m} \leq \kappa_{max}. \tag{2-21}$$

Bound on recycling

$$\phi_{s_t,m_t} \leq \theta_{TU} \cdot \overline{\phi}_{t,in} \quad \forall t, \tag{2-22}$$

$$\phi_{s_t,m_t} \leq \theta_{PU} \cdot \overline{\phi}_{p,in} \quad \forall p. \tag{2-23}$$

The total number of pipe connections in Equation (2-21) is bounded by the minimum number κ_{min} and maximum number κ_{max}.
If recycling around a treatment unit is not allowed, the binary parameter θ_{TU} is zero and the flow from a unit to the same unit, ϕ_{s_t,m_t} is bounded by zero. The same holds for the process units.

Total network contamination balance (redundant)

Network contaminant mass balance:

$$\sum_{w} \phi_w \cdot \rho_{w,c} + 1000 \sum_{p} \omega_{p,c} = \sum_{d} \phi_d \cdot \rho_{d,c} + \sum_{t} \phi_t \cdot (\rho_{t,c,in} - \rho_{t,c,out}) \quad \forall c. \tag{2-24}$$

This constraint represents the total network contaminant mass balance. It is already implied by other constraints and is therefore redundant. Sometimes it is useful though, to include Equation (2-24) for algorithmic purposes, see e.g. [Karuppiah and Grossmann, 2006].

Objective function

The objective of minimizing annual network costs consists of five terms concerning treatment unit costs, pipe connection costs and costs for intake of source water. Both investment costs and operating costs are included. The latter depend on the amount of operation hours per year. These five terms are:

i Pipe connection investment costs dependent on the number of connections and flow rate through the pipe.

ii Pipe connection operating costs dependent on flow rate through the pipe.

iii Source water operating costs dependent on the amount of water intake.

iv Treatment unit investment costs dependent on flow rate through treatment unit.

v Treatment unit operating costs dependent on flow rate through treatment unit.

This leads to the objective's mathematical formulation:

$$z = \min_{\phi,\rho,\psi} \quad \underbrace{AR \cdot \left(\sum_{\substack{(s,m) \\ \in PC_{free}}} \pi_{s,m,fix} \cdot \psi_{s,m} + \sum_{\substack{(s,m) \\ \in PC_{fixed}}} \pi_{s,m,fix} + \sum_{(s,m)} \pi_{s,m,var} \cdot (\phi_{s,m})^{\gamma} \right)}_{\text{i}} \tag{2-25}$$

$$+ \quad \underbrace{H \sum_{s} \sum_{m} \pi_{s,m,op} \cdot \phi_{s,m}}_{\text{ii}} + \underbrace{H \sum_{w} \phi_w \cdot \pi_w}_{\text{iii}}$$

$$+ \quad \underbrace{AR \sum_{t} \pi_{t,IC} \cdot (\phi_t)^{\alpha}}_{\text{iv}} + \quad \underbrace{H \sum_{t} \pi_{t,OC} \cdot \phi_t}_{\text{v}} \tag{2-26}$$

Note that in many cases in literature, the cost terms concerning pipe connections (i and ii) are excluded from the objective function.

The mathematical definition of the objective concludes the formulation of the TFM in Equations (2-1) - (2-25).

2-3 Model validation

In order to validate the TFM formulation, it is applied to the four cases used in [Karuppiah and Grossmann, 2006]. They solved the MINLP TWWN problem with the global solver BARON 7.2. The global solver used here is BARON 15 [Sahinidis, 1996]. The solution and solution value from the TFM are compared with the ones it is based on from [Karuppiah and Grossmann, 2006]. These results should display the smallest possible deviation. If this is the case, the TFM model is verified as a representative MINLP model.

These test instances K1, K2, K3 and K4 consist of one single source and sink, up to five process units and up to three treatment units. The instances are specified in Appendix E-1. Since the experiments from Karuppiah and Grossmann did not include costs with respect to pipe connections, the corresponding terms i and ii are not included in the objective out during this comparison. The maximum running time is set to 10800 seconds. Table 2-1 shows the solution values for the instances K1, K2 and K3. The fourth instance, instance K4, is left out during this comparison, since the solver exceeded the maximum running time. Figure 2-1 shows the optimal network configuration for instance K1 as found both in literature, and with solving the TFM. It shows that the difference between the two solutions is negligible for all three instances.

Table 2-1: MINLP models objective value comparison

Instance	Pipe costs	Literature soln. ($/year)	TFM soln. ($/year)	Difference (%)
K1	no	584016.90	584016.97	0.00
K2	no	381751.35	381751.35	0.00
K3	no	874057.37	874057.36	0.00

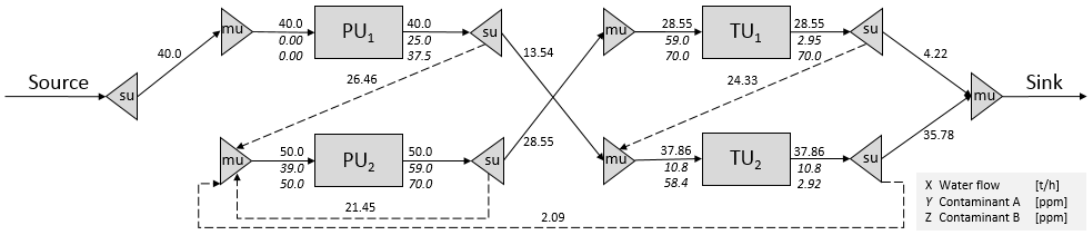


Figure 2-1: Optimal network Instance K1 for TFM

It is interesting to note that solving the TFM for Instance K2 obtains a different solution than the one from [Karuppiah and Grossmann, 2006]. At least three different solutions can be found with the same global optimal objective value. These solutions are shown in Figure 2-2 to 2-4. The solution in Figure 2-2 is from [Karuppiah and Grossmann, 2006], the other two solutions from Figure 2-3 and 2-4 were found by solving the TFM with BARON. Since pipe costs are excluded from the optimization, all costs are determined by the intake of source water, the number of treatment units and the flow rates through each treatment unit. For all network configurations shown in Figure 2-2 to 2-4, these

three cost factors are equal. Therefore the existence of multiple optima is caused by the multiple possibilities of directing water flows through a network with a fixed topology.

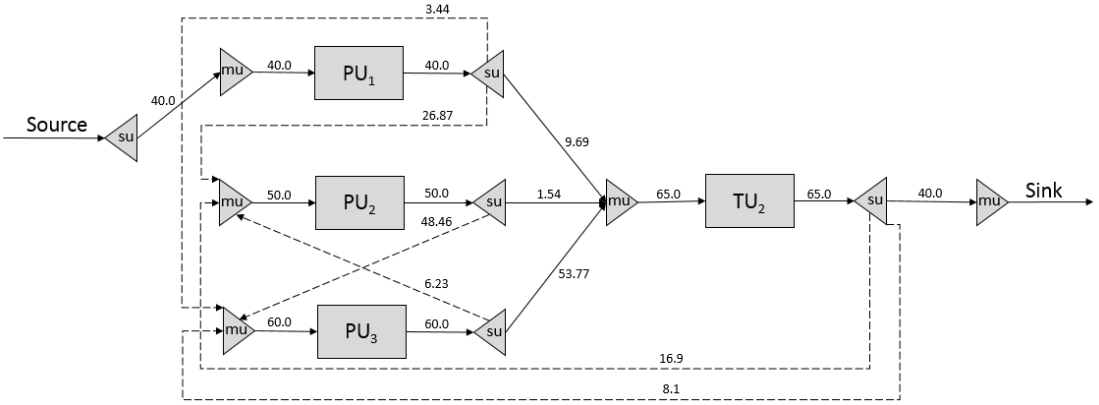


Figure 2-2: Optimal water flow Instance K2 - [Karuppiah and Grossmann, 2006]

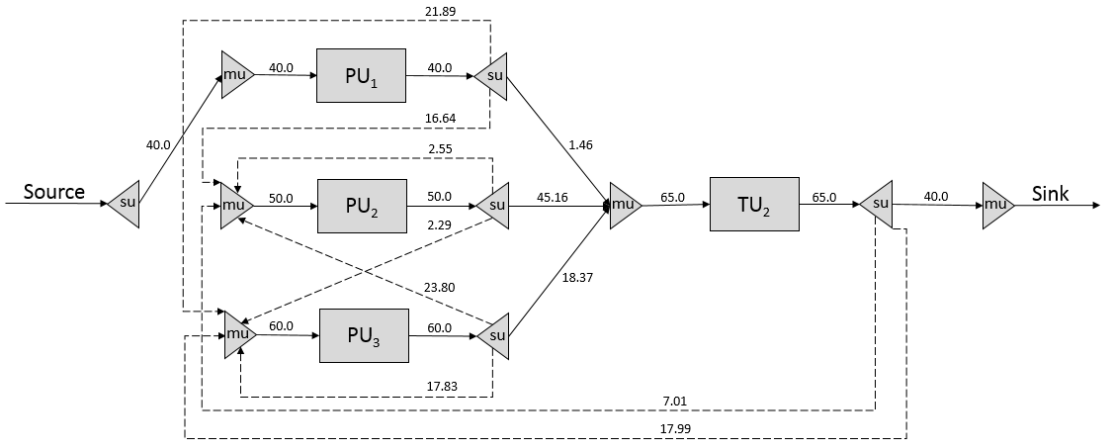


Figure 2-3: Optimal water flow Instance K2 - TFM

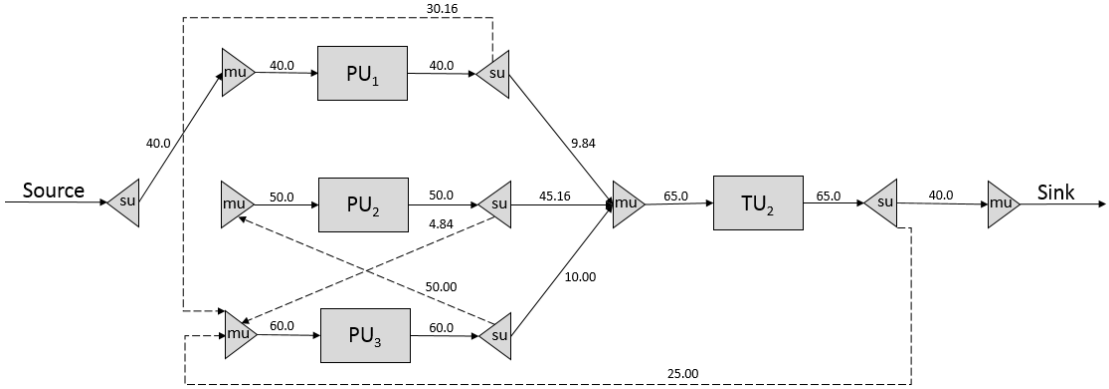


Figure 2-4: Alternative optimal water flow Instance K2 - TFM

2-4 Nonlinear characteristics of the MINLP model

Now that the TFM is verified as benchmark MINLP model. It is interesting to analyze the different nonlinear elements. Since it is these elements that solution methods struggle with, the type of nonlinearities determine which alternative methods would be suitable to solve the TWWN problem. The TFM formulation contains three different types of nonlinearities:

1. Equations containing a product of a continuous and binary variable in Equation (2-17).
2. Equations containing bilinear terms: terms that are linear in each of its arguments. These occur in Equations (2-6), (2-15) and (2-24).
3. Exponential terms in the objective function z, Equation (2-25).

The following paragraphs discuss these nonlinear elements and possibilities to deal with them.

Product of continuous and binary variables

The formulation of this type of nonlinearity is used in Equation (2-17) with two non-negative continuous variables and one binary variable:

$$\rho_{s_u,m,c} = \rho_{u,out,c}\psi_{s_u,m} \quad \forall u, c \text{ s.t. } (s_u, m) \in P_{free}. \tag{2-27}$$

This equation ensures contaminant concentration preservation in an existing pipe connection leaving a unit. If no pipe connection exists, the binary variable ψ is zero. Hence, the contaminant concentration is forced to be zero too. As stated in Section 2-1 on page 8, this type of nonlinearity can be reformulated into a set of linear constraints such that the resulting model formulation fits the general formulation of (MINLP). The upper and lower bounds of the continuous contaminant concentration variables, $\rho^L_{u,out,c}$ and $\rho^U_{u,out,c}$, are used to rewrite Equation (2-17) by three equations:

$$\begin{aligned}
&\rho_{s_u,m,c} \leq \rho^U_{u,out,c}\psi_{s_u,m}, \\
&\rho_{s_u,m,c} \geq \rho_{u,out,c} - (1-\psi_{s_u,m})\rho^U_{u,out,c}, \qquad \forall c, (s_u, m) \in P_{free} \\
&\rho_{s_u,m,c} \leq \rho_{u,out,c} + (1-\psi_{s_u,m})\rho^U_{u,out,c}.
\end{aligned} \tag{2-28}$$

Here, each splitter s_u corresponds to the unit u which is located right after the splitter. Replacing the constraint from Equation (2-17) with the three constraints from Equation (2-28), removes this type of nonlinearity from the MINLP model.

Bilinear terms

The bilinear terms contain a product of a flow rate variable and a contaminant concentration variable. These terms are used in balance equations around units and mixers, occurring in Equations (2-6), (2-15), (2-24). They can be generalized as:

$$\phi_1\rho_{1,c} = \phi_2\rho_{2,c} + \phi_3\rho_{3,c} + \ldots \quad \forall c. \tag{2-29}$$

The characteristics of this type of equation can be stated as follows:

- Both the left-hand side and right-hand side contain bilinear terms.
- The right-hand side contains multiple bilinear terms.
- All variables occur only once.

In contrast to Constraints (2-27), these nonlinearities cannot be reformulated to linear equations. Another property of Equation (2-29) is the fact that it is *nonseparable*. In contrast to a nonseparable function, a *separable* multivariate function can be reformulated to a sum of terms, consisting of at most one variable, i.e. univariate terms [Belotti et al., 2013]. For example the bilinear function

$$z = xy,$$

can be reformulated as:

$$\log(z) = \log(x) + \log(y).$$

In this case, one dimensional nonlinearities are left to be dealt with after reformulation. However, Equation (2-29) is nonseperable due to the multiple terms at the right hand side. Moreover, the *log* operator is undefined at zero. Therefore, the equation holds the property of being nonseparable. This property is of importance when selecting suitable model reformulations for the TWWN problem.

Due to the fact that the bilinear terms are the most frequent occurring nonlinearities in the MINLP TWWN problem, these equations guide the search for a suitable solution method.

Exponential terms

Besides the model constraints, the objective function contains nonlinear terms as well. In specific, the following two terms describe the investment costs of the treatment units and pipe costs respectively:

$$AR \sum_{s} \sum_{m} \pi_{s,m,var} (\phi_{s,m})^{\gamma} + AR \sum_{t} \pi_{t,IC} (\phi_t)^{\alpha}.$$

Both terms contain a nonlinear element describing a flow rate variable to the power of an investment factor. By replacing these terms by a linear approximation, the only nonlinear elements remaining in the MINLP model are the bilinear terms. This situation provides a good starting position for a comparison of solution methods. In order to linearize the objective, the two terms are approximated by the linear under estimators $\hat{\phi}_t$ and $\hat{\phi}_{s,m}$, illustrated in Figure 2-5 for $\hat{\phi}_t$.

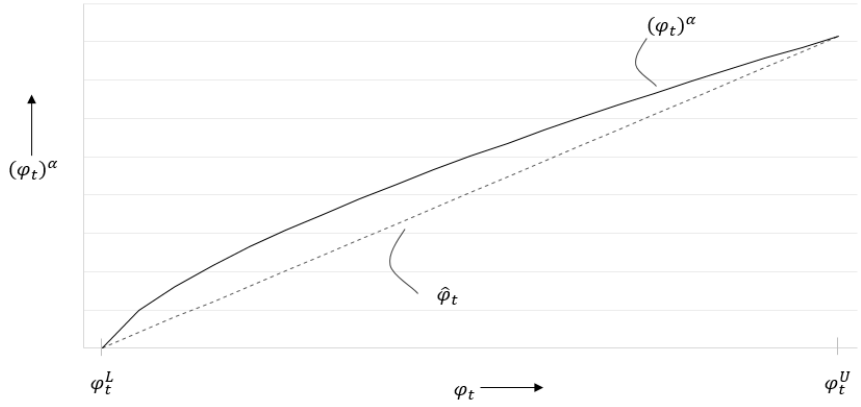


Figure 2-5: Cost factor under estimator $\hat{\phi}_t$ for flow rate ϕ_t and investment discount factor α

These estimators are defined by two equations in form of the linear equation $y \geq ax + b$ with slope a and intersection with the y-axis b:

$$\hat{\phi}_t \geq (\phi_t^L)^\gamma + \left(\frac{(\phi_t^U)^\gamma - (\phi_t^L)^\gamma}{\phi_t^U - \phi_t^L} \right) (\phi_t - \phi_t^L) \qquad \forall t, \tag{2-30}$$

$$\hat{\phi}_{s,m} \geq (\phi_{s,m}^L)^\alpha + \left(\frac{(\phi_{s,m}^U)^\alpha - (\phi_{s,m}^L)^\alpha}{\phi_{s,m}^U - \phi_{s,m}^L} \right) (\phi_{s,m} - \phi_{s,m}^L) \qquad \forall s, m. \tag{2-31}$$

The effect of these linear relaxations is a linear objective to replace the original one from Equation (2-25):

$$\begin{aligned} \min_{\phi,\rho,\psi} \quad & \underbrace{AR \left(\sum_{\substack{(s,m) \\ \in PC_{free}}} \pi_{s,m,fix} \psi_{s,m} + \sum_{\substack{(s,m) \\ \in PC_{fixed}}} \pi_{s,m,fix} + \sum_{(s,m)} \pi_{s,m,var} \hat{\phi}_{s,m} \right)}_{\text{i}} \\ & + \underbrace{H \sum_s \sum_m \pi_{s,m,op} \phi_{s,m}}_{\text{ii}} + \underbrace{H \sum_w \phi_w \pi_w}_{\text{iii}}. \\ & + \underbrace{AR \sum_t \pi_{t,IC} \hat{\phi}_t}_{\text{iv}} + \underbrace{H \sum_t \pi_{t,OC} \phi_t}_{\text{v}} \end{aligned} \tag{2-32}$$

For the remainder of this thesis, this linearized objective function is used for all solution methods. The complete formulation of the TFM with linear objective function in Equations (2-1) - (2-24), (2-30) - (2-32) is stated in Appendix B.

The effects of linearizing the objective function are shortly assessed next, based on instances K1, K2 and K3, and including all five cost terms in the objective. First, the instances were solved using the TFM as stated in Appendix B. Next, the nonlinear objective function was used for the same solution. The solution values, obtained by both objectives, are stated Table 2-2. The last column represents the relative cost underestimation due the linearized objective function. These results show that the annual costs are underestimated with 2.5-5%, using the linear objective function.

Table 2-2: Impact of using a linearized objective function for the TFM

Instance	Soln. ($/year)		Relative under-estimation *(%)*
	Lin. obj.	Nonlin. obj.	
K1	5.804E+5	5.940E+5	2.34
K2	3.781E+5	3.946E+5	4.37
K3	8.696E+5	8.940E+5	2.79

In addition, the TFM with nonlinear objective was used to solve the instances, in order to observe whether a different solution is optimal for the linear objective function. For all three instances, the solution for both objective functions was the same. Even though this short assessment does not prove the effects of using a linear objective function, it does provide a first indication that it is representative for the solution of the TFM with the original nonlinear objective function.

2-5 Summary of the mathematical models

The TFM provides a verified MINLP formulation of the TWWN problem. It is based on modeling water flow rates and contaminant concentrations. The model contains three types of nonlinearities:

- Constraints that include a product of a continuous and binary variable.
- Bilinear terms of two continuous variables.
- Exponential terms in the objective function.

The first type of nonlinearity can be eliminated from the model by the introduction of three alternative constraints. The exponential terms are replaced by linear underestimators. The presence of the bilinear terms is therefore binding for the selection of suitable solution methods in Chapter 4 and 5.

Chapter 3

Computational complexity

Up to now, solving the MINLP TWWN problem has been classified as "difficult to solve", but without clear definition of "difficult". This chapter elaborates on the computational complexity in more detail. In order to do so, definitions from computational complexity theory are used.

In computational complexity theory, problems can be divided in different complexity classes. This classification is based on the rate of growth of computation time needed when the the size of the problem input increases. Two important classes are class $\mathcal{P}$ and class $\mathcal{NP}$. Class $\mathcal{P}$ contains all problems that can be solved in polynomial time. For the problems in class $\mathcal{NP}$, this might not be the case. A third class is the class of NP-hard problems. A definition of this class is given in [Pinedo, 2012], Appendix D:

Definition 3.1. *A problem P, either a decision problem or an optimization problem, is called NP-hard if the entire class of $\mathcal{NP}$ problems polynomially reduces to P.*

All problems that are NP-hard can be classified to be as least as hard as the hardest problem in class $\mathcal{NP}$. A fundamental concept used here is *problem reduction*. A problem P is said to reduce to P', if for any instance of P, an equivalent instance of P' can be constructed. Once a problem P is reduced to P', this implies that if a polynomial time algorithm exists for P', it also exists for P. This concept enables to show difficulty equivalences between problems based on reduction.

The remainder of this chapter aims to prove that it is not guaranteed that the MINLP TWWN problem can be solved in polynomial time. The proof is based on a reduction from the related *pooling problem* to the TWWN problem. It shows that the pooling problem can be formulated as a special case of the TFM from Section 2-2. In particular, this proves that the TWWN problem is NP-hard.

3-1 The pooling problem

The pooling problem arises in the application of refinery processes [Gupte et al., 2016]. The problem objective is to minimize costs such that raw materials are mixed in pools before being sent to the final demand outputs while meeting certain specification criteria. This bilinear optimization problem is proven to be strongly NP-hard in Section 3, Proposition 1, from [Alfaki and Haugland, 2013]. This proof holds even for a fixed number of pools. For this proof, a polynomial reduction from the NP-hard *maximum independent vertex set* (MIVS) problem is used.

There are different formulations of the pooling problem. The one used here is given in Section 2.2 from [Gupte et al., 2016]. The p-formulation [Haverly, 1978] of the generalized pooling problem is defined as follows:

Definition 3.2. *Given is a directed graph $G = (\mathcal{N}, \mathcal{A})$ with a set of nodes $\mathcal{N}$ and set of arcs $\mathcal{A}$. Set $\mathcal{N}$ is partitioned into three nonempty subsets $I, L, J \subset \mathcal{N}$. These subsets contain inputs exclusively in I, pools in L and outputs in J, see Figure 3-1. Connections are only allowed from inputs to pools and outputs, from pools to pools and from pools to outputs:*

$$\mathcal{A} \subseteq (I \times L) \cup (I \times J) \cup (L \times L) \cup (L \times J).$$

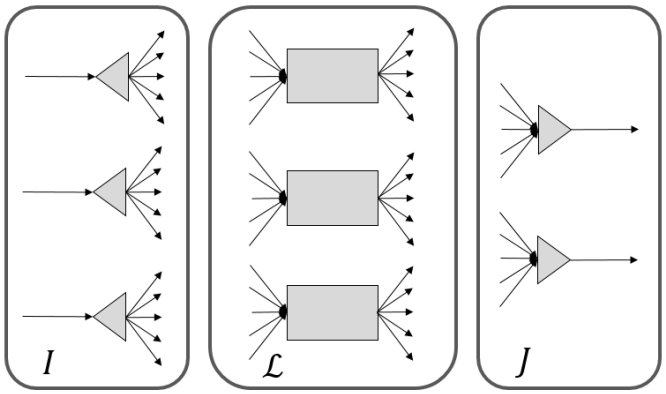


Figure 3-1: Set definition and superstructure pooling problem

For each arc $(i, j) \in \mathcal{A}$ a variable cost $c_{i,j}$ is defined for using this connection. Every node $i \in \mathcal{N}$ has a maximum incoming or outgoing flow capacity C_i. C_i denotes the maximum supply for input $i \in I$, C_l is considered the volumetric size of a pool for $l \in L$, and C_j defines the maximum intake at output j in$J$. In addition the upper bound on each arc $(i, j) \in \mathcal{A}$ is given by u_{ij}.

The level of specification (spec) k in the raw material at input $i \in I$ is given by $\lambda_{i,k}$ for $k \in K$, which is the set of specifications. Upper and lower bounds on a required spec level $k \in K$ at output $j \in J$ are given by $\mu_{jk}^{\max}$ and $\mu_{jk}^{\min}$. A flow on arc $(i, j) \in \mathcal{A}$ is given by y_{ij} and the concentration value of spec $k \in K$ in flow $y_{ij} \in \mathcal{A}$ is given by p_{ik}. The cost factor of using arc $(i, j) \in \mathcal{A}$ is defined by c_{ij} per unit of flow y_{ij}.

The objective of the pooling optimization problem is to find a minimum cost feasible flow in graph G:

$$\min_{y,p} \sum_{(i,j)\in\mathcal{A}} c_{ij} y_{ij}, \tag{3-1}$$

satisfying capacity constraints:

$$\sum_{j \in L \cup J} y_{ij \leq C_i} \qquad i \in I, \tag{3-2}$$

$$\sum_{j \in L \cup J} y_{lj \leq C_l} \qquad l \in L, \tag{3-3}$$

$$\sum_{i \in I \cup L} y_{ij \leq C_j} \qquad j \in J, \tag{3-4}$$

$$0 \leq y_{ij} \leq u_{ij} \qquad (i,j) \in \mathcal{A}, \tag{3-5}$$

flow balance constraint:

$$\sum_{i \in I \cup L} y_{il} = \sum_{j \in L \cup J} y_{lj} \qquad l \in L, \tag{3-6}$$

spec output constraints:

$$\sum_{i \in I} \lambda_{ik} \cdot y_{ij} + \sum_{l \in L} p_{lk} \cdot y_{lj} \leq \mu_{jk}^{\max} \sum_{i \in I \cup L} y_{ij} \quad j \in J, k \in K, \tag{3-7}$$

$$\sum_{i \in I} \lambda_{ik} \cdot y_{ij} + \sum_{l \in L} \cdot p_{lk} y_{lj} \geq \mu_{jk}^{\min} \sum_{i \in I \cup L} y_{ij} \quad j \in J, k \in K, \tag{3-8}$$

and spec tracking constraint:

$$\sum_{i \in I} \lambda_{ik} \cdot y_{il} + \sum_{l' \in L} p_{l'k} \cdot y_{l'l} = p_{lk} \sum_{j \in L \cup J} y_{lj} \quad l \in L, k \in K. \tag{3-9}$$

Equation (3-9) represents the spec balance equation around pool $l \in L$ for spec $k \in K$.

3-2 Reduction to TWWN problem

In order to fit Definition 3.2 to the TWWN MINLP, all sets, indices, variables and parameters in the TWWN program are defined such that the program stated in Equations (3-1) - (3-9) is equivalent to a special case of the TFM from Equations (2-1) - (2-25). Since the pooling problem is a special case of the TFM, the TWWN parameters are defined such that the pooling problem remains. Both the translation from the pooling elements to the TFM and the remaining parameter definitions are stated below.

Sets and indices

The relation between the sets and indices from the two programs is stated in Table 3-1. The remaining sets from the TFM are the set of process units PU, and the sets of pipe connections PC_{fixed} and PC_{free}. The set of process units is defined as empty, $PU := \emptyset$, as is the set of optional pipe connections: $PC_{free} := \emptyset$, such that set PC is equal to set PC_{fixed}.

Each splitter s and each mixer m are associated with the corresponding network unit.

Table 3-1: Sets and indices translation

TWWN			Pooling		
Description	Set	Index	Description	Set	Index
Contaminants	C	c	Spec	K	k
Sources	WI	w	Inputs	I	i
Treatment units	TU	t	Pools	L	l
Sinks	WO	d	Outputs	J	j

Variables

The translation between the variables from the two programs is stated in Table 3-2. Some variables from the TWWN problem are not defined in the pooling problem. In the following sections it is shown that the constraints regarding these variables can be substituted by other constraints, resulting a formulation, solely based on variables from table 3-2. In addition, the TFM contains a binary variable $\psi_{s,m}$ which represents the pipe existence between splitter s and mixer m for optional pipe connections. Since this variable is not present in the $p-$formulation of the pooling problem, the relevant constraints should not appear in the spcieal case of the pooling problem. This is accomplished by choosing the set of variable pipe connections PC_{free} to be empty. The constraints concerned are Equation (2-17), (2-19) and (2-21). In the following paragraphs it is shown that these constraints are indeed absent in the resulting model formulation.

Table 3-2: Variables translation

TWWN		Pooling	
Description	Variable	Description	Variable
Water flow	ϕ_{s_w,m_t}	Flow	y_{il}
Water flow	ϕ_{s_w,m_d}	Flow	y_{ij}
Water flow	ϕ_{s_t,m_d}	Flow	y_{lj}
Water flow	$\phi_{s_t,m_{t'}}$	Flow	$y_{ll'}$
Contaminant concentration	$\rho_{s_t,m_d,c}$	Spec	p_{lk}
Contaminant concentration	ρ_{s_t,m'_t}	Spec	p_{lk}
Contaminant concentration	ρ_{s_t,m_d}	Spec	p_{lk}

Parameters

The translation between the parameters from the two programs is stated in Table 3-3. In addition, some parameters from the TFM will be fixed in case of the pooling instance. All parameters related to process units are zero, since no process units are defined. Additionally, the parameters used in the objective function are fixed such that

Table 3-3: Parameters translation

TWWN		Pooling	
Description	Parameter	Description	Parameter
Max intake source	$\overline{\phi}_w$	Available supply input	C_i
Contaminant intake source	$\rho_{w,c}$	Spec level input	λ_{ik}
Max flow rate TU	$\overline{\phi}_t$	Pool capacity	C_l
Max discharge flow sink	$\overline{\phi}_d$	Max intake at output	C_j
Min contaminant discharge sink	$\underline{\rho}_{d,c}$	Min spec level output	$\mu_{jk}^{\min}$
Max contaminant discharge sink	$\overline{\rho}_{d,c}$	Max spec level output	$\mu_{jk}^{\max}$
Operational pipe costs	$\pi_{s,m,op}$	Cost factor arc	c_{ij}

the resulting objective yiels the one given in Equation (3-1). The definitions of the remaining parameters are stated in Table 3-4.

Table 3-4: Remaining TWWN parameters definition

Parameter	Description	Value
π_w	Price of water intake	0
$\pi_{t,IC}$	Investment costs TU	0
$\pi_{t,OC}$	Operational costs TU	0
$\underline{\phi}_t$	Min flow rate TU	0
$\overline{\rho}_{t,c,in}$	Max contaminant entering TU	sufficiently large
$\overline{\rho}_{t,c,out}$	Max contaminant leaving TU	sufficiently large
$\lambda_{t,c}$	Removal ratio TU	0
$\underline{\phi}_d$	Min discharge flow sink	0
H	Operation hours	arbitrary
AR	Investment factor TU	arbitrary
κ_{min}	Min number of pipe connections	0
κ_{max}	Max number of pipe connections	0
α	Investment discount factor TU	arbitrary
γ	Investment discount factor pipes	arbitrary
$\pi_{s,m,fix}$	Fixed cost pipes	0
$\pi_{s,m,var}$	Variable cost pipes	0
θ_{TU}	Recycling around TU	1
θ_{PU}	Recycling around PU	1

Constraints

In this special case of representing the pooling problem as an instance from the TFM, the constraints are a result of the variable and parameter definitions stated above. The equivalence between the constraints from the pooling problem and the constraints from the TFM is shown in Table 3-5, using substitution of constraints and the definitions as stated above.

Table 3-5: Constraint translation

TWWN		Pooling	
Description	Constr	Description	Constr
Flow bound source	(2-1),(2-16)	Capacity constraint intake	(3-2)
Flow bound TU	(2-7),(2-10),(2-16)	Capacity constraint pool	(3-3)
Flow bound sink	(2-12),(2-14)	Capacity constraint output	(3-4)
Flow bound pipe	(2-20),(2-22)	Capacity contraint arc	(3-5)
Flow balance	(2-10),(2-14),(2-16)	Flow balance pool	(3-6)
Flow and contaminant bound and balance	(2-13),(2-14),(2-15)	Spec output constraint	(3-7),(3-8)
Flow and contaminant balance	(2-10), (2-11),(2-15) (2-16),(2-18)	Spec tracking around pool	(3-9)

Constraint (3-6) from the pooling problem is the result of substitution of constraint (2-14) and (2-16) in constraint (2-10), and using the variable and parameter translations from the previous paragraphs:

$$\underbrace{\sum_{i \in I \cup L} y_{il}}_{(2-14)} = \overbrace{y_{l,in}}^{(2-10)} = \underbrace{y_{l,out}}_{(2-16)} = \underbrace{\sum_{j \in L \cup J} y_{lj}}_{(2-16)} \quad l \in L.$$

With the same reasoning, constraints (3-7) and (3-8) are both the result of substituting constraints (2-13) and (2-15) in (2-14):

$$\underbrace{\sum_{i \in I} \lambda_{ik} \cdot y_{ij} + \sum_{i \in L} p_{lk} \cdot y_{lj}}_{(2-15)} = \overbrace{p_{j,in,k} y_{j,in}}^{(2-10)} = \underbrace{p_{j,in,k} \sum_{i \in I \cup L} y_{ij}}_{(2-16)} \leq \mu_{jk}^{max} \underbrace{\sum_{j \in L \cup J} y_{ij}}_{(2-16)} \quad j \in J, k \in K,$$

$$\underbrace{\sum_{i \in I} \lambda_{ik} \cdot y_{ij} + \sum_{i \in L} p_{lk} \cdot y_{lj}}_{(2-15)} = \overbrace{p_{j,in,k} y_{j,in}}^{(2-10)} = \underbrace{p_{j,in,k} \sum_{i \in I \cup L} y_{ij}}_{(2-16)} \geq \mu_{jk}^{min} \underbrace{\sum_{j \in L \cup J} y_{ij}}_{(2-16)} \quad j \in J, k \in K.$$

Finally, spec tracking constraint (3-9) is the result of combining constraints (2-10), (2-11), (2-15), (2-16) and (2-18):

$$\underbrace{\sum_{i \in I} \lambda_{ik} \cdot y_{il} + \sum_{l' \in L} p_{l'k} \cdot y_{l'l} = p_{l,in,k} \cdot y_{l,in}}_{(2-15)} =_{\substack{(2-10)\\(2-11)}} p_{l,out,k} \cdot y_{l,out}$$

$$=_{\substack{(2-16)\\(2-18)}} p_{lk} \sum_{j \in I \cup J} y_{lj} \qquad l \in L, k \in K.$$

The remaining constraints from the TFM that are not in accordance with one of the constraints from the pooling problem are either redundant or implied otherwise by the pooling problem definition. These constraints are specified in Table 3-5.

Table 3-6: TWWN constraints which are absent or redundant in pooling problem

Description	Constr	Reason
Contaminant intake bound TU	(2-8)	Definition $\overline{\rho}_{t,c,in}$
Contaminant output bound TU	(2-9)	Definition $\overline{\rho}_{t,c,out}$
PU Constraints	(2-2),(2-3),(2-4), (2-5),(2-6),(2-23)	Empty set PU
Contaminant balance splitters	(2-17)	Empty set PC_{free}
Contaminant balance splitters	(2-18)	Parameter $\lambda_{i,k}$ implies constraint (2-18)
Flow bound pipe	(2-19)	Empty set PC_{free}
Number of pipe connections	(2-21)	Definition κ_{min} and κ_{max}
Total network balance	(2-24)	Redundant

Objective function

By definition of all parameters and variables as stated in Table 3-2, 3-3 and 3-4, the objective function (2-25) from the TFM as pooling problem becomes:

$$\min_{\phi,\rho} \sum_{s} \sum_{m} \pi_{s,m,op} \phi_{s,m}. \tag{3-10}$$

This is in agreement with the objective function from the pooling problem stated in Definition 3.2. This completes the pooling problem defined as a special case of the TWWN formulation as presented in the TFM. Thus the following holds:

Theorem 1. *The TWWN problem is NP-hard.*

Proof. The NP-hardness of the pooling problem is shown in Proposition 1, page 901-202 of [Alfaki and Haugland, 2013]. This chapter contains the proof that the pooling problem is a special case of the TWWN problem on page 23-27. Thus the TWWN problem is NP-hard. □

Chapter 4

Nonlinear solution methods

In the previous chapters, the MINLP formulation of the TWWN was analyzed: the TFM. Solution methods that take this model as input, need a nonlinear solver. For linear programs, a wide variety of solvers is available. These solvers have rapidly improved and specialized over time, to state of the art solvers. For nonlinear optimization however, no such variety of solvers is available. The issue of finding global optimal solutions for large scale problems remains an open research area, see [Tawarmalani and Sahinidis, 2005]. Additional complexities such as non convexity and integer characteristics make the problems too complex to solve even for small problem instance sizes.

Two search algorithms that might be suitable to solve the TFM are discussed here: *Outer Approximation (OA)* and *spatial Branch-and-Bound* (SB&B). OA is designed to solve convex MINLP models by using decomposition, outer approximation and relaxation [Duran and Grossmann, 1986]. It is used in the solver DICOPT [Grossmann et al., 2002] and in the *AIMMS Outer Approximation (AOA)* solver in AIMMS [Bisschop and Roelofs, 2006]. The method of solving the TFM with AOA, results in the solution method called **TFM-AOA**. This search algorithm is discussed in Section 4-1. The second method uses SB&B to solve the TFM. This is the steering algorithm in the *Branch-And-Reduce Optimization Navigator (BARON)*: [Tawarmalani and Sahinidis, 2005]; a computational system which solves non linear optimization problems to global optimality using cutting planes and a continuous branch and bound strategy. The method of solving the TFM with SB&B, **TFM-BARON**, is discussed in Section 4-2. A summary of the nonlinear solution methods is stated in Section 4-3.

4-1 Outer approximation

OA is based on a finite iterative process in which relaxed MILP and NLP problems are solved consecutively, using decomposition, outer approximation and relaxation. It is closely related to the generalized Benders decomposition method, using a cutting-plane approach using both primal and dual optimality information. One main difference however, is that OA only uses primal information for the linear relaxation.

For convex problems with integer variables, OA converges to the global optimum. However, convergence is not guaranteed in case of non convex problems [Fletcher and Leyffer, 1994]. This section describes the basic workings of the method for the class of convex MINLP programs. Given by:

$$\begin{aligned} z = \min_{x,y} \quad & c^\intercal y + f(x) \\ \text{s.t.} \quad & g(x) + By \leq 0 \\ & x \in X \subset \mathbb{R}^n \\ & y \in Y \subset \mathbb{Z}^m_+, \end{aligned} \tag{P}$$

where both the functions $f : \mathbb{R}^n \mapsto \mathbb{R}$ and $g : \mathbb{R}^n \mapsto \mathbb{R}^p$ are assumed to be convex and continuously differentiable. The function $g(x)$ is defined as:

$$g(x) = \Big[g_1(x)g_2(x)\dots g_p(x)\Big]^\intercal.$$

The set $X := \{x : x \in \mathbb{R}^n, A_1x \leq a_1\}$ is assumed to be a closed polyhedral set, and the set $Y := \{y : y \in Y \in \mathbb{Z}^m_+, A_2y \leq a_2\}$ a nonnegative finite discrete set. The matrix B may contain zero row vectors, resulting in constraints which only involve continuous variables. Note that (P) is linear with respect to its discrete variables. This characteristic, together with the convexity of the nonlinear functions, is exploited in the algorithm.

The OA algorithm

The method creates an increasingly tighter relaxation on the original MINLP program in each iteration, which is then called the *Master Program (MP)*. This MP is created each iteration k by finding a solution for both the NLP and MIP subproblems and replacing the nonlinear functions $f(x)$ and $g(x)$ by supporting half-spaces in the current solution point x^k. It starts with fixing an initial solution y^k in iteration k. This solution y^k should satisfy the following integer constraints:

$$\begin{aligned} & y^k \in Y \subset \mathbb{Z}^m_+, \\ & g(x) + By^k \leq 0 \quad \text{for some } \ x \in X. \end{aligned} \tag{IP}$$

Now, the following NLP program is solved:

$$\begin{aligned} z(y^k) = \min_{x} \quad & c^\intercal y^k + f(x) \\ \text{s.t.} \quad & g(x) + By^k \leq 0 \\ & x \in X \subset \mathbb{R}^n. \end{aligned} \tag{NLP^k}$$

The feasible solution, x^k, provides an upper bound (x^U, y^U) for the optimal solution (x^*, y^*). Next, the Master Program is created by linearizing (P) around solution x^k. This is done by defining the supporting half-spaces at x^k. The supporting half-spaces are defined as follows:

Definition 4.1. *For every convex $f(x)$ and for any $x^k \in X \subset \mathbb{R}^n$ the supporting half-space is defined by:*

$$f(x) \geq f(x^k) + \nabla f(x^k)^\intercal (x - x^k),$$

where $\nabla f(x^k)$ is the gradient vector of $f(x^k)$ with dimension n.

Thus, for any $\alpha \in \mathbb{R}$:

$$\alpha \geq f(x),$$

may be relaxed to:

$$\alpha \geq f(x^k) + \nabla f(x^k)^\intercal (x - x^k).$$

This yields the MIP Master Program:

$$\begin{aligned}
\min_{x,y,z} \quad & z \\
\text{s.t.} \quad & z \geq c^\intercal y + f(x^i) + \nabla f(x^i)^\intercal (x - x^i) && \forall x^i \in T^k \\
& g_j(x^i) + \nabla g_j(x^i)^\intercal (x - x^i) + (By)_j \leq 0 \quad \forall i \in T^k \;\; \forall j \in \{1, 2, \ldots, p\} \\
& x \in X \subset \mathbb{R}^n \\
& y \in Y \subset \mathbb{Z}_+^m \\
& z \in \mathbb{R}.
\end{aligned} \tag{MP^k}$$

where $T^k := \{x^i \text{ optimal solution to program (NLP}^i), i = 1, 2, \ldots, k\} \subseteq T$. With each iteration, more half-spaces are added to the Master Program, resulting in an increasingly tight, linear relaxation of (P).

Solving this program results in either a solution, or an infeasible program. In the latter case the global optimum has been found in this iteration for y^k and x^k after solving (NLP^k). In case a solution is found for (MP^k), this yields a lower bound (x^L, y^L) on the global optimum. Otherwise an updated fixed integer part is taken as new input for problem (IP).

One problem that may occur is that no feasible solution can be found for (NLP^k). In that case, an integer cut is added to (MP^k) in order to eliminate y^k from future iterations. The complete algorithm can now be presented as the flow diagram in Figure 4-1. It was shown in [Duran and Grossmann, 1986] that the solutions found for (MP^k) in each iteration are a monotonic non-decreasing sequence on the optimal value of the original problem (P). Moreover, the algorithm terminates in a finite number of iterations, due to the fact that the set of feasible integer solutions is finite by assumption, and revisiting integer combinations from previous iterations is ruled out. For more details about OA and its convergence, the reader is referred to [Duran and Grossmann, 1986].

OA applied to the TWWN problem

The OA method is implemented as a solver in DICOPT [Grossmann et al., 2002] and in AIMMS [Bisschop and Roelofs, 2006] as *AIMMS Outer Approximation*. Since OA exploits the linearity of the integer variables, it is of value to check whether the TFM can be written in the format of (P). This format requires all terms containing integer variables to be linear. The TFM violates this format in constraints (2-17). However, these constraint can each easily be replaced by the set of linear Equations (2-28). After this substitution the TFM can be written in the format of program (P), using Equations (2-1)-(2-16),(2-19)-(2-25),(2-28). For this model formulation of the TWWN problem, OA can be applied.

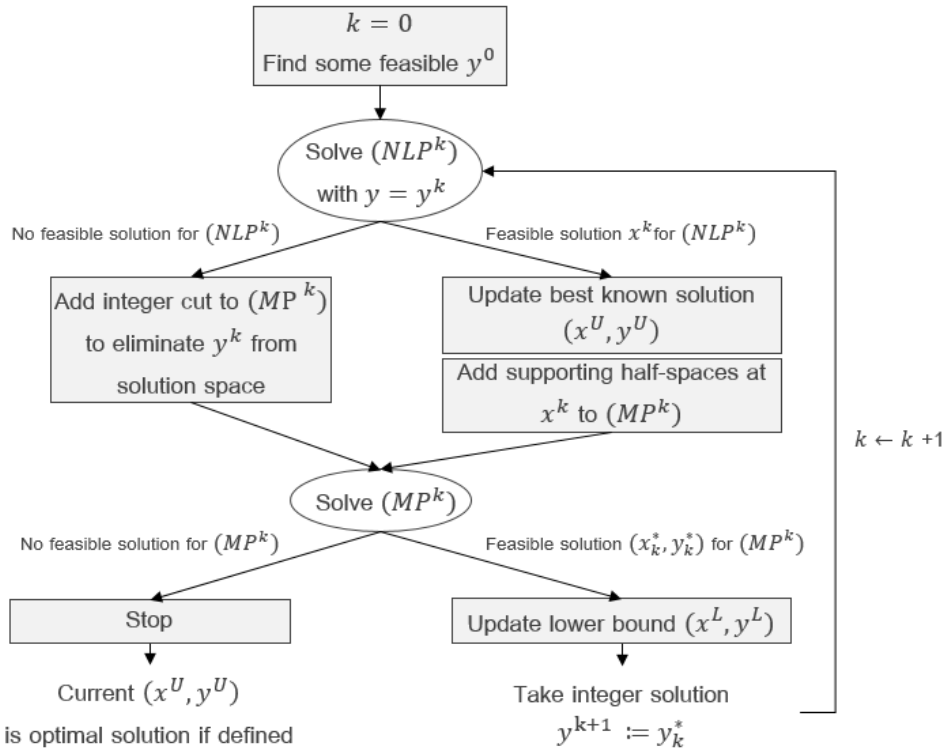


Figure 4-1: OA algorithm

4-2 Spatial branch-and-bound

In order to quantify the difficulty of solving the MINLP TWWN problem, a suitable nonlinear solver was selected. In previous research, BARON has more than once been the benchmark for solving the TWWN problem [Karuppiah and Grossmann, 2006, Wicaksono and Karimi, 2008, Ahmetović, 2011] and the closely related pooling problem [Pham et al., 2009, Gupte et al., 2016, Alfaki and Haugland, 2013]. For smaller instances of the TWWN problem, the solver proved to obtain a global optimum within reasonable time [Karuppiah and Grossmann, 2006]. However, in many other cases, no global solution was found within the limits on running times [Karuppiah and Grossmann, 2006, Ahmetović, 2011].

The following system is designed to globally solve nonlinear non convex optimization problems [Sahinidis, 1996] with general formulation (Q) from [Sahinidis, 2002]:

$$\begin{aligned} z = \min_{x,y} \quad & f(x) \\ \text{s.t.} \quad & g(x) \leq 0 \\ & x_i \in \mathbb{R}^n \quad i = 1, \ldots, n_d \\ & x_i \in \mathbb{Z}^m \quad i = n_d + 1, \ldots, n, \end{aligned} \tag{Q}$$

where $f : \mathbb{R}^n \mapsto \mathbb{R}$, and $g : \mathbb{R}^n \mapsto \mathbb{R}^m$.

The SB&B algorithm

As the name BARON (Branch and Reduce Optimization Navigator) suggests, the system is based on two ground principles: spatial branching on continuous variables, and variable range reduction. While both principles are known solution strategies, BARON enhanced these principles and created a new framework involving constraint propagation and duality techniques. By doing so, BARON was the first global branch-and-bound solver addressing nonlinear and MINLP problems. The main steps and components of the solver are described in [Sahinidis, 1996, Sahinidis, 2002]:

1. Branching on continuous variables using node selection rules such that convergence is guaranteed.
   The main rule to select a node, is based on *bound-improving*. In each iteration, the node with the current lowest bound is selected. The selected variable is partitioned at the location of the last relaxation solution. The bound-improving rule is alternated with selection of a variable with the widest range remaining. The combination of these rules guarantees convergence. See [Horst and Tuy, 1996].

2. Creating a linear relaxation at each node.
   The integer part of (Q) is relaxed by dropping the integrality conditions. Relaxation of the nonlinear part, however, is more complex; It involves a polyhedral outer-approximation algorithm and convexification prior to this algorithm, in case the nonlinear part is not convex.
   BARON combines several convexification strategies:

   - *Factorable programming relaxations* can be applied to functions containing sums and products of univariate functions. With the introduction of new variables, these functions are decomposed, after which straightforward outer-approximations of the resulting parts can be applied. It may however, result in a large relaxation gap.
   - A second method that does not result in such a large relaxation gap, is *convex extensions*. This method constructs a convex function that contains the original continuous function and corresponds to the original function values. In [Tawarmalani and Sahinidis, 2002], a methodology for creating these extensions is provided. This method can be applied to several multilinear functions among others.

   After application of these convexification methods, a polyhedral outer approximation scheme is applied, introduced as the *sandwich algorithm* in [Tawarmalani and Sahinidis, 2004]. In each step of the algorithm, a supporting hyperplane is added to the convex nonlinear function at the location where the relaxation error is maximal. The algorithm terminates when the required accuracy is obtained, resulting in a polyhedral relaxation of problem at the current node.

3. Solving the linear relaxation of the problem, including range reduction of the continuous variables in pre- and post processing steps.
   Prior to solving the linear relaxation, *feasibility-based range reduction* reduces the feasible range of the variables, using information from the problem constraints. A linear program is solved, based on individual constraints.
   BARON incorporates two range reduction strategies:

- Solving the complete set of according linear constraints. This method yields the maximal range reduction, but is computationally expensive. At each node, BARON applies this strategy to a few auspicious variables.
- Approximated range reduction, which is commonly used in the application of constraint programming and solving MIPs, see [Hooker, 2011]. This method does not guarantee maximal range reductions, but takes less computational effort. Therefore this method is applied at each node for all variables.

Following the range reduction as preprocessing step, the linear relaxation at the node is solved, resulting in a lower bound of problem (Q). The solution obtained is used for additional range reduction: *optimality-based range reduction*, based on [Ryoo and Sahinidis, 1995]. For variables at their bound in the linear relaxed problem, the Lagrange multiplier of the dual solution is used. Combining this information with the primal information, results in a feasible range reduction. In case a variable is not at its bounds, some additional steps could still obtain a range reduction, using the information of the current solution.

4. Local optimization of problem (Q) restricted to the feasible domain of the current node, obtaining an upper bound on the global optimum.

These steps are combined in an iterative framework, resulting in an algorithm that iteratively improves the current best upper and lower bounds, converging to the global optimum of problem (Q).

SB&B applied to the TWWN problem

Different studies showed that BARON outperforms other nonlinear solvers and it is therefore a promising choice for different kinds of nonlinear problems, including MINLP programs [Lastusilta et al., 2007, Nowak and Vigerske, 2008].

In [Lastusilta et al., 2007], four different MINLP solvers were compared: sBB, Dicopt, GAMS/αECP and BARON. The solver sBB combines branch-and-bound with NLP solvers integrated in modeling language GAMS. DICOPT [Viswanathan and Grossmann, 1990] is based on outer approximation techniques. GAMS/αECP [Westerlund and Pörn, 2002] however, is based on solving a sequence of MILP problems until a feasible solution is found. The comparison of these solvers was based on a test set of 250 problems from the GAMS MINLP model library [gam, 2016b], given a running time limit of 1000 seconds. The results showed that BARON obtained the most accurate results in case a feasible solution was found. The running time however, was relatively high. An interesting observation was that there were more than 80 cases for which only one solver obtained a feasible or optimal solution. This emphasizes the importance of selecting a suitable solver for each problem.

In [Nowak and Vigerske, 2008], BARON was chosen to compare with the branch-and-cut algorithm *LaGO (Lagrangian Global Optimizer)*, which is designed to solve MINLP problems. The comparison was based on 127 MINLPs and 77 Mixed Integer all-Quadratic Programs (MIQQP) from the GAMS model libraries [gam, 2016a, gam, 2016b]. Over 50% of these test instances were non convex. Overall, BARON outperformed LaGO both on running time and solution quality. For less than 5% of the test instances, LaGO

found a solution within an hour, where BARON did not. The reversed was true for almost 25% of the cases.

In summary, BARON showed to be a high performance MINLP solver in general, and was selected as MINLP solver in advanced research on pooling problems and TWWN problems.

4-3 Summary of nonlinear solution methods

In this section, two nonlinear solver strategies are explained which can be applied to solve the TWWN problem, based on the MINLP TFM.

TFM-AOA obtains an upper bound when applied to the TWWN problem. An advantage is the acceptable running times, making the method suited for finding a local optimum of the TWWN problem. However, its distance from the global optimum is unknown.

The method **TFM-BARON** uses BARON as a global solver. A drawback is the possible exploding running time when increasing the instance size for the TWWN problem. In case the algorithm terminates within reasonable time, the obtained solution is guaranteed to be the global optimum.

Chapter 5

Linear solution methods

The nonlinear solution methods discussed in the previous chapter, use the TFM formulation as input for a nonlinear solver. Linearization is incorporated in nonlinear solution methods as well, though only applied locally. However, the methods in this section are based on different linear reformulations of the TFM, resulting in a MIP formulation. These models can be solved with a linear solver. For this research, the linear solver that is used to solve all MIP models is IBM ILOG CPLEX 12.6.2 [ILOG, 2014]. The different linearization techniques, that result in a MIP formulation, can be categorized as: relaxation, approximation and discretization. The theoretical background and implementation of these techniques are explained in this chapter.

The first solution method, **CH-CPLEX**, incorporates a linearized model, CH, obtained by a relaxation technique. This technique is known as the *Convex Hull method*. The technique uses linear over- and underestimators in order to approximate the nonlinearities in the formulation [McCormick, 1976]. It is implemented in a global branch-and-bound algorithm in the *APOGEE* software [Misener et al., 2011] and is described in Section 5-1. Several, smaller, convex hulls can be added to tighten the relaxation. This entails selecting breakpoints, which specify the start and end of each convex hull. As expected, each break point requires a binary variable in the mathematical program to represent it. In order to speed up the algorithm, an alternative formulation can be used, reducing the additional number of binary variables to a logarithmic increase with respect to the number of breakpoints added. A detailed description of this model, CH_{log} and the corresponding solution method, **CH_{log}-CPLEX**, starts at page 42.

A second linearization technique concerns the use of *Piecewise Linear (PWL) Approximation*. This technique defines and solves a linear approximation of the original nonlinear model. A research on existing literature resulted in the most suited MIP formulations for the TWWN problem: PWL and PWL_{log}. These model formulations are derived and stated in Section 5-2. The corresponding solution methods are referred to as **PWL-CPLEX** and **PWL_{log}-CPLEX** respectively.

The final linearization technique discussed, is based on discretization. This technique is introduced in [Gupte et al., 2013] for the application of the pooling problem. The original MINLP model is approximated by discretizing part of the continuous variables.

Reformulation of the approximated model results in a MIP problem, $\text{Discr}_{\text{log}}$, that can be solved using linear solvers. For this linearization technique, a formulation is used based on a logarithmic number of additional variables and constraints. All solutions obtained by the solution method **$\text{Discr}_{\text{log}}$-CPLEX** are feasible for the original MINLP model and provide an upper bound on the global optimum. The discretization strategy is stated in Section 5-3.

5-1 Convex hull relaxation

Convex hull relaxation relies on linear relaxations of bilinear terms by creating tight *convex hulls*. These can be considered linear envelopes enclosing the nonlinear term. The use of linear relaxation for bilinear programs was introduced in [Balas, 1985] and was applied to waste water treatment problems in [Karuppiah and Grossmann, 2006]. Since then, different MIP formulations of the convex hull relaxation have been defined and compared for several bilinear programming applications. For existing models and twelve other formulations, an extended comparison was done in [Wicaksono and Karimi, 2008]. Both uniform and random interval lengths were taken into account. They compared fifteen formulations on two case studies from integrated water systems synthesis and distillation column sequencing. Moreover, in [Gounaris et al., 2009] fifteen formulations were compared on benchmark problems, some different formulations than those in [Wicaksono and Karimi, 2008]. It was shown in [Hasan and Karimi, 2010] that for any bilinear program a uniform interval distribution is the best strategy.

Three categories of formulations are distinguished in [Gounaris et al., 2009, Wicaksono and Karimi, 2008]: the *BigM Class, Convex Combination Class,* and *Incremental Cost Class.* These classes differ in the definition of the additional binary variables, and the use of a large M coefficient. Both [Gounaris et al., 2009] and [Wicaksono and Karimi, 2008] showed that the best running times can be obtained using the convex combination and incremental cost formulations. For the pooling problem, the best convex combination based models outperformed the best incremental cost based models on most instances in [Gounaris et al., 2009] for both formulations of the pooling problem [Haverly, 1978].

In previous work that applied the convex hull method to the TWWN problem [Karuppiah and Grossmann, 2006], a convex combination based formulation from [Gounaris et al., 2009] was used. This formulation is the base of the first linear relaxation method that is discussed in this section: CH-CPLEX. The formulation of the CH model starts at page 40.

A tool that incorporates the use of convex hulls is APOGEE [Misener et al., 2011]. As the name suggests, the tool *Algorithms for Pooling-problem global Optimization in GEneral and Extended classes* (APOGEE) is developed in order to solve all cases of pooling problems, of which the general case is described in Chapter 3-1. APOGEE based their novel formulation on the incremental cost based formulation **nf4r** from [Gounaris et al., 2009]. This formulation can be found in Appendix D. It is used for a novel formulation, based on an alternative formulation from [Vielma and Nemhauser, 2008]. Previously, this formulation was used for piecewise linear approximation only. It uses a logarithmic number of additional binary variables and constraints, instead of a linear number. This novel formulation, hereafter called the CH_{log} model, is derived in this chapter at page 42.

The remainder of this chapter is organized as follows: first, the general idea of linear relaxation is explained, followed by the concept of domain partitioning. Then, a formulation of the models CH and CH_{log} is stated, followed by the application of these models of solving the TWWN problem.

Linear relaxation

The tightest linear relaxation of a bilinear term was introduced in [McCormick, 1976]. This relaxation is based on the bilinear convex and concave envelope and it can generally be described as a relaxation of the set:

$$S := \left\{(xy, x, y) \in \mathbb{R}^3_+ \mid x \in [x^L, x^U], y \in [y^L, y^U]\right\}.$$

This set is relaxed by replacement of the bilinear term xy by z and the below definition of its convex linear underestimators:

$$\begin{aligned} z &\geq x^L y + y^L x - x^L y^L, \\ z &\geq x^U y + y^U x - x^U y^U, \end{aligned} \tag{5-1}$$

and convex linear overestimators:

$$\begin{aligned} z &\leq x^L y + y^U x - x^L y^U, \\ z &\leq x^U y + y^L x - x^U y^L, \end{aligned} \tag{5-2}$$

yielding the relaxed convex linear set of S:

$$\begin{aligned} C(S) := \Big\{(z, x, y) \in \mathbb{R}^3_+ \mid\ & z \geq x^L y + y^L x - x^L y^L, \\ & z \geq x^U y + y^U x - x^U y^U, \\ & z \leq x^L y + y^U x - x^L y^U, \\ & z \leq x^U y + y^L x - x^U y^L \Big\}. \end{aligned} \tag{5-3}$$

Note that $C(S)$ is the convex hull of S in $\mathbb{R}^3$. These four inequalities create an envelope around a bilinear term and are commonly referred to as the *McCormick envelopes*.

The construction of the convex hull is depicted in Figure 5-1, where the cross-section $x = y$ of the 3D - space of function $z = xy$ is illustrated. The tightness of the linear relaxation can be improved by partitioning of the domain, such that smaller convex hulls can be defined for each interval. This is explained in the following paragraphs.

Domain partitioning

Since the tightness of the relaxation from (5-3) depends on the size of the domain $[x^L, x^U] \times [y^L, y^U]$, one of the variables can be partitioned in order to create tighter relaxations for each interval separately.

The definition of the convex hull method applied over disjunctive intervals is as follows: in case x is the variable that is partitioned in N_p uniform intervals, the following points define the values of x at each of the domain boundaries:

$$x^L = x_1 < x_2 < \cdots < x_{N_p} < x_{N_p+1} = x^U.$$

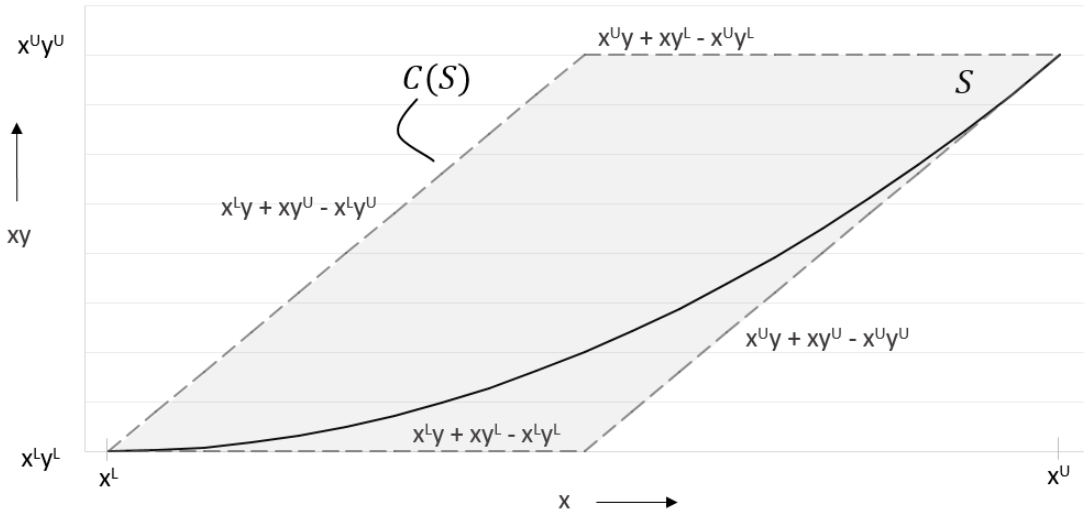


Figure 5-1: Construction of convex hull $C(S)$

The convex hull relaxation from (5-3) can now be applied over each interval $[x_{n_p}, x_{n_p+1}]$ for any $n_p \in \{1, 2, \ldots, N_p\}$, yielding the following linear estimators from [Karuppiah and Grossmann, 2006]. For $x \in [x_{n_p}, x_{n_p+1}]$ it should hold that:

$$\begin{aligned} z &\geq x_{n_p} y + y^L x - x_{n_p} y^L, \\ z &\geq x_{n_p+1} y + y^U x - x_{n_p+1} y^U, \\ z &\leq x_{n_p} y + y^L x - x_{n_p+1} y^L, \\ z &\leq x_{n_p} y + y^U x - x_{n_p} y^U. \end{aligned} \tag{5-4}$$

Equations 5-4 are the base of the CH model, which is to be formulated in the next section.

CH model

The partitioned convex hull relaxation from equations 5-4 can be reformulated such that it can be included in a MIP program.It is equivalent to formulation **ch** from [Gounaris et al., 2009]. Incorporated in an optimization program, the set of constraints from (5-4) yields a small convex envelope at the particular interval in which the optimal solution is located, instead of one convex hull around the whole solution space.

For each $n_p \in \{1, 2, \ldots, N_p\}$, both a π_{n_p} and χ_{n_p} are introduced, such that:

$$\pi_{n_p} = \begin{cases} x & \text{if} \quad x \in [x_{n_p}, x_{n_p+1}] \\ 0 & \text{else,} \end{cases}$$

and

$$\chi_{n_p} = \begin{cases} y & \text{if} \quad x \in [x_{n_p}, x_{n_p+1}] \\ 0 & \text{else.} \end{cases}$$

This is done by definition of the indicator variable λ_{n_p} and constraining it to:

$$\lambda_{n,p} = \begin{cases} 1 & \text{if} \quad x \in [x_{n_p}, x_{n_p+1}] \\ 0 & \text{else.} \end{cases}$$

These variables and constraints can now be implemented as a MIP, using the equations from (5-4), which now hold for the interval containing x. This results in partitioned convex hull MIP program:

$$\begin{aligned} x &= \pi_1 + \pi_2 + \cdots + \pi_{N_p}, \\ y &= \chi_1 + \chi_2 + \cdots + \chi_{N_p}, \end{aligned}$$

$$\begin{aligned} z &\geq \sum_{n_p=1}^{N_p} x_{n_p}\chi_{n_p} + y^L\pi_{n_p} - x_{n_p}y^L\lambda_{n_p}, \\ z &\geq \sum_{n_p=1}^{N_p} x_{n_p+1}\chi_{n_p} + y^U\pi_{n_p} - x_{n_p+1}y^U\lambda_{n_p}, \\ z &\leq \sum_{n_p=1}^{N_p} x_{n_p+1}\chi_{n_p} + y^L\pi_{n_p} - x_{n_p+1}y^L\lambda_{n_p}, \\ z &\leq \sum_{n_p=1}^{N_p} x_{n_p}\chi_{n_p} + y^U\pi_{n_p} - x_{n_p}y^U\lambda_{n_p}, \end{aligned} \tag{5-5}$$

$$\begin{aligned} x_{n_p}\lambda_{n_p} \leq \pi_{n_p} \leq x_{n_p+1}\lambda_{n_p} \quad & \forall n_p \in \{1, 2, \ldots, N_p\}, \\ y^L\lambda_{n_p} \leq \chi_{n_p} \leq y^U\lambda_{n_p} \quad & \forall n_p \in \{1, 2, \ldots, N_p\}, \end{aligned}$$

$$\sum_{n_p=1}^{N_p} \lambda_{n_p} = 1; \quad \lambda_{n_p} \in \{0, 1\} \quad \forall n_p \in \{1, 2, \ldots, N_p\}.$$

For each interval n_p, one additional binary variable λ_{n_p} and two continuous variables, π_{n_p} and χ_{n_p}, are defined. The convex hull over the whole domain, and the convex hull defined for each interval are illustrated in Figure 5-2, which shows the cross-section $y = c \in (y^L, y^U)$ of the 3D-space of function $z = xy$. This figure illustrates that the increasing the number of intervals improves the tightness of the linear relaxation.

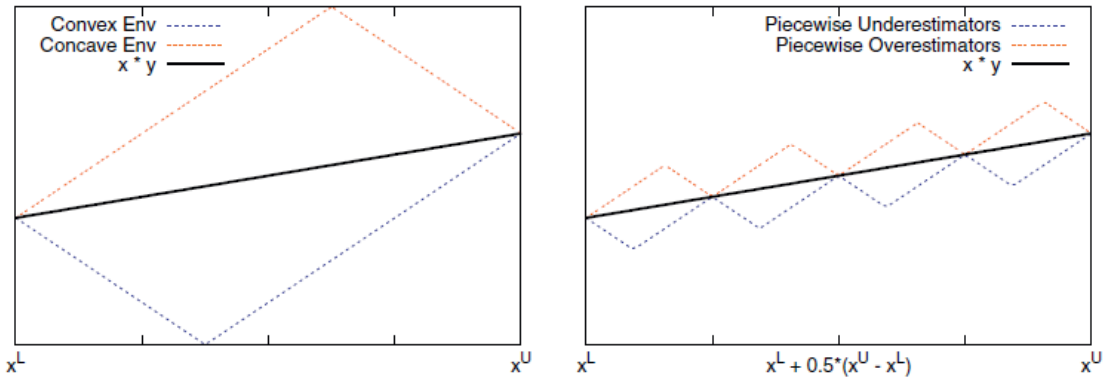


Figure 5-2: Convex hull of xy over a single interval (left) and partitioned interval for $N_p = 4$ (right) [Misener et al., 2011]

CH_{log} model

The partitioned linear relaxation from (5-5) yields a number of additional binary and continuous variables that scale linearly with the number of intervals N_p. The model stated in this section uses additional binary variables which scale logarithmically with the number of intervals, instead of linearly. This scaling is obtained by using a binary encoding of the interval number. APOGEE adapted this reformulation in such a way that it applies to bilinear programs. This reformulation is based on formulation **nf4r** [Gounaris et al., 2009], see Appendix D. The reformulation and the effects on solving time are stated next.

The main idea of the CH_{log} model, is that only $\lceil \log_2(N_p) \rceil$ variables are needed to define N_p intervals. This is obtained by using a binary representation of the intervals. A total of $N_l = \lceil \log_2(N_p) \rceil$ binary variables is needed:

$$\lambda_{n_l} \in \{0,1\}^{N_l}, \quad \lambda = [\lambda_1, \lambda_2, \ldots, \lambda_{N_l}]^\intercal .$$

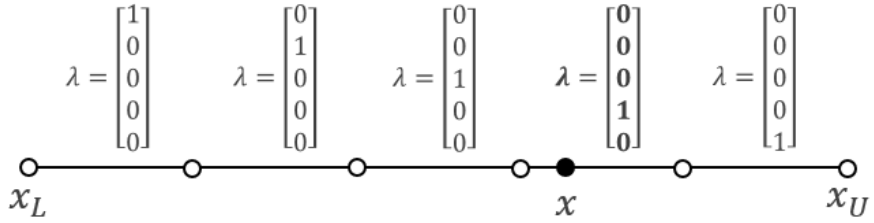
In the linear formulation of **nf4r**, (D-7) - (D-8), the binary variable λ represents the interval in which x lies. For the logarithmic formulation, λ represents the the largest interval boundary point smaller than x. An example for $N_p = 5$, where x lies in the fourth interval, is shown in Figure 5-3. Figure 5-3a shows that using the linear formulation yields

$$\lambda = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 \end{bmatrix}^\intercal ,$$

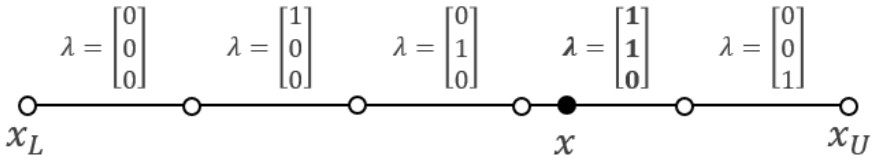
indicating that x lies in the fourth interval. The logarithmic formulation yields the vector:

$$\lambda = \begin{bmatrix} 1 & 1 & 0 \end{bmatrix}^\intercal ,$$

which indicates that x lies in the interval after the third breakpoint (i.e. $1 \times 1 + 1 \times 2 + 0 \times 4 = 3$) . Note that the most left breakpoint at x^L is considered to be the breakpoint zero. The intervals and corresponding values of λ are illustrated in Figure 5-3b. The



(a) Linear number of binary variables



(b) Logarithmic number of binary variables

Figure 5-3: Activation of fourth interval for $N_p = 5$ and $N_l = \lceil \log_2(N_p) \rceil = 3$

continuous switch Δy_{n_l} is now defined for each n_l:

$$\Delta y_{n_l} = \begin{cases} y - y^L & \text{if} \quad \lambda_{n_l} = 1 \\ 0 & \text{elsewhere.} \end{cases}$$

In addition, the introduction of one additional variable for each n_l is needed. The variable s_{n_l} defines *continuous slack* and is introduced since multiple λ_{n_l} can obtain a value of 1, as can be seen in Figure 5-3b.

The variable s_{n_l} can obtain the values:

$$s_{n_l} = \begin{cases} y - y^L & \text{if} \quad \lambda_{n_l} = 0 \\ 0 & \text{if} \quad \lambda_{n_l} = 1. \end{cases} \tag{5-6}$$

The definition of the continuous switch, continuous slack and binary variable λ from Equations (D-2) and (5-6) is obtained by the constraints:

$$\Delta y_{n_l} \leq (y^U - y^L)\lambda_{n_l} \quad \forall n_l \in \{1, 2, \ldots, N_l\}, \tag{5-7}$$
$$\Delta y_{n_l} = (y - y^L) - s_{n_l} \quad \forall n_l \in \{1, 2, \ldots, N_l\}, \tag{5-8}$$
$$s_{n_l} \leq (y^U - y^L)(1 - \lambda_{n_l}) \quad \forall n_l \in \{1, 2, \ldots, N_l\}. \tag{5-9}$$

The uniform interval length a for each interval $n_l \in \{1, 2, \ldots, N_p\}$, a, is defined as:

$$a = \frac{x^U - x^L}{N_p}.$$

Next, the interval containing x is bounded by the constraints:

$$x^L + \sum_{n_l=1}^{N_l} a 2^{n_l - 1} \lambda_{n_l} \leq x \leq x^L + a + \sum_{n_l=1}^{N_l} a 2^{n_l - 1} \lambda_{n_l}, \tag{5-10}$$

and one additional constraint which limits the right interval bound to be at most x^U. This constraint is only non redundant in case $\mod(\lceil \log_2(N_p) \rceil) \neq 2$:

$$x^L + a + \sum_{n_l=1}^{N_l} a 2^{n_l - 1} \lambda_{n_l} \leq x^U. \tag{5-11}$$

The reformulation of the partitioned convex hulls as formulated in (5-4) can now be stated as follows:

$$z \geq x y^L + x^L (y - y^L) + \left[\sum_{n_l=1}^{N_l} a 2^{n_l - 1} \Delta y_{n_l} \right], \tag{5-12}$$

$$z \geq x y^U + (x^L + a)(y - y^U) + \left[\sum_{n_l=1}^{N_l} a 2^{n_l - 1} \left(\Delta y_{n_l} - (y^U - y^L)\lambda_{n_l} \right) \right], \tag{5-13}$$

$$z \leq x y^L + (x^L + a)(y - y^L) + \left[\sum_{n_l=1}^{N_l} a 2^{n_l - 1} \Delta y_{n_l} \right], \tag{5-14}$$

$$z \leq xy^U + x^L(y - y^U) + \left[\sum_{n_l=1}^{N_l} a2^{n_l-1}\left(\Delta y_{n_l} - (y^U - y^L)\lambda_{n_l}\right)\right]. \tag{5-15}$$

For the example of five intervals with x in the fourth interval illustrated in Figure 5-3, the following values hold for different n_l: Substitution of these values in Constraint

n_l	1	2	3
λ_{n_l}	1	1	0
Δy_{n_l}	$y - y^L$	$y - y^L$	0
s_{n_l}	0	0	$y - y^L$
2^{n_l-1}	1	2	4

(5-12) indeed yields the first convex underestimator from Equations (5-4):

$$\begin{aligned} z &\geq xy^L + x^L(y - y^L) + a(y - y^L) + 2a(y - y^L) \\ &= xy^L + x^L(y - y^L) + 3ay - 3ay^L \\ &= (x^L + 3a)y + xy^L - (3a + x)y^L \\ &= x_4 y + y^L x - x_4 y^L. \end{aligned}$$

The other convex estimators can be deduced in the same way.

Constraints (5-7)-(5-15) represent the MIP implementation of the partitioned $\mathbf{CH_{log}}$ formulation, using a logarithmic number of binary variables, continuous variables and constraints.

Effect on computational performance

The effect of using the alternative logarithmic formulation of the linear relaxations was analyzed in [Misener et al., 2011]. They incorporated both the linear and alternative formulation in their global algorithm on twenty-five pooling problem instances. This algorithm is described in Section 5-1. The instances were solved for $N_p = 3, 4, 5, 6, 8, 12, 16$. They concluded that the logarithmic formulation outperforms the linear formulation from (5-5) on computational effort, especially when using a higher number of intervals ($N_p \geq 8$). For a smaller number of intervals, the linear formulation performed better. A possible explanation for this result was not given. A short analysis on the additional number of constraints, however, shows that, up to four intervals, the total number of constraints is 2-8% less for the linear formulation. For the case of $N_p = 8$, the logarithmic formulation needs 4-5% less constraints. Since the difference in additional binary variables is not that significant for $N_p = 4$, the use of additional constraints might explain the worse results for the logarithmic formulation. In addition, they showed it is beneficial to use a number of intervals, which is a power of 2.

The conclusion from [Misener et al., 2011] is that, when solving the bilinear pooling problem with the algorithm from Section 5-1, the logarithmic formulation is best to use in case the problem requires a higher number of partition intervals. Otherwise a linear formulation is recommended.

No results were presented on the performance of solving the MIP, separated from the other steps in the algorithm. Therefore, additional analysis is needed in order to compare the performance of both formulations isolated, when solving TWWN instances. applied

CH_{log} applied in APOGEE

The algorithm incorporated in APOGEE uses the linear relaxations described in the previous sections in a continuous branch-and-bound tree search [Misener et al., 2011]. It uses a strategy to select a new node and defines variable bounds for each new node using optimality-based bounds tightening [Belotti et al., 2009]. The upper bound (UB) is updated each time a feasible solution for the relaxed MIP program is obtained. The algorithm terminates in case one of the following holds:

1. The running time limit is reached: in this case the current upper bound UB is returned as solution, if present.
2. Branch-and-bound tree is empty while $UB \leq \infty$: in this case the best solution is returned.
3. Branch-and-bound tree is empty while $UB = \infty$: in this case an infeasibility notice is reported.

CH and CH_{log} applied to the TWWN problem

The application of linearly relaxed convex hulls is thoroughly tested for the use on pooling problems in [Misener et al., 2011]. Since this type of problem is a special case of the TWWN problem, as shown in Appendix 3-2, this method promises to be a suited method for the TWWN problem as well.

The model CH is used to solve the TWWN in [Karuppiah and Grossmann, 2006] for several instances. They concluded that it is best to partition the flow variable in intervals instead of the concentration variable. This conclusion was verified in [Misener et al., 2011]. The logarithmic formulation CH_{log} has not been applied to the TWWN up to now. Therefore, an analysis on the differences between the linear and logarithmic convex hull formulations for the TWWN problem is of value.

5-2 PWL approximation

In this section, the PWL approximations, most suited for the TWWN problem, are formulated. In case of a two-dimensional function $g(x)$, the PWL approximation results in line segments, connecting points on the original function $g(x)$: a PWL approximation of a three dimensional surface $f(x, y)$ results in triangles with its vertices on the original surface, see Figure 5-4. The resulting approximation is integrated in a MIP by constraining a solution point to be on the surface of any triangle.

To obtain this approximation, first, the domain is partitioned in triangles. The result is a *triangulation* of the domain. At the vertices of this triangulation, the breakpoints, the function values represent the vertices of the PWL approximation. The triangular planes connecting every set of corresponding vertices yield the linear approximation of the original function $f(x, y)$.

Many ways of triangulating the domain are possible. The importance of choosing a suited breakpoint strategy was pointed out in [Lin and Tsai, 2015] and different strategies were

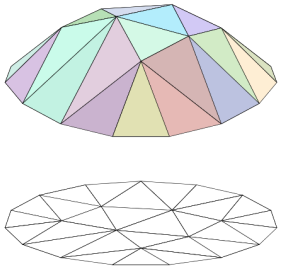


Figure 5-4: Piecewise linear approximation of 3D surface [Commons, 2007]

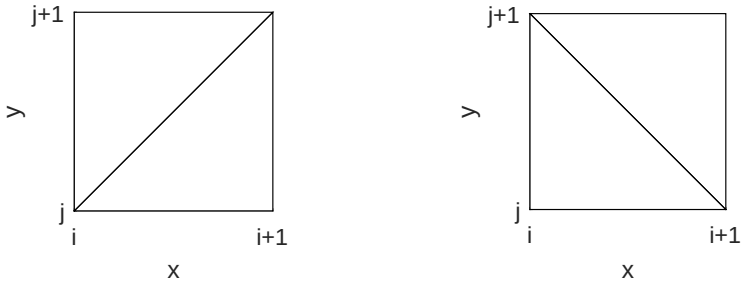
proposed in [Lundell, 2009]. For the TWWN problem and related pooling problem, a uniform breakpoint distribution was used in every single case. This research will therefore only focus on PWL approximation using a uniform breakpoint distribution.

For more details and a specific formulation of all models, the reader is referred to literature [D'Ambrosio et al., 2010, Vielma et al., 2010, Geißler et al., 2012]. These researches compare the available PWL approximation methods for multivariate problems.

In the remainder of this section, the domain triangulations are specified and the derivation of the MIP formulations PWL and PWL_{log} is provided.

Domain triangulation

This section describes the different ways of triangulating the domain with uniform intervals. This is discussed in literature [D'Ambrosio et al., 2010, Vielma et al., 2010, Geißler et al., 2012]. The triangulation determines whether the function is under or overapproximated. For some model formulations, different triangulations are possible, while for other models, only one triangulation is possible. The different triangulations are compared below. For one domain part $[x_i, x_{i+1}] \times [y_j, y_{j+1}]$, the two possible triangulations are shown in Figure 5-5.

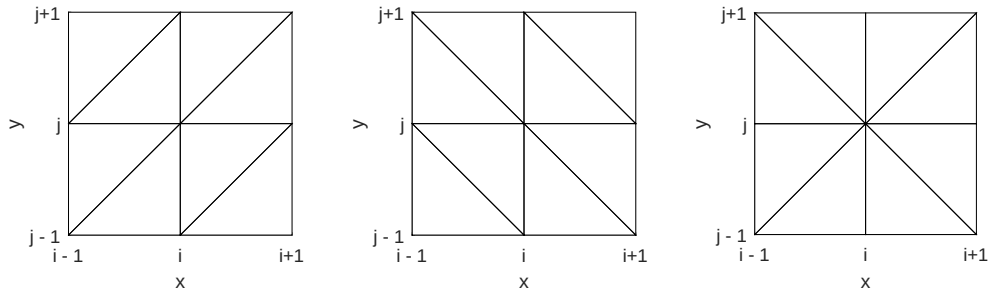


(a) Triangulation I **(b)** Triangulation II

Figure 5-5: Two triangulations for the interval $[x_i, x_{i+1}] \times [y_j, y_{j+1}]$

For more than one interval, three different triangulations can be defined: Triangulation I from Figure 5-5a repeated for each interval, Triangulation II from Figure 5-5b repeated for each interval, or using the two triangulations alternating. The latter is called the *Union Jack* triangulation in [Todd, 1977]. The three options are shown in Figure 5-6.

Each triangulation obtains a different PWL approximation of the surface, which is illustrated by the following example: define $f(x, y) = xy$ for $x, y \in [0, 1]$, shown in Figure



(a) Triangulation I **(b)** Triangulation II **(c)** Alternating

Figure 5-6: Three triangulations for 2×2 uniform intervals in the domain $[x_{i-1}, x_{i+1}] \times [y_{j-1}, y_{j+1}]$

5-7. For each triangulation from Figure 5-6, the PWL approximation of the surface is shown in Figure 5-8. Figure 5-8a shows that Triangulation I yields a linear overestimator for $f(x, y)$. Triangulation II in Figure 5-8b results in a linear underestimator. The Union Jack triangulation in Figure 5-8c and 5-8d alternates between a linear under- and overestimator for each interval.

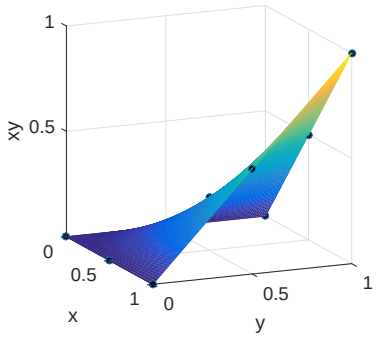
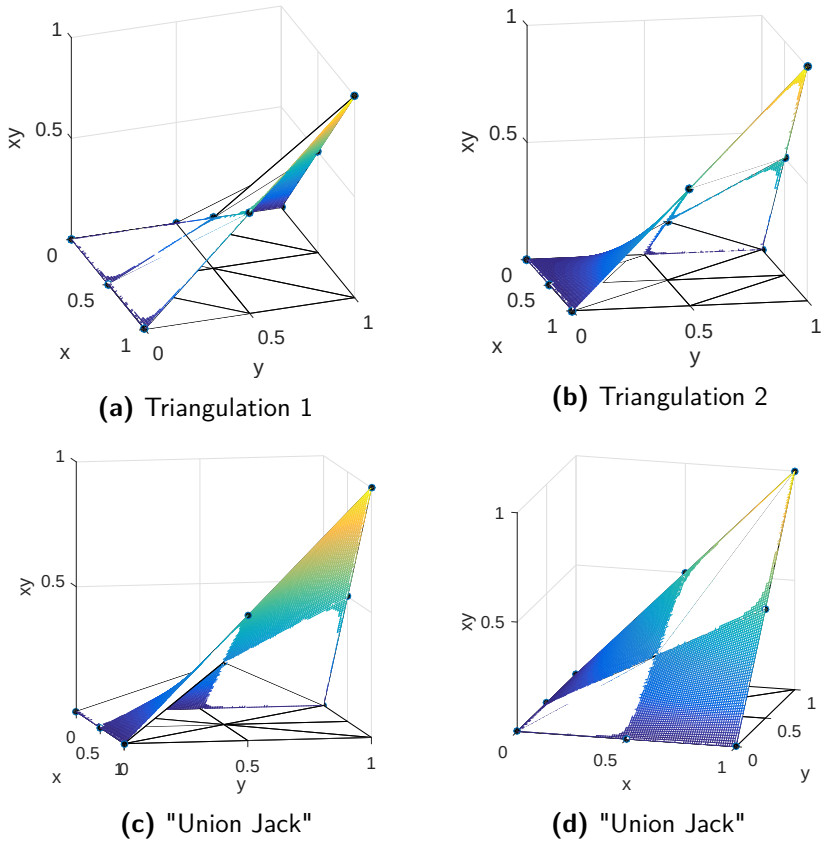


Figure 5-7: $f(x, y) = xy$ for $x, y \in [0, 1]$

Now, the different MIP formulations can be derived for the corresponding domain triangulations. The linearization techniques used to obtain these MIP formulations are stated in the following paragraphs.



(a) Triangulation 1 **(b)** Triangulation 2

(c) "Union Jack" **(d)** "Union Jack"

Figure 5-8: Three PWL approximations for 2×2 intervals in the domain $[0, 1] \times [0, 1]$

PWL model

The multivariate model PWL formulates a PWL approximation by imposing each point to be on the surface of a triangle that has its vertices on the original function [Dantzig, 1960, Lee and Wilson, 2001]. The solution point is formulated as a convex combination of the corresponding three vertices and is based on the *Convex combination model* (*CC*) from [Vielma et al., 2010]. The formulation used here is based on the explicit formulation from [D'Ambrosio et al., 2010]. The model is presented for Triangulation I from Figure 5-6a.

The domain $[x^L, x^U] \times [y^L, y^U]$ for a function $z(x, y)$ is considered. Both the x-domain and y-domain are partitioned. For $N_c \cdot M_c$ intervals, the breakpoints (x_i, y_j) are defined for $i \in \{1, 2, \ldots, N_c, N_c + 1\}$ and $j \in \{1, 2, \ldots, M_c, M_c + 1\}$, where:

$$x^L = x_1 < x_2 < \cdots < x_{N_c} < x_{N_c+1} = x^U,$$

and

$$y^L = y_1 < y_2 < \cdots < y_{M_c} < y_{M_c+1} = y^U.$$

The function $z(x, y)$ can be approximated by a convex combination of the function values of the breakpoints (x_i, y_j). The constraints that impose this approximation for $\alpha_{i,j} \in [0, 1] \ \forall i, j$ are:

$$x = \sum_{i=1}^{N_c+1} \sum_{j=1}^{M_c+1} \alpha_{i,j} x_i, \tag{5-16}$$

$$y = \sum_{i=1}^{N_c+1} \sum_{j=1}^{M_c+1} \alpha_{i,j} y_j, \tag{5-17}$$

$$z(x,y) = \sum_{i=1}^{N_c+1} \sum_{j=1}^{M_c+1} \alpha_{i,j} z(x_i, y_j), \tag{5-18}$$

$$\sum_{i=1}^{N_c+1} \sum_{j=1}^{M_c+1} \alpha_{i,j} = 1. \tag{5-19}$$

Here, the continuous variables $\alpha_{i,j}$ represent the weight of the corresponding vertex at (x_i, y_j). Since exactly one triangle has to be selected, only the breakpoints of the corresponding triangle may have a positive weight. This single triangle is selected by introducing one binary variable for each triangle. Since each interval contains two triangles, two binary variables $\lambda^U_{i,j}$ and $\lambda^L_{i,j}$ define the respectively upper and lower triangle for each interval, illustrated in Figure 5-9.

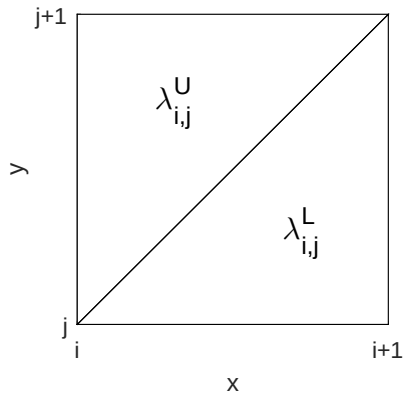


Figure 5-9: Definition of upper triangle and lower triangle

Now exactly one triangle from the domain is selected by the constraint:

$$\sum_{i=1}^{N_c} \sum_{j=1}^{M_c} \left(\lambda^L_{i,j} + \lambda^U_{i,j} \right) = 1. \tag{5-20}$$

What remains, is to allow only the vertices of the selected triangle to have positive weights. The constraint that ensures this is:

$$\alpha_{i,j} \leq \lambda^U_{i,j} + \lambda^L_{i,j} + \lambda^U_{i,j-1} + \lambda^L_{i-1,j} + \lambda^U_{i-1,j-1} + \lambda^L_{i-1,j-1} \quad \forall i,j, \tag{5-21}$$

with boundary values:

$$\lambda^L_{0,j} = \lambda^U_{0,j} = \lambda^L_{i,0} = \lambda^U_{i,0} = \lambda^L_{i,M_c+1} = \lambda^U_{i,M_c+1} = \lambda^L_{N_c+1,j} = \lambda^U_{N_c+1,j} = 0. \tag{5-22}$$

On the right-hand side of Constraints (5-21), exactly one binary variable is nonzero for $\alpha_{i,j}$ in case the vertex at (x_i, y_j) is part of the selected triangle.

Constraints (5-16) - (5-22) yield the MIP PWL model. For each bilinear term, the model uses an additional $2N_cM_c$ binary variables and $(N_c+1)(M_c+1)$ continuous variables.

For the Union Jack triangulation, another linearization technique is used in order to obtain a MIP model, called PWL_{log}. This technique is also based on convex combination

formulation, but needs less additional binary variables and constraints. Since the increase in binary variables and constrains scale logarithmically with the number of intervals, the model is referred to as PWL_{log}. This model is explained in the following paragraph.

PWL_{log} model

The model PWL_{log} reduces the number of additional binary variables of PWL. The approach that is used to improve the convex hull formulation in Section 5-1, is in fact derived from the PWL_{log} formulation *Log* in [Vielma and Nemhauser, 2008]. It is based on a branching scheme for this particular triangulation with only a logarithmic number of binary variables.

As in Section 5-2, consider $N_c \cdot M_c$ uniform intervals for the domain $[x^L, x^U] \times [y^L, y^U]$. The breakpoints that define the interval boundaries x_i are:

$$x^L = x_1 < x_2 < \cdots < x_{N_c} < x_{N_c+1} = x^U,$$

and y_j:

$$y^L = y_1 < y_2 < \cdots < y_{M_c} < y_{M_c+1} = y^U.$$

In order to define the intervals, a logarithmic number of binary variables is used:

$$\lceil log_2(N_c) \rceil = N_l$$

binary variables λ_n for $n \in \{1, 2, \ldots, N_l\}$, and

$$\lceil log_2(M_c) \rceil = M_l$$

binary variables λ_m for $m \in \{1, 2, \ldots, M_l\}$.

Since the PWL_{log} model is also based on the convex combination formulation, Constraints (5-16)-(5-19) still hold. As opposed to the PWL model, where the main idea is to select one triangle, and to constrain only the values $\alpha_{i,j}$ of the corresponding vertices to nonzero; the setup for the PWL_{log} method is different: three values $\alpha_{i,j}$ are selected that may be nonzero, according to the following rules:

1. For the x-dimension, it holds that: in case $\alpha_{i,j}$ is nonzero for x_i, then any other $\alpha_{i,j}$ can only be nonzero for x_i, and either the adjacent x_{i-1} or x_{i+1}. The same holds for the other dimension: if $\alpha_{i,j}$ is nonzero for y_j, then any other $\alpha_{i,j}$ can only be nonzero for y_j, and either the adjacent y_{j-1} or y_{j+1}.

2. In any triangle exactly one vertex is present where i and j are both not odd nor even, indicated by the squares and diamonds in Figure 5-10. Therefore, from the three selected $\alpha_{i,j}$, it holds for exactly one $\alpha_{i,j}$ that both i and j are nor odd nor even.

These rules results in the selection of one triangle and the according nonzero values $\alpha_{i,j}$. The MIP implementation of rule 2 requires the introduction of one additional binary variable μ and two subsets of breakpoints (x_i, y_j) :

$$\begin{aligned} \Delta_L &:= \{(x_i, y_j) \quad | \ i \text{ is even and } j \text{ is odd}\}, \\ \Delta_R &:= \{(x_i, y_j) \quad | \ i \text{ is odd and } j \text{ is even}\}. \end{aligned}$$

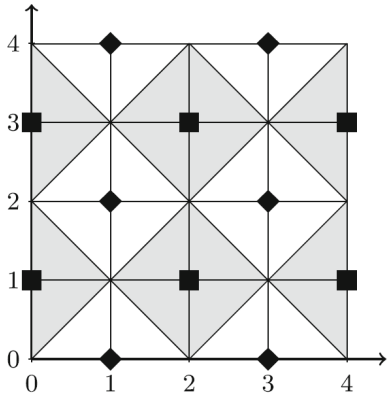


Figure 5-10: Selection of one triangle [Vielma and Nemhauser, 2008]

Set Δ_L is illustrated in Figure 5-10 by the rectangle shaped breakpoints, and Set Δ_R by the diamond shaped breakpoints. The following constraints enforce phase 2:

$$\sum_{(i,j)|(x_i,y_j)\in\Delta_L} \alpha_{i,j} \leq \quad \mu, \tag{5-23}$$

$$\sum_{(i,j)|(x_i,y_j)\in\Delta_R} \alpha_{i,j} \leq \quad (1-\mu). \tag{5-24}$$

According to the statements from rule 1, the variables $\alpha_{i,j}$ have to be constrained such that, in any dimension, at most two variables are nonzero. In addition, these two variables have to be adjacent. This property is called the *Special Ordered Set of type 2 (SOS2)* property. Implementation of this property using only a logarithmic number of binary variables, requires a SOS2 compatible binary representation of the intervals, which is called the *Gray code*. This representation has been used before for similar purposes, but was introduced in [Vielma and Nemhauser, 2008] for the alternating triangulation. A binary formulation holding the Gray code property, differs exactly in one digit in each subsequent interval. It is shown that, for each number of intervals, such a Gray code formulation can be obtained. Table 5-1 illustrates the difference between a natural binary encoding and Gray code for four intervals.

Table 5-1: Natural binary code and Gray code formulation

Interval	Binary		Gray	
	λ_2	λ_1	λ_2	λ_1
1	0	0	0	0
2	0	1	0	1
3	1	0	1	1
4	1	1	1	0

Figure 5-11 illustrates that the Gray code of two subsequent intervals are equal in all but one digits. This property enables that a set of statements can exclusively activate

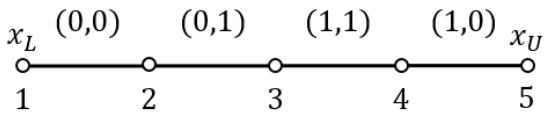


Figure 5-11: Binary interval representation Gray code in x-direction

two adjacent values $\alpha_{i,j}$. For the x-direction of example of $N_c = 4$ from Table 5-1, the following four statements suffice:

$$\begin{aligned} &\text{If } \lambda_1 = 0 \text{ then} && \alpha_{3,j} = 0 \ \ \forall j \\ &\text{If } \lambda_1 = 1 \text{ then} && \alpha_{1,j} = \alpha_{5,j} = 0 \ \ \forall j \end{aligned}$$

$$\begin{aligned} &\text{If } \lambda_2 = 0 \text{ then} && \alpha_{4,j} = \alpha_{5,j} = 0 \ \ \forall j \\ &\text{If } \lambda_2 = 1 \text{ then} && \alpha_{1,j} = \alpha_{2,j} = 0 \ \ \forall j \end{aligned}$$

Formulating these statements for any number of intervals as a MIP formulation results, in the following equations:

$$\sum_{j=1}^{M_c+1} \sum_{i|x_i \in S_X^+(n)} \alpha_{i,j} \leq \quad \lambda_n \qquad \forall n, \tag{5-25}$$

$$\sum_{j=1}^{M_c+1} \sum_{i|x_i \in S_X^0(n)} \alpha_{i,j} \leq \quad (1-\lambda_n) \qquad \forall n, \tag{5-26}$$

where

$$\begin{aligned} S_X^+(n) &:= \{x_i \mid \lambda_n = 1 \text{ for each adjacent interval}\} \quad \forall n, \\ S_X^0(n) &:= \{x_i \mid \lambda_n = 0 \text{ for each adjacent interval}\} \quad \forall n. \end{aligned}$$

Implementing the example for $N_c = 4$ indeed yields:

$$\begin{aligned} S_X^+(1) &:= \{x_3\}, \\ S_X^0(1) &:= \{x_1, x_5\}, \end{aligned}$$

$$\begin{aligned} S_X^+(2) &:= \{x_4, x_5\}, \\ S_X^0(2) &:= \{x_1, x_2\}, \end{aligned}$$

such that Constraints (5-25) and (5-26) impose the SOS2 property:

$$\begin{aligned} \sum_{j=1}^{M_c+1} \alpha_{3,j} &\leq \quad \lambda_1, \\ \sum_{j=1}^{M_c+1} \alpha_{1,j} + \alpha_{5,j} &\leq \quad (1-\lambda_1), \\ \sum_{j=1}^{M_c+1} \alpha_{4,j} + \alpha_{5,j} &\leq \quad \lambda_1, \\ \sum_{j=1}^{M_c+1} \alpha_{1,j} + \alpha_{2,j} &\leq \quad (1-\lambda_1). \end{aligned}$$

The same set definitions and constraints hold for the y-dimension:

$$\begin{aligned} S_Y^+(m) &:= \{y_j \mid \lambda_m = 1 \text{ for each adjacent interval}\}, \\ S_Y^0(m) &:= \{y_j \mid \lambda_m = 0 \text{ for each adjacent interval}\}. \end{aligned}$$

$$\sum_{i=1}^{N_c+1} \sum_{j \in S_Y^+(m)} \alpha_{i,j} \leq \lambda_m \qquad \forall m, \tag{5-27}$$

$$\sum_{i=1}^{N_c+1} \sum_{j \in S_Y^0(m)} \alpha_{i,j} \leq (1 - \lambda_m) \qquad \forall m. \tag{5-28}$$

This concludes the SOS2 constrains for both dimensions. In summary, Constraints (5-16) - (5-19), (5-23) - (5-28) define the MIP PWL_{log} model.

PWL and PWL_{log} applied to TWWN problem

Bilinearities are the main nonlinear component, making the use of PWL approximation promising for the TWWN problem. Since the triangulation models proved to be both accurate and flexible for solving MINLP multivariate problems [D'Ambrosio et al., 2010, Vielma et al., 2010, Geißler et al., 2012], this group of models is selected to be applied to the TWWN problem. Among the different models available within this group, the model *Log* from [Vielma et al., 2010] showed to outperform other models with respect to running time. Therefore, *Log* was the base of the model PWL_{log}, and the *Convex combination* model, *CC*, it is based on, will be used for the PWL model.

5-3 Discretization

Another linearization technique is based on discretization of part of the continuous variables. In the special case of a bilinear program, one variable from the bilinear term is discretized. Reformulation techniques can be applied to model the resulting MINLP as a MIP problem. A solution obtained by solving the MIP problem, guarantees to be feasible for the MINLP TWWN problem. This method is introduced in [Gupte et al., 2013] for the application of the bilinear pooling problem and followed by additional research in [Gupte et al., 2016]. In this section, the most promising reformulation technique is explained, which is hereafter named $\text{Discr}_{\text{log}}$.

Discretization relaxation

For each bilinear term of two continuous variables, one variable has to be discretized. Consider the bilinear equation $z = xy$, where $x \in [x^L, x^U]$ and $y \in [y^L, y^U]$. The set representation of this equation is:

$$S = \left\{(z, x, y) \in \mathbb{R}_+^3 \mid z = xy, x \in [x^L, x^U], y \in [y^L, y^U]\right\}.$$

Let us choose variable x to discretize. This yields subset $S^x \subset S$:

$$S^x := \left\{(z, x, y) \in \mathbb{R}_+ \times \mathbb{Z}_+ \times \mathbb{R}_+ \mid z = xy, x \in \{x^L, x^U\}, y \in [y^L, y^U]\right\}.$$

Set S^x still consists of nonlinear equations. Different reformulation methods are available to obtain a MIP formulation. All reformulations integrate the use of *McCormick envelopes* from Equation (5-3). They vary in the number of additional binary variables, continuous variables and constraints.

The *unary reformulation* of S^x introduces an additional binary variable, and a McCormick envelope for each integer in the range $\{x^L, x^U\}$. A second reformulation, the *log unary reformulation* reduces the additional number of binary variables logarithmically, by using the method introduced in [Vielma and Nemhauser, 2008]. It does, however, uses more additional continuous variables and constraints. A third method uses a base-2 expansion of the discretized variable, called the *binary reformulation*. This method requires only a logarithmic number of additional constraints and variables, both binary and continuous.

These three possible reformulations were compared on a variety of pooling problem instances in [Gupte et al., 2016]. Since the p-formulation of the pooling problem, as defined in Chapter 3-1, is in accordance with the TWWN formulation from Section 2-2, the focus is on the results found for this $p-$formulation. The experiments showed the best results for discretizing the flow variable for the p-formulation of the pooling problem, in combination with the *binary reformulation*, which is to be explained next.

binary reformulation

The binary reformulation uses an additional number of binary variables that scales logarithmically with number of discrete values of x: for the general case of $0 \leq x^L < x^U$, a total of

$$D := \lceil log_2(x^U) \rceil$$

binary variables is needed to formulate all possible integer values of x. Additionally, the discretization level can be refined to $\frac{1}{2^F}$ by introduction of F additional binary variables λ_d. Now $d \in \{-F, -F+1, .., D\}$, where $-F < D$.

The reformulation is obtained by two steps: first, set S^x is reformulated using a base-2 expansion of x. Then, the obtained formulation is linearized using the constraints from Equation (5-3).

The base-2 expansion of x is defined as:

$$x = \sum_{d=-F}^{D} 2^d \lambda_d, \tag{5-29}$$

where $\lambda_d \in \{0, 1\}\ \forall d$, and

$$x^L \leq \sum_{d=-F}^{D} 2^d \lambda_d \leq x^U. \tag{5-30}$$

Substitution of Equation (5-29) in $z = xy$ yields:

$$z = y \sum_{d=-F}^{D} 2^d \lambda_d. \tag{5-31}$$

Constraints (5-29) - (5-31) conclude the reformulation of set S^x with the base-2 expansion of x. Now, the MIP formulation of the discretized model $\text{Discr}_{\text{log}}$ can be formulated.

Discr$_{\text{log}}$ model

In order to obtain a MIP reformulation of Constraints (5-29) - (5-31), Equation (5-31) has to be linearized. Note that, w.l.o.g, this equation can be rewritten as:

$$z = \sum_{d=-F}^{D} (2^d \lambda_d y). \tag{5-32}$$

The bilinear terms $\lambda_d y$ can be linearized using the convex hull relaxation from Equation (5-3). Substitution of $\gamma_d := \lambda_d y$ in Equation (5-32) and introduction of the McCormick envelope constraints yields:

$$z = \sum_{d=-F}^{D} (2^d \gamma_d), \tag{5-33}$$

and

$$\begin{aligned} \gamma_d \geq \quad & \lambda_d^L y + y^L \lambda_d - \lambda_d^L y^L, \\ \gamma_d \leq \quad & \lambda_d^L y + y^U \lambda_d - \lambda_d^L y^U, \\ \gamma_d \geq \quad & \lambda_d^U y + y^U \lambda_d - \lambda_d^U y^U, \\ \gamma_d \leq \quad & \lambda_d^U y + y^L \lambda_d - \lambda_d^U y^L, \end{aligned} \qquad \forall d \in \{-F, -F+1, \ldots, D\}$$

Since $\lambda_d^L = 0$ and $\lambda_d^U = 1$, it holds that:

$$\begin{aligned} \gamma_d &\geq y^L \lambda_d, \\ \gamma_d &\leq y^U \lambda_d, \\ \gamma_d &\geq y + y^U \lambda_d - y^U \\ &= y + y^U (\lambda_d - 1), \\ \gamma_d &\leq y + y^L \lambda_d - y^L \\ &= y + y^L (\lambda_d - 1). \end{aligned} \tag{5-34}$$

Because of the binary variable in the bilinear term, only two cases can occur: $\lambda_d = 0$ or $\lambda_d = 1$. In case $\lambda_d = 0$, the first two constraints from (5-34) yield $\gamma_d = 0$. The other constraints are redundant. In case $\lambda_d = 1$, the last two constraints yield $\gamma_d = y$ and the first two constraints are redundant. This is exactly what is stated in Equation (5-32). This shows the following:

$$(5\text{-}32) \Leftrightarrow (5\text{-}33) \wedge (5\text{-}34).$$

In summary, the MIP formulation of set S^x is given by Constraints (5-29), (5-30), (5-33), (5-34), concluding the **Discr$_{\text{log}}$** model.

Discr$_{\text{log}}$ applied to the TWWN problem

Based on the results from [Gupte et al., 2016], the discretization as stated in the previous paragraphs, looks promising for the pooling problem application. Since the TWWN problem formulation is similar to the p-formulation of the pooling problem, these results indicate the same holds for application to the TWWN problem. Therefore, the binary reformulation based on a base-2 expansion of the flow variable, is applied to the MINLP problem. For the pooling problem, a discretization was applied in [Gupte et al., 2016]

such that only the integer values of the flow variable were allowed, i.e., $F = 0$ and $D = \lceil log_2(x^U) \rceil$. This level of discretization is accordingly the configuration applied to the TWWN problem.

5-4 Summary of linear solution methods

In this chapter, suitable linearization techniques were explained to reformulate a MINLP problem to a MIP problem. The resulting MIP problems can be solved with a linear solver.

The technique of linear relaxation is suitable for bilinear problems. It defines convex hulls around each nonlinear term in order to obtain a tight linear relaxation. In the case of bilinear terms, this is the tightest linear relaxation possible. By using partitioning of the domain, the tightness of the relaxation is further improved. The resulting MIP formulation is called the CH model. The time needed to solve CH can be reduced by using an alternative formulation of the relaxation, CH_{log}, using only a logarithmic number of additional binary variables. One drawback of these methods is that solutions of both **CH-CPLEX** and **CH_{log}-CPLEX** are not feasible for the original TWWN problem. It merely provides a lower bound on the global optimum.

In addition to relaxation, PWL approximation techniques can be applied. The resulting models are flexible in choosing the number of intervals in two dimensions. The MIP formulations based on convex combinations of the vertices showed to be the most promising. For different triangulations of the domain, different PWL techniques can be applied. For a first domain triangulation, Triangulation 1, the technique results in the PWL model. A drawback of this model is the rapid growth in additional binary variables for an increasing number of intervals. In order to reduce the number of additional binary variables, a MIP formulation using only a logarithmic number of binary variables can be obtained for the Union Jack triangulation: the PWL_{log} model. Solutions obtained for both methods **PWL-CPLEX** and **PWL_{log}-CPLEX** are not feasible for the original MINLP problem. Further more, the solutions are not guaranteed to be either an upper or lower bound on the global optimum.

The third technique introduced in this chapter is based on discretization of a part of the variables. The resulting MIP model, $\text{Discr}_{\text{log}}$, uses a base-2 expansion of the discretized variable. For the pooling problem it was shown to be most effective to discretize the flow variable. Hence, when applied to the TWWN problem, the flow variable is chosen to discretize. The method **$\text{Discr}_{\text{log}}$-CPLEX** results in a feasible upper bound for the global optimum of the MINLP TWWN problem.

Chapter 6

Experiments

The nonlinear and linear solution methods, introduced in Chapter 4 and 5 are used to optimize the TWWN. This chapter presents and evaluates the results for the different methods, instances and interval partitioning schemes. The set of instances consist of ten cases of TWWN problems, both theoretical and from application. The experimental setup is stated in Section 6-1. The actual results and their evaluation are presented in Section 6-2. To conclude, Section 6-3 discusses the main findings of this chapter.

The models are implemented and solved in AIMMS [Bisschop and Roelofs, 2006]. To solve the MIP problems, IBM ILOG CPLEX 12.6.2 [ILOG, 2014] is used. For the MINLP model, both BARON 15 [Sahinidis, 2016] and AOA [Hunting, 2011] are used. A 64-bit operating system is used with an Intel(R) Core(TM) i7-5600 CPU 2.60 GHz processor and 8.00 GB of RAM. The system operates under Windows 7.

6-1 Experimental setup

An overview of the methods that are tested is illustrated in Figure 1-4. Some additional information is stated in Table 6-1. For the PWL-CPLEX method, initially two different domain triangulations were considered, as defined in Chapter 5-2. For the sake of simplicity, only the results of Triangulation 1 are included in this report. In general, Triangulation 1 slightly outperforms the other triangulation and therefore Triangulation 2 is left out of further analysis.

Partitioning schemes

For the linear solution methods, different configurations are tested with respect to the number of partitioning intervals.

Both models CH and CH_{log} require partitioning of a single dimension. In Chapter 5-1 on page 45, the range of the flow variable was selected to partition. The partitioning schemes considered, consist of N uniform intervals over domain of each flow variable, where $N \in \{1, 2, 4, 8, 16, 32\}$. This results in a total of six configurations.

Table 6-1: Method overview

Method	Introduced in Chapter	Nr of dimensions that can be partitioned	Solution provides	Feasible for TFM model
Nonlinear methods				
TFM-SB&B	4-2	0	Global optimum	Yes
TFM-OA	4-1	0	Local optimum	Yes
Linear methods				
CH-CPLEX	5-1, p39	1	Lower bound	No
CH_{log}-CPLEX	5-1, p42	1	Lower bound	No
PWL-CPLEX	5-2, p48	2	Approximation	No
PWL_{log}-CPLEX	5-2, p50	2	Approximation	No
$Discr_{log}$-CPLEX	5-3	1	Upper bound	Yes

For the piecewise linear approximation methods PWL and PWL_{log}, both the flow variable and contamination variable may be partitioned. The different partitioning schemes are defined for N uniform intervals for the domain of the flow variables and M uniform intervals for the domain of the contaminant variables. The different configurations are defined as an element of set S_{conf} as defined below:

$$S_{conf} := \left\{ (N, M) \in \{1, 2, 4, 8, 16, 32\}^2 \mid NM \leq 32 \right\}. \tag{6-1}$$

This results in a total of 21 possible configurations.

For $Discr_{log}$, the continuous flow variables are redefined as discrete variables which are allowed to hold all integer values within the variable bounds. This results in one configuration.

Instances

The data set used for the experiments consists of ten problem instances from different sources. Some of them are slightly modified to fit the TFM as formulated in Section 2-2. The set is put together in such a way that a variety of network problems is taken into consideration. A summary of the instances is stated in Table 6-2.

The first nine instances are theoretical cases, used in previous research. Instance R1 however, represents a real life TWWN problem. Note that all instances include both TU's and PU's, except for instance B1, J1 and R1. Instance B1 is a water using network and Instance J1 and R1 are water treatment networks. All data is provided in detail in Appendix E, including the slight modifications.

Table 6-2: Intance overview

Instance	PU's	TU's	Sources	Sinks	Contaminants
K1	2	2	1	1	2
K2	3	3	1	1	2
K3	4	2	1	1	2
K4	5	3	1	1	3
A1	6	3	4	1	3
A2	2	3	1	1	4
B1	10	0	1	1	1
J1	0	2	7	1	5
T1	1	3	3	1	3
R1	0	7	2	3	3

Termination criteria

In case no additional remarks are made, the following termination criteria are used:

- The relative termination tolerance for CPLEX is set to $\epsilon = 1e - 003$.
- The relative termination tolerance for BARON is set to $\epsilon = 1e - 003$.
- The time limit for the MIP problems is set to 10800 seconds. If the time limit is the binding termination condition, the current best upper bound is taken as solution.

Evaluation criteria

The different methods are tested with respect to the following evaluation criteria:

- Solution quality: a solution is said to be accurate, if it is guaranteed to be within 1.5% of the global optimal solution.
- Running time: running time is said to be acceptable, if it is in the order of magnitude of 10^2 seconds.
- Time needed to close the relative gap to 5%, 2% and 1%.
- Relative gap at termination.
- Number of pipe connections in resulting network.

6-2 Results

Detailed results and analysis are presented here, though a reader who is only interested in major findings can directly go to page 70 for a summary. A general overview of the best performances per method is stated in Table 6-3. It shows the corresponding running times, partitioning schemes if used, and accuracy of the solution compared to the global optima, highlighted with the stars. In case no global optimum is found, the best upper bound is used as benchmark value. The solutions which are feasible for the TFM are highlighted with the f-superscripts.

For all but instances B1 and T1, a feasible upper bound is found that is guaranteed to be within 1.5% of the global optimum. Using TFM-SB&B, five instances are solved to global optimality. One should note that BARON did need a high quality initial solution, provided by AOA, and many trials in order to do so. BARON could not close the gap in case the initial gap was significant. Even with an initial upper bound provided by AOA, BARON showed no big improvements over time.

Another method that provides a feasible solution for the TWWN problem, is method TFM-AOA. It finds an upper bound for each instance within 5 minutes. In some cases this upper bound is equal to the global optimal solution. Due to random initialization of AOA, it might take several runs to obtain the solution quality as stated in Table 6-3.

A third method that yields a feasible solution for the TFM is $\text{Discr}_{\text{log}}$-CPLEX. The solutions, obtained by this method, show similar upper bounds as for TFM-AOA. However, the running time exceeded 10800 seconds for five instances. Note that in addition to the solving time, $\text{Discr}_{\text{log}}$-CPLEX needs additional implementation to obtain the MIP formulation, which TFM-AOA does not need.
For instance T1, the discretization method could not be directly applied, due to the fixed noninteger flows that are required for the processing units, see Appendix E-2, page 102. Therefore, the discretization model was solved, using the enclosing two integer values of the flow, hence, obtaining an infeasible solution for the TFM.
In order to improve the upper bound found by $\text{Discr}_{\text{log}}$-CPLEX, a post solve step can be executed. This is done by resetting the flow variables to the original continuous range, and fixing the flow variables in the model that have value zero. Then, BARON is used as solver to solve the remaining MINLP model. Due to the decrease in number of variables and constraints, the chances of finding a solution within acceptable time are reasonable. The results of this post solve step are stated in Table 6-4 on page 62. The running times of $\text{Discr}_{\text{log}}$-CPLEX can be found in Table 6-3. The results show that the post solve yields the global optimum for four instances In addition, for instance K3 an upper bound is obtained that is equal to the best found for either TFM-AOA or TFM-SB&B. The relative improvement with respect to the upper bound found by $\text{Discr}_{\text{log}}$-CPLEX is limited to less than 1%. Note that the postsolve step resulted in a feasible upper bound for instance T1 as well.

Table 6-3: Best performances per method

	TFM-SB&B			**TFM-AOA**			**Discr$_{log}$-CPLEX**			**CH-CPLEX**			
	LB ($/year)	Soln. ($/year)	Time (s)	UB ($/year)	Time (s)	% from global	UB ($/year)	Time (s)	% from global	LB ($/year)	Time (s)	N	% from global
Instance													
K1	5.804E+5	5.804E+5*	370.75	5.804E+5[f]	33.15	0.00	5.831E+5[f]	19.83	0.47	5.714E+5	471.54	32	-1.56
K2	3.781E+5	3.781E+5*	20.89	3.781E+5[f]	191.12	0.00	3.781E+5[f]	157.83	0.00	3.707E+5	5705.57	32	-1.94
K3	8.678E+5	8.697E+5[f]	>10800	8.697E+5[f]	143.12	≤0.21[a]	8.712E+5[f]	228.70	≤0.38[a]	8.679E+5	407.94	16	≤0.40[a]
K4	1.014E+6	1.027E+6[f]	>10800	1.025E+6[f]	721.07	≤1.02[a]	2.527E+6[f]	>10800	>100.00[b]	1.014E+6	1341.03	4	≤-1.04
A1	8.404E+5	-	>10800	8.517E+5[f]	203.25	≤1.35[b]	-	>10800	-	8.403E+5	103.58	2	≤-1.33[b]
A2	5.097E+5	5.097E+5*	32.24	5.106E+5[f]	256.92	0.09	5.102E+5[f]	3.07	0.09	5.069E+5	1.70	32	-0.55
B1	7.840E+5	1.388E+5[f]	>10800	1.351E+6[f]	100.00	≤72.26[c]	1.391E+6[f]	>10800	≥2.97[b]	7.839E+5	2066.86	16	≤41.95[b]
J1	1.927E+6	1.927E+6*	53.77	1.927E+6[f]	19.42	0.00	1.942E+6[f]	63.35	0.78	1.912E+6	183.21	32	-0.81
T1	1.081E+5	5.321E+5	>10800	5.321E+5	58.67	≤23.85[a]	5.581E+5	>10800	≤29.89[a]	1.335E+5	41.61	4	≤-45.75[b]
R1	4.945E+7	4.945E+7*	2.31	4.945E+7*	458.21	0.00	4.945E+7[f]	>10800	0.02	4.945E+7	0.11	1	0.00

	CH$_{log}$-CPLEX				**PWL-CPLEX**[d]				**PWL$_{log}$-CPLEX**[d]			
	LB ($/year)	Time (s)	N	% from global	Approx. ($/year)	Time (s)	N	% from global	Approx. ($/year)	Time (s)	N	% from global
Instance												
K1	5.714E+5	6.16	32	-1.56	5.680E+5	5504.68	8	-2.14	5.749E+5	6121.70	16	-0.95
K2	3.707E+5	67.41	32	-1.94	3.653E+5	189.20	4	-3.39	3.737E+5	7062.48	8	-1.17
K3	8.713E+5	8.70	16	≤0.40[a]	8.682E+5	2058.96	4	≤-0.18[b]	8.682E+5	2058.96	4	≤-0.18[b]
K4	1.014E+6	3329.86	8	≤-1.02	-	>10800	-	-	1.014E+6	0.86	2	≤-1.03[b]
A1	8.404E+5	53.79	2	≤-1.33[b]	-	-	-	-	8.403E+5	1.69	2	≤-1.33[b]
A2	5.069E+5	1.70	32	-0.55	5.097E+5	394.48	4	0.00	5.097E+5	3.90	4	0.00
B1	7.839E+5	830.86	16	≤-41.95[b]	7.875E+5	9705.65	1	≤-41.69[b]	7.875E+5	9772.53	1	≤-41.69[b]
J1	1.912E+6	3.65	32	-0.81	1.915E+5	1.98	1	-0.65	1.926E+5	294.58	8	-0.09
T1	4.297E+5	2356.33	16	≤-19.26[b]	4.904E+5	57.36	1	-7.84	4.904E+5	11.33	1	-7.84
R1	4.945E+7	0.11	1	0.00	4.945E+7	0.39	1	0.00	4.945E+7	0.17	1	0.00

[a] Based on best solution found by CH$_{log}$; [b] Based on best solution found by OA or SB&B; [c] Based on LB found by SB&B; [d] Solely based on configurations where $M = 1$; [f] Solution feasible for TFM; * Global optimum;

Table 6-4: $\text{Discr}_{\text{log}}$ - Results for integer flows and post solve improvement

	$\text{Discr}_{\text{log}}$ UB ($/year)	Post solve UB ($/year)	Running time post solve *(s)*	Relative Improvement *(%)*
Instance				
K1	5.831E+5	5.809E+5	23.07	-0.38
K2	3.781E+5	3.781E+5*	157.83	0.00
K3	8.712E+5	8.697E+5[a]	16.51	-0.17
K4	2.527E+6	2.527E+6	>10800	0.00
A1	-	-	-	-
A2	5.102E+5	5.097E+5*	0.76	-0.09
B1	1.391E+6	1.390E+6	8,073.07	-0.06
J1	1.942E+6	1.927E+6*	29.06	-0.77
T1	5.581E+5	5.550E+5	>10800	-0.55
R1	4.945E+7	4.945E+7*	0.73	-0.02

* Global optimum;
[a] Equal to best found upper bound by SB&B or OA.

The methods CH-CPLEX, CH_{log}-CPLEX, PWL-CPLEX and PWL_{log}-CPLEX, do not provide a feasible solution for the TWWN model. However, they are suited to find an approximation of the optimal network costs. The results in Table 6-3 show that for at least some partitioning levels, an accurate approximation of the global optimal solution can be obtained, by any of these methods. Detailed results of these methods are shown in Figure 6-1 and Figure 6-2. These figures show the running times and lower bounds per instance and for each interval partitioning. The results include values with a relative gap, named x in the legend, up to 1% at termination. Solution values with non-zero relative gap at termination are marked with a cross. For the models PWL and PWL_{log}, M is fixed to value 1. Note that models CH and CH_{log} yield equal solution values in case a solution is found within the time limits, as expected.

In Figure 6-1, some general observations can be made with respect to running time. For these ten instances, the methods based on CH_{log} and PWL_{log} outperform their linear counter methods. For almost every partitioning configuration, CH_{log} is fastest, followed by CH, PWL_{log} and PWL respectively. These differences in running time can be explained by the number of constraints and the number of binary variables, used in the model formulation. PWL uses, on average, 1.5 times the number of constraints used in CH, and up to eight times the number of binary variables, regardless of the interval partitioning scheme. While the differences are smaller for the logarithmic formulations, PWL_{log} uses up to 1.25 times the number of constraints used in CH_{log} and up to three times the number of binary variables. For increasing N, the differences become smaller. Using model PWL_{log} instead of PWL, results in 60% less constraints and up to 85% less binary variables. Model CH_{log} needs up to 50% less constraints and up to 60% less binary variables than CH.

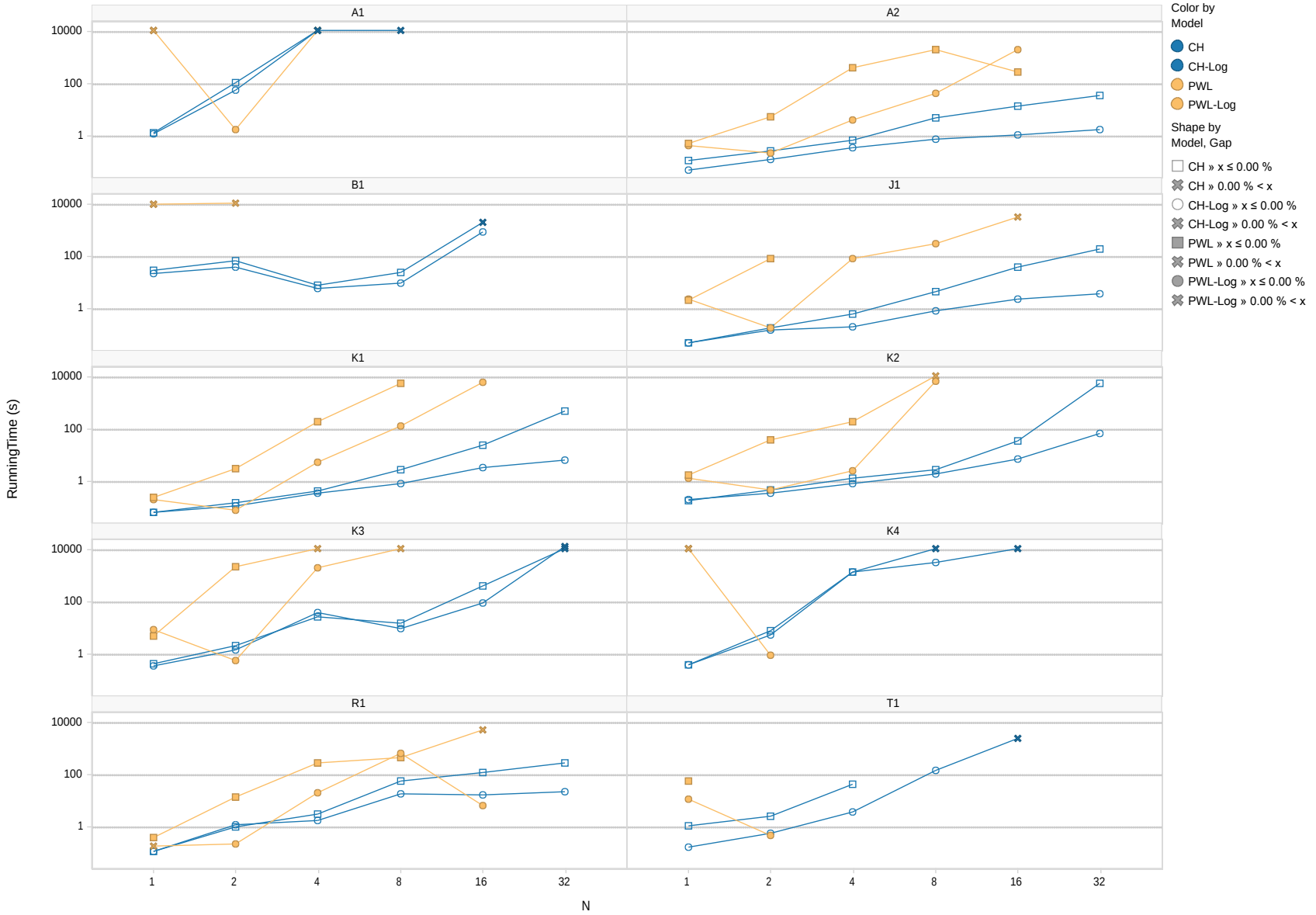


Figure 6-1: Running times for CH, CH_{log}, PWL and PWL_{log} for increasing N

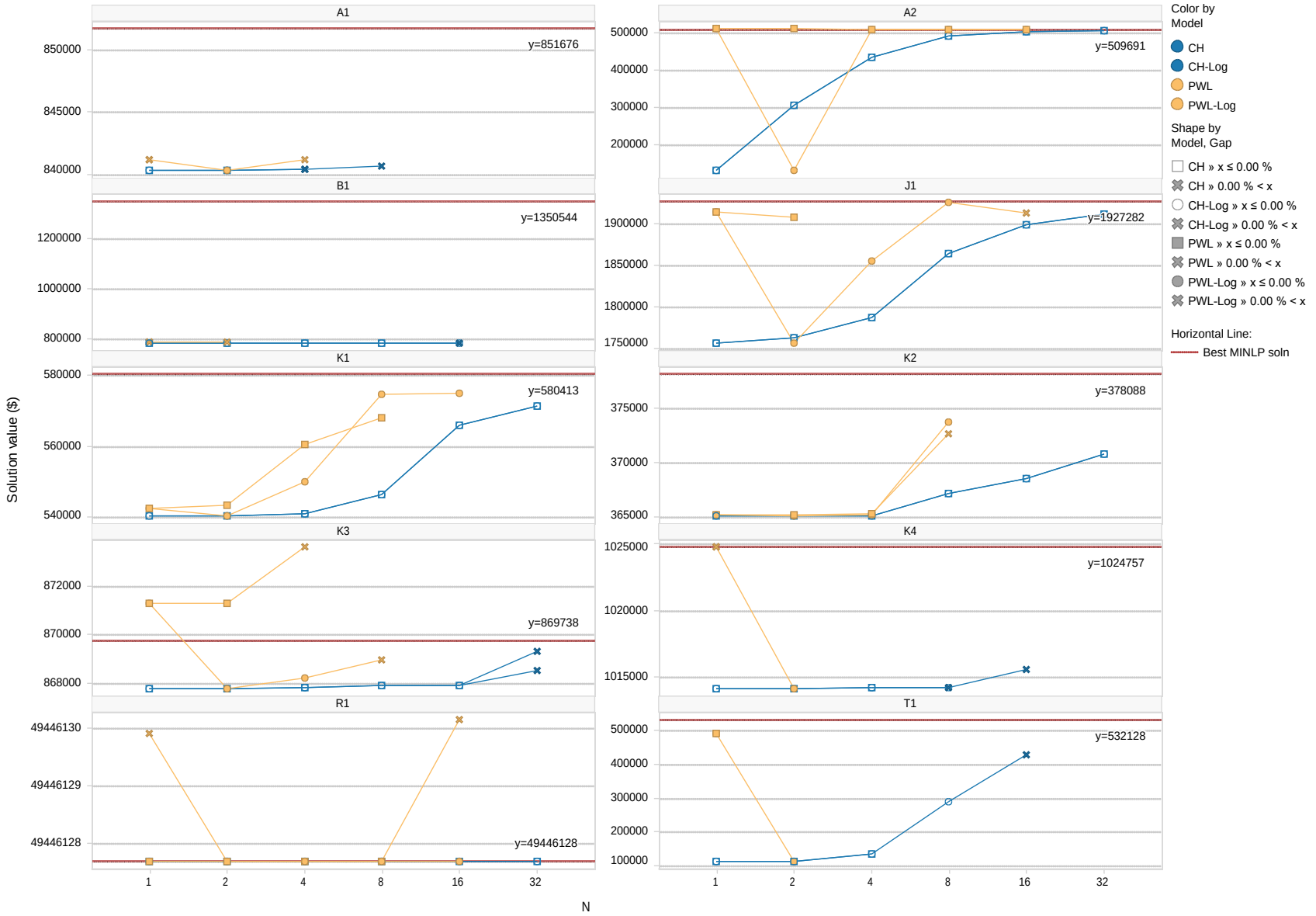


Figure 6-2: Solution values for CH, CH_{log}, PWL and PWL_{log} for increasing N

A comparison of solving the both linear relaxation models, shows that the efficient formulation of CH_{log} becomes of significant value for $N \geq 8$. This is illustrated by the results in Table 6-5. Here, both relaxation methods are compared with respect to number of completed runs and running time.

Table 6-5: CH-CPLEX and CH_{log}-CPLEX - Average running time for each N

	Nr. solutions $< 10800s$		Average running time *(s)* [a]	
	CH	CH_{log}	CH	CH_{log}
N				
1	10	10	3.50	**2.71**
2	10	10	20.29	**11.23**
4	9	9	158.02	**154.71**
8	7	**9**	15.20	**5.74**
16	7	**8**	385.61	**143.56**
32	5	5	1335.61	**20.12**

[a] Average taken over instances for which both CH and CHV terminated after less than 10800 seconds.

Figure 6-2 shows that some instances were easier to approximate, using linear solution methods, than others. For instances A2, J1 K4, T1 and R1, PWL-CPLEX finds an accurate approximation for partitioning level $N = 1$, which might not improve for increasing N. For almost every partitioning level N, the best approximations are obtained by the solutions methods PWL-CPLEX and PWL_{log}-CPLEX. For increasing N, this difference diminishes, and it occurs more often that only the linear relaxation based methods obtain a solution. For CH-CPLEX and CH_{log}-CPLEX, a larger value of N is needed to ensure a tight lower bound on the global optimal solution. However, due to the relatively high speed of the method CH_{log}-CPLEX, higher partitioning schemes can be solved while maintaining acceptable running times. The following paragraphs elaborate more on the effect of using different partitioning schemes.

The average improvement of the solution quality, obtained by doubling the previous value of N, is stated in Table 6-6 for both linear relaxation based methods.

Table 6-6: CH-CPLEX and CH_{log}-CPLEX - Relative improvement of solution value for increasing N

	CH						**CH_{log}**					
N	1	2	4	8	16	32	1	2	4	8	16	32
	-	13.48	6.10	2.74	1.21	0.57	-	13.48	6.10	15.03	7.16	0.57

For CH_{log}-CPLEX, on average, every increase of N up to the value of 16, results in an significant improvement of solution quality. For CH-CPLEX, the average improvement decreases with every increase of N. Table 6-7 states, for each instance, the minimal number of intervals N needed, to approximate the best feasible solution for the TFM up to 5%, 2% and 1% respectively. Due to missing accurate information on the global

optimal value of instance B1 and T1, the results for these instances are unknown. These results show that for the remaining instances, an approximation within 5% of the best found feasible solution can be found within four seconds.

Table 6-7: CH_{log} - Minimum N needed to approximate the best found feasible solution up to 5%, 2% and 1%

Final gap	5%		2%		1%	
	min. N	Time *(s)*	min. N	Time *(s)*	min. N	Time (s)
Instance						
K1	16	3.20	32	6.16	-	-
K2	1	0.19	32	67.41	-	-
K3	1	0.33	1	0.33	1	0.33
K4	1	0.36	1	0.36	-	-
A1	1	1.15	1	1.15	-	-
A2	8	0.72	16	1.03	32	1.70
B1						
J1	8	0.80	16	2.15	32	3.65
T1						
R1	1	0.11	1	0.11	1	0.11

Both PWL-CPLEX and PWL_{log}-CPLEX perform worse than CH-CPLEX and CH_{log}-CPLEX with respect to running time, with the exception of configurations where $N = 2$. However, for the corresponding solution values in Figure 6-2, this particular configuration for $N = 2$ obtains low quality solutions for PWL_{log}-CPLEX. It is not directly explainable why this odd jump occurs consistently over a number of instances. Starting from $N = 2$, better solution approximations are obtained for each increase in N. For five instances, using any $N > 1$ did not obtain a better approximation for PWL-CPLEX and PWL_{log}-CPLEX than $N = 1$.

The results presented for the methods PWL-CPLEX and PWL_{log}-CPLEX so far, were limited to $M = 1$. This is because no clear improvement is obtained for either running time or approximation accuracy for increased M. Table 6-8 states the accuracy of the approximations per partitioning schemes that terminated before the time limit was reached. One can observe that no better approximations are found for any partitioning scheme with $M > 1$ and any value of N, than for $M = 1$ and some N. The running times increase for configurations with increasing $M \cdot N$.

Table 6-8: PWL and PWL_{log} - Percentage from best feasible solution for different partitioning configurations

		PWL						**PWL_{log}**					
	N	1	2	4	8	16	32	1	2	4	8	16	32
Instance	M												
A1	1								-1.33				
	2							-1.33	-1.33				
	4							-1.04	-1.33				
	8												
A2	1	0.58	0.58	0.00	0.00	0.00		0.58	-74.33	0.00	0.00	0.00	
	2	0.58	0.58	0.00				-74.33	-74.33	-15.65	-74.33	-74.33	
	4	0.58						-0.46	-40.02	-20.12	-3.74		
	8	0.58						-0.46	-0.22				
	16	0.92						-74.33	-40.02				
B1	1	-41.69						-41.69					
	2							-42.11	-42.11	-42.1	-42.07	-41.95	
	4								-42.11	-42.1	-42.07		
	16								-42.11				
J1	1	-0.65	-1.01					-0.65	-8.88	-3.72	-0.09	-0.75	
	2	8.54	0.04					-8.88	-8.88	-7.29	-8.88	-1.65	
	4	24.36						0.19	-8.52	-7.64	-3.27		
	8							9.85					
	16							-8.88					
K1	1	-6.55	-6.39	-3.42	-2.14			-6.55	-6.90	-5.22	-0.97	-0.95	
	2	-6.60	-6.02	-2.33				-6.90	-6.90	-6.80	-6.90	-2.48	
	4	-6.55						-6.90	-6.90	-6.83	-5.89		
	8	-2.28						-3.41	-1.60				
	16							-6.90	-6.90				
K2	1	-3.40	-3.40	-3.39				-3.40	-3.44	-3.40	-1.17		
	2	-1.27						-3.44	-3.44	-3.44	-3.44	-2.72	
	4							-3.44	-3.44	-3.44	-3.01		
	8							-1.27					
	16							-3.44					
K3	1	0.18	0.18					0.18	-0.23	-0.18			
	2							-0.23	-0.23	-0.22	-0.23		
	4							-0.21	-0.23	-0.22	-0.21		
	8							-0.21					
	16							-0.23					
K4	1								-1.03				
	2							-1.03	-1.03				
	4									-1.03			
	8								-1.03				
	16								-1.03				
R1	1	0.00	0.00	0.00	0.00	0.00		0.00	0.00	0.00	0.00	0.00	
	2								0.00				
	4								0.00				
T1	1	-7.84						-7.84	-78.79				
	2	-6.14						-78.79	-78.79	-74.9	-78.79		
	4							-8.17	-78.63	-75.71			

So far, the results based on solution value quality and running times have been stated. What remains to look at, is the solution itself. One of the main characteristics of the solution is the number of pipe connections, that is selected in the optimal configuration. Figure 6-4 shows the resulting number of pipes found by each method for each instance per interval partitioning. Even though no clear common behavior is observed, for some instances, the number of pipes found by the linear solution methods gives a rough indication of the number of pipes in a quality feasible solution. Overall, the solutions found by PWL-CPLEX and PWL_{log}-CPLEX provide the best indication of the number in the best found feasible network configuration. This information could be used in other methods that obtain feasible solutions, to specify the model parameter κ_{max}.

To conclude the results, a few findings are stated on the real life case, instance R1. It appears that for the real life case, R1, all methods, except $\text{Discr}_{\text{log}}$, are able to find the global optimal solution within a relative short amount of time. No partitioning of the domains is needed in order to obtain an accurate outcome. This might explained by the very distinct options the instance contains. Since the costs on the sinks are relatively high compared to other costs, these are binding. By choosing one effluent option, the according treatment facilities needed are as good as fixed. One of the optimal solutions found by SB&B is illustrated in Figure 6-3. All optimal networks found by the models use sink 1.

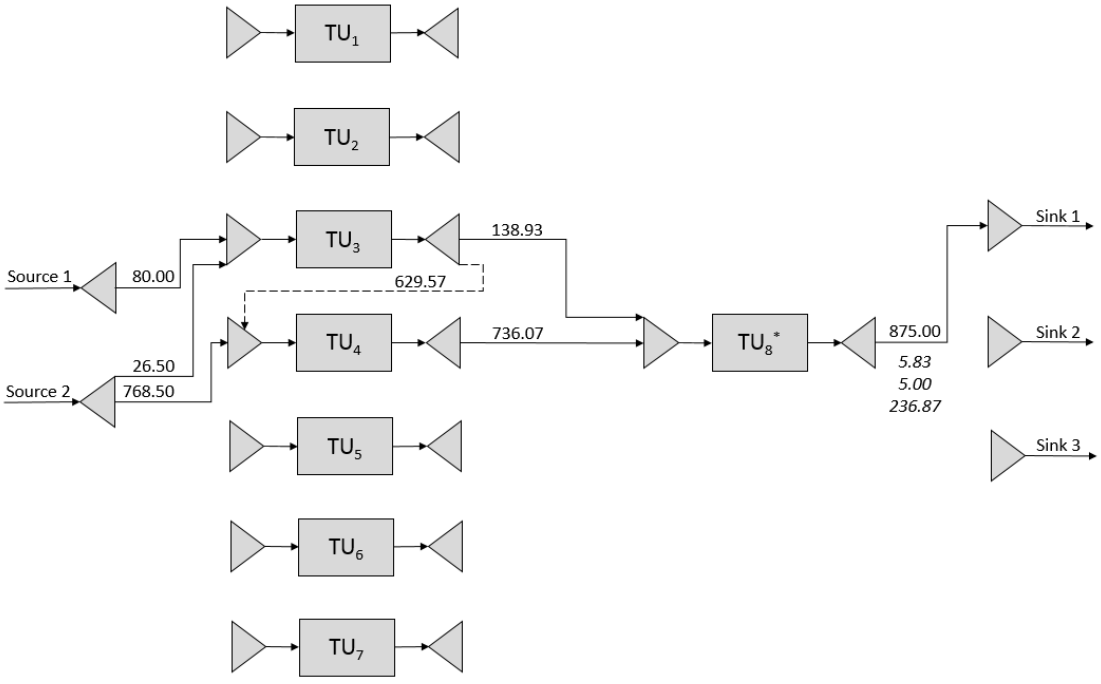


Figure 6-3: Best found solution for instance R1

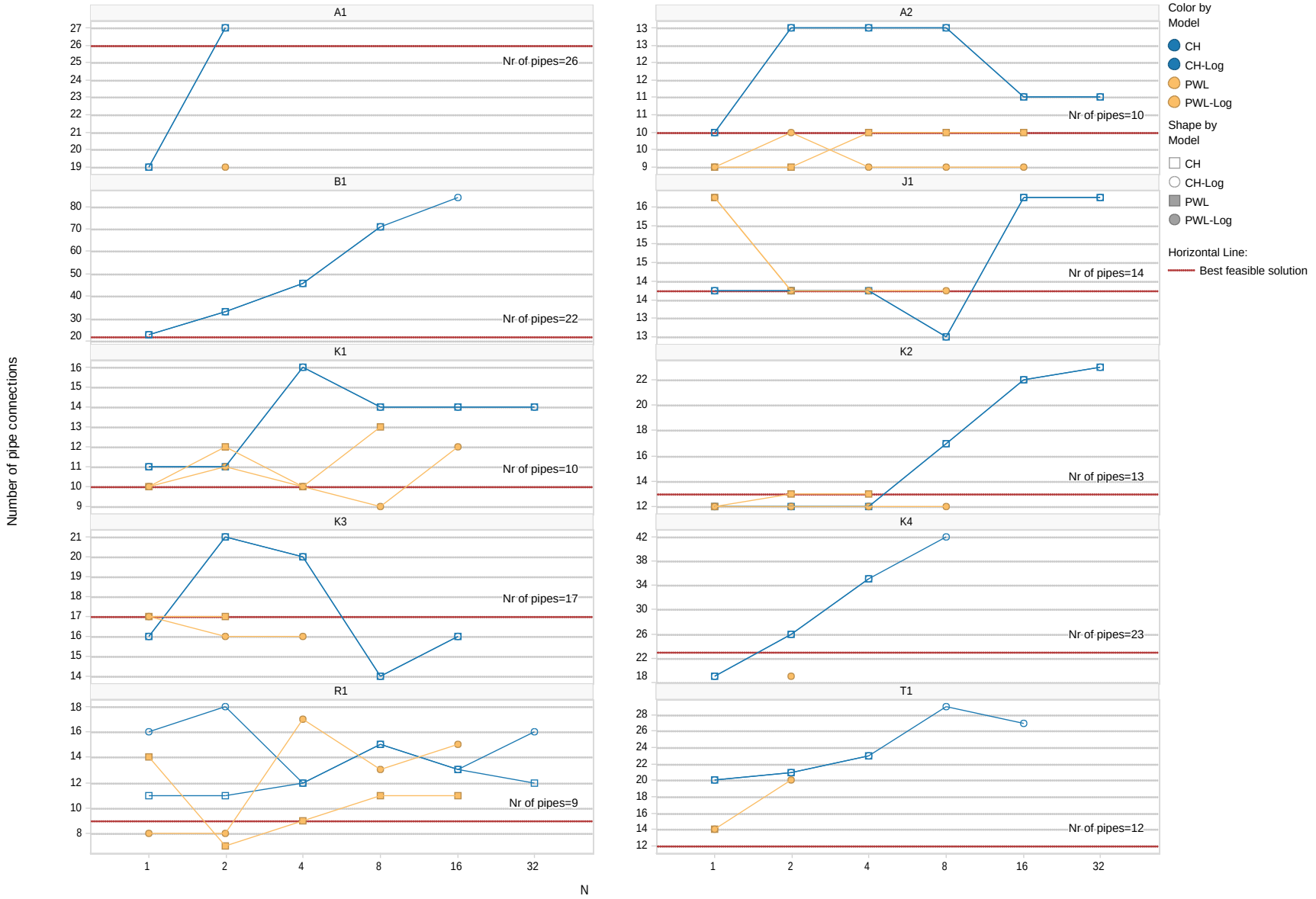


Figure 6-4: Number of pipe connections for CH, CH_{log}, PWL and PWL_{log}

6-3 Results summary

Running the different solution methods on a set of test instances, showed that finding a high quality solution with TFM-SB&B is not guaranteed within reasonable time. As an alternative, the methods TFM-AOA and $\text{Discr}_{\text{log}}$-CPLEX both provide an accurate upper bound. In most cases, TFM-AOA obtained a solution more quickly. $\text{Discr}_{\text{log}}$-CPLEX can be slightly enhanced by using a postsolve technique, which improves the upper bound.

Other methods aim to approximate the optimal network costs without obtaining a feasible solution for the TFM. All four methods CH-CPLEX, CH_{log}-CPLEX, PWL-CPLEX and PWL_{log}-CPLEX are able to obtain a tight approximation on the global optimal solution for the right interval partitioning N. With respect to running time, the methods using a logarithmic formulation technique outperform their linear counter method. In addition, CH-CPLEX and CH_{log}-CPLEX outperform PWL-CPLEX and PWL_{log}-CPLEX. Overall, CH_{log}-CPLEX is the fastest method, where the difference becomes significant for $N \geq 8$. Furthermore, this method is guaranteed to obtain a better lower bound for each increasing N.

For PWL-CPLEX and PWL_{log}-CPLEX, increasing the interval partitioning for the contamination dimension, M, did not show to improve the approximation quality.

An analysis on the solutions showed that the number of pipe connections in the best feasible network configuration can be roughly indicated by the solutions of PWL-CPLEX and PWL_{log}.

All methods were able to find a very accurate solution value for the real life instance. In fact, this instance showed to be one of the most easy to solve.

Chapter 7

Conclusion and recommendations

This thesis investigated suitable solution methods for solving the MINLP total waste water network problem. The motivation was that optimal solutions are difficult to obtain within reasonable time. In specific, this thesis addressed the following questions:

1. What are the characteristics of the MINLP formulation of the TWWN problem that affect choosing a suitable solution method?
2. What modeling approaches exist in literature and how can they be implemented to solve the TWWN problem?
3. What is the performance of the proposed methods when applied to different problem instances?

A suitable solution method is considered to be the combination of a model and solver, that provides an accurate solution to the TWWN problem within reasonable time, within 1.5% of the global optimal solution.

In order to find the most suited methods, first the MINLP formulation of the TWWN problem was analyzed. It was proven that the TWWN problem is NP-hard. Based on the nonlinear characteristics, a literature review resulted in possibly suitable methods. Five of these methods include a linearization of the MINLP model to a MIP, solved by a linear solver. The other two methods are based on solving the MINLP model with a nonlinear solver. The performance of the methods was tested on the basis of ten TWWN instances. Based on these results, the research question will be answered. The following sections answer the sub-questions and conclude on the main research question.

1. What are the characteristics of the MINLP formulation of the TWWN problem that affect choosing a suitable solution method?

It was shown, that the TWWN problem is NP-hard. Hence, it is not guaranteed that the problem can be solved within polynomial time. This supports the need for alternative solution methods, besides solving the MINLP problem with a global solver. Whether a solution method is based on an alternative modeling approach, or on an alternative

solver, in both cases it is of importance that it suits the specific nonlinear elements of the MINLP model. Three different types of nonlinearities are present in the TFM:

- Constraints that include a product of a continuous and binary variable.
- Constraints that consist of bilinear terms of two continuous variables.
- Exponential terms in the objective function.

The first type of nonlinear term was reformulated as a set of three linear constraints. The objective function was linearized using a linear underestimator. The presence of the bilinear terms was binding for the selection of suitable solution methods.

2. What modeling approaches exist in literature and how can they be implemented to solve the TWWN problem?

A literature review showed that a variety of alternative methods may be applicable to the TWWN problem and related problems. An overview of these methods was stated in Figure 1-4. Besides finding the global optimum solution of the TFM with a spacial Branch-and-Bound solver, an outer approximation algorithm can be used to find an upper bound for the TFM. These two methods were named TFM-BARON and TFM-AOA respectively. Another way of obtaining an upper bound is to discretize the continuous flow variables and reformulate the resulting problem to a MIP by using additional binary variables and constraints. The method that solves this MIP with CPLEX is called $\text{Discr}_{\text{log}}$-CPLEX.

Other methods included relaxation of the bilinear terms in the TFM, using convex hulls. The method that solves the resulting MIP with CPLEX provides a lower bound on the optimal solution value. The lower bound can be improved by partitioning of one dimension of the search domain, which results in a tighter relaxation. Two related methods were considered. They differ in the formulation technique: CH-CPLEX and CH_{log}-CPLEX.

Finally, two approximation methods were considered: PWL-CPLEX and PWL_{log}-CPLEX. The base of these methods is a MIP model obtained by piecewise linear approximation of the nonlinear model elements. The model accuracy can be improved by partitioning one or two dimensions of the search domain. While CH-CPLEX and PWL-CPLEX formulate the MIP with a linear increase in additional binary variables and constraints for increasing partitioning schemes of the search domain, CH_{log}-CPLEX and PWL_{log}-CPLEX use a logarithmic increase.

3. What is the performance of the proposed methods when applied to different problem instances?

The performance of each of the methods, mentioned in previous answer, was tested and evaluated on the basis of several criteria. These criteria were: (i) solution quality, (ii) running time, (iii) time needed to close the relative gap, (iv) relative gap at termination, and (v) number of pipe connections in the resulting network. In addition, it was taken into account whether the method finds an upper bound, approximation or lower bound on the global optimum. A varied set of instances was used to test the methods, including

networks containing either processing units or treatment units exclusively, networks with several sources, several sinks and a real life case.

The experiments showed that finding the global optimal solution of the MINLP of the TFM is quite difficult using one single method. For five out of ten instances, TFM-SB&B found the global optimum within reasonable time. Two instances were not even solved with a relative gap up to 1.5%. The methods $\text{Discr}_{\text{log}}$-CPLEX and TFM-AOA both provided upper bounds with comparable performance. The upper bounds found by TFM-AOA proved to be closer to the optimal solution value in most cases and solved faster. This method was a good trade-off between speed and solution quality. Due to random initialization of the solver AOA, it has taken a few runs to find the accuracy as stated in Table 6-3.

Other methods aimed to approximate the optimal network costs without obtaining a feasible solution for the TFM. All four methods CH-CPLEX, CH_{log}-CPLEX, PWL-CPLEX and PWL_{log}-CPLEX were able to obtain a tight approximation on the global optimal solution for the right interval partitioning N. Overall, CH_{log}-CPLEX was the fastest method, especially for $N \geq 8$. For a partitioning level of $N = 16$, CH_{log}-CPLEX guaranteed a lower bound not more than 2% from the global optimal solution for at least eight instances. For this configuration, the running times varied from one second to one minute.

Even though PWL-CPLEX and PWL_{log}-CPLEX provided accurate approximations for some configurations, the methods were not guaranteed to find either an upper or lower bound on the global optimum. Using PWL and PWL_{log}-CPLEX, the best approximations were obtained for partitioning levels $M = 1$ and $N = 1$ or $N = 8$.

The experiments showed that there are suitable methods available for solving different variations of the TWWN problem, though no accurate approximation was obtained for the instances B1 and T1, by any method or combination of methods. These instances may contain elements that made them harder to solve. Instance B1 was different from the other instances since it was a network containing only processing units, with no tight upper bound on the maximal flows. Instance T1 had a large complexity since it contained three sources, a processing unit and three treatment units. For other network configurations consisting of one source, one or multiple sinks and a number of processing and treatment units, an accurate solution has been obtained.

In order to assess the applicability of the computational methods, also a problem arising in application was considered: instance R1. All methods except $\text{Discr}_{\text{log}}$-CPLEX found the global optimal solution. It was not needed to use domain partitioning to do so. This instance was one of the easier ones to solve, which indicated that it is acceptable to extend the conclusions made for the literature cases, to real life cases as well.

To conclude, an answer can be stated for the main research question:

What solution methods are suitable for solving the MINLP total waste water network problem?

Since the method CH_{log}-CPLEX resulted in an accurate lower bound, it provided a good indicator that an upper bound found by either TFM-SB&B, TFM-AOA or $Discr_{log}$-CPLEX was close to optimal. Out of these three methods, TFM-AOA showed to be the most suitable to obtain an upper bound. By combining the information from CH_{log}-CPLEX and TFM-AOA, the global optimum was tightly approximated from below and above respectively, solving the TWWN problem accurately within reasonable time.

Recommendations for practical use

Using a global solver to solve the MINLP TWWN problem to global optimality is time consuming and does not guarantee to converge within reasonable time at all. Therefore, it is recommended to use alternative solution methods to approximate the global optimum as accurately as possible: a combination of TFM-AOA and CH_{log}-CPLEX provides an accurate approximation, since the former method provides an accurate upper bound, while the latter obtains an accurate lowerbound. It is effective to use a partitioning level of $N = 16$ for obtaining the lower bound with CH_{log}-CPLEX.

During this research, it proved to be important to invest time in obtaining logic variable bounds as tight as possible, by using the network characteristics. By doing so, TFM-AOA has a smaller search space and the probability of finding an accurate upper bound increases accordingly. In addition, the relaxations used in CH_{log}-CPLEX will become more tight and therefore its lower bound is more accurate.

The performance of the different solution methods indicate that some TWWN instances are easier to solve than others. Therefore, it is recommended to define the problem in which preferably not more than one of the following factors is unknown:

- The number of treatment units needed.
- The number of sources and sinks that are used.
- The order in which the process units are operated.

Instances containing a significant amount of network elements and more of these questions incorporated, might not be solvable by any method.

Recommendations for future work

As an extension of this research, the following factors could be varied in order to improve the method performances:

- As suggested in [Karuppiah and Grossmann, 2006], among others, constraints can be added to reduce the search space and speed up the solving time. One redundant constraint is added, but there are additional possibilities to explore: a short network analysis could determine how many pipe connections would be present in a possible optimal network design. Setting the parameters κ_{max} and κ_{min} slightly above and below this estimation respectively, may cut off parts the search space with a disproportional number of pipe connections. For the instances used in this thesis, default values of $\kappa_{min} = 0$ and $\kappa_{max} = 100$ were used.
- Zooming in on the settings of the methods as applied in this research, there are additional options to consider. For the method Discr, only one configuration with discretized intervals is considered. More research can be done on the effect of either increasing or reducing the discretization precision. An increased precision could result in more accurate upper bounds, but could also take additional running time. The opposite would be expected for a reduced precision.

In addition, one element that has not been elaborated on in this research, is the effect of linearizing the objective function. The original exponential cost terms in the objective function have been fixed to their linear underestimator, in order to compare the different solutions methods on a common ground. The effects on the solution and solution value of this linearization are not yet clear.

This thesis considered solely treatment units which remove contamination with a certain removal rate. Both in literature and practice, another type of treatment is used as well: reducing contamination up to a fixed threshold concentration, independent of the incoming contaminant concentration, see Example 6 from [Ahmetović, 2011] and Instance R1 in Appendix E-3. It would be of added value to incorporate this treatment option.

To conclude, it would be interesting to evaluate the methods and configurations based on a larger set of test instances. Due to the relative small set of ten test instances, it is not possible to detect patterns and to extend the conclusions to groups of a specific type of instance. Increasing the instance set would provide insight in which particular characteristics of a TWWN make it more difficult or easy to solve by specific methods.

Appendices

Appendix A

Component Based MINLP model

In this appendix a second formulation of the MINLP model is stated for a TWWN; the CBM. This model was analyzed as an optional MINLP formulation. Since this model was found to be inferior to the TFM, this section is added solely for comparison.

The model definition and formulation are stated in Appendix A-1. A comparison between this second model formulation and the earlier stated TFM from Section 2-2 can be found in Appendix A-2. This comparison resulted in a decision to continue this research using the TFM. The arguments leading to this decision are stated in Appendix A-3.

A-1 Model formulation

The CBM is based on modeling the component flow rates of separate contaminants [Galan and Grossmann, 1998, Galan and Grossmann, 1999]. This model uses contaminant flow rates f (kg/h), instead of contaminant concentrations ρ (ppm). In addition, the model contains one extra variable: the splitter fraction.

The variables used in this model are stated in Table A-1, the additional network parameters in Table A-2. The cost related parameters are equal to the TFM parameters in Table 4. After the definitions, the constraints are stated. The objective is equal to the one stated in Equation (2-25) from the TFM.

Table A-1: Variables CBM

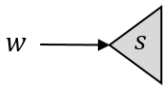
Variable	Description	# Variables
Continuous non negative variables		
ϕ_w	water intake from source w (ton/h)	w
$\phi_{p,in}$	flow rate into process unit p (ton/h)	p
$\phi_{p,out}$	flow rate out of process unit p (ton/h)	p
$\phi_{t,in}$	flow rate into treatment unit t (ton/h)	t
$\phi_{t,out}$	flow rate out of treatment unit t (ton/h)	t
ϕ_d	water flow rate in discharge of sink d (ton/h)	d
$\phi_{s,m}$	water flow rate through from splitter s to mixer m (ton/h)	$(w+p+t)(p+t+d)$
$f_{p,c,in}$	flow rate contaminant c entering process unit p (kg/h)	pc
$f_{p,c,out}$	flow rate contaminant c exiting process unit p (kg/h)	pc
$f_{t,c,in}$	flow rate contaminant c entering treatment unit t (kg/h)	tc
$f_{t,c,out}$	flow rate contaminant c exiting treatment unit t (kg/h)	tc
$f_{s,m,c}$	flow rate contaminant c from splitter s to mixer m (kg/h)	$c(w+p+t)(p+t+d)$
$f_{d,c}$	flow rate contaminant c in discharge of sink d (kg/h)	cd
$\zeta_s^m \in [0,1]$	split fraction from splitter s to mixer m $(-)$	$(w+p+t)(p+t+d)$
Binary variables		
$\psi_{s,m}$	existance of connection splitter s to mixer m (-)	$(w+p+t)(p+t+d)$

Table A-2: Additional parameters CBM

Parameter	Description
Water sources	
$f_{w,c}$	flow rate from source w of contaminant c (kg/h)
Process units	
$\overline{f}_{p,c,in}$	maximum flow rate of contaminant c entering process unit p (kg/h)
$\overline{f}_{p,c,out}$	maximum flow rate of contaminant c exiting process unit p (kg/h)
Treatment units	
$\overline{f}_{t,c,in}$	maximum flow rate of contaminant c entering treatment unit t (kg/h)
$\overline{f}_{t,c,out}$	maximum flow rate of contaminant c exiting treatment unit t (kg/h)
Water sinks	
$\overline{f}_{d,c}$	maximum discharge flow rate of contaminant c in sink d (kg/h)
$\underline{f}_{d,c}$	minimum discharge flow rate of contaminant c in sink d (kg/h)

Constraints

Water source

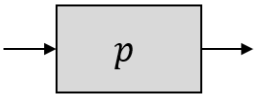


Source

Bound on intake:

$$\phi_w \leq \overline{\phi}_w \quad \forall w. \tag{A-1}$$

Process unit



Process unit

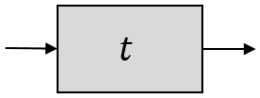
Bounds on intake and output:

$$\underline{\phi_p} \leq \phi_{p,in} \leq \overline{\phi}_p \quad \forall p, \tag{A-2}$$
$$f_{p,c,in} \leq \overline{f}_{p,c,in} \quad \forall p, c, \tag{A-3}$$
$$f_{p,c,out} \leq \overline{f}_{p,c,out} \quad \forall p, c, \tag{A-4}$$

Balance around process unit:

$$\phi_{p,out} = \phi_{p,in} + \phi_{p,\Delta} \quad \forall p, \tag{A-5}$$
$$f_{p,c,out} = f_{p,c,in} + \omega_{p,c} \quad \forall p, c. \tag{A-6}$$

Treatment unit



Treatment unit

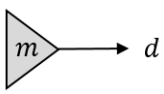
Bounds on intake and output:

$$\underline{\phi_t} \leq \phi_{t,in} \leq \overline{\phi}_t \quad \forall t, \tag{A-7}$$
$$f_{t,c,in} \leq \overline{f}_{t,c,in} \quad \forall t, c, \tag{A-8}$$
$$f_{t,c,out} \leq \overline{f}_{t,c,out} \quad \forall t, c. \tag{A-9}$$

Balance around treatment unit:

$$\phi_{t,in} = \phi_{t,out} \quad \forall t, \tag{A-10}$$
$$f_{t,c,out} = (1 - \frac{\lambda_{t,c}}{100}) \cdot f_{t,c,in} \quad \forall t, c. \tag{A-11}$$

Water sink

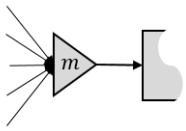


Sink

Bounds on intake:

$$\underline{\phi}_d \leq \phi_d \leq \overline{\phi}_d \quad \forall d, \tag{A-12}$$
$$\underline{f}_{d,c} \leq f_{d,c} \leq \overline{f}_{d,c} \quad \forall d, c. \tag{A-13}$$

Unit mixer



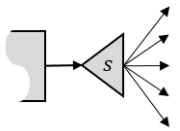
Mixer

Balance around mixer:

$$\phi_{u,in} = \sum_{s} \phi_{s,m_u} \quad \forall u, \tag{A-14}$$

$$f_{u,c,in} = \sum_{s} f_{s,m_u,c} \quad \forall u, c. \tag{A-15}$$

Unit splitter



Splitter

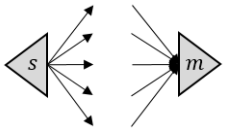
Balance around splitter:

$$\sum_{m} \zeta_{s_u,m} = 1 \quad \forall u, \tag{A-16}$$

$$\zeta_{s_u,m} \cdot \phi_{u,out} = \sum_{m} \phi_{s_u,m} \quad \forall u, \tag{A-17}$$

$$\zeta_{s_u,m} \cdot f_{u,c,out} = f_{s_u,m,c} \quad \forall u, c. \tag{A-18}$$

Pipe constraints



Pipe

Bound on flow rate:

$$\phi_{s,m} \leq \overline{\phi}_{s,m} \psi_{s,m} \quad \forall (s,m) \in PC_{free}. \tag{A-19}$$

$$\phi_{s,m} \leq \overline{\phi}_{s,m} \quad \forall (s,m) \in PC_{fixed}. \tag{A-20}$$

Bound on number of connections:

$$\kappa_{min} \leq |P_{fixed}| + \sum_{\substack{(s,m) \\ \in PC_{free}}} \psi_{s,m} \leq \kappa_{max}. \tag{A-21}$$

Bound on recycling

$$\phi_{s,m} \leq \theta_{TU} \cdot \overline{\phi}_{t,in} \quad \forall t, \tag{A-22}$$

$$\phi_{s,m} \leq \theta_{PU} \cdot \overline{\phi}_{p,in} \quad \forall p. \tag{A-23}$$

Redundant cuts

Network flow balance:

$$\sum_{w} \phi_w + \sum_{p} \phi_{p,\Delta} = \sum_{d} \phi_d. \tag{A-24}$$

Network contaminant balance:

$$\sum_{w} f_{w,c} + \sum_{p} \omega_{p,c} = \sum_{d} f_{d,c} + \sum_{t} (f_{t,c,in} - f_{t,c,out}) \quad \forall c. \tag{A-25}$$

This concludes the formulation of the Component Based MINLP model in equations (2-25), (A-1) - (A-25).

A-2 Model comparison

The formulation of the MINLP model greatly influences all following steps from solving the model up to finding appropriate linearization methods. Therefore a detailed comparison is required in order to select the most suitable formulation. The main differences between the two models are stated in Table A-3, under the assumption that all pipe connections from any splitter to any mixer are variable, so $P_{fixed} = \emptyset$.

The number of nonlinear equations for the TFM is obtained by counting the number of nonlinear equations (2-6), (2-15), (2-17), (2-24), summed over all related indices. These nonlinearities contain bilinear terms and product terms with a continuous and binary variable. For the CBM the same approach is used for the bilinear equations (A-17), (A-18). The number of bilinear terms is counted by summing all nonlinear terms in the equations stated above over all relevant indices. For The CBM this yields the same number of nonlinear terms as the number of equations, since all nonlinearities consist of a single bilinear term. The total number of variables is counted by adding all values in the last column from Table 2 and Table A-1. The difference in number of variables between the two models shows to be equal to the number of possible splitter to mixer connections in the network.

In order to compare the differences more explicitly, Table A-3 shows the number of nonlinearities, bilinear terms and number of variables for random but representative small- and larger instances with a varying number of contaminants. This model comparison shows significantly less nonlinear equations for the TFM for all cases and this difference increases with both the instance size and number of contaminants. However, the number of total bilinear terms is less for the CBM in most cases. It's interesting to see that this difference in number of bilinear terms does not increase drastically with the total number of bilinear terms for increasing instance sizes. Thus it is concluded that the CBM contains approximately 10% less bilinear terms for different instance sizes.

Since linearizing nonlinearities creates additional variables, constraints and computational effort, a MINLP model with the least possible nonlinearities is desired. It is interesting to compare the examples' values in Table A-4 with a statement made in literature [Karuppiah and Grossmann, 2006]; In section 4 is said that a model like The

Table A-3: Comparison MINLP models

Characteristic	TFM	CBM
Location nonlin	mixers	splitters
	process units	
	objective function	
Type nonlin eqn	multiple bilinear terms	single bilinear term
	product of continuous and binary variable	
# nonlin eqns	$c(2p+t+d+1)$ $+c(p+t)$	$(p+t+d)(w+(c+1)(p+t))$
# bilinear terms	$c((1+t+p)(p+t+d)$ $+2(t+p)+d)$	$(p+t+d)(w+(c+1)(p+t))$
# Variables	$(c+3)(w+p+t)(p+t+d)$ $+(c+1)(2p+2t+d)+w$	$(c+2)(w+p+t)(p+t+d)$ $+(c+1)(2p+2t+d)+w$
Unique variables	contaminant concentration ρ	contaminant flow f
		split fraction ζ
magnitude variables	water flow rate ϕ: order 2	water flow rate ϕ: order 2
	contaminant concentration ρ: order 2	contaminant flow f: order 1
		split fraction ζ: order 0

TFM contains fewer bilinearities than a model formulation as the CBM. This is in contradiction with 3 out of 4 examples discussed here.

A final measure of comparison is the order of magnitude of the different variable types [Karuppiah and Grossmann, 2006]. In case variables are divergent with respect to order of magnitude, additional numerical difficulties may occur. For the TFM, both the flow rate and contaminant concentration variables obtain values within the same range. For the CBM however, the order of magnitude differs among the variable types from the $[0, 1]$ range on the split fraction to order 2 variables on the water flow rates.

A-3 Choice of MINLP model

With respect to all differences discussed above, both models have their advantages and disadvantages. Since the number of bilinear terms is from the same order of magnitude, this is not a binding decision criterion. This does not outweighs the importance of a numerical solvability of the model. This is why the TFM is chosen to implement as MINLP model formulation and used as benchmark to compare with different linearized MILP models. This model choice is in accordance with the conclusion drawn in an earlier comparison [Karuppiah and Grossmann, 2006].

Table A-4: Example configurations

Instance characteristic	TFM	CBM
$w = 1, p = 2, t = 2, d = 1, c = 3$		
Number of nonlinear equations	**36**	85
Number of bilinear terms	102	**85**
# Variables	**137**	162
$w = 1, p = 2, t = 2, d = 1, c = 7$		
Number of nonlinear equations	**84**	165
Number of bilinear terms	238	**165**
# Variables	**298**	323
$w = 3, p = 5, t = 5, d = 2, c = 3$		
Number of nonlinear equations	**84**	516
Number of bilinear terms	**462**	516
# Variables	**869**	1025
$w = 3, p = 5, t = 5, d = 2, c = 7$		
Number of nonlinear equations	**196**	996
Number of bilinear terms	1078	**996**
# Variables	**1581**	1637

Appendix B

TFM with linear objective

In this appendix the complete TFM with linear objective is stated. The original TFM is formulated in Section 2-2 and the nonlinear objective is derived in Section 2-4.

$$
\begin{aligned}
\min_{\phi,\rho,\psi} \quad & AR \sum_t \pi_{t,IC} \cdot \hat{\phi}_t + H \sum_t \pi_{t,OC} \cdot \phi_t \\
& + AR \left(\sum_{\substack{(s,m) \\ \in PC_{free}}} \pi_{s,m,fix} \cdot \psi_{s,m} + \sum_{\substack{(s,m) \\ \in PC_{fixed}}} \pi_{s,m,fix} \right) \\
& + AR \sum_{(s,m)} \pi_{s,m,var} \cdot \hat{\phi}_{s,m} \\
& + H \sum_{(s,m)} \pi_{s,m,op} \cdot \phi_{s,m} + H \sum_w \phi_w \cdot \pi_w \\
s.t. \quad & \hat{\phi}_t \geq (\phi_t^L)^\alpha + \left(\frac{(\phi_t^U)^\alpha - (\phi_t^L)^\alpha}{\phi_t^U - \phi_t^L} \right) (\phi_t - \phi_t^L) && \forall t \\
& \hat{\phi}_{s,m} \geq (\phi_{s,m}^L)^\gamma + \left(\frac{(\phi_{s,m}^U)^\gamma - (\phi_{s,m}^L)^\gamma}{\phi_{s,m}^U - \phi_{s,m}^L} \right) (\phi_{s,m} - \phi_{s,m}^L) && \forall s,m \\
& \phi_w \leq \overline{\phi}_w && \forall w \\
& \underline{\phi_p} \leq \phi_{p,in} \leq \overline{\phi}_p && \forall p \\
& \underline{\phi_t} \leq \phi_{t,in} \leq \overline{\phi}_t && \forall t \\
& \underline{\phi_d} \leq \phi_d \leq \overline{\phi}_d && \forall d \\
& \phi_{s,m} \leq \overline{\phi}_{s,m} \cdot \psi_{s,m} && \forall (s_u,m) \in PC_{free} \\
& \phi_{s,m} \leq \overline{\phi}_{s,m} && \forall (s,m) \in PC_{fixed} \\
& \rho_{p,c,in} \leq \overline{\rho}_{p,c,in} && \forall p,c
\end{aligned}
$$

$$\rho_{p,c,out} \leq \overline{\rho}_{p,c,out} \qquad \forall p, c$$

$$\rho_{t,c,in} \leq \overline{\rho}_{t,c,in} \qquad \forall t, c$$

$$\underline{\rho}_{d,c} \leq \rho_{d,c} \leq \overline{\rho}_{d,c} \qquad \forall d, c$$

$$\phi_{u,out} = \sum_{m} \phi_{s_u,m} \qquad \forall u$$

$$\phi_{u,in} = \sum_{s} \phi_{s,m_u} \qquad \forall u$$

$$\phi_{p,out} = \phi_{p,in} + \phi_{p,\Delta} \qquad \forall p$$

$$\phi_{t,in} = \phi_{t,out} \qquad \forall t$$

$$\phi_{u,in} \cdot \rho_{u,c,in} = \sum_{s} \phi_{s,m_u} \cdot \rho_{s,m_u,c} \qquad \forall u, c$$

$$\phi_{p,out} \cdot \rho_{p,c,out} = \phi_{p,in} \cdot \rho_{p,c,in} + 1000 \cdot \omega_{p,c} \qquad \forall p, c$$

$$\rho_{t,c,out} = (1 - \frac{\lambda_{t,c}}{100}) \cdot \rho_{t,c,in} \qquad \forall t, c$$

$$\rho_{s_u,m,c} = \rho_{u,c,out} \cdot \psi_{s_u,m} \qquad \forall u, c \text{ s.t. } (s_u, m) \in PC_{free}$$

$$\rho_{s_u,m,c} = \rho_{u,c,out} \qquad \forall u, c \text{ s.t. } (s_u, m) \in PC_{fixed}$$

$$\kappa_{min} \leq |PC_{fixed}| + \sum_{\substack{(s,m) \\ \in PC_{free}}} \psi_{s,m} \leq \kappa_{max}$$

$$\phi_{s_t,m_t} \leq \theta_{TU} \cdot \overline{\phi}_{t,in} \qquad \forall t$$

$$\phi_{s_p,m_p} \leq \theta_{PU} \cdot \overline{\phi}_{p,in} \qquad \forall p$$

$$\rho_{w,c} + 1000 \sum_{p} \omega_{p,c} = \sum_{d} \phi_d \cdot \rho_{d,c} + \sum_{t} \phi_t (\rho_{t,c,in} - \rho_{t,c,out}) \qquad \forall c$$

Appendix C

Variable bounds

In this appendix, the definition of the upper- and lower bounds on the network variables is stated. In Section 5 the use of bounds on variables is introduced as important component of linear solving methods. In addition, these bounds are incorporated as redundant constraints in the MINLP problem.

Table C-1: Variable lower bounds

Variable	Condition	Lower bound
ϕ_w	$\|WI\| > 1$	$\underline{\phi}_w$
ϕ_w	$\|WI\| = 1$	$\max\left\{\min_p(\underline{\phi}_p), \min_d(\underline{\phi}_d), \underline{\phi}_w\right\}$
$\phi_{p,in}$		$\underline{\phi}_p$
$\phi_{p,out}$		$\underline{\phi}_p + \phi_{p,\Delta}$
$\phi_{t,in}$		$\underline{\phi}_t$
$\phi_{t,out}$		$\underline{\phi}_t$
ϕ_d	$\|WO\| > 1$	0
ϕ_d	$\|WO\| = 1$	$\sum_w \phi_w^L + \sum_p \phi_{p,\Delta}$
$\phi_{s,m}$		0
$\rho_{p,c,in}$		0
$\rho_{p,c,out}$		$\frac{1000 \cdot \omega_{p,c}}{\overline{\phi}_p + \phi_{p,\Delta}}$
$\rho_{t,c,in}$		0
$\rho_{t,c,out}$		0
$\rho_{s,m,c}$		0
$\rho_{d,c}$		$\underline{\rho}_{d,c}$

Table C-2: Variable upper bounds

Variable	Condition	Upper bound
ϕ_w	$\|PU\| > 0$	$\sum_p \overline{\phi_p}$
ϕ_w	$\|PU\| = 0$	$\min\left\{\overline{\phi_w}, \max(\sum_t \overline{\phi}_t, \sum_d \overline{\phi_d})\right\}$
$\phi_{p,in}$		$\overline{\phi}_p$
$\phi_{p,out}$		$\overline{\phi}_p + \phi_{p,\Delta}$
$\phi_{t,in}$	$\theta_t = 1$	$\min\left\{\overline{\phi}_t, 2\max\left(\sum_p \phi^U_{p,out}, \sum_w \phi^U_w\right)\right\}$
$\phi_{t,in}$	$\theta_t = 0$	$\min\left\{\overline{\phi}_t, \max\left(\sum_p \phi^U_{p,out}, \sum_w \phi^U_w\right)\right\}$
$\phi_{t,out}$		$\phi^U_{t,in}$
ϕ_d		$\min\left\{\sum_w \phi^U_w + \sum_p \phi_{p,\Delta}, \overline{\phi_d}\right\}$
$\phi_{s_{u1},m_{u2}}$		$\min\left\{\overline{\phi}_{s,m}, \phi^U_{u1,out}, \phi^U_{u2,in}\right\}$
$\rho_{p,c,in}$		$\overline{\rho}_{p,c,in}$
$\rho_{p,c,out}$		$\frac{\underline{\phi}_p \overline{\rho}_{p,c,in} + 1000\cdot\omega_{p,c}}{\underline{\phi}_p + \phi_{p,\Delta}}$
$\rho_{t,c,in}$		$\min\left\{\overline{\rho}_{t,c,in}, \max\left(\max_p(\rho^U_{p,c,out}), \max_w(\rho_w)\right)\right\}$
$\rho_{t,c,out}$		$(1-\lambda_{t,c})\rho^U_{t,c,in}$
$\rho_{d,c}$		$\overline{\rho}_{d,c}$
$\rho_{s_u,m,c}$	$u \in PU$ or $u \in TU$	$\rho^U_{u,out}$
$\rho_{s_u,m,c}$	$u \in WI$	$\rho_{w,c}$

Appendix D

Convex hull formulation

In this Appendix, the MIP implementation of the incremental cost based convex hull formulation, **nf4r** from [Gounaris et al., 2009] is demonstrated. Formulation **nf4r** is based on the use of the binary variable λ_{n_p} for each
$n_p \in \{1, 2, \ldots, N_p\}$ such that

$$\lambda_{n_p} = \begin{cases} 1 & \text{if} \quad x \in [x_{n_p}, x_{n_p+1}] \\ 0 & \text{elsewhere.} \end{cases} \tag{D-1}$$

In addition a continuous variable Δy_{n_p} is introduced:

$$0 \leq \Delta y_{n_p} \leq y^U - y^L,$$

which yields values

$$\Delta y_{n_p} = \begin{cases} y - y^L & \text{if} \quad \lambda_{n_p} = 1 \\ 0 & \text{elsewhere.} \end{cases} \tag{D-2}$$

The MIP formulation is defined as follows. The first constraint fixes only one interval $[x_{n_p}, x_{n_p+1}]$ to be active:

$$\sum_{n_p=1}^{N_p} \lambda_{n_p} = 1. \tag{D-3}$$

This interval contains x:

$$\sum_{n_p=1}^{N_p} x_{n_p} \lambda_{n_p} \leq x \leq \sum_{n_p=1}^{N_p} x_{n_p+1} \lambda_{n_p}. \tag{D-4}$$

For this active interval, Δy_{n_p} should obtain value $y - y^L$, and zero otherwise. This is enforced by the constraints:

$$0 \leq \Delta y_{n_p} \leq (y^U - y^L)\lambda_{n_p}, \tag{D-5}$$

and

$$y = y^L + \sum_{n_p=1}^{N_p} \Delta y_{n_p}. \tag{D-6}$$

The introduction of the variable Δy_{n_p}, referred to as a *continuous switch* in [Misener et al., 2011], results in a reduced relaxation size compared to other formulations, such as the one formulated in Section 5-1 and used in [Karuppiah and Grossmann, 2006]. The piecewise linear convex envelopes are defined by two underestimators and two overestimators, see (5-4). In case of multiple intervals, they are written as:

$$\begin{aligned} z \geq\ & xy^L + \sum_{n_p=1}^{N_p} x_{n_p} \Delta y_{n_p}, \\ z \geq\ & xy^U + \sum_{n_p=1}^{N_p} x_{n_p+1} \Delta y_{n_p} \\ & - (y^U - y^L) \sum_{n_p=1}^{N_p} x_{n_p+1} \lambda_{n_p}. \end{aligned} \tag{D-7}$$

The two overestimators are defined by:

$$\begin{aligned} z \leq\ & xy^L + \sum_{n_p=1}^{N_p} x_{n_p+1} \Delta y_{n_p}, \\ z \leq\ & xy^U + \sum_{n_p=1}^{N_p} x_{n_p} \Delta y_{n_p} \\ & - (y^U - y^L) \sum_{n_p=1}^{N_p} x_{n_p} \lambda_{n_p}. \end{aligned} \tag{D-8}$$

Substitution of $\lambda_{n_p} = 1$ and $\Delta y_{n_p} = y - y^L$ in (D-7) indeed yields the recognizable convex underestimators from (5-4) for interval $[x_{n_p}, x_{n_p+1}]$:

$$\begin{aligned} z \geq\ & xy^L + x_{n_p}(y - y^L) \\ & = x_{n_p} y + y^L x - x_{n_p} y^L, \\ z \geq\ & xy^U + x_{n_p+1}(y - y^L) - (y^U - y^L) x_{n_p+1} \\ & = x_{n_p+1} y + y^U x - x_{n_p+1} y^U. \end{aligned}$$

The same substitution would obtain the corresponding convex overestimators.

Equations (D-3) - (D-8) represent formulation **nf4r**.

Appendix E

Instance specifications

In this appendix the instance data as for the experiments is stated. This data is gathered from different sources to obtain a diverse set of ten instaces. A summary is given in Section 1-3, Table 6-2. The remainder of this appendix contains the input data. In case input parameters are absent in the tables below, they are not defined for the particular instance. Appendix E-1 contains data for four instances from [Karuppiah and Grossmann, 2006], used to validate the model, followed by five additional instances in Appendix E-2. The real life case is stated in Appendix E-3. For all instances holds that $P_{fixed} = \emptyset$.

E-1 Base cases

The four instances from [Karuppiah and Grossmann, 2006] are stated in this section. Throughout this thesis, these instances are referred to as instance K1, K2, K3 and K4 respectively. The superstructures for all four instances are illustrated in Figure E-1, E-2, E-3 and E-4. For each of these instances one source and one sink is present with the same requirements, stated in Table E-1 and E-4. Data for the process units and treatment units is given in Table E-2 and Table E-3 respectively. Additional parameters are stated in Table E-5.

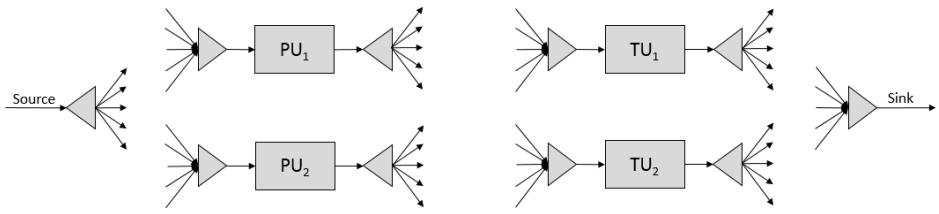


Figure E-1: Superstructure instance K1

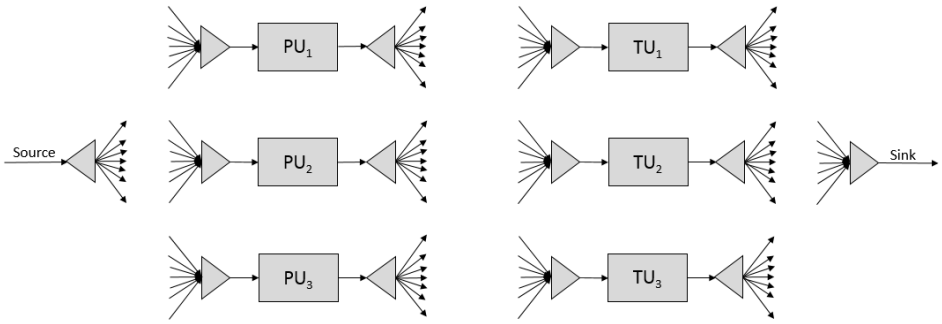


Figure E-2: Superstructure instance K2

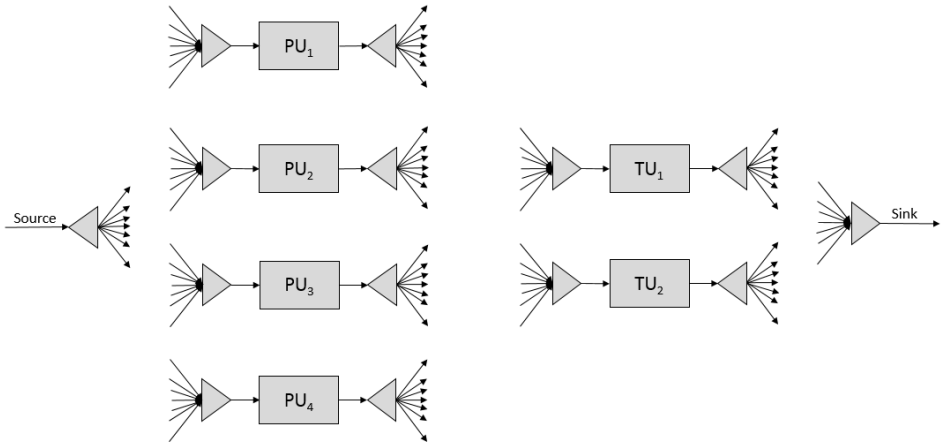


Figure E-3: Superstructure instance K3

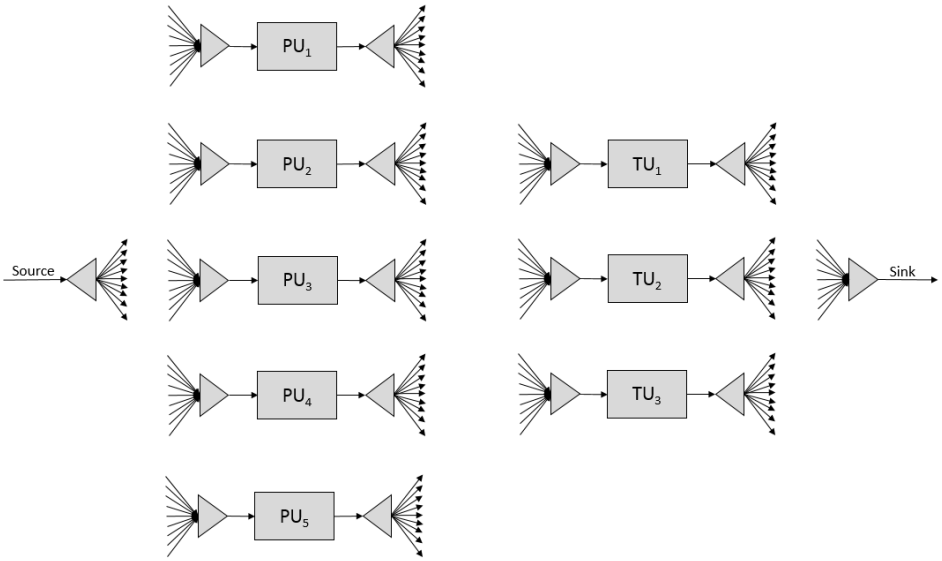


Figure E-4: Superstructure instance K4

Table E-1: Source data base cases

Unit	Flow rate (ton/h)		Cont. conc. (ppm)			Intake costs ($/t$)
	Min	Max	A	B	C	
Instance K1, K2, K3, K4						
Source1	-	-	0	0	0	1

Table E-2: PU data base cases

Unit	Flow rate (ton/h)	Discharge load (kg/h)			Cont. conc. in max (ppm)		
		A	B	C	A	B	C
Instance K1							
PU1	40	1	1.5		0	0	
PU2	50	1	1		50	50	
Instance K2							
PU1	40	1	1.5		0	0	
PU2	50	1	1		50	50	
PU3	60	1	1		50	50	
Instance K3							
PU1	40	1	1.5		0	0	
PU2	50	1	1		50	50	
PU3	60	1	1		50	50	
PU4	70	2	2		50	50	
Instance K4							
PU1	40	1	1.5	1	0	0	0
PU2	50	1	1	1	50	50	50
PU3	60	1	1	1	50	50	50
PU4	70	2	2	2	50	50	50
PU5	80	1	1	0	25	25	25

Table E-3: TU data base cases

Unit	Removal ratio (%)			ICC	OCC
	A	B	C		
Instance K1					
TU1	95	0		16800	1
TU2	0	95		12600	0.0067
Instance K2					
TU1	95	0		16800	1
TU2	80	90		24000	0.033
TU3	0	95		12600	0.0067
Instance K3					
TU1	95	0		16800	1
TU2	0	0		12600	0.0067
Instance K4					
TU1	95	0	0	16800	1
TU2	0	0	95	9500	0.04
TU3	0	95	0	12600	0.0067

Table E-4: Sink data base cases

Unit	Flow rate (ton/h)		Max cont. discharge (ppm)		
	Min	Max	A	B	C
Instance K1, K2, K3, K4					
Sink1	-	-	10	10	10

Table E-5: Other data base cases

Parameter	Description	Value
$\overline{\phi}_{s,m}$	Maximum pipe flow (ton/h)	-
κ_{min}	Minimum number of pipes $(-)$	-
κ_{max}	Maximum number of pipes $(-)$	-
θ_{TU}	Recycling around TU's allowed $(-)$	1
θ_{PU}	Recycling around PU's allowed	1
α	Cost exponent for TU $(-)$	0.7
$\pi_{s,m,fix}$	Fixed pipe cost (\$)	6
$\pi_{s,m,var}$	Variable pipe cost (\$)	100
$\pi_{s,m,op}$	Operating pipe cost $(\$/t)$	0.006
γ	Cost exponent for pipes $(-)$	0.6
H	Total operation time $(h/year)$	8000
AR	Annualized investment factor TU$(-)$	0.1

E-2 Additional cases

The set of four base cases is extended with five more cases: instance A1 and A2 from [Ahmetović, 2011], instance B1 from [Bagajewicz and Savelski, 2001], instance J1 from [Poplewski and Jeżowski, 2007] and instance T1 from [Takama et al., 1980]. Instance A2 was slightly modified: originally the treatment units from this instance removed contamination up to a fixed concentration. To fit the TFM this type of treatment was changed to treatment with a removal ratio.

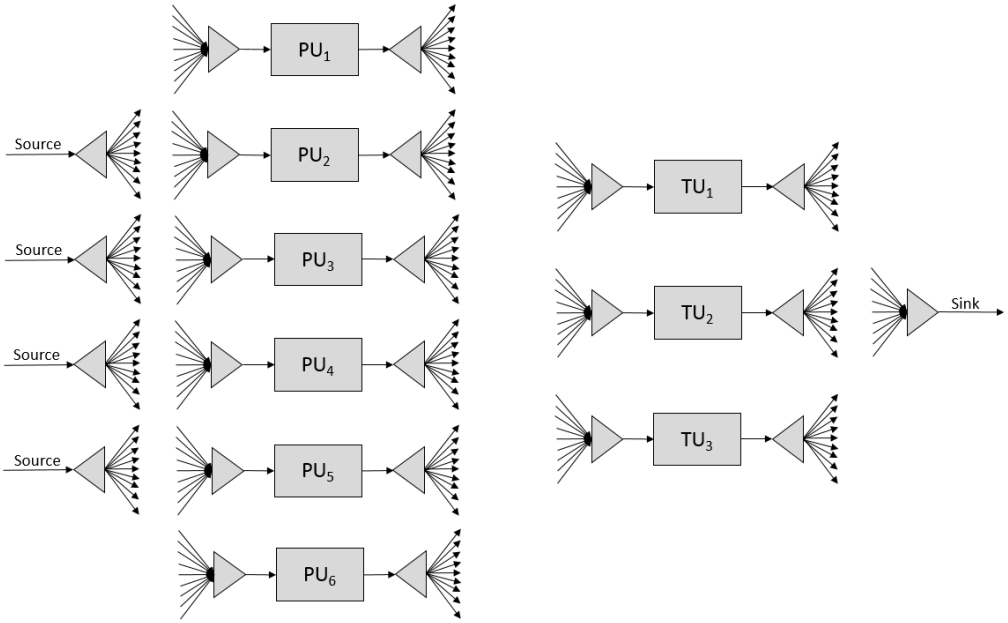


Figure E-5: Superstructure instance A1

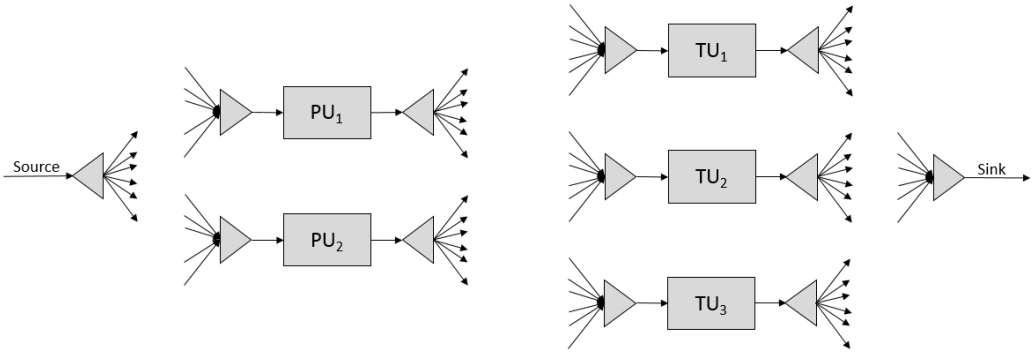


Figure E-6: Superstructure instance A2

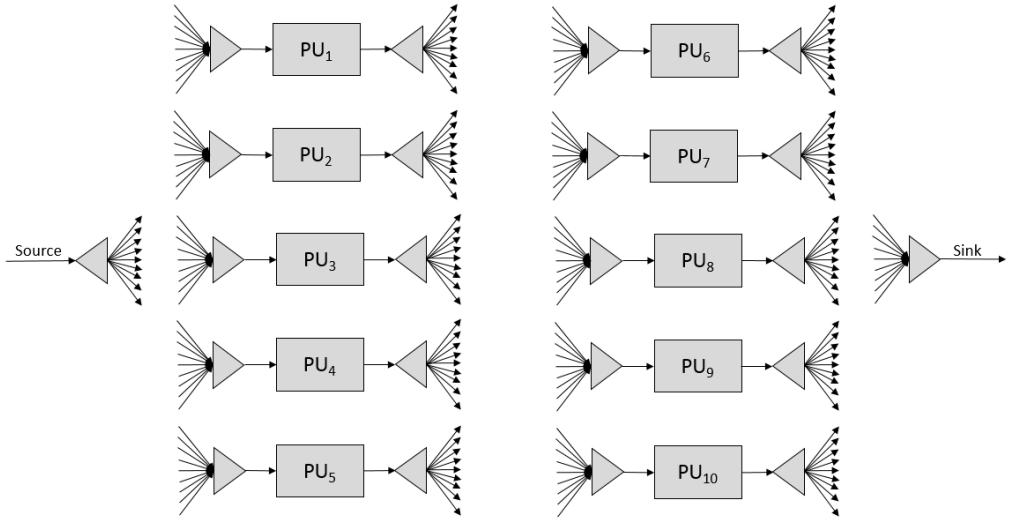


Figure E-7: Superstructure instance B1

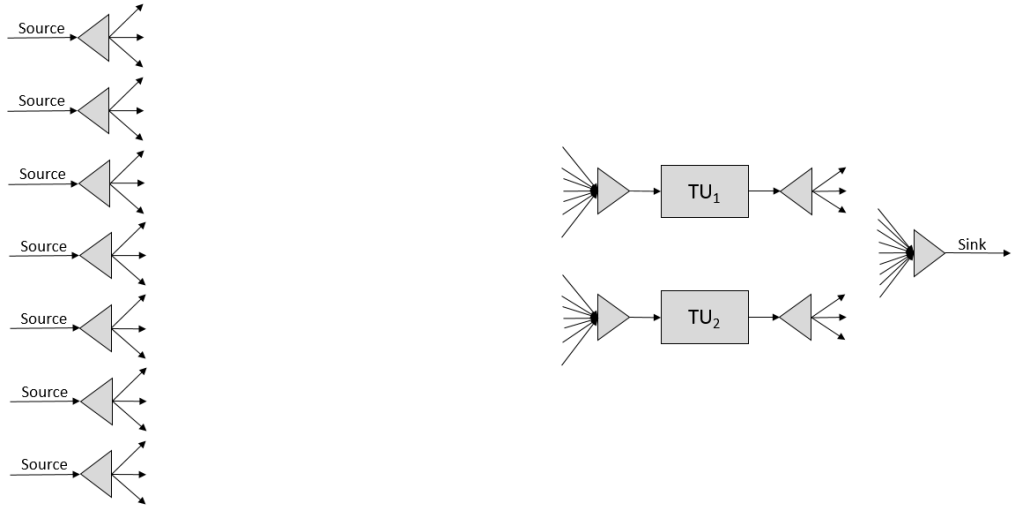


Figure E-8: Superstructure instance J1

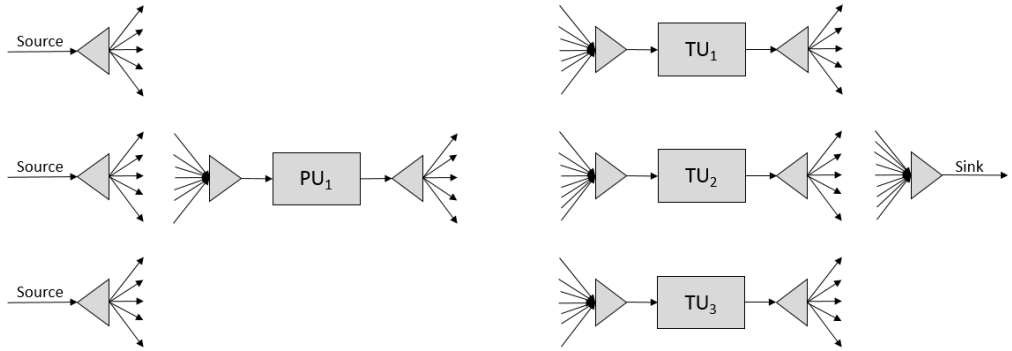


Figure E-9: Superstructure instance T1

Table E-6: Source data additional cases

Unit	Flow rate (ton/h)		Cont. conc. (ppm)					Intake costs ($/t$)
	Min	Max	A	B	C	D	E	
Instance A1								
Source1			0	0	0			1
Source2			25	35	35			0.5
Source3			45	40	40			0.2
Source4			50	50	50			0.15
Instance A2								
Source1			100	100	100	100		0.1
Instance B1								
Source1			0					1
Instance J1								
Source1	18	18	1390	10	250	200	400	0
Source2	25	25	14000	110	400	600	2800	0
Source3	50	50	25	100	1350	2500	3115	0
Source4	60	60	8550	800	45	220	230	0
Source5	36	36	500	300	600	500	500	0
Source6	12	12	50	1500	400	200	100	0
Source7	8	8	2300	12500	200	1000	200	0
Instance T1								
Source1			0	0				0.3

Table E-7: Sink data additional cases

Unit	Max cont. discharge (ppm)				
	A	B	C	D	E
Instance A1					
Sink1	10	10	10		
Instance A2					
Sink1	10	10	10	10	
Instance B1					
Sink1	800				
Instance J1					
Sink1	150	200	140	175	200
Instance T1					
Sink1	2	2	5		

Table E-8: TU data additional cases

Unit	Removal ratio (%)					Cont. conc. in max (ppm)					ICC	OCC
	A	B	C	D	E	A	B	C	D	E		
Instance A1												
TU1	95	0	0								16800	1
TU2	0	0	95								9500	0.04
TU3	0	95	0								12600	0.0067
Instance A2												
TU1	90	90	90	90		100	100	100	100		10000	0.1
TU2	100	100	100	100		10	10	10	10		25000	1.15
TU3	95	95	95	95							30000	1.8
Instance J1												
TU1	99	70	80	60	55						1	1
TU2	90	88	55	85	90						1	1
Instance T1												
TU1	99.9	0	0								16800	1
TU2	0	95	0								4800	0
TU3	90	90	97								12600	0.0067

Table E-9: PU data additional cases

Unit	Flow rate (ton/h)		Water	Discharge				Cont. conc.			
	Min	Max	Added (ton/h)	load (kg/h)				in max (ppm)			
				A	B	C	D	A	B	C	D
Instance A1											
PU1	40	40	0	1	1.5	1		25	25	25	
PU2	50	50	0	1	1	1		50	50	50	
PU3	60	60	0	1	1	1		50	50	50	
PU4	70	70	0	2	2	2		50	50	50	
PU5	80	80	0	1	1	0		25	25	25	
PU6	90	90	0	1	1	0		10	10	10	
Instance A2											
PU1	40	40	0	1	1.5	1	1	0	0	0	0
PU2	50	50	0	1	1	1	1	50	50	50	50
Instance T1											
PU1	45.8	45.8	0	17.9	0.5	1.2		0	0	0	
PU2	32.7	32.7	0	536	3.3	0.5		500	20	50	
PU3	56.5	56.5	0	1.3	5.7	2.0		20	120	50	

Table E-10: PU data case B1

Unit	Flow rate (ton/h)		Water	Discharge	Cont. conc. max (ppm)	
	Min	Max	Added (ton/h)	load (kg/h)	In	Out
				A	A	A
Instance B1						
PU1	25	-	0	2	25	80
PU2	32	-	0	2.9	25	90
PU3	20	-	0	4	25	200
PU4	30	-	0	0	50	100
PU5	37.5	-	0	30	50	800
PU6	6.25	-	0	5	400	800
PU7	3.33	-	0	2	400	600
PU8	10	-	0	1	0	100
PU9	66.67	-	0	20	50	300
PU10	21.67	-	0	6.5	150	300

Table E-11: Other data additional cases

Parameter	Description	Value
Instance A1, A2, B1, J1, T1		
$\overline{\phi}_{s,m}$	Maximum pipe flow (ton/h)	-
κ_{min}	Minimum number of pipes $(-)$	-
κ_{max}	Maximum number of pipes $(-)$	-
θ_{TU}	Recycling around TU's allowed $(-)$	1
θ_{PU}	Recycling around PU's allowed	1
α	Cost exponent for TU $(-)$	0.7
$\pi_{s,m,fix}$	Fixed pipe cost (\$)	6
$\pi_{s,m,var}$	Variable pipe cost (\$)	100
$\pi_{s,m,op}$	Operating pipe cost $(\$/t)$	0.006
γ	Cost exponent for pipes $(-)$	0.6
H	Total operation time $(h/year)$	8000
AR	Annualized investment factor TU $(-)$	0.1

E-3 Real life case

Besides the other cases from literature, one real life case is included in the experiments: instance R1, illustrated in Figure E-10. This treatment network consists of two contaminated input streams, eight treatment units and three possible outlets. Due to confidentiality considerations, no more details on the origins are included. The case was almost one-on-one ready to implement in the TFM. However, for treatment units TU1 to TU4, a treatment with a linear removal rate is used instead of treatment up to a fixed level. In addition, one main aspect of instance R1 is significant costs on the use of each sink. These costs are stated in Table E-12. Since the TFM does not take these costs into consideration, they have to be incorporated otherwise. The costs for each sink are therefore translated to pipe connections right before the corresponding sink. In case there would exist only one pipe connection to every sink, these costs could be levied over the use of these pipes. In order to obtain this network structure, a dummy treatment unit is introduced: TU8*. This treatment unit serves as an additional mixer in the network, collecting the network flows.

Table E-12: Sink cost requirements

Unit	Operational costs $(\$/year)$	Investment costs (\$)
Sink1	0	148590
Sink2	60000000	720090
Sink3	90000000	0

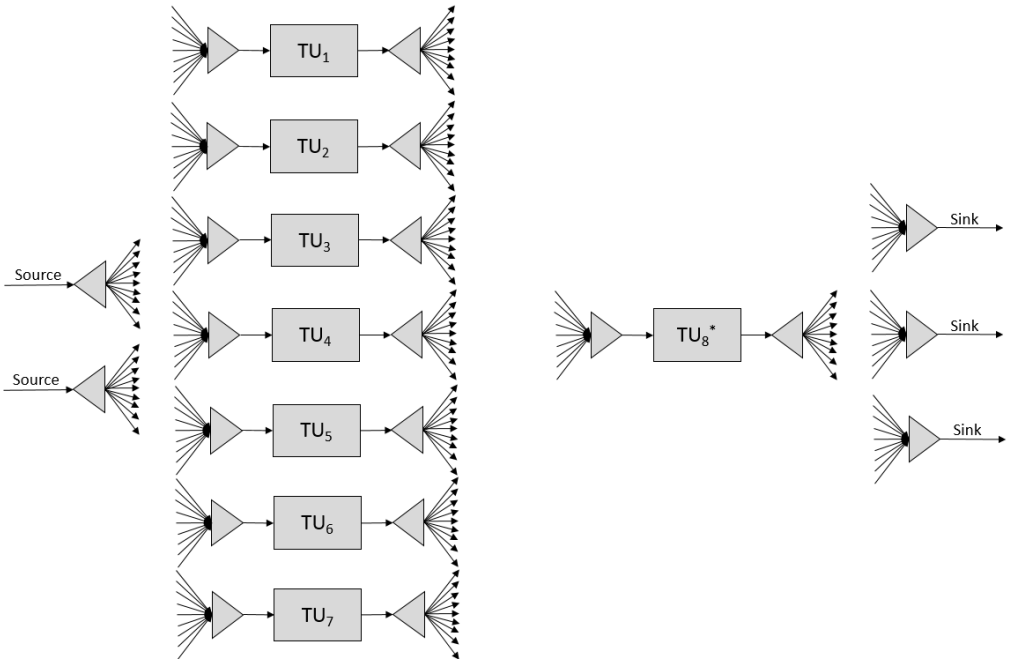


Figure E-10: Superstructure instance R1

Now a single flow can be directed from this treatment unit to any sink. In this way the sink cost components are implied by the costs levied on the pipe connections from TU8* to the sinks. The specifications of the different treatment units and the dummy treatment unit TU8* are stated in Table E-15. The input parameters for the sources and sinks can be found in Table E-13 and E-14. The cost specifications and remaining parameters are stated in Table E-16.

Table E-13: Source data instance R1

Unit	Flow rate (ton/h)		Cont. conc. (ppm)			Intake costs ($/ton$)
	Min	Max	A	B	C	
Source1	80	100	100	1000	7	0
Source2	795	927	100	10000	260	0

Table E-14: Sink data instance R1

Unit	Flow rate (ton/h)		Max cont. discharge (ppm)		
	Min	Max	A	B	C
Sink1	0	5086.8	15	5	-
Sink2	0	5086.8	30	5	-
Sink3	0	5086.8	-	-	-

Table E-15: TU data instance R1

Unit	Removal ratio (%)			Flow rate (ton/h)		Max cont. conc (ppm)						ICC	OCC
				Min	Max	In			Out				
	A	B	C			A	B	C	A	B	C		
TU1	50	99.75	0	11.25	768.5	100	10000	260	50	25	260	2300000	0.07
TU2	50	99.75	0	11.25	768.5	100	10000	260	50	25	260	2300000	0.07
TU3	85	99.75	0	11.25	768.5	100	10000	260	15	25	260	2300000	0.07
TU4	85	99.75	0	11.25	768.5	100	10000	260	15	25	260	2300000	0.07
TU5	95	80	0	11.25	333.3	50	25	260	5	25	260	1670000	0.05
TU6	95	80	0	11.25	333.3	50	25	260	5	25	260	1670000	0.05
TU7	95	80	0	11.25	333.3	50	25	260	5	25	260	1670000	0.05
TU8*	0	0	0	0	-	-	-	-	-	-	-	0	0

Table E-16: Other data instance R1

Parameter	Description	Value
$\overline{\phi}_{s,m}$	Maximum pipe flow (ton/h)	-
κ_{min}	Minimum number of pipes ($-$)	-
κ_{max}	Maximum number of pipes ($-$)	-
θ_{TU}	Recycling around TU's allowed (-)	1
α	Cost exponent for TU ($-$)	0.66
$\pi_{s,m,fix}$ $\forall m \notin MO$	Fixed pipe cost (\$)	0
$\pi_{s,m,fix}$ for m_{sink1}	Fixed pipe cost (\$)	148600
$\pi_{s,m,fix}$ for m_{sink2}	Fixed pipe cost (\$)	60000000
$\pi_{s,m,fix}$ for m_{sink3}	Fixed pipe cost (\$)	90000000
$\pi_{s,m,var}$ $\forall m \notin MO$	Variable pipe cost (\$)	0
$\pi_{s,m,var}$ for m_{sink1}	Variable pipe cost (\$)	148590
$\pi_{s,m,var}$ for m_{sink2}	Variable pipe cost (\$)	720090
$\pi_{s,m,var}$ for m_{sink3}	Variable pipe cost (\$)	0
$\pi_{s,m,op}$	Operating pipe cost (\$/t)	0.006
γ	Cost exponent for pipes ($-$)	0.66
H	Total operation time ($h/year$)	8000
AR	Annualized investment factor TU ($-$)	0.1

References

[gam, 2016a] (2016a). Gams global library. http://www.gamsworld.org/global/globallib.htm.

[gam, 2016b] (2016b). Gams minlp library. http://www.gamsworld.org/minlp/minlplib.htm.

[Ahmetović, 2011] Ahmetović, Eand Grossmann, I. (2011). Global superstructure optimization for the design of integrated process water networks. *AIChE journal*, 57(2):434–457.

[Alfaki and Haugland, 2013] Alfaki, M. and Haugland, D. (2013). Strong formulations for the pooling problem. *Journal of Global Optimization*, 56(3):897–916.

[Bagajewicz and Savelski, 2001] Bagajewicz, M. and Savelski, M. (2001). On the use of linear models for the design of water utilization systems in process plants with a single contaminant. *Chemical engineering research and design*, 79(5):600–610.

[Balas, 1985] Balas, E. (1985). Disjunctive programming and a hierarchy of relaxations for discrete optimization problems. *SIAM Journal on Algebraic Discrete Methods*, 6(3):466–486.

[Belotti et al., 2013] Belotti, P., Kirches, Leyffer, S., Linderoth, J., Luedtke, J., and Mahajan, A. (2013). Mixed-integer nonlinear optimization. *Acta Numerica*, 22:1–131.

[Belotti et al., 2009] Belotti, P., Lee, J., Liberti, L., Margot, F., and Wächter, A. (2009). Branching and bounds tighteningtechniques for non-convex minlp. *Optimization Methods & Software*, 24(4-5):597–634.

[Bisschop and Roelofs, 2006] Bisschop, J. and Roelofs, M. (2006). *Aimms-User's Guide*. Lulu. com.

[Commons, 2007] Commons, W. (2007). Piecewise linear function 2d. https://commons.wikimedia.org/wiki/File:Piecewise_linear_function2D.svg.

[D'Ambrosio et al., 2010] D'Ambrosio, C., Lodi, A., and Martello, S. (2010). Piecewise linear approximation of functions of two variables in milp models. *Operations Research Letters*, 38(1):39–46.

[Dantzig, 1960] Dantzig, G. B. (1960). On the significance of solving linear programming problems with some integer variables. *Econometrica, Journal of the Econometric Society*, pages 30–44.

[Duran and Grossmann, 1986] Duran, M. A. and Grossmann, I. E. (1986). An outer-approximation algorithm for a class of mixed-integer nonlinear programs. *Mathematical programming*, 36(3):307–339.

[Fletcher and Leyffer, 1994] Fletcher, R. and Leyffer, S. (1994). Solving mixed integer nonlinear programs by outer approximation. *Mathematical programming*, 66(1-3):327–349.

[Furman and Androulakis, 2008] Furman, K. C. and Androulakis, I. P. (2008). A novel minlp-based representation of the original complex model for predicting gasoline emissions. *Computers & Chemical Engineering*, 32(12):2857–2876.

[Galan and Grossmann, 1998] Galan, B. and Grossmann, I. (1998). Optimal design of distributed wastewater treatment networks. *Industrial & Engineering Chemistry Research*, 37(10):4036–4048.

[Galan and Grossmann, 1999] Galan, B. and Grossmann, I. (1999). Optimization strategies for the design and synthesis of distributed wastewater treatment networks. *Computers & Chemical Engineering*, 23:S161–S164.

[Geißler et al., 2012] Geißler, B., Martin, A., Morsi, A., and Schewe, L. (2012). Using piecewise linear functions for solving minlps. In *Mixed Integer Nonlinear Programming*, pages 287–314. Springer.

[Gounaris et al., 2009] Gounaris, C. E., Misener, R., and Floudas, C. A. (2009). Computational comparison of piecewise- linear relaxations for pooling problems. *Industrial & Engineering Chemistry Research*, 48(12):5742–5766.

[Grossmann et al., 2002] Grossmann, I. E., Viswanathan, J., Vecchietti, A., Raman, R., Kalvelagen, E., et al. (2002). Gams/dicopt: A discrete continuous optimization package. *GAMS Corporation Inc.*

[Gupte et al., 2013] Gupte, A., Ahmed, S., Cheon, M. S., and Dey, S. (2013). Solving mixed integer bilinear problems using milp formulations. *SIAM Journal on Optimization*, 23(2):721–744.

[Gupte et al., 2016] Gupte, A., Ahmed, S., Dey, S. S., and Cheon, M. S. (2016). Relaxations and discretizations for the pooling problem. *Journal of Global Optimization*, (Online First):1–39.

[Halim et al., 2015] Halim, I., Adhitya, A., and Srinivasan, R. (2015). A novel application of genetic algorithm for synthesizing optimal water reuse network with multiple objectives. *Chemical Engineering Research and Design*.

[Hasan and Karimi, 2010] Hasan, M. and Karimi, I. (2010). Piecewise linear relaxation of bilinear programs using bivariate partitioning. *AIChE journal*, 56(7):1880–1893.

[Haverly, 1978] Haverly, C. A. (1978). Studies of the behavior of recursion for the pooling problem. *ACM SIGMAP Bulletin*, (25):19–28.

[Hooker, 2011] Hooker, J. (2011). *Logic-based methods for optimization: combining optimization and constraint satisfaction*, volume 2. John Wiley & Sons.

[Horst and Tuy, 1996] Horst, R. and Tuy, H. (1996). Global optimization: Deterministic approaches springer. *Heidelberg, 3rd enlarged edition*.

[Hunting, 2011] Hunting, M. (2011). The aimms outer approximation algorithm for minlp. *Paragon Decision Technology, Haarlem*.

[ILOG, 2014] ILOG, I. (2014). Ibm ilog cplex optimization studio cplex userâĂŹs manual. `http://www.ibm.com/support/knowledgecenter/SSSA5P_12.6.1/ilog.odms.studio.help/pdf/usrcplex.pdf`.

[Karuppiah and Grossmann, 2006] Karuppiah, R. and Grossmann, I. (2006). Global optimization for the synthesis of integrated water systems in chemical processes. *Computers & Chemical Engineering*, 30(4):650–673.

[Lastusilta et al., 2007] Lastusilta, T., Bussieck, M. R., and Westerlund, T. (2007). Comparison of some high-performance minlp solvers. *Chemical Engineering Transactions*, 11:125–130.

[Lee and Wilson, 2001] Lee, J. and Wilson, D. (2001). Polyhedral methods for piecewise-linear functions i: the lambda method. *Discrete applied mathematics*, 108(3):269–285.

[Lin and Tsai, 2015] Lin, M. and Tsai, J. (2015). Comparisons of break points selection strategies for piecewise linear approximation. *International Journal of Mechanical Engineering and Robotics Research*, 4(3):247.

[Lundell, 2009] Lundell, A. (2009). *Transformation techniques for signomial functions in global optimization*. Åbo Akademi University.

[McCormick, 1976] McCormick, G. P. (1976). Computability of global solutions to factorable nonconvex programs: Part iâĂŤconvex underestimating problems. *Mathematical programming*, 10(1):147–175.

[Misener and Floudas, 2010] Misener, R. and Floudas, C. (2010). Global optimization of large-scale generalized pooling problems: quadratically constrained minlp models. *Industrial & Engineering Chemistry Research*, 49(11):5424–5438.

[Misener et al., 2011] Misener, R., Thompson, J. P., and Floudas, C. A. (2011). Apogee: Global optimization of standard, generalized, and extended pooling problems via linear and logarithmic partitioning schemes. *Computers & Chemical Engineering*, 35(5):876–892.

[Nowak and Vigerske, 2008] Nowak, I. and Vigerske, S. (2008). Lago: a (heuristic) branch and cut algorithm for nonconvex minlps. *Central European Journal of Operations Research*, 16(2):127–138.

[Pham et al., 2009] Pham, V., Laird, C., and El-Halwagi, M. (2009). Convex hull discretization approach to the global optimization of pooling problems. *Industrial & Engineering Chemistry Research*, 48(4):1973–1979.

[Pinedo, 2012] Pinedo, M. L. (2012). *Scheduling: theory, algorithms, and systems*. Springer Science & Business Media.

[Poplewski and Jeżowski, 2007] Poplewski, G. and Jeżowski, J. (2007). A simultaneous approach for designing optimal wastewater treatment network. *Chem. Eng. Trans*, 12:321–326.

[Quesada and Grossmann, 1992] Quesada, I. and Grossmann, I. E. (1992). An lp/nlp based branch and bound algorithm for convex minlp optimization problems. *Computers & chemical engineering*, 16(10):937–947.

[Quesada and Grossmann, 1995] Quesada, I. and Grossmann, I. E. (1995). Global optimization of bilinear process networks with multicomponent flows. *Computers & Chemical Engineering*, 19(12):1219–1242.

[Ryoo and Sahinidis, 1995] Ryoo, H. S. and Sahinidis, N. V. (1995). Global optimization of nonconvex nlps and minlps with applications in process design. *Computers & Chemical Engineering*, 19(5):551–566.

[Sahinidis, 2016] Sahinidis, N. (2016). Baron user manual v. 16.4.7. `http://www.minlp.com/downloads/docs/baron%20manual.pdf`.

[Sahinidis, 1996] Sahinidis, N. V. (1996). Baron: A general purpose global optimization software package. *Journal of global optimization*, 8(2):201–205.

[Sahinidis, 2002] Sahinidis, N. V. (2002). Global optimization and constraint satisfaction: The branch-and-reduce approach. In *Global Optimization and Constraint Satisfaction*, pages 1–16. Springer.

[Takama et al., 1980] Takama, N., Kuriyama, T., Shiroko, K., and Umeda, T. (1980). Optimal water allocation in a petroleum refinery. *Computers & Chemical Engineering*, 4(4):251–258.

[Tawarmalani and Sahinidis, 2002] Tawarmalani, M. and Sahinidis, N. V. (2002). Convex extensions and envelopes of lower semi-continuous functions. *Mathematical Programming*, 93(2):247–263.

[Tawarmalani and Sahinidis, 2004] Tawarmalani, M. and Sahinidis, N. V. (2004). Global optimization of mixed-integer nonlinear programs: A theoretical and computational study. *Mathematical programming*, 99(3):563–591.

[Tawarmalani and Sahinidis, 2005] Tawarmalani, M. and Sahinidis, N. V. (2005). A polyhedral branch-and-cut approach to global optimization. *Mathematical Programming*, 103(2):225–249.

[Todd, 1977] Todd, M. J. (1977). Union jack triangulations. *Fixed points: algorithms and applications*, pages 315–336.

[Vielma et al., 2010] Vielma, J., Ahmed, S., and Nemhauser, G. (2010). Mixed-integer models for nonseparable piecewise-linear optimization: Unifying framework and extensions. *Operations research*, 58(2):303–315.

[Vielma and Nemhauser, 2008] Vielma, J. P. and Nemhauser, G. L. (2008). Modeling disjunctive constraints with a logarithmic number of binary variables and constraints. In *Integer Programming and Combinatorial Optimization*, pages 199–213. Springer.

[Viswanathan and Grossmann, 1990] Viswanathan, J. and Grossmann, I. E. (1990). A combined penalty function and outer-approximation method for minlp optimization. *Computers & Chemical Engineering*, 14(7):769–782.

[Wang and Smith, 1994] Wang, Y. and Smith, R. (1994). Wastewater minimisation. *Chemical Engineering Science*, 49(7):981 – 1006.

[Westerlund and Pörn, 2002] Westerlund, T. and Pörn, R. (2002). Solving pseudo-convex mixed integer optimization problems by cutting plane techniques. *Optimization and Engineering*, 3(3):253–280.

[Wicaksono and Karimi, 2008] Wicaksono, D. S. and Karimi, I. (2008). Piecewise milp under-and overestimators for global optimization of bilinear programs. *AIChE Journal*, 54(4):991–1008.