Algorithmische Optimierung von Fluidsystemen

Algorithmic Optimization of Fluid Systems

Master-Thesis von Marlene Luka Utz

Juni 2014

TECHNISCHE UNIVERSITÄT DARMSTADT

Algorithmische Optimierung von Fluidsystemen
Algorithmic Optimization of Fluid Systems

Vorgelegte Master-Thesis von Marlene Luka Utz

1. Gutachten: PD Dr. Ulf Lorenz
2. Gutachten: Prof. Dr. Marc E. Pfetsch

Tag der Einreichung:

Erklärung zur Master-Thesis

Hiermit versichere ich, die vorliegende Master-Thesis ohne Hilfe Dritter nur mit den angegebenen Quellen und Hilfsmitteln angefertigt zu haben. Alle Stellen, die aus Quellen entnommen wurden, sind als solche kenntlich gemacht. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Darmstadt, den 30. Juni 2014

(Marlene Luka Utz)

Danksagung

Ich möchte Herrn PD Dr. Ulf Lorenz für die interessante Aufgabenstellung und Herrn Prof. Dr. Marc Pfetsch für die Zweitbegutachtung danken. Durch inhaltliche Diskussionen mit beiden hat meine Arbeit an thematischer Vielfalt gewonnen.

Mein besonderer Dank gilt den Mitarbeitern - Lena Altherr, Thorsten Ederer, Phillip Pöttgen und Thomas Opfer - der TOR Arbeitsgruppe des Fachgebietes Fluidsystemtechnik der Technischen Universität Darmstadt, die mir in Gesprächen stets hilfreiche Anregungen geben konnten.

Zusätzlich möchte ich Herrn Jonas Kobbert für die Hilfe beim Setzten von Graphiken und meinen zahlreichen Korrekturlesern meinen Dank aussprechen.

Auch meinem Freund möchte ich von ganzem Herzen danken, dass er in den schwierigen Zeiten, die von schlaflosen Nächten geprägt waren, immer Ruhe bewahrte und mir mit Rat und Tat zur Seite stand. Nicht zu vergessen sind Familie und Freunde, die mir stets Kraft gaben und die richtigen Worte fanden.

Inhaltsverzeichnis

1 Einleitung **1**

2 Problemstellung **3**
2.1 Das zu planende Wasserversorgungssystem - Beschreibung und Anforderungen 3
2.2 Physikalische Größen und Zusammenhänge 3

3 Mathematisches Modell **9**
3.1 Graphentheoretische Modellierung 9
3.2 Lineares gemischtes ganzzahliges Optimierungsmodell 12
3.2.1 Parameter 13
3.2.2 Variablen 13
3.2.3 Zielfunktion 14
3.2.4 Nebenbedingungen 14
3.2.5 Diskrete Füllstände 24

4 Serien-parallele Netzwerke **27**
4.1 Grundlegende Definitionen 27
4.2 Baumdarstellung 28

5 Dynamische Optimierung **35**
5.1 Mathematische Grundlagen 35
5.1.1 Lösbare Probleme 35
5.1.2 Das Lösungsprinzip 37
5.2 Lösen des Aussteuerungsproblems 40

6 Primale Heuristiken **45**
6.1 Tabu-Suche 45
6.1.1 Mathematische Grundlagen 45
6.1.2 Finden von Kaufentscheidungen 47
6.2 Ameisenalgorithmus 52
6.2.1 Lösungssuche 52
6.2.2 Pheromonupdate 55

7 Duale Schranken **57**
7.1 Relaxierung der Zeitkopplung 57
7.2 Relaxierung der Pumpen-Kennfelder 58

8 Auswertung **61**
8.1 Testinstanzen 61
8.2 Laufzeitverhalten der Dynamischen Optimierung 64
8.3 Verhalten der Tabu-Suche 67

9 Fazit **73**

Anhang **75**

Literaturverzeichnis **82**

1 Einleitung

In Zeiten immer knapper werdender fossiler Brennstoffe tritt der Energieverbrauch technischer Anlagen und Prozesse immer mehr in den Fokus. In ganz Europa werden jährlich etwa 2000-3000 TW h Strom verbraucht, davon circa 30% bei dem Betrieb von Fluidsystemen (vgl.[20]). Aus diesem Grund wird seit einiger Zeit untersucht, wie sich die Methoden der mathematischen Optimierung im Kontext der Planung und des Betreibens solcher Systeme einsetzen lassen.

Dazu wird in dieser Arbeit die Planung von Wasserversorgungssystemen mit Volumenspeichern als beispielhafte Problemstellung betrachtet. Die Anforderungen an das zu planende System und die physikalischen Gesetze, die in diesen Systemen gelten, werden in Kapitel 2 erläutert.

In Kapitel 3 wird gezeigt, wie sich ein Wasserversorgungssystem als Graph darstellen lässt und wie der Zustand des Systems in diesem abgebildet werden kann. Die Systemzustände in den einzelnen Zeitschritten hängen dabei von den vorherigen Füllständen des Speichers ab. Zudem wird ein lineares gemischt-ganzzahliges Programm aufgestellt, das für ein gegebenes Nachfrageprofil einen Systemaufbau und eine Pumpeneinstellung liefert, deren Anschaffungs- und auf lange Sicht dominierenden Stromkosten minimal sind.

Das aufgestellte MIP ist allerdings mit den Standardlösungsverfahren der diskreten Optimierung schon bei wenigen Zeitschritten nicht mehr in akzeptabler Zeit lösbar. Aus diesem Grund werden in dieser Arbeit zwei Verfahren ausgearbeitet, die trotz der hohen Komplexität des Problems in akzeptabler Zeit „gute“ Lösungen finden. Die Idee ist dabei, das Problem in zwei einzelne Entscheidungen zu zerlegen: Das Finden eines Systemaufbaus und das Finden einer Pumpen- und Ventileinstellung.

Dafür wird in Kapitel 5 die Dynamische Optimierung als ein effizientes Lösungsverfahren für das Finden von optimalen Pumpen- und Ventileinstellungen für einen gegebenen Systemaufbau vorgestellt, nachdem in Kapitel 4 die mathematischen Grundlagen zu serien-parallelen Netzwerken gegeben wurden.

In Kapitel 6 werden zwei Heuristiken erläutert, die im Rahmen diese Arbeit implementiert wurden und die „gute“ Systemaufbauten finden. Beide Heuristiken verwenden zur Bewertung ihrer Lösungen die vorgestellte und ebenfalls implementierte Dynamische Optimierung. Die erste Heuristik ist ein Ameisenalgorithmus, der das Verhalten von Ameisen bei der Futtersuche imitiert. Die Lösungen, die von diesem Algorithmus gefunden werden, sind Systemaufbauten, bei denen alle Komponenten seriell oder parallel zueinander verschaltet sind. Solche Systemaufbauten lassen sich als serien-parallele Netzwerke darstellen.

Die zweite, in Kapitel 6 betrachtete Heuristik ist eine lokale Suche, die ausgehend von einer Lösung ihre Nachbarschaft durchsucht und zur besten gefundenen Nachbarschaftslösung wechselt, um dort mit der Suche fortzufahren. Genauer wird in dieser Arbeit eine Tabu-Suche verwendet, die durch Verbote versucht, ein Zurückkehren zu bereits besuchten Lösungen zu verhindern.

Um die Güte von Lösungen abzuschätzen, ohne dass die optimale Lösung bekannt ist, können duale Schranken verwendet werden. Zwei solcher Schranken für das MIP werden in Kapitel 7 erläutert.

Zuletzt fasst Kapitel 8 die Ergebnisse der Untersuchung von Laufzeit und Lösungsgüte der implementierten Verfahren zusammen.

2 Problemstellung

Für eine Siedlung soll ein möglichst günstiges und energiesparendes Wasserversorgungssystem geplant werden. Dabei steht aufgrund von Versorgungsengpässen das Wasser nicht in ausreichender Menge und eventuell nicht mit ausreichendem Druck zur Verfügung. Dies führt zum einen dazu, dass Wasser gespeichert werden muss, damit zu Engpasszeiten ausreichend Wasser zur Deckung des Bedarfs vorhanden ist. Zum anderen muss der Druck des Wassers so erhöht werden, dass das Wasser die Abnehmer erreicht. In diesem Kapitel sollen zunächst die genauen Anforderungen an das zu planende System formuliert werden. Anschließend werden die physikalischen Größen und Zusammenhänge beschrieben, die in Wasserversorgungssystemen zu beobachten sind und bei der Planung beachtet werden müssen.

2.1 Das zu planende Wasserversorgungssystem - Beschreibung und Anforderungen

Das System muss sicherstellen, dass die Abnehmer zu jedem Zeitpunkt genügend Wasser entnehmen können. In Vorbereitung der Planung wird für jeden zu versorgenden Abnehmer ein Lastprofil erarbeitet. Zur Vereinfachung werden nahezu konstante Nachfragen durch ihren Mittelwert ersetzt. Zusammen stellen die Lastprofile die Anforderung an das System dar und geben an, wann, wie lange und wie viel Wasser nachgefragt wird. Der in den Lastprofilen betrachtete Zeitraum wird so in Abschnitte unterteilt, dass in jedem Abschnitt die Nachfrage konstant ist. Für jeden Zeitabschnitt ist eine maximal zur Verfügung stehende Wassermenge vorgegeben. In Abbildung 2.1 sind zwei beispielhafte Lastprofile und die aus diesen folgende Einteilung in Zeitabschnitte zu sehen.

Für das zu entwickelnde System steht eine Menge aus Speichern mit freier Oberfläche zur Verfügung, deren Grundfläche und Höhe vorgegeben ist. Diese dürfen, ebenso wie eine gegebene Menge an Pumpen, im System verbaut werden.

Die ausgewählten Komponenten werden in einem festgelegten Bereich, beispielsweise in einem Pumpenhaus, aufgestellt und passend verbunden. Jeder Abnehmer ist einzeln an das Pumpenhaus angeschlossen und die dazu benötigten Rohre sind vorinstalliert. Es ist möglich, den Speicher sowohl im Pumpenhaus als auch an einem Standort außerhalb aufzustellen. Wird der Speicher nicht im Pumpenhaus aufgestellt, ist zusätzlich ein geodätischer Höhenunterschied zwischen Pumpenhaus und Speicherstandort anzugeben.

Die Pumpen und Speicher sollen so ausgewählt und zu einem System verschaltet werden, dass jederzeit der benötigte Druck erreicht und der Bedarf an Wasser gedeckt werden kann und zudem die Summe aus Anschaffungskosten und Stromkosten über einen gegebenen Zeitraum minimal ist.

2.2 Physikalische Größen und Zusammenhänge

Wasserversorgungssysteme gehören zu der Klasse der Fluidsysteme mit strömenden Flüssigkeiten und folgen somit den Gesetzmäßigkeiten der Hydrostatik und Hydrodynamik. Der Zustand eines solchen Systems lässt schon durch seine Zustandsgrößen beschreiben (vgl. [11]). In dieser Arbeit werden folgende Zustandsgrößen betrachtet:

- Q Volumenstrom in $\mathrm{m}^3\,\mathrm{h}^{-1}$
- v mittlere Strömungsgeschwindigkeit in $\mathrm{m\,s}^{-1}$

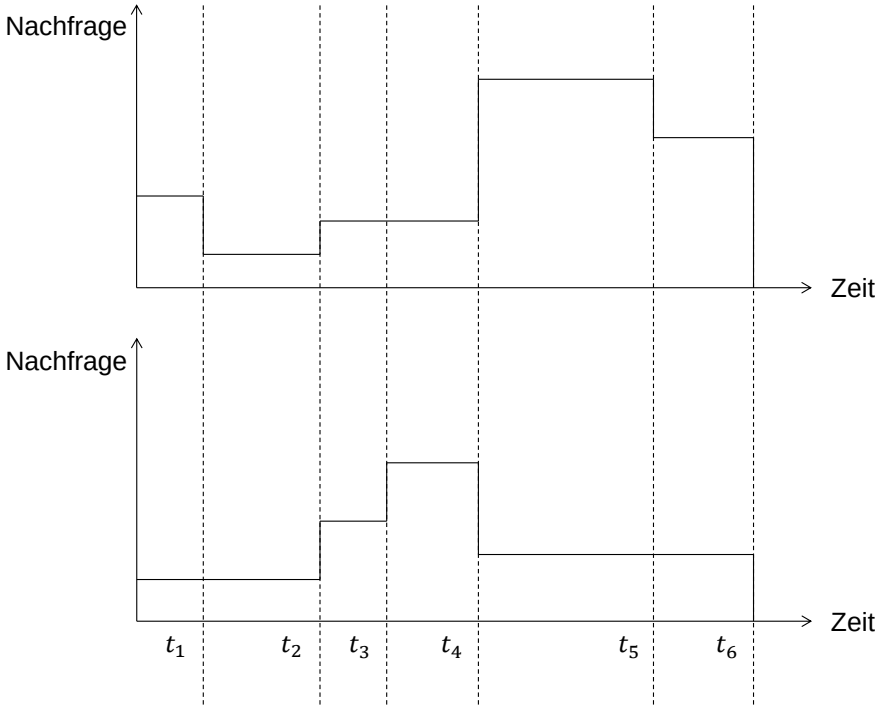

Abbildung 2.1: Beispielhafte Lastprofile, die angeben wann wie lange wie viel Wasser verbraucht wird

- H Druck in mWS (Meter Wassersäule).

Zustandsgrößen lassen sich in *intensive* und *extensive Zustandsgrößen* einteilen. Intensive Zustandsgrößen sind von der Masse des Systems unabhängig, das heißt sie verändern sich durch Teilung des Systems nicht. Die Masse eines Systems entspricht der im System beinhalteten Stoffmenge. Eine intensive Zustandsgröße ist beispielsweise der Druck H, der an den einzelnen Stellen des Systems zu beobachten ist. Im Gegensatz zu intensiven Zustandsgrößen verhalten sich die extensiven Größen proportional zur Masse des Systems. Ein Beispiel hierfür ist - etwa bei inkompressiblen Flüssigkeiten - das Volumen.

Verändern sich diese Zustandsgrößen im Zeitverlauf nicht, so befindet sich das System in einem *stationären Zustand* bzw. in einem *thermodynamischen Gleichgewicht* (vgl. [11]). Ausgehend von einem Gleichgewicht lässt sich die Zustandsänderung eines Systems nur durch die äußere Beeinflussung der Zustandsgrößen herbeiführen. Danach strebt das System wieder einen Gleichgewichtszustand an. Sind Zustandsänderungen eine Abfolge mehrerer Gleichgewichtszustände, so werden sie als *quasistationär* oder *quasistatisch* bezeichnet. Ein Prozess, der zu einer quasistationären Zustandsänderung führt, wird ebenfalls quasistationär genannt [11].

Das Betreiben eines Wasserversorgungssystems kann als ein quasistationärer Prozess betrachtet werden. Die Änderung der Zustandsgrößen erfolgt in diesem Fall durch die Änderung der Wassernachfrage, beispielsweise durch das Öffnen eines Wasserhahns. Das Wasserversorgungssystem erreicht schon nach kurzer Zeit wieder einen stationären Zustand. Der Zeitanteil, in dem sich das System in keinem stationären Zustand befindet, ist vernachlässigbar.

Befindet sich ein Fluidsystem in einem Gleichgewichtszustand, so gilt entlang jeder Stromlinie das *Bernoullische Gesetz [13]*, welches besagt, dass die Gesamtenergie einer strömenden Flüssigkeit entlang einer Stromlinie konstant ist:

$$H_1 + h_1 + \frac{v_1^2}{2g} = H_2 + h_2 + \frac{v_2^2}{2g} = \text{konstant.} \tag{2.1}$$

H	Druckhöhe der Flüssigkeit in mWS
v	Strömungsgeschwindigkeit in $\mathrm{m\,s^{-1}}$
g	Erdbeschleunigung in $\mathrm{m\,s^{-2}}$
h	geodätische Höhe in m

In Abbildung 2.2 ist ein Rohrquerschnitt mit einer eingezeichneten Stromlinie zu sehen.

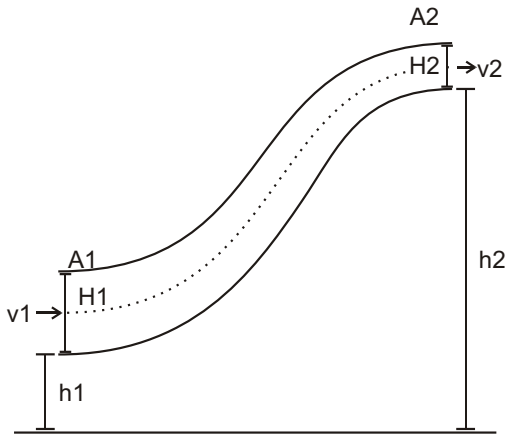

Abbildung 2.2: Querschnitt eines Rohres, mit einer eingezeichneten Stromlinie (angelehnt an [13])

Wird ein System betrachtet, in dem keine wesentlichen geodätischen Höhenunterschiede vorhanden sind, so gilt:

$$H_1 + \frac{v_1^2}{2g} = H_2 + \frac{v_2^2}{2g} = \text{konstant.} \tag{2.2}$$

In diesen beiden Formen des Bernoullischen Gesetzes - (2.1) und (2.2) - wird der Rohrreibungsverlust $\Delta\mathscr{H} > 0$ vernachlässigt. Wird dieser zusätzlich berücksichtigt, so gilt:

$$H_1 + h_1 + \frac{v_1^2}{2g} = H_2 + h_2 + \frac{v_2^2}{2g} + \Delta\mathscr{H} = \text{konstant.} \tag{2.3}$$

Der Rohrreibungsverlust ist abhängig von der Strömungsgeschwindigkeit und der Beschaffenheit des Rohres. Dieser kann durch die *Darcy-Weißbach Gleichung* [9] berechnet werden.

$$\Delta\mathscr{H} = \lambda \cdot \frac{L}{D} \cdot \frac{v^2}{2g} \tag{2.4}$$

$\Delta\mathscr{H}$	Rohrreibungsverlust in mWS
λ	Rohrreibungszahl
L	Länge des Rohres in m
D	Durchmesser des Rohres in m

Unabhängig vom Zustand, in dem sich das System befindet, gilt das *Kontinuitätsgesetz* (vgl. [4]). Dieses Gesetz besagt, dass für alle Teilsysteme der einfließende Volumenstrom dem ausfließenden Volumenstrom entspricht. Bei Systemen mit Volumenspeichern wird der Volumenstrom, der in den Speicher fließt, als aus dem System strömender Volumenstrom betrachtet und der Volumenstrom, der aus dem Speicher fließt, als in das System fließender Volumenstrom. Der Speicher liegt somit außerhalb der Systemgrenze. Der Volumenstrom ist das Produkt aus mittlerer Strömungsgeschwindigkeit und der Querschnittsfläche des Rohres, d.h. die mittlere Strömungsgeschwindigkeit ändert sich mit dem Rohrdurchmesser.

$$Q = 3600 \cdot A \cdot v$$

A	Rohrquerschnittsfläche in m^2
v	mittlere Strömungsgeschwindigkeit in $\mathrm{m\,s}^{-1}$

Die zu betrachtenden Zustandsgrößen bei der Beschreibung einer Pumpe sind der Volumenstrom Q, der durch die Pumpe fließt, die Druckerhöhung ΔH, die von der Pumpe erzeugt wird, und die Leistungsaufnahme P, die durch Betreiben der Pumpe entsteht. Im Allgemeinen gilt für Pumpen, dass mit steigendem Volumenstrom die erzeugte Druckerhöhung sinkt. Bei drehzahl-regelbaren Pumpen gilt zusätzlich, dass die Druckerhöhung bei festem Volumenstrom mit der eingestellten Drehzahl n steigt. Die Druckerhöhung ist somit abhängig vom Volumenstrom und gegebenenfalls von der Drehzahl. Die Leistungsaufnahme einer Pumpe steigt sowohl mit dem geförderten Volumenstrom als auch mit der eingestellten Drehzahl; somit steigt bei steigender Fördermenge auch der Stromverbrauch. Die genauen Zusammenhänge zwischen Volumenstrom, Drehzahl und Druck bzw. Leistungsaufnahme werden in der Pumpenkennlinie (drehzahl-fest) bzw. dem Pumpenkennfeld (drehzahl-regelbar) abgebildet, die von den Pumpenherstellern zur Verfügung gestellt werden. Ein beispielhaftes Kennfeld ist in Abbildung 2.3 zu sehen.

Für drehzahl-regelbare Kreiselpumpen kann berechnet werden, wie sich, ausgehend vom Betriebspunkt der Pumpe, mit Q_1, ΔH_1 und P_1 die Drehzahländerung von n_1 zu n_2 auf den Betriebspunkt auswirkt. Diese Aussage ergibt sich aus den Skalierungsgesetzen für Kreiselpumpen [9]:

$$\frac{Q_1}{Q_2} = \frac{n_1}{n_2} \qquad \frac{\Delta H_1}{\Delta H_2} = \frac{n_1^2}{n_2^2} \qquad \frac{P_1}{P_2} = \frac{n_1^3}{n_2^3} \tag{2.5}$$

Somit stellen die Skalierungsgesetze folgende Zusammenhänge dar:

$$Q \sim n \qquad \Delta H \sim n^2 \qquad P \sim n^3 \tag{2.6}$$

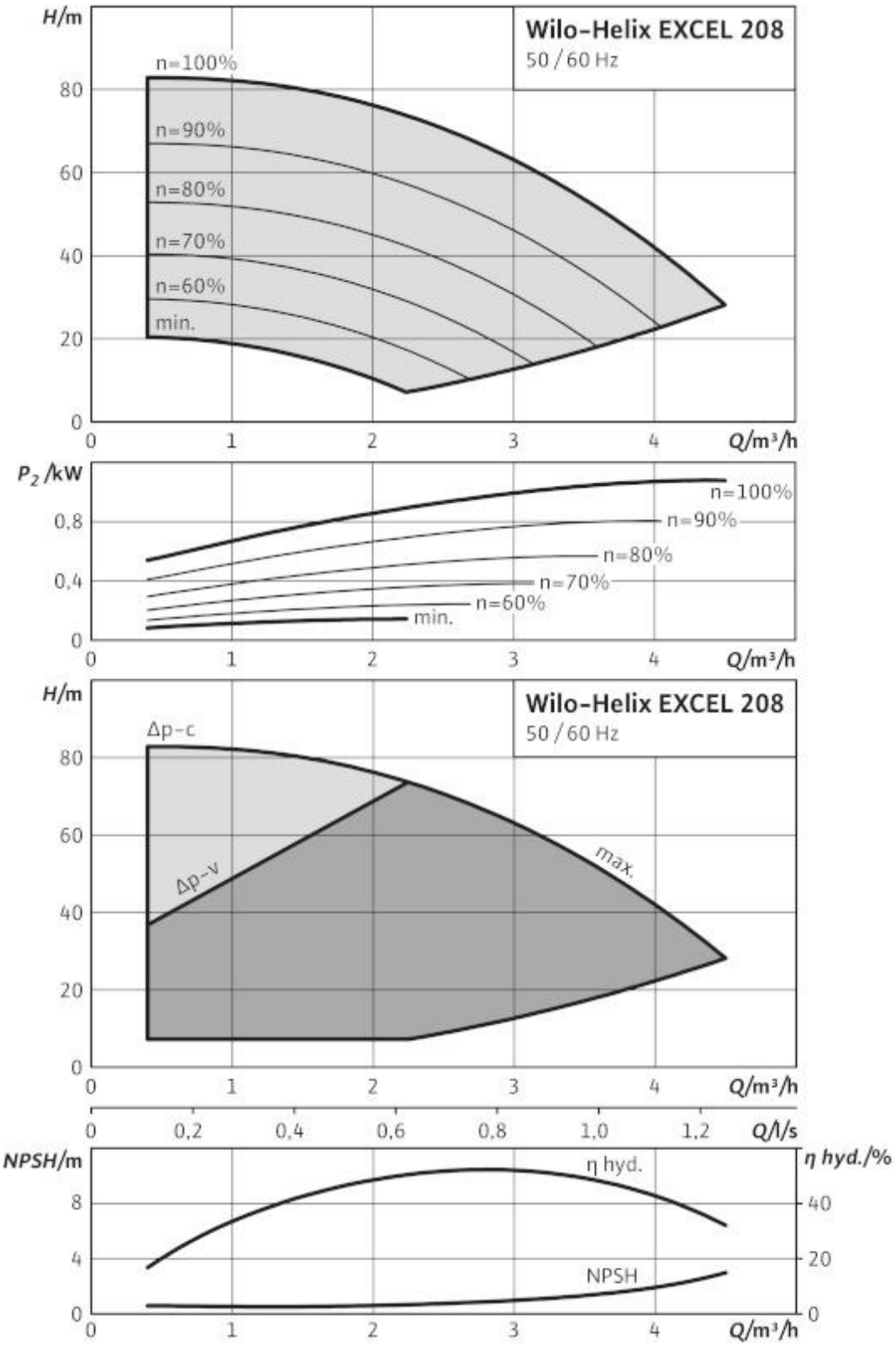

Abbildung 2.3: Beispielhaftes Kennfeld: Die Kennfelder stellen den Zusammenhang zwischen Volumenstrom, Drehzahl und Druckerhöhung bzw. Leistungsaufnahme einer Pumpe dar [22].

3 Mathematisches Modell

In Abschnitt 3.1 wird zunächst erläutert, wie sich der Aufbau und der Zustand eines Wasserversorgungssystems graphentheoretisch modellieren lassen. Zudem wird gezeigt, wie ein solcher Aufbau, ausgehend von einem vollständigen Graphen, gefunden werden kann. Anschließend werden die Annahmen beschrieben, die es ermöglichen, dass das Betreiben eines solchen Systems mit Speichern als eine Folge von stationären Zuständen dargestellt werden kann. Daraus wird abgleitet, wie sich ein Systemaufbau für den Betrieb mit verschiedenen stationären Zuständen bestimmen lässt. Im Abschnitt 3.2 wird auf Grundlage dieser Aussagen ein gemischt ganzzahliges Programm aufgestellt, mit dessen Hilfe ein Aufbau und eine Belegung der Zustandsgrößen für alle Zeitabschnitte und deren jeweiligen stationären Zustand gefunden werden kann. Dabei werden die Kosten, die aus Anschaffungskosten und Stromkosten resultieren, minimiert.

3.1 Graphentheoretische Modellierung

Ein Wasserversorgungssystem lässt sich als gerichteter Graph $G = (V, E)$ darstellen, genauer als Netzwerk mit ausgezeichneter Quelle q und ausgezeichneten Senken $s_1, ..., s_n$, wobei n die Anzahl der zu versorgenden Abnehmer ist. Für jede Pumpe p_i und jeden Speicher sp_j enthält das Netzwerk einen Knoten.

Im Folgenden wird die Interpretation des Graphen als reales System durch Übersetzung von Schaltbildern erklärt. Ein Quellenknoten stellt den Wasserzugang des betrachteten Systems dar und ist somit der Zulauf, der in Schaltplänen, wie in Abbildung 3.1, dargestellt wird. Ein Senkenknoten entspricht dem Anschluss eines Abnehmers an das Pumpenhaus und ist somit ein Ablauf des betrachteten Systems, der in Schaltplänen, wie in Abbildung 3.1, dargestellt wird. Vom Anschluss bis zum Abnehmer selbst führt ein vorinstalliertes Rohr.

Abbildung 3.1: Quellen- und Senkenknoten mit gegenübergestellter Schaltbilddarstellung

In der Realität entspricht ein Pumpenknoten einer Pumpe mit vorgeschaltetem Absperrventil. An das Ventil ist ein eingehendes Rohr angeschlossen und an die Pumpe ein ausgehendes Rohr. In einem Schaltbild des Wasserversorgungssystems wird dies wie in Abbildung 3.2 dargestellt.

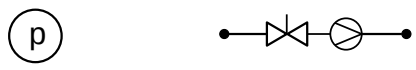

Abbildung 3.2: Pumpe mit vorgeschaltetem Absperrventil und angeschlossenen Rohren

Ein Speicherknoten entspricht einem Speicher mit vor- und nachgeschaltetem Absperrventil. An die Ventile ist jeweils ein Rohr angeschlossen. Die entsprechende Darstellung im Schaltbild ist in Abbildung 3.3 zu sehen.

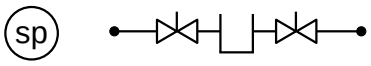

Abbildung 3.3: Speicher mit vor-und nachgeschaltetem Absperrventil und angeschlossenen Rohren

Eine Kante im Graph bedeutet, dass die den Knoten zugeordneten Elemente verbunden sind. Zwei Elemente werden miteinander verbunden, indem ihre Rohre zusammengeschaltet werden. Alle Rohre haben unterstelltermaßen den gleichen Durchmesser. Die Kante gibt gleichzeitig noch die Fließrichtung des Wassers an. Dies ist keine Einschränkung, da die Fließrichtung vorab bekannt ist.

Im Folgenden werden beispielhaft Ausschnitte aus Schaltplänen gezeigt, deren Elemente auf unterschiedliche Weise verbunden sind.

Der Ausschnitt des Schaltplans in Abbildung 3.4(a) zeigt zwei Pumpen und einen Speicher, die an die Quelle angeschlossen sind. Das Rohrsystem, das zwischen der Quelle und den Komponenten liegt, besteht aus einem Rohr, das sich in drei Rohre verzweigt.

Der entsprechende Schaltbildausschnitt für eine Pumpe und einen Speicher, die an eine Senke angeschlossen sind, ist in Abbildung 3.4(b) zu sehen. Hier werden zwei Rohre zu einem Rohr zusammengeführt.

Eine ähnliche Verschaltung von zueinander parallelen Pumpen in Reihe vor einem Speicher ist in Abbildung 3.4(c) zu sehen.

Beispiel 3.1. *In Abbildung 3.5 ist der Schaltplan eines Wasserversorgungssystems, bestehend aus einem Speicher, fünf Pumpen und einer Senke, dargestellt. Hier sind die Pumpen p_1 und p_2 parallel zueinander vor den Speicher geschaltet. Die Pumpen p_3, p_4 und p_5 sind parallel zueinander nach dem Speicher geschaltet und sie sind an die Senke angeschlossen. Dem Schaltbild ist in der Abbildung seine Graphendarstellung gegenübergestellt.*

Im Graph lässt sich nicht nur der Aufbau des Wasserversorgungssystems abbilden, sondern zusätzlich auch der Zustand des Systems. Dazu werden den Knoten und Kanten Zustandsgrößen zugeordnet. Befindet sich das System in einem stationären Zustand, so sind diese Zustandsgrößen konstant und es gelten die physikalischen Gesetze für Systeme im stationären Zustand, wie in Kapitel 2.1 beschrieben.

Jedem Pumpenknoten wird ein Volumenstrom Q_i zugeordnet, der angibt, wie viel Wasser durch die Pumpe strömt. Des Weiteren wird den Pumpenknoten eine Druckerhöhung ΔH_i zugewiesen, die die vom Bauteil erzeugte Druckerhöhung darstellt. Zusätzlich haben diese Knoten noch eine Drehzahl n_i und eine Leistungsaufnahme P_i, die für ein gegebenes Volumenstrom-Druckerhöhungs-Paar angeben, welche Drehzahl eingestellt werden muss und wie hoch der Stromverbrauch dabei ist.

Für die Speicherknoten werden zwei Volumenströme Q_{In} und Q_{Out} angegeben, um die Wassermengen darzustellen, die in den Speicher hinein- bzw. aus dem Speicher herausströmen. Aus dem Anfangsfüllstand und diesen Volumenströmen ergibt sich, wie viel Wasser am Ende des betrachteten Zeitraumes gespeichert ist. Jedem Speicherknoten wird zusätzlich ein eingehender Druck H_{In} und ein ausgehender Druck H_{Out} zugeordnet. Diese beiden Werte geben an, wie hoch der Druck am Anfang bzw. am Ende des Rohres ist, das in bzw. aus dem Speicher führt.

Allen Kanten (i, j) wird ein Volumenstrom Q_{ij} zugeordnet, der angibt, wie viel Wasser von Element i zu Element j fließt. Die Elemente i bzw. j können Pumpen, Puffer, Anschlüsse von Abnehmern oder Quellen sein. Zusätzlich wird den Kanten ein Druck H_{ij} zugeordnet. Dieser entspricht dem Druck an der

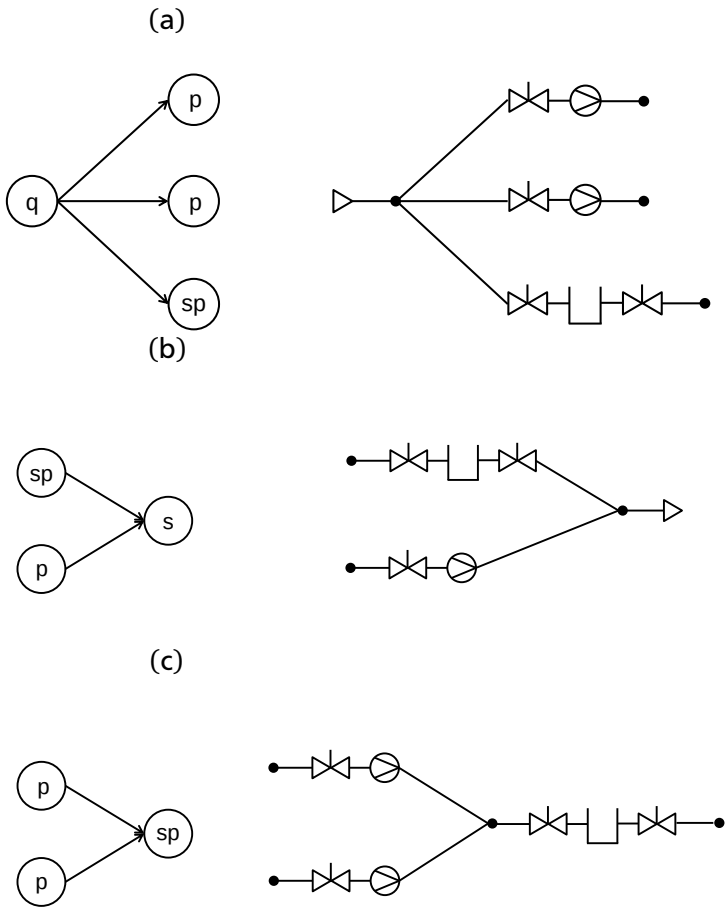

Abbildung 3.4: Beispielhafte Ausschnitte aus Schaltplänen, deren Elemente auf unterschiedliche Weise verbunden sind

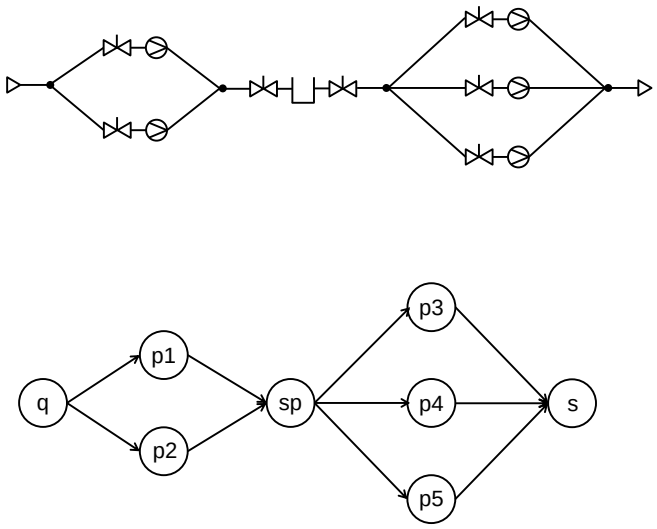

Abbildung 3.5: Schaltplan eines Wasserversorgungssystems mit fünf Pumpen, einem Speicher und einer Senke sowie zugehöriger Graphendarstellung

Stelle, an denen die Rohre zu den Elementen der Knoten i und j verbunden sind. Die Rohre, die die

Elemente innerhalb des Pumpenhauses verbinden, sind im Vergleich zu den Rohren, die zu den Senken oder gegebenfalls zum außerhalb liegenden Speicher führen, sehr kurz. Aus diesem Grund wird der Reibungsverlust in den Rohren innerhalb des Pumpenhauses nicht beachtet.

Den Senken- und Quellknoten wird ebenfalls ein Volumenstrom Q_s bzw. Q_q zugewiesen, der angibt, wie viel Wasser durch sie hindurch in das System fließt bzw. es verlässt. Dem Quellknoten wird ein Druck H_q zugeordnet, der die Höhe des Wasserdrucks beim Eintreten in das System angibt. Auch den Senkenknoten wird ein Druck zugeordnet. Hierbei handelt es sich um den Druck des Wassers an den Anschlüssen der Abnehmer.

Bei der Planung eines Wasserversorgungssystems sollen Pumpen und Speicher so ausgewählt und verschaltet werden, dass das System allen Anforderungen genügt. Die gestellten Anforderungen können typischerweise durch subjektive Einschätzungen variieren. Die minimale Anforderung ist allerdings, dass den Abnehmern immer genügend Wasser entnehmen können. Wird bei den Anforderungen nun davon ausgegangen, dass eine konstante Nachfrage durch das System gedeckt werden soll, so entspricht das Finden eines Aufbaus nun dem Finden eines Graphen, dessen Zustandsgrößen im stationären Zustand die Nachfrage decken. Hierfür sei $\hat{G} = (V, E)$ ein Graph, dessen Knotenmenge aus einem Knoten für die Quelle, einem Knoten für jeden Abnehmer und einem Knoten für jede auswählbare Pumpe bzw. jeden auswählbaren Speicher besteht. Die Kantenmenge beinhaltet alle Kanten zwischen zwei unterschiedlichen Knoten mit Ausnahme der Kanten, die aus einer Senke heraus- oder in die Quelle hineinführen. Das Auswählen eines Teilgraphen von $\hat{G}$ entspricht in diesem Fall dem Auswählen und Verschalten der Komponenten zu einem Systemaufbau.

Das Betreiben eines Wasserversorgungssystems ohne Speicher kann als ein quasistatischer Prozess betrachtet werden. Da die Übergänge zwischen den einzelnen stationären Zuständen sehr kurz sind, können die Zustandsänderungen als eine Folge von stationären Zuständen betrachtet werden, die voneinander unabhängig sind. Enthält das System Speicher und wird deren Speicherdruck als konstant angenommen, so kann auch das Betreiben von solchen Wasserversorgungssystemen als Folge von stationären Zuständen betrachtet werden. Die Zustände sind dann über die Füllstände der Speicher gekoppelt. Somit kann in der mathematischen Modellierung des Problems eine Folge anhängigen stationärer Zustände betrachtet werden.

3.2 Lineares gemischtes ganzzahliges Optimierungsmodell

In diesem Abschnitt wird ein lineares gemischt ganzzahliges Optimierungsmodell (MIP) vorgestellt, das die in Kapitel 2 aufgestellten Bedingungen berücksichtigt und die geforderte Zielsetzung verfolgt. Mit Hilfe des MIP lässt sich zum einen eine Verschaltung der Komponenten finden, d.h. ein Teilgraph aus $\hat{G}$ auswählen und zum anderen eine Belegung der entsprechenden Zustandsgrößen in diesem Teilgraph finden. Hierzu sollen zuerst die gegebenen Parameter beschrieben und die Variablen des Optimierungsmodells vorgestellt werden. Anschließend werden die benötigten Nebenbedingungen und die Zielfunktion erläutert.

Die vereinfachenden Annahmen dabei sind:

- Auf den Rohren des Systems entsteht kein Rohrreibungsverlust. Ausgeschlossen davon sind die Rohre, die zu bzw. aus den Speichern führen und die Rohre, die vom Pumpenhaus zu den Abnehmern führen.
- Der Speicherdruck ist über einen betrachteten Zeitschritt konstant und entspricht dem mittleren Füllstand.
- Nichtlineare Zusammenhänge werden linearisiert.

3.2.1 Parameter

Als Grundlage für die Optimierung seien folgende Parameter gegeben:

Sp	Menge an verbaubaren Speichern
B	Menge an verbaubaren Pumpen
q	anzuschließende Quelle
S	Menge aller zu versorgenden Abnehmer
$\bar{Q}^t_{min,s}$	minimaler Volumenstrom, der beim Abnehmer im Zeitschritt t ankommen muss
$\bar{Q}^t_{max,q}$	maximaler Volumenstrom, der im Zeitschritt t aus der Quelle entnommen werden kann
$K = B \cup Sp$	Menge aller verbaubaren Komponenten
E	Menge aller (potentiellen) Kanten im Graph
$\bar{H}^t_{min,s}$	minimaler Druck, mit dem das Wasser beim Abnehmer im Zeitschritt t ankommen muss
$H_q \geq 0$	Startdruck am Beginn des System
$st^0_i = 0$	Füllhöhe des Speicher i am Anfang des Planungszeitraumes
G_j	Grundfläche des Speichers j
T	Anzahl der betrachteten Zeitschritte
Δ_t	Dauer eines Zeitschrittes

3.2.2 Variablen

Die folgenden Variablen werden benötigt, um Aufbau und Zustand des optimalen Systems zu beschreiben:

$x_i \in \{0, 1\}$	gibt an, ob der Knoten zur Komponente $i \in K$ ausgewählt („gekauft“) wird.
$x_{ij} \in \{0, 1\}$	gibt an, ob die Kante zwischen den Knoten i und j ausgewählt („gekauft“) wird.
$y^t_i \in \{0, 1\}$	gibt an, ob der Pumpenknoten i im Zeitschritt t einen positiven Volumenstrom hat bzw. ob der Pumpenknoten „verwendet“ wird. Wird eine Pumpenknoten nicht verwendet, so ist das vorgeschaltete Absperrventil der zugehörigen Pumpe geschlossen.
$y^t_{In,i} \in \{0, 1\}$	gibt an, ob der Speicherknoten i im Zeitschritt t einen positiven eingehenden Volumenstrom hat. Ist der eingehende Volumenstrom des Knotens null, so wird der Speicher nicht befüllt und das vorgeschaltete Absperrventil ist geschlossen.
$y^t_{Out,i} \in \{0, 1\}$	gibt an, ob der Speicherknoten i im Zeitschritt t einen positiven ausgehenden Volumenstrom hat. Ist der ausgehende Volumenstrom des Konten null, so wird der Speicher nicht geleert und das nachgeschaltete Absperrventil ist geschlossen.
$Q^t_i \geq 0$	gibt den Volumenstrom des Pumpenknotens i während des Zeitschritts t an.
$Q^t_{In,i} \geq 0$	gibt den eingehenden Volumenstrom des Speicherknotens i während des Zeitschritts t an.
$Q^t_{Out,i} \geq 0$	gibt den ausgehenden Volumenstrom des Speicherknotens i während des Zeitschritts t an.
$Q^t_s \geq Q^t_{min,s}$	gibt den Volumenstrom des Senkenknotens s während des Zeitschritts t an.

$0 \leq Q_q^t \leq Q_{max,q}^t$	gibt den Volumenstrom des Quellknotens q während des Zeitschritts t an.
$Q_{ij}^t \geq 0$	gibt den Volumenstrom auf der Kante zwischen den Knoten i und j während des Zeitschritts t an.
$H_{ij}^t \geq 0$	gibt den Druck der Kante zwischen den Knoten i und j während des Zeitschritts t an.
$\Delta H_i^t \geq 0$	gibt die Druckerhöhung des Pumpenknotens i während des Zeitschritts t an.
$H_s^t \geq 0$	gibt den Druck des Senkenknotens s während des Zeitschritts t an.
$H_{In,i}^t \geq 0$	gibt den eingehenden Druck des Speichers i während des Zeitschritts t an.
$H_{Out,i}^t \geq 0$	gibt den ausgehenden Druck des Speichers i während des Zeitschritts t an.
$n_i^t \geq 0$	gibt die Drehzahl des Pumpenknotens i während des Zeitschritts t an.
$P_i^t \geq 0$	gibt die Leistungsaufnahme des Pumpenknotens i während des Zeitschritts t an.
$st_i^t \geq 0$	gibt die Füllhöhe zum Speicherknoten i am Ende von Zeitschritt t an.

3.2.3 Zielfunktion

Die Kosten für die Anschaffung und den Strom des resultierenden Systems sollen so niedrig wie möglich sein. Sei für die Berechnung der Kosten

- $\mathcal{K}_i$ Anschaffungskosten für die Pumpe i mit zugehörigem Absperrventil bzw. für den Speicher mit zugehörigen Absperrventilen
- κ Preis für eine Kilowattstunde
- J Anzahl an Wiederholungen des betrachteten Lastverlaufes

gegeben. Daraus resultiert die zu minimierende Zielfunktion:

$$\sum_{i \in K} \mathcal{K}_i \cdot x_i + J \cdot \kappa \sum_{t=1}^{T} \sum_{i \in B} \cdot \Delta_t \cdot P_i^t.$$

3.2.4 Nebenbedingungen

Die hier beschriebenen Nebenbedingungen stellen sicher, dass alle Anforderungen aus Kapitel 2 erfüllt und die physikalischen Gesetze eingehalten werden.

Kaufentscheidungen

Die Kaufentscheidungen für das zu planende System sollen so getroffen werden, dass die ein- und ausgehenden Rohre einer Komponente an andere Rohre angeschlossen sind und die Flussrichtung eindeutig ist:

$$\sum_{(i,\,j)\in E} x_{ij} \geq x_i \quad \forall i \in K \tag{3.1}$$

$$\sum_{(j,\,i)\in E} x_{ji} \geq x_i \quad \forall i \in K \tag{3.2}$$

$$x_{ij} \leq x_i \quad \forall i \in K,\ (i,\ j) \in E \tag{3.3}$$

$$x_{ji} \leq x_i \quad \forall i \in K,\ (i,\ j) \in E \tag{3.4}$$

$$x_{ji} + x_{ij} \leq 1 \quad \forall\, (j,\ i) \in E. \tag{3.5}$$

(3.1), (3.2): Knoten dürfen nur dann gekauft werden, wenn mindestens eine herausführende und eine eingehende Kante gekauft wurde.
(3.3), (3.4): Kanten dürfen nur gekauft werden, wenn die Start- und Endknoten gekauft wurden.
(3.5): Bauteile dürfen nur in einer Richtung miteinander verbunden werden.

Flusserhaltung

Die folgenden Nebenbedingungen stellen sicher, dass die Volumenströme im System dem Kontinuitätsgesetz folgen. Für die Speicher gilt, dass sich die Füllmenge am Ende des betrachteten Zeitraumes aus der Anfangsfüllmenge und den ein- und ausfließenden Volumenströmen sowie der Dauer des Zeitschrittes ergibt:

$$\sum_{i\in K} Q^t_{qi} = Q^t_q \quad \forall t = 1...T \tag{3.6}$$

$$\sum_{i\in K} Q^t_{is} = Q^t_s \quad \forall s \in S,\ t = 1...T \tag{3.7}$$

$$\sum_{(i,j)\in E} Q^t_{ij} = Q^t_j \quad \forall i \in B,\ t = 1...T \tag{3.8}$$

$$\sum_{(j,i)\in E} Q^t_{ji} = Q^t_j \quad \forall i \in B,\ t = 1...T \tag{3.9}$$

$$\sum_{(i,j)\in E} Q^t_{ij} = Q^t_{In,i} \quad \forall i \in Sp,\ t = 1...T \tag{3.10}$$

$$\sum_{(j,j)\in E} Q^t_{ji} = Q^t_{Out,i} \quad \forall i \in Sp,\ t = 1...T \tag{3.11}$$

$$\Delta_t \cdot (Q^t_{In,i} - Q^t_{Out,i}) + G_i \mathrm{st}^{t-1}_i = G_i \mathrm{st}^t_i \quad \forall i \in Sp,\ t = 1...T. \tag{3.12}$$

(3.6): Der Volumenstrom des Quellknotens entspricht der Summe der Volumenströme auf den Kanten, die aus dem Knoten führen.
(3.7): Der Volumenstrom der Senkenknoten entspricht der Summe der Volumenströme auf den Kanten, die in den Knotenführen.
(3.8), (3.9): Der Volumenstrom des Pumpenknoten entspricht zum einen der Summe der Volumenströme auf den Kanten, die in den Knoten führen, und zum anderen der Summe der Volumenströme auf den Kanten, die aus dem Knoten führen.
(3.10): Der eingehende Volumenstrom des Speicherknotens entspricht der Summe der Volumenströme auf den Kanten die in den Knoten führen.

(3.11): Der ausgehende Volumenstrom des Speicherknotens entspricht der Summe der Volumenströme auf den Kanten die aus dem Knoten führen.
(3.12): Die Füllmenge eines Speicherknotens am Ende eines Zeitschrittes ergibt sich aus der Endfüllmenge des vorherigen Zeitschrittes, den eingehenden und ausgehenden Volumenströmen, sowie der Dauer des Zeitschrittes.

Zusammenhang zwischen Volumenstrom und Kaufentscheidung

Zwei Bauteile müssen verbunden sein, damit zwischen ihnen Wasser fließen kann, d.h. eine Kante muss „gekauft“ sein, wenn sie einen positiven Volumenstrom haben soll. Auch Pumpen- und Speicherknoten müssen „gekauft“ sein, damit durch sie positive Volumenströme führen können. Dieser Zusammenhang lässt sich mit Hilfe der „Big-M“-Methode [2] als lineare Ungleichung darstellen. Dabei ist M eine genügend große Zahl, welche mindestens dem maximalen Volumenstrom, der auf einer Kante auftreten kann, entsprechen muss.

$$\text{Die Kante } (i,j) \text{ ist nicht „gekauft“, d.h. } x_{ij} = 0 \Rightarrow Q_{ij}^t = 0 \quad \forall t.$$

$$M \cdot x_{ij} \geq Q_{ij}^t \quad \forall\,(i,\ j) \in E,\ t = 1...T. \tag{3.13}$$

Gemeinsam mit den Nebenbedingungen für die Kaufentscheidungen und der Flusserhaltung sorgt diese Nebenbedingung dafür, dass auch für Pumpen- und Speicherknoten gilt, dass sie „gekauft“ sein müssen, wenn ihr Volumenstrom positiv sein soll.

Druckverlauf

In Wasserversorgungssystemen, die sich in einem stationären Zustand befinden, gilt entlang von Stromlinien das Bernoullische Gesetz (vgl. 2.1). Es wird angenommen, dass die Rohre, mit denen die Komponenten verbunden sind, alle den gleichen Durchmesser haben. Aus diesem Grund ist bei gegebenem Volumenstrom die mittlere Strömungsgeschwindigkeit überall gleich. Da im Pumpenhaus keine nennenswerten geodätischen Höhenunterschiede vorhanden sind und der Rohrreibungsverlust vernachlässigt wird, gilt somit für je zwei Stellen in einem zusammenhängenden Rohrsystem, dass die an diesen Stellen zu beobachtenden Drücke H_1 und H_2 identisch sind, d.h.

$$H_1 = H_2.$$

In allen Rohren, die zu einem Anschluss eines Abnehmers führen, herrscht ein identischer Druck H_s^t. Insbesondere herrscht dieser Druck auch an den Stellen, an denen die Rohre der Komponenten mit dem Anschluss verbunden sind. Somit gilt für alle „gekauften“ Kanten, die zum Senkenknoten führen, dass ihnen Drücke H_s^t entsprechen, das heißt also:

$$\text{Die Kante } (j,s) \text{ ist „gekauft“, d.h. } x_{js} = 1 \Rightarrow H_{js}^t = H_s^t.$$

Auch dieser Zusammenhang lässt sich mit Hilfe der „Big-M“-Methode als lineare Ungleichung darstellen:

$$M \cdot x_{js} - H_{js}^t + H_s^t \leq M \quad \forall (j, s) \in E,\ t = 1...T \tag{3.14}$$
$$-M \cdot x_{js} - H_{js}^t + H_s^t \geq -M \quad \forall (j, s) \in E,\ t = 1...T \tag{3.15}$$

In allen Rohren, die aus dem Quellknoten führen, herrscht der Anfangsdruck H_q, das bedeutet also:

Die Kante (q, j) ist „gekauft“, d.h. $x_{qj} = 1 \Rightarrow H_{qj}^t = H_q$

$$M \cdot x_{qj} - H_{qj}^t + H_q \leq M \quad \forall (q,\ j) \in V,\ t = 1...T \tag{3.16}$$
$$-M \cdot x_{qj} - H_{qj}^t + H_q \geq -M \quad \forall (q,\ j) \in V,\ t = 1...T. \tag{3.17}$$

Auf allen „gekauften“ Kanten, die aus dem Speicherknoten i führen, herrscht der gleiche Druck $H_{Out,i}^t$, das heißt also:

Die Kante (i, j) ist „gekauft“, d.h. $x_{ij} = 1 \Rightarrow H_{ij}^t = \mathrm{H}_{Out,i}^t$

$$M \cdot x_{ij} - H_{ij}^t + \mathrm{H}_{Out,i}^t \leq M \quad \forall (i,\ j) \in E,\ t = 1...T \tag{3.18}$$
$$-M \cdot x_{ij} - H_{ij}^t + \mathrm{H}_{Out,i}^t \geq M \quad \forall (i,\ j) \in E,\ t = 1...T. \tag{3.19}$$

Analog herrscht der gleiche Druck $H_{In,i}^t$ auf allen „gekauften“ Kanten, die in den Speicherknoten i führen:

Die Kante (j, i) ist „gekauft“, d.h. $x_{ji} = 1 \Rightarrow H_{ji}^t = \mathrm{H}_{In,i}^t$

$$M \cdot x_{ji} - H_{ji}^t + H_{In,i}^t \leq M \quad \forall (j,\ i) \in E,\ t = 1...T \tag{3.20}$$
$$-M \cdot x_{ji} - H_{ji}^t + H_{In,i}^t \geq \cdot M \quad \forall (j,\ i) \in E,\ t = 1...T. \tag{3.21}$$

Stromlinien werden durch Pumpen unterbrochen, d.h. die Bernoullische Gleichung entlang vom Stromlinien gilt im hier betrachteten Beispiel nur in Rohrsystemen.

Für jede Pumpe gilt der folgende Zusammenhang bezüglich des jeweiligen Drucks der „gekauften“ eingehenden und herausführenden Rohre: Sie unterscheiden sich um die Druckerhöhung des Bauteils, was für die Pumpenknoten bedeutet:

Der Pumpenknoten i wird verwendet und die Kanten (j, i), (i, k) sind „gekauft“, d.h. $y_i^t = 1, x_{ji} = 1$, $x_{ik} = 1 \Rightarrow H_{ij}^t + \Delta H_j^t = H_{jk}^t$.

Mit Hilfe der „Big-M“-Methode lässt sich dieser Zusammenhang wie folgt als lineare Ungleichung darstellen:

$$M \cdot (x_{ji} + x_{ik} + y_i) - \Delta H_i^t - H_{ji}^t + H_{ik}^t \leq \ 3 \cdot M \quad \forall\,(j,\ i)\,,\ (i,\ k) \in E,\ t = 1...T \tag{3.22}$$

$$-M \cdot (x_{ji} + x_{ik} + y_i) - \Delta H_i^t - H_{ji}^t + H_{ik}^t \geq -3 \cdot M \quad \forall\,(j,\ i)\,,\ (i,\ k) \in E,\ t = 1...T. \tag{3.23}$$

Da diese Nebenbedingung für alle Paare an „gekauften“ herein- und herausführenden Kanten gilt, folgt dass auf allen hinein- bzw. herausführenden Kanten identische Drücke herrschen.

Zusammenhang zwischen Volumenstrom und den Absperrventilen

Der Volumenstrom durch eine Pumpe ist dann null, wenn das zugehörige Absperrventil geschlossen ist. Für den Pumpenknoten bedeutet dies:

Der Pumpenknoten i wird nicht verwendet, d.h $y_i^t = 0 \Rightarrow Q_i^t = 0$,

$$Q_i^t - M \cdot y_i^t \leq 0 \quad \forall i \in B,\ t = 1...T. \tag{3.24}$$

(3.25)

Ein Pumpenknoten darf nur dann verwendet werden, wenn er „gekauft“ wurde.

Der Pumpenknoten i ist nicht „gekauft“, d.h. $x_i = 0 \Rightarrow y_i^t = 0$,

$$y_i^t \leq x_i \quad \forall i \in B,\ t = 1...T. \tag{3.26}$$

Ist das Absperrventil vor dem Speicher geschlossen, so fließt kein Wasser in den Speicher.

Der Speicherknoten i wird nicht befüllt, d.h. $y_{In,i}^t = 0 \Rightarrow Q_{In,i}^t = 0$,

$$Q_{In,i}^t - M \cdot y_{In,i}^t \leq 0 \quad \forall i \in Sp,\ t = 1...T. \tag{3.27}$$

Ist das Absperrventil nach dem Speicher geschlossen, so kann kein Wasser aus dem Speicher entnommen werden.

Der Speicherknoten i wird nicht geleert, d.h. $y_{In,i}^t = 0 \Rightarrow Q_{In,i}^t = 0$

$$Q_{Out,i}^t - M \cdot y_{Out,i}^t \leq 0 \quad \forall i \in Sp,\ t = 1...T \tag{3.28}$$

Ist eine Kante „gekauft“, die mit einem Speicher- oder Pumpenknoten inzident ist, hängt der Volumenstrom von der Einstellung der Absperrventile ab. Sind angrenzende Ventile geöffnet, so ist der Volumenstrom echt größer als null. Diese Tatsache wird abgebildet, indem gefordert wird, dass der Volumenstrom in solchen Fällen größer gleich $\epsilon = \frac{1}{M}$ ist.

Die Kante (i,j) ist „gekauft“ und der Speicher i wird geleert, d.h. $x_{ij} = 1, y^t_{In,i} = 1$
$\Rightarrow Q^t_{In,i} \geq \epsilon,$

$$MQ^t_{ij} \geq y^t_{In,i} + (x_{ij} - 1) \quad \forall i \in Sp, (i,j) \in E \; t = 1...T. \tag{3.29}$$

Die Kante (j,i) ist „gekauft“ und der Speicher i wird geleert, d.h. $x_{ji} = 1, y^t_{Out,i} = 1$
$\Rightarrow Q^t_{ji} \geq \epsilon,$

$$MQ^t_{ji} \geq y^t_{Out,i} + (x_{ji} - 1) \quad \forall i \in Sp, (j,i) \in E \; t = 1...T. \tag{3.30}$$

Die Kante (i,j) ist „gekauft“ und die Pumpe i wird verwendet, d.h. $x_{ij} = 1, y^t_{In,i} = 1$
$\Rightarrow Q^t_{ij} \geq \epsilon,$

$$MQ^t_{ij} \geq y^t_{i} + (x_{ij} - 1) \quad \forall i \in B, (i,j) \in E \; t = 1...T. \tag{3.31}$$

Die Kante (j,i) ist „gekauft“ und die Pumpe i wird verwendet, d.h. $x_{ji} = 1, y^t_{Out,i} = 1$
$\Rightarrow Q^t_{ji} \geq \epsilon,$

$$MQ^t_{ji} \geq y^t_{i} + (x_{ji} - 1) \quad \forall i \in B, (j,i) \in E \; t = 1...T. \tag{3.32}$$

Zusammenhang zwischen Volumenstrom, Drehzahl und Druckerhöhung bzw. Leistungsaufnahme der Pumpe

In dieser Arbeit werden alle nichtlinearen Zusammenhänge durch lineare Splines approximiert, so zum Beispiel die nichtlinearen Pumpenkennlinien, die den Zusammenhang zwischen Volumenstrom, Dreh-

zahl und Druckerhöhung bzw. Leistungsaufnahme darstellen. Die aus der Approximation entstehenden stückweise linearen Funktionen können mit verschiedenen Methoden in MIP abgebildet werden (vgl. [21, 10, 16]). Die für diese Arbeit ausgewählte Methode ist die Konvexkombinationsmethode, genauer gesagt wird hier die aggregierte Konvexkombinationsmethode verwendet.

Eine stückweise lineare Funktion f ist durch ihre Stützstellen (x_k, f_k) mit $k \in \mathscr{S}$ definiert. Ist $f : \mathbb{R} \to \mathbb{R}^n$ so unterteilen die Stützstellen den Definitionsbereich in Intervalle. Ein Paar von benachbarten Stützstellen wird Segment bzw. Simplex genannt. Die Funktionswerte auf den Intervallen ergeben sich aus einer Konvexkombination der angrenzenden Stützpunkte.

Die Konvexkombinationsmethode bildet diesen Zusammenhang in einem MIP ab. Hierfür wird zuerst eine Binärvariable eingeführt, die einen Simplex auswählt. Zusätzlich wird für jede Stützstelle eine kontinuierliche Variable benötigt, die den Anteil der Stützstelle an der Konvexkombination darstellt. Außerdem müssen Nachbarschaftsbedingungen formuliert werden, die sicherstellen, dass nur Stützstellen an der Konvexkombination beteiligt sind, die zum gewählten Simplex gehören.

Für den Fall, dass der Definitionsbereich von f nicht ein- sondern zweidimensional ist, wird der Definitionsbereich trianguliert. Somit sind die Simplizes Dreiecke. Auch hier sind Binärvariablen zur Auswahl eines Dreiecks benötigt und kontinuierliche Variablen für die Konvexkombination erforderlich.

Für die Linearisierung der Pumpenkennfelder werden die folgenden Parameter und Variablen benötigt:

Parameter:

Sx	Menge aus Simplizes
$\mathscr{S}$	Menge an Stützstellen $(k,l),\ k,l = 1...m$
$\hat{Q}_{kl}(i)$	Volumenstrom an der Stützstelle (k,l) für Pumpe i
$\Delta\hat{H}_{kl}(i)$	Druckerhöhung an der Stützstelle (k,l) für Pumpe i
$\hat{n}_{kl}(i)$	Drehzahl an der Stützstelle (k,l) für Pumpe i
$\hat{P}_{kl}(i)$	Leistungsaufnahme an der Stützstelle (k,l) für Pumpe i.

Variablen:

$0 \leq \lambda^t_{kl}(i) \leq 1$	Anteil der Stützstelle an der Konvexkombination
$a^t_{sx}(i) \in \{0,1\}$	gibt an, welcher Simplex für die Pumpe i im Zeitschritt t ausgewählt ist.

Um sicherzustellen, dass nur ein Simplex gewählt wird und nur dessen Stützstellen an der Konvexkombination beteiligt sind, werden die folgenden Nebenbedingungen benötigt:

$$\sum_{sx \in Sx} a^t_{sx}(i) = y^t_i \qquad \forall i \in B,\ t = 1...T \tag{3.33}$$

$$\sum_{k,l=1...m} \lambda^t_{kl}(i) = y^t_i \qquad \forall i,\ t = 1...T \tag{3.34}$$

$$\sum_{sx:\,(k,l)\in sx} a^t_{sx}(i) \geq \lambda^t_{kl}(i) \quad \forall i \in B,\ k,l = 1...m,\ t = 1...T. \tag{3.35}$$

Wird der Pumpenknoten i verwendet, so ergeben sich die Werte für deren Volumenstrom, Druckerhöhung und Drehzahl aus der Konvexkombination der Werte an den Stützpunkten des ausgewählten Simplex. Wird er dagegen nicht verwendet, so sind alle Werte der betrachteten Variablen für diesen Knoten null.

$$Q_i^t = \sum_{k,l=1...m} \lambda_{kl}^t(i) \cdot \hat{Q}_{kl}(i) \qquad \forall i \in B,\ t = 1...T \tag{3.36}$$

$$\Delta H_i^t = \sum_{k,l=1...m} \lambda_{kl}^t(i) \cdot \Delta\hat{H}_{kl}(i) \qquad \forall i \in B,\ t = 1...T \tag{3.37}$$

$$n_i^t = \sum_{k,l=1...m} \lambda_{kl}^t(i) \cdot \hat{n}_{kl}(i) \qquad \forall i \in B,\ t = 1...T \tag{3.38}$$

$$P_i^t = \sum_{k,l=1...m} \lambda_{kl}^t(i) \cdot \hat{P}_{kl}(i) \qquad \forall i \in B,\ t = 1...T \tag{3.39}$$

Zusammenhang zwischen Druck und Volumenstrom für die Senken

Im Rohr, das zum Abnehmer führt, geht durch Reibung und Höhenunterschied Druck verloren. Wie hoch der Druck H_1 bzw. H_s am Anfang des Rohres sein muss, damit das Wasser beim Abnehmer mit einem Druck von H_2 ankommt, lässt sich mit Hilfe der Bernoullischen Gleichung (vgl. 2.1(2.3)) berechnen:

$$H_1 + h_1 + \frac{v_1^2}{2g} = H_2 + h_2 + \frac{v_2^2}{2g} + \Delta\mathscr{H}.$$

Da für das Rohr zum Abnehmer immer der gleicher Durchmesser angenommen wird, ist die Strömungsgeschwindigkeit am Anfang und am Ende des Rohrs gleich groß $v_1 = v_2$. Es gilt also:

$$H_1 = H_2 + (h_2 - h_1) + \Delta\mathscr{H}.$$

Der Rohrreibungsverlust $\Delta\mathscr{H}$ lässt sich durch die Darcy-Weißbach Gleichung (vgl. 2.1(2.4)) bestimmen. Dieser nichtlineare Zusammenhang wird hier ebenfalls linearisiert und mit Hilfe der aggregierten Konvexkombinationsformulierung dargestellt. Hierfür werden die folgenden Parameter und Variablen benötigt:
Parameter:

Sx	Menge aus Simplizes
$\mathscr{S}$	Menge an Stützstellen $k,\ k = 1...m$
$\hat{Q}_k(s)$	Volumenstrom an der Stützstelle k für die Senke s
$\hat{H}_k(s)$	Druck an der Stützstelle k für die Senke s
$h_s = h_2 - h_1$	geodätischer Höhenunterschied zwischen Pumpenhaus und Abnehmer.

Variablen:

$y_s^t \in \{0, 1\}$	gibt an, ob der Senkenknoten s im Zeitschritt t einen positiven Volumenstrom hat.
$0 \le \lambda_k^t(s) \le 1$	Anteil der Stützstelle an der Konvexkombination.
$a_{sx}^t(s) \in \{0, 1\}$	gibt an, welcher Simplex für die Senke s im Zeitschritt t ausgewählt ist:

$$\sum_{sx \in Sx} a_{sx}^t(s) = y_s^t \qquad \forall s \in S,\ t = 1...T \tag{3.40}$$

$$\sum_{k=1...m} \lambda_k^t(s) = y_s^t \qquad \forall s \in S,\ t = 1...T \tag{3.41}$$

$$\sum_{sx:\, k \in sx} a_{sx}^t(s) \geq \lambda_k^t(s) \quad \forall s \in S,\ k = 1...m,\ t = 1...T. \tag{3.42}$$

Hat der Senkenknoten einen positiven Volumenstrom, so muss der Druck des Senkenknotens größer gleich der Konvexkombination der Werte an den Stützpunkten des ausgewählten Simplex zuzüglich h_s und $\bar{H}_{min,s}^t$ sein. Der Volumenstrom ergibt sich aus der Konvexkombination der Werte an den Stützstellen. Hat der Senkenknoten dagegen keinen positiven Volumenstrom, ist sein Druck nur durch die Bedingungen (3.14) und (3.15) festgelegt:

$$Q_s^t \leq M \cdot y_s^t \qquad \forall s \in S, t = 1...T \tag{3.43}$$

$$Q_s^t = \sum_{k=1...m} \lambda_k^t(s) \cdot \hat{Q}_k(s) \qquad \forall s \in S,\ t = 1...T \tag{3.44}$$

$$H_s^t \geq \sum_{k=1...m} \lambda_k^t(s) \cdot \hat{H}_k(s) + h_s + \bar{H}_{min,s}^t + M \cdot y_s^t - M \quad \forall s \in S,\ t = 1...T. \tag{3.45}$$

Zusammenhang zwischen Druck und Volumenstrom für die Speicher

Das Wasser wird durch ein Rohr unten in den Speicher eingeleitet. Dazu muss der Wasserdruck am Speichereingang dem Speicherdruck, abgeleitet aus der Füllhöhe des Speichers in Metern, entsprechen. In diesem Modell wird vereinfachend angenommen, dass der Speicherdruck im gesamten Zeitschritt konstant ist und sich aus dem mittleren Speicherstand $\bar{st}$ ergibt:

$$\bar{st}_i^t = \frac{st_i^{t-1} + st_i^t}{2}.$$

Im Rohr, das zum Speicher führt, geht durch Reibung und Höhenunterschied Druck verloren. Wie hoch der Druck H_1 bzw. $H_{In,i}$ am Anfang des Rohres sein muss, damit das Wasser im Speicher mit einem Druck von H_2 ankommt, lässt sich ebenfalls durch

$$H_1 = H_2 + (h_2 - h_1) + \Delta\mathscr{H}$$

berechnen. Dieser nichtlineare Zusammenhang wird mithilfe der aggregierten Konvexkombinationsmethode linearisiert.

Hierfür werden zusätzlich die folgenden Parameter und Variablen benötigt:

Parameter:

Sx	Menge aus Simplizes
$\mathscr{S}$	Menge an Stützstellen $k,\ k = 1...m$

$\hat{Q}_{In,k}(i)$	Volumenstrom an der Stützstelle k für den Speicher i
$\hat{H}_{In,k}(i)$	Druck an der Stützstelle k für den Speicher i
$h_i = h_2 - h_1$	geodätischer Höhenunterschied zwischen Pumpenhaus und Speicher

Variablen:

$\bar{st}_i^t$	der mittlere Füllstand des Speichers i im Zeitschritt t
$0 \leq \lambda_{In,k}^t(i) \leq 1$	Anteil der Stützstelle an der Konvexkombination
$a_{In,sx}^t(i) \in \{0,1\}$	gibt an, welcher Simplex für den Speicher i im Zeitschritt t ausgewählt ist:

$$\sum_{sx \in Sx_{In}} a_{In,sx}^t(i) = y_{In,i}^t \qquad \forall y_{In,i}^t \in Sp,\ t = 1...T \tag{3.46}$$

$$\sum_{k=1...m} \lambda_{In,k}^t(i) = y_{In,i}^t \qquad \forall i \in Sp,\ t = 1...T \tag{3.47}$$

$$\sum_{sx:\, k \in sx} a_{In,sx}^t(i) \geq \lambda_{In,k}^t(i) \qquad \forall i \in Sp,\ k = 1...m,\ t = 1...T. \tag{3.48}$$

Hat der Speicherknoten einen positiven eingehenden Volumenstrom, so ergibt sich dieser aus der Konvexkombination der Werte an den Stützpunkten des ausgewählten Simplex. Der eingehende Druck des Speicherknotens ergibt sich aus der Konvexkombination zuzüglich des geodätischen Höhenunterschieds und des mittleren Füllstandes. Hat der Speicherknoten dagegen keinen positiven Volumenstrom, so ist der eingehende Druck des Knotens nur durch die Bedingungen (3.20) und (3.21) festgelegt:

$$Q_{In,s}^t = \sum_{k=1...m} \lambda_{In,k}^t(i) \cdot \hat{Q}_{In,k}(i) \qquad \forall i,\ t = 1...T \tag{3.49}$$

$$H_{In,i}^t \geq \sum_{k=1...m} \lambda_{In,k}^t(i) \cdot \hat{H}_{In,k}(i) + h_i + \bar{st}_i^t + M \cdot y_{In,i}^t - M \qquad \forall i \in Sp,\ t = 1...T \tag{3.50}$$

$$H_{In,i}^t \leq \sum_{k=1...m} \lambda_{In,k}^t(i) \cdot \hat{H}_{In,k}(i) + h_i + \bar{st}_i^t - M \cdot y_{In,i}^t + M \qquad \forall i \in Sp,\ t = 1...T. \tag{3.51}$$

Das Wasser wird durch ein Rohr zurück in das System geleitet. Der Druck H_2 bzw. $H_{Out,i}$, mit dem das Wasser wieder in das System gelangt, ergibt sich aus dem Speicherdruck abzüglich des Rohrreibungsverlustes:

$$H_2 = H_1 - \Delta\mathscr{H}.$$

Zur Linearisierung dieses Zusammenhangs werden die folgenden Parameter und Variablen benötigt:

Parameter:

Sx_{Out}	Menge aus Simplizes
$\mathscr{S}$	Menge an Stützstellen $k,\ k = 1...m$
$\hat{Q}_{Out,k}(i)$	Volumenstrom an der Stützstelle k für den Speicher i

$\hat{H}_{Out,k}(i)$ Druck an der Stützstelle k für den Speicher i

Variablen:

$0 \leq \lambda^t_{Out,k}(i) \leq 1$ Anteil der Stützstelle an der Konvexkombination
$a^t_{Out,sx}(i) \in \{0,1\}$ gibt an welcher Simplex für den Speicher i im Zeitschritt t ausgewählt ist:

$$\sum_{sx \in Sx_{Out}} a^t_{Out,sx}(i) = y^t_{Out,i} \qquad \forall i \in Sp,\ t = 1...T \tag{3.52}$$

$$\sum_{k=1...m} \lambda^t_{Out,k}(i) = y^t_{Out,i} \qquad \forall i \in Sp,\ t = 1...T \tag{3.53}$$

$$\sum_{sx:k \in sx} a^t_{Out,sx}(i) \geq \lambda^t_{Out,k}(i) \qquad \forall i \in Sp,\ k = 1...m,\ t = 1...T. \tag{3.54}$$

Hat der Speicherknoten einen positiven Volumenstrom, so ergibt sich dieser auch hier aus der Konvexkombination der Werte an den Stützpunkten des ausgewählten Simplex. Der ausgehende Druck des Speicherknotens dagegen ergibt sich aus dem mittleren Füllstand und dem geodätischen Höhenunterschied abzüglich des Rohrreibungsverlustes, der sich aus der Konvexkombination ergibt. Hat der Speicherknoten keinen positiven Volumenstrom, so ist der ausgehende Druck des Speichers nur durch die Bedingungen (3.18) und (3.19) festgelegt:

$$Q^t_{Out,s} = \sum_{k=1...m} \lambda^t_{Out,k}(i) \cdot \hat{Q}_{Out,k}(i) \qquad \forall i \in sx,\ t = 1...T \tag{3.55}$$

$$H^t_{Out,i} \geq h_i + \bar{st}^t_i - \sum_{k=1...m} \lambda Out^t_k(i) \cdot \hat{H}_{Out,k}(i) + M \cdot y^t_{Out,i} - M \qquad \forall i \in sx,\ t = 1...T \tag{3.56}$$

$$H^t_{Out,i} \leq h_i + \bar{st}^t_i - \sum_{k=1...m} \lambda^t_{Out,k}(i) \cdot \hat{H}_{Out,k}(i) - M \cdot y^t_{Out,i} + M \qquad \forall i \in sx,\ t = 1...T. \tag{3.57}$$

3.2.5 Diskrete Füllstände

Im oben beschriebenen Modell sind die Füllstände der Speicher kontinuierliche Entscheidungsvariablen. Sollen dagegen diskrete Füllstände betrachtet werden gelten zusätzliche Nebenbedingungen. Vor der Erläuterung dieser Nebenbedingungen sollen zuerst die in diesem Fall zusätzlich benötigten Parameter und Variablen eingeführt werden:
Parameter:

N_i Anzahl an Füllständen für Speicher i
$P_i = \{p_{i,1}, ..., p_{i,N_i}\}$ Menge an auswählbaren Füllständen für den Speicher i

Variablen:

$s^t_{i,n} \in \{0,1\}$ gibt an, ob der Füllstand n für Speicher i im Zeitschritt t ausgewählt wird.

Für jeden „gekauften“ Speicher i darf nur ein Füllstand ausgewählt werden. Der Füllstand ergibt sich aus der Auswahl n und dem dieser Auswahl zugeordneten Stand $p_{i,n}$:

$$\sum_{n=1...N} s^t_{i,n} \cdot p_{i,n} = st^t_i \quad \forall i \in Sp,\ t = 1...T \tag{3.58}$$

$$\sum_{n=1...N} s^t_{i,n} = x_i \quad \forall i \in Sp,\ t = 1...T. \tag{3.59}$$

4 Serien-parallele Netzwerke

In diesem Kapitel werden serien-paralle Netzwerke betrachtet, die die Grundlage für den später verwendeten Ameisenalgorithmus bilden. Hierfür werden zuerst einige grundlegende Definitionen gegeben. Auf dieser Grundlage wird dann erläutert, wie sich alle serien-parallelen Netzwerke einer bestimmten Größe erzeugen lassen. Aus dieser Vorgehensweise lässt sich anschließend die Anzahl an serien-parallelen Netzwerken einer bestimmten Größe ableiten und daraus ihre Repräsentation in einer Baumdarstellung.

4.1 Grundlegende Definitionen

Definition 4.1. *(minimales Netzwerk)*
*Sei $G_{min} = (V, E)$ ein Netzwerk bestehend aus einem Knoten p, der Quelle q und der Senke s, d.h. $V = \{p, q, s\}$. Die Kanten verbinden die Quelle mit dem Knoten und diesen mit der Senke. Somit ist die Kantenmenge gegeben durch $E = \{(q, p), (p, s)\}$. Ein Netzwerk dieser Art wird **minimales Netzwerk** genannt. In diesem Kontext werden Knoten q und s Kopplungsknoten und der Knoten p Hauptknoten genannt.*

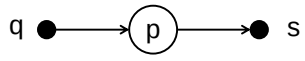

Abbildung 4.1: Minimales Netzwerk, bestehend aus einem Hauptknoten und zwei Kopplungsknoten

Definition 4.2. *(paralleles Verschalten von Netzwerken)[8]*
*Seien $G_1 = (V_1, E_1)$ und $G_2 = (V_2, E_2)$ zwei Netzwerke mit Quellen q_1 bzw. q_2 und Senken s_1 bzw. s_2. Zwei **Netzwerke werden parallel verschaltet**, indem ihre Quellen bzw. Senken miteinander identifiziert werden. Die so entstandenen Knoten sind wieder Kopplungsknoten. Sei $G = (V, E)$ das Netzwerk, das durch das parallele Verschalten der beiden Netzwerke entsteht.*

$$G = G_1 \circ_{par} G_2$$

Dann ist $V = \{(V_1 \setminus \{s_1, q_1\}) \cup (V_2 \setminus \{s_2, q_2\}) \cup \{s, q\}\}$ die Knotenmenge des entstandenen Netzwerkes und $E = \{E_1 \cup E_2\}$ ist die Kantenmenge, wobei die Kanten $(q_1, v_1) \in E_1$ und $(q_2, v_2) \in E_2$ durch (q, v_1) bzw. (q, v_2) und die Kanten $(v_1, s_1) \in E_1$, $(v_2, s_2) \in E_2$ durch (v_1, s) bzw. (v_2, s) ersetzt werden.

*Das so entstandene Netzwerk G wird **paralleles Netzwerk** genannt.*

Beispiel 4.1. *Das parallele Verschalten zweier minimaler Netzwerke ergibt das in Abbildung 4.2 zu sehende Netzwerk.*

Bemerkung 4.1. *Zwei parallele Netzwerke sind gleich, wenn es sich um eine parallele Verschaltung der gleichen Teilnetze handelt. Die Reihenfolge der Verschaltung ist dabei irrelevant.*

Definition 4.3. *(serielles Verschalten von Netzwerken)[8]*
Seien $G_1 = (V_1, E_1)$ und $G_2 = (V_2, E_2)$ zwei Netzwerke mit Quellen q_1 bzw. q_2 und Senken s_1 bzw. s_2. Zwei

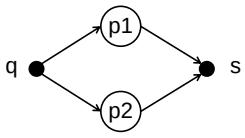

Abbildung 4.2: Netzwerk als Ergebnis der parallelen Verschaltung zweier minimaler Netzwerke

***Netzwerke werden seriell verschaltet**, indem die Senke des einen Netzwerks mit der Quelle des anderen Netzwerks identifiziert wird. Der so entstandene Knoten ist wieder ein Kopplungsknoten. Sei $G = (V, E)$ das Netzwerk, das durch das serielle Verschalten der beiden Netzwerke entsteht.*

$$G = G_1 \circ_{ser} G_2$$

*Dann gilt $V = \{(V_1 \setminus \{s_1\}) \cup (V_2 \setminus \{s_2\}) \cup \{k\}\}$ ist die Knotenmenge des entstandenen Netzwerkes und $E = \{E_1 \cup E_2\}$ die Kantenmenge , wobei alle Kanten $(v_1, s_1) \in E_1$ durch (v_1, k) ersetzt werden und alle die Kanten $(q_2, v_2) \in E_2$ durch (k, v_2) ersetzt werden. Das so entstandene Netzwerk G wird **serielles Netzwerk** genannt.*

Beispiel 4.2. *Das serielle Verschalten zweier minimaler Netzwerke ergibt das in Abbildung 4.3 zu sehende Netzwerk.*

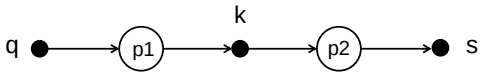

Abbildung 4.3: Netzwerk als Ergebnis der seriellen Verschaltung zweier minimaler Netzwerke

Bemerkung 4.2. *Zwei serielle Netzwerke sind gleich, wenn es sich um eine serielle Verschaltung der gleichen Teilnetze handelt. Die Reihenfolge der Verschaltung ist dabei irrelevant.*

Serien-parallele Netzwerke sind rekursiv über minimale Netzwerke und das serielle bzw. parallele Verschalten definiert.

Definition 4.4. *(serien-paralleles Netzwerk)[8]*

- *Das minimale Netzwerk aus Definition 4.1 ist ein serien-paralleles Netzwerk.*
- *Ein serien-paralleles Netzwerk (kurz sp-Netzwerk) entsteht durch das serielle und parallele Verschalten von serien-parallelen Netzwerken.*

In sp-Netzwerken gibt es zwei Typen von Knoten, zum einen die Kopplungsknoten, welche zum Verschalten der Netzwerke dienen, und zum anderen die Hauptknoten, welche unabhängig von der Verschaltung erhalten bleiben.

Definition 4.5. *(Ordnung von Netzwerken)[15]*
*Ein serien-paralleles Netzwerk mit n Hauptknoten wird **Netzwerk der Ordnung** n genannt.*

4.2 Baumdarstellung

Serien-parallele Netzwerke lassen sich rekursiv durch serielle und parallele Verschaltung aus dem minimalen Netzwerk gemäß Definition 4.1 erstellen. Eine mögliche Vorgehensweise, um alle sp-Netzwerke

einer bestimmten Ordnung zu erhalten, wurde von P. A. McMahon aufgezeigt[1] [15]. Eine dadurch inspirierte, leicht abweichende Vorgehensweise wird in diesem Kapitel vorgestellt. Aus dieser Vorgehensweise lässt sich die Anzahl an sp-Netzwerken der Ordnung n ableiten. Ausgehend von dieser Anzahl wird dann der Zusammenhang zwischen sp-Netzwerken und Bäumen ohne innere Knoten des Grads 2 (engl. series-reduced trees) hergestellt.

Definition 4.6. *(Konjugieren von serien-parallelen Netzwerken)[15]*

- *Das minimale Netzwerk ist selbst-konjugiert.*
- *Sei $G = G_1 \circ_{par} G_2$ ein paralleles Netzwerk. Die Konjugation $\bar{G}$ von G ist ein Netzwerk, das aus der seriellen Verschaltung der Konjugationen der Teilnetzwerke G_1 und G_2 besteht, d.h.*

$$\bar{G} = \bar{G}_1 \circ_{ser} \bar{G}_2$$

- *Sei $G = G_1 \circ_{ser} G_2$ ein serielles Netzwerk. Die Konjugation $\bar{G}$ zu G ist ein Netzwerk, das aus der parallelen Verschaltung der Konjugationen der Teilnetzwerke G_1 und G_2 besteht, d.h.*

$$\bar{G} = \bar{G}_1 \circ_{par} \bar{G}_2$$

Das Konjugieren von sp-Netzwerken ist, wie die Definition dieser Netzwerke, auch rekursiv gegeben.

Die sp-Netzwerke der Ordnung 2 entstehen durch die serielle Verschaltung zweier minimaler Netzwerke und der Konjugation des so entstandenen Netzwerkes (vgl. Abbildung 4.4).

Abbildung 4.4: Alle sp-Netzwerke der Ordnung 2, wobei das linke ein serielles und das rechte ein paralleles Netzwerk ist

Alle seriellen Netzwerke dritter Ordnung können erzeugt werden, indem zum einen ein paralleles Netzwerk der Ordnung 2 mit dem minimalen Netzwerk seriell verschaltet wird und zum anderen drei minimale Netzwerke seriell verschaltet werden. Um alle parallelen Netzwerke der Ordnung 3 zu erhalten, werden die seriellen Netzwerke der Ordnung 3 konjugiert (vgl. Abbildung 4.5).

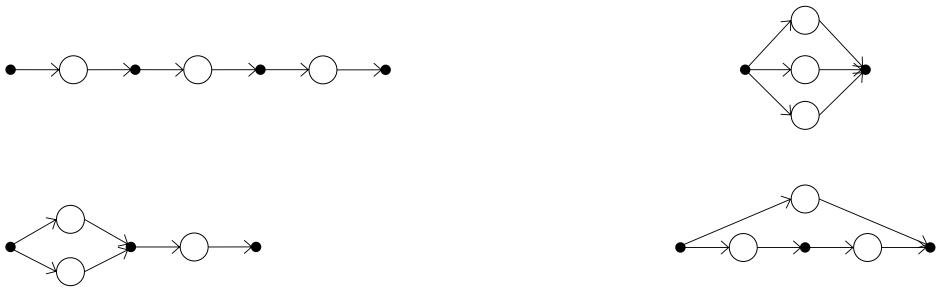

Abbildung 4.5: Alle sp-Netzwerke der Ordnung 3, wobei die linke Spalte alle seriellen und die rechte Spalte alle parallelen Netzwerke enthält

[1] Das minimale Netzwerk aus Definition 4.1 entspricht im Originalpaper einem sogenannten „line branch“.

Alle seriellen Netzwerke der Ordnung 4 lassen sich durch die serielle Verschaltung der folgenden Netzwerke erzeugen:

- Je ein paralleles Netzwerk der Ordnung 3 mit einem minimalen Netzwerk
- Zwei parallele Netzwerke der Ordnung 2 miteinander
- Ein paralleles Netzwerk der Ordnung 2 mit zwei minimalen Netzwerken
- Vier minimale Netzwerke

Alle parallelen Netzwerke der Ordnung 4 ergeben sich durch das Konjugieren von seriellen Netzwerken der Ordnung 4 (vgl. Abbildung 4.6).

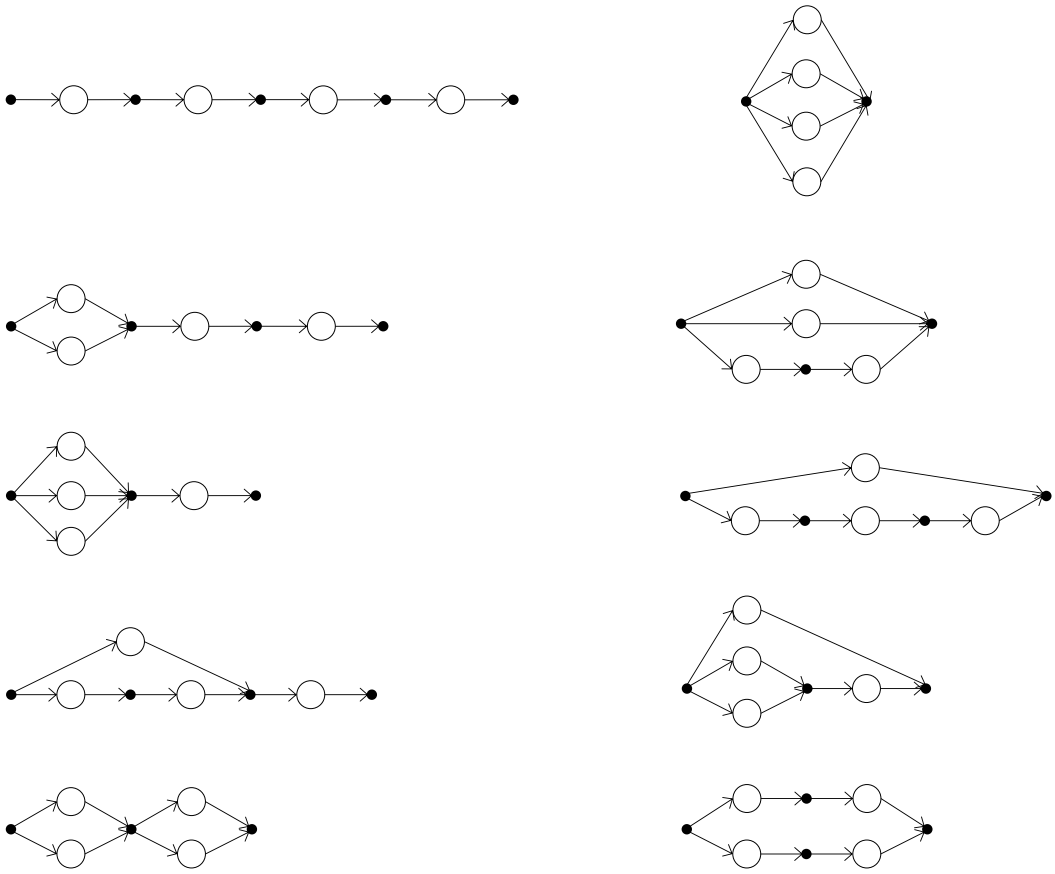

Abbildung 4.6: Alle sp-Netzwerke der Ordnung 4, wobei die linke Spalte alle seriellen und die rechte Spalte alle parallelen Netzwerke enthält

Das Vorgehen, um alle sp-Netzwerke der Ordnung 5 zu erhalten, ist analog. Alle seriellen Netzwerke der Ordnung 5 lassen sich durch serielle Verschaltung von parallelen Netzwerken niedrigerer Ordnung erzeugen. Die zu verschaltenden Netzwerke ergeben sich durch die folgende Auswahl:

- Je ein paralleles Netzwerk der Ordnung 4 mit einem minimalen Netzwerk
- Je ein paralleles Netzwerk der Ordnung 3 mit einem parallelen Netzwerk der Ordnung 2
- Je ein paralleles Netzwerk der Ordnung 3 mit zwei minimalen Netzwerken

- Zwei parallele Netzwerke der Ordnung 2 mit einem minimalen Netzwerk
- Ein paralleles Netzwerk der Ordnung 2 mit drei minimalen Netzwerken
- Vier minimale Netzwerke.

Anschließend werden alle so entstandenen Netzwerke konjugiert, um alle sp-Netzwerke der Ordnung 5 zu erhalten.

Wird diese Vorgehensweise für sp-Netzwerke der Ordnung n verallgemeinert, so führt dies zu Satz 4.1.

Bemerkung 4.3. *Eine Partition der Zahl n lässt sich durch $p = ((s_1, \lambda_1), ..., (s_k, \lambda_k))$ mit $s_i, \lambda_i \in \mathbb{N}$, $i = 1...k$ und s_i, paarweise verschieden, repräsentieren, falls $n = \sum_{i=1}^{k} \lambda_i s_i$.*

Satz 4.1. *Alle sp-Netzwerke der Ordnung $n \geq 2$ lassen sich erzeugen, indem die folgenden zwei Schritte befolgt werden:*

(i) Für jede Partition $p = ((s_1, \lambda_1), ...(s_k, \lambda_k))$ der Zahl n mit $s_i < n$ wird das Folgende getan:

Es werden jeweils λ_i-viele Netzwerke aus der Menge der parallelen Netzwerke der Ordnung s_i gewählt (Ziehen mit Zurücklegen). Diese Netzwerke werden dann seriell verschaltet. Dieses Vorgehen wird für alle möglichen Kombinationen aus Netzwerken wiederholt, die durch Ziehen mit Zurücklegen ohne Beachtung der Reihenfolge entstehen.

(ii) Alle Netzwerke, die aus (i) entstanden sind, werden konjugiert.

Beweis. *Zuerst soll gezeigt werden, dass sich jedes serielle Netzwerk der Ordnung n durch (i) erzeugen lässt.*

IA $n = 2$
Es gibt nur ein serielles Netzwerk der Ordnung 2. Dieses entsteht durch die serielle Verschaltung zweier minimaler Netzwerke.
Die Partition von 2 mit Summanden echt kleiner 2 ist durch $n = 2 \cdot 1$ eindeutig gegeben. Die Menge aus Netzwerken der Ordnung 1 beinhaltet nur das minimale Netzwerk; somit lassen sich alle seriellen Netzwerke der Ordnung 2 mit der im Schritt (i) beschriebene Vorgehensweise erzeugen.

IV Es lassen sich alle seriellen Netzwerke der Ordnung $k = 2...n$ mit der in Schritt (i) beschriebenen Vorgehensweise erzeugen.

IS $n \to n + 1$

Jedes serielle Netzwerk G der Ordnung n lässt sich per Definition in zwei sp-Netzwerke G_1 und G_2 der Ordnung l bzw. m zerlegen mit $l + m = n$, da $n \geq 2$, d.h. $G = G_1 \circ_{ser} G_2$.

1. *Fall: Beide Netzwerke G_1 und G_2 sind parallele Netzwerke. Dann lässt sich das Netzwerk G mit der im 1. Schritt beschriebenen Vorgehensweise erzeugen. Die zugehörige Partition ist in diesem Fall durch $(l, 1), (m, 1)$ gegeben.*
2. *Fall: Nur eines der beiden Netzwerke G_1 oder G_2 ist ein serielles Netzwerk, sei O.B.d.A. G_1 seriell und somit G_1 parallel. Dann lässt sich G_1 mit der in Schritt (i) beschriebenen Vorgehensweise erzeugen, d.h. G_1 lässt sich als serielle Verschaltung von parallelen Netzwerken darstellen. Sei die Partition hierzu durch $p_1 = ((s_1, \lambda_1), ..., (s_i, \lambda_i))$ gegeben. Dann lässt sich G durch die Partition $p = ((s_1, \lambda_1), ..., (s_i, \lambda_i), (m, 1))$ erzeugen. Entspricht einer der Werte $s_1, ..., s_i$ m wird das zugehörige λ angepasst.*
3. *Fall: Beide Netzwerke G_1 und G_2 sind serielle Netzwerke. Dann lassen sich sowohl G_1 als auch G_2 mit der in Schritt (i) beschriebenen Vorgehensweise erzeugen, d.h. es existieren Partitionen $p_1 = ((s_{11}, \lambda_{11}), ..., (s_{1i}, \lambda_{1i}))$ und $p_2 = ((s_{21}, \lambda_{21}), ..., (s_{2j}, \lambda_{2j}))$ durch die G_1 bzw. G_2 entstehen. Somit entsteht G aus der Partition $p = ((s_{11}, \lambda_{11}), ..., (s_{1i}, \lambda_{1i}), (s_{21}, \lambda_{21}), ..., (s_{2j}, \lambda_{2j}))$. Treten zwei Werte in beiden Partitionen auf, werden die λ addiert.*

Da es zu jedem seriellen Netzwerk ein konjugiertes paralleles Netzwerk gibt und umgekehrt, ergibt die Konjugation der seriellen Netzwerke aus Schritt i alle parallelen Netzwerke der Ordnung n und somit werden durch die Schritte (i) und (ii) alle sp-Netzwerke der Ordnung n erzeugt.

□

Aus Satz 4.1 lässt sich die Anzahl B_n an seriellen Netzwerken der Ordnung n ableiten. Für die Auswahl aus λ_i-vielen Netzwerken aus der Menge der parallelen Netzwerke der Ordnung s_i gibt es

$$\binom{B_{s_i} - \lambda_i}{\lambda_i}$$

viele mögliche Kombinationen. Somit gibt es für jede Partition $p = ((s_1, \lambda_1), (s_2, \lambda_2), ..., (s_k, \lambda_k))$

$$\binom{B_{s_1} - \lambda_1}{\lambda_1} \cdot \binom{B_{s_2} - \lambda_2}{\lambda_2} \cdots \binom{B_{s_k} - \lambda_k}{\lambda_k}$$

viele Möglichkeiten, Netzwerke zu wählen. Insgesamt gilt somit:

$$B_n = \sum_{p \in P(n)} \prod_{i=1}^{k_p} \binom{B_i - \lambda_i}{\lambda_i},$$

wobei $P(n)$ die Menge an Partitionen der Zahl n mit mindestens zwei Summanden ist.

Diese Anzahl entspricht - wie in [5] gezeigt und in [3] erläutert - der Anzahl an Bäumen mit n Blättern, die keinen inneren Knoten vom Grad 2 besitzten. Solche Bäume haben keine inneren Knoten, die nur eine eingehende und eine ausgehende Kante besitzen.

Aus dem Umstand, dass es für jedes serielle Netzwerk der Ordnung n einen solchen Baum mit n Blättern gibt, folgt, dass ein serielles Netzwerk durch einen Baum repräsentiert werden kann. Da es durch Konjugation zu jedem seriellen Netzwerk ein zugehöriges paralleles Netzwerk gibt, beschreibt jeder Baum genau zwei Netzwerke. Die hier erläuterte Vorgehensweise beschreibt die Übersetzung der Baumdarstellung in ein serielles Netzwerk, wie sie in [15] vorgeschlagen wird.

Jedes Blatt im Baum entspricht einem minimalen Netzwerk. Alle weiteren Knoten im Baum stellen ein paralleles bzw. serielles Verschalten der Teilnetzwerke dar. Auf welche Art verschaltet wird, hängt von der Interpretation und der Stufe ab, auf der sich der Knoten befindet. Werden serielle Netzwerke betrachtet und beginnt die Nummerierung der Stufen an der Wurzel mit null, so stellen alle Knoten auf einer geraden Stufe eine serielle Verschaltung der Teilnetze dar. Alle Knoten auf einer ungeraden Stufe stellen eine parallele Verschaltung der Teilnetze dar. Da es zu jedem seriellen Netzwerk ein eindeutiges konjugiertes paralleles Netzwerk gibt, kann auf diese Art auch ein paralleles Netzwerk aus einem Baum erstellt werden. In diesem Fall ist das Vorgehen beim Verschalten der Netzwerke invers.

Dieser Vorgehensweise folgend sind die Baumdarstellungen für sp-Netzwerke der Ordnung 2, 3 und 4 in den Abbildungen 4.7 dargestellt.

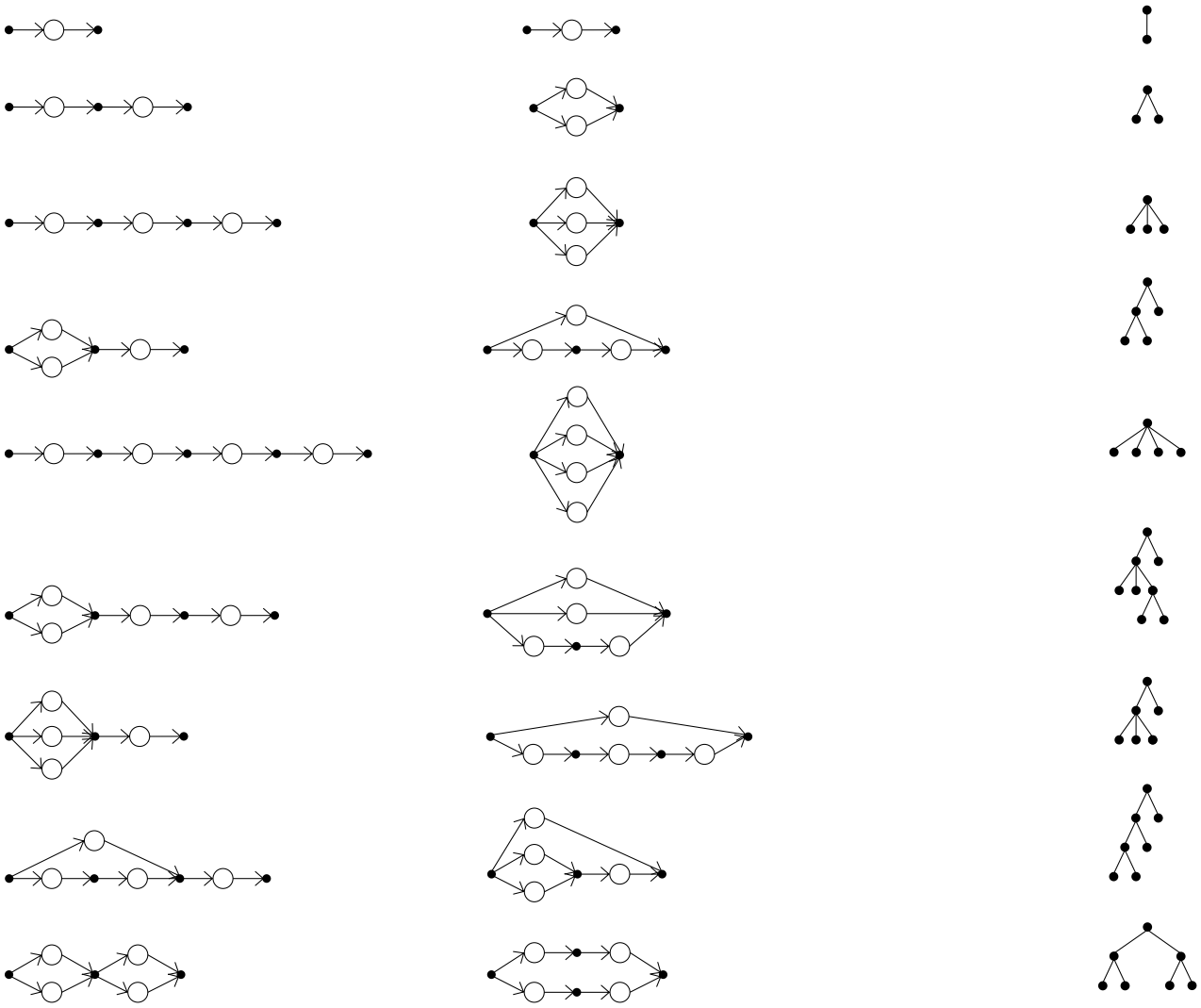

Abbildung 4.7: Alle Paare aus zueinander konjugierten Netzwerken von der Ordnung 1 bis 4 mit entsprechender Baumdarstellung

5 Dynamische Optimierung

In diesem Kapitel wird die Dynamische Optimierung als Möglichkeit eine effiziente Lösungsmethode vorgestellt. Dazu werden in dem in Kapitel 3 vorgestellten Modell alle Kaufentscheidungen festgelegt. Das nun zu lösende Problem ist ein Aussteuerungsproblem, dessen Lösung eine kostenminimale Festlegung der Pumpendrehzahlen mit resultierenden Volumenströmen, Druckerhöhungen, Speicherständen etc. darstellt. Um einem hohen Rechenaufwand, der bei der Betrachtung von Lastprofilen mit vielen Zeitabschnitten entsteht, entgegenzuwirken, werden die Füllstände diskretisiert. Dann wird die mittels Dynamischer Optimierung gefundene Lösung des angepassten Problems in Abschnitt 5.2 vorgestellt. Vorab wird die Dynamische Optimierung in Abschnitt 5.1 im Allgemeinen erläutert.

5.1 Mathematische Grundlagen

Die Dynamische Optimierung (DO) wird zur Lösung von Optimierungsproblemen verwendet, die aus Folgen von abhängigen Entscheidungsproblemen bestehen. Hierzu werden zunächst Lösungen für die kleinsten Teilprobleme gesucht, indem nur die im Teilproblem zu treffenden Entscheidungen berücksichtigt werden. Aus diesen Ergebnissen werden Lösungen für nächstgrößere Probleme erstellt. Dieser Vorgang der Problemerweiterung wird solange wiederholt, bis eine Lösung für das Gesamtproblem gefunden wird.

Bezeichnend für die DO ist, dass die optimalen Lösungen von größeren Teilproblemen immer auch optimale Lösungen für die kleineren darin enthaltenen Teilprobleme sind. Die im Folgenden gegebenen Definitionen und Aussagen sind angelehnt an Domschke [6].

5.1.1 Lösbare Probleme

Um ein Optimierungsproblem mit DO lösen zu können, muss es wie in nachfolgender Definition 5.1 darstellbar sein. An dieser Stelle sollen nur Optimierungsprobleme mit zu minimierender Zielfunktion betrachtet werden. Die Aussagen gelten analog für Maximierungsprobleme und ergeben sich durch Multiplikation der Zielfunktion mit minus eins.

Definition 5.1. *(Erste allgemeine Form der Dynamischen Optimierung (AG-DO1))[6]*
Seien folgende Begriffe definiert:

$\mathbf{n}$	*Anzahl an **Stufen**, in die der Entscheidungsprozess zerlegt werden kann*
$\mathbf{z_k}$	***Zustandsvariable**, die angibt in welchem Zustand sich das betrachtete System nach der Stufe k befindet*
$\mathbf{Z_k}$	***Zustandsmenge**: Menge aller Zustände, in denen sich das System nach der Stufe k befinden kann*
$\mathbf{z_0 = a}$	*vorgegebener (ggf. mehrdimensionaler) **Anfangszustand** des Systems*
$\mathbf{z_n = b}$	*vorgegebener (ggf. mehrdimensionaler) **Endzustand** des Systems*
$\mathbf{x_k}$	*(ggf. mehrdimensionale) **Entscheidungsvariable** des Modells, die angibt, welche Entscheidung in Stufe k getroffen wird*

$\mathbf{X_k(z_{k-1})}$ ***Entscheidungsmenge**: Menge aller Entscheidungen, die in Stufe k, vom Zustand z_{k-1} ausgehend, getroffen werden können*

$\mathbf{tr_k(z_{k-1}, x_k)}$ ***Transformationsfunktion**, beschreibt, in welchen Zustand z_k das System, ausgehend vom Zustand z_{k-1}, durch Treffen der Entscheidung x_k überführt wird*

$\mathbf{f_k(z_{k-1}, x_k)}$ ***Stufenbezogene Zielfunktion**, die den Einfluss auf die Zielfunktion beschreibt, den die Entscheidung x_k ausgehend vom Zustand z_{k-1} hat. Dabei darf f_k nur vom Zustand z_{k-1} und der Entscheidung x_k abhängen.*

Dann ist die erste allgemeine Form der Dynamischen Optimierung gegeben durch:

$$\begin{aligned} \min \quad & F(x_1, \cdots, x_n) = \sum_{k=1}^{n} f_k(z_{k-1}, x_k) \\ s.t. \quad & z_k = tr_k(z_{k-1}, x_k) && \forall k = 1...n \\ & z_0 = a \\ & z_n = b \\ & z_k \in Z_k && \forall k = 1...n \\ & x_k \in X_k(z_{k-1}) && \forall k = 1...n \end{aligned}$$

Ist für ein gegebenes Zustandspaar z_{k-1} und z_k die für die Überführung des Systems in einen neuen Zustand zu treffende Entscheidung x_k eindeutig, existiert eine äquivalente Definition für die allgemeine Form der Dynamischen Optimierung. Die Transformationsfunktion aus Definition 5.1 gibt an, in welchen Zustand das System übergeführt wird, wenn ausgehend von einem Zustand eine Entscheidung getroffen wurde. Eine Alternative wäre eine Entscheidungsfunktion, die angibt, welche Entscheidung getroffen werden muss, um das System von einem gegebenen in einen anderen gewünschten Zustand zu überführen. Dies führt zur zweiten allgemeinen Form der Dynamischen Optimierung.

Definition 5.2. *(Zweite allgemeine Form der Dynamischen Optimierung (AG-DO2))*
Sei die Entscheidungsfunktion wie folgt definiert:

$\mathbf{e_k(z_{k-1}, z_k)}$ ***Entscheidungsfunktion**: Sie beschreibt, welche eindeutige Entscheidung x_k getroffen werden muss, damit sich das System nach Stufe k im Zustand z_k befindet, wenn es sich vor Stufe k im Zustand z_{k-1} befunden hat.*

Dann ist die zweite allgemeine Form der Dynamischen Optimierung gegeben durch:

$$\begin{aligned} \min \quad & F(x_1, \cdots, x_n) = \sum_{k=1}^{n} f_k(z_{k-1}, x_k) \\ s.t. \quad & x_k = e_k(z_{k-1}, z_k) && \forall k = 1...n \\ & z_0 = a && \forall k = 1...n \\ & z_n = b && \forall k = 1...n \\ & z_k \in Z_k && \forall k = 1...n \\ & x_k \in X_k(z_{k-1}) && \forall k = 1...n. \end{aligned}$$

Bemerkung 5.1. *Die beiden Darstellungen AG-OD1 und AG-OD2 sind im Fall einer eindeutigen Entscheidung äquivalent. Lässt sich ein solches Optimierungsproblem in der Form aus AG-DO1 darstellen, so ist dies auch in der Form aus AG-DO2 möglich und umgekehrt.*

Denn ist bekannt, in welchem Zustand sich das System ausgehend vom Zustand z_{k-1} nach der Entscheidung x_k befindet, so ist auch bekannt, welche Entscheidung getroffen werden muss um das System von z_{k-1} in z_k zu überführen. Andererseits kann aus dem Wissen, welche Entscheidung zu treffen ist, um das System von z_{k-1} in z_k zu überführen, abgeleitet werden, in welchem Zustand sich das System ausgehend vom Zustand z_{k-1} nach der Entscheidung x_k befindet.

Sind die zu betrachtenden Mengen Z_k und X_k endlich, so lässt sich ein durch DO lösbares Optimierungsproblem als Graph darstellen.

Definition 5.3. *(Graphdarstellung)[6]*
Sei $G = (V, E)$ die Graphdarstellung zum zu lösenden Optimierungsmodell. Dann ist $V = \{a\} \cup Z_1 \cup Z_2 \cup ... \cup \{b\}$, wobei jeder Knoten $z_k \in Z_k$ einem Zustand in der Stufe k entspricht. Für jede Entscheidung x_k, die das System vom Zustand z_{k-1} in den Zustand z_k überführt, enthält E eine Kante (z_{k-1}, z_k). Der Übergang zwischen den beiden Zuständen erzeugt die Kosten $f_k(z_{k-1}, x_k)$. Diese werden als Kantengewichte gesetzt.

Gibt es in der Graphendarstellung des Optimierungsproblems keinen Weg von z_0 nach z_n, ist es nicht möglich, das System vom Zustand z_0 in den Zustand z_n zu überführen. Daraus folgt, dass das Optimierungsproblem keine Lösung hat.

5.1.2 Das Lösungsprinzip

Optimierungsprobleme, die in der Form AG-DO1 oder AG-DO2 darstellbar sind und endliche Zustands- und Entscheidungsmengen besitzen, lassen sich mit Algorithmus 1 lösen. Für diese Probleme seien die nachfolgenden Begriffe definiert.

Definition 5.4. *(optimale Politik)[6]*
*Eine Folge $(x_h,\ x_{h+1}, ...,\ x_l)$ von Entscheidungen, die ein System von einem Zustand z_{h-1} in den Zustand z_l überführt, wird als **Politik** bezeichnet. Eine Politik $(x_h^*,\ x_{h+1}^*, ...,\ x_l^*)$, die $\sum_{i=h}^{l} f_i(z_{i-1}, x_i)$ minimiert, so wird sie **optimale Politik** genannt.*

Definition 5.5. *[6]*

1. *Das Finden einer optimalen Politik für AG-DO1 wird als $\mathbf{P_0(z_0 = a)}$ bezeichnet.*
2. *Das Finden einer optimalen Politik vom Zustand z_k bis zum Endzustand wird als $\mathbf{P_k(z_k)}$ bezeichnet.*
3. *Die Lösung (optimale Politik) des Problems $\mathbf{P_k(z_k)}$ ist $\xi_k^*(z_k) = (x_{k+1}^*, ...,\ x_n^*)$.*
4. *Der optimale Zielfunktionswert des Problems $\mathbf{P_k(z_k)}$ ist $\mathbf{F_k^*(z_k)} = F_k(\xi_k^*(z_k))$.*

Aus der Eigenschaft, dass die stufenbezogene Zielfunktion $f_k(z_{k-1}, x_k)$ nur vom Zustand z_{k-1} und der Entscheidung x_k abhängt, ergibt sich folgende Aussage über optimale Politiken.

Satz 5.1. *(Bellman'sches Optimalitätsprinzip)[6]*
Sei $x_1^*, ..., x_n^*$ eine Lösung von $P_0(z_0 = a)$ und sei z_{j-1}^* der Zustand, in dem sich das System in der Stufe $j-1$ befindet, wenn diese Politik befolgt wird. Dann gilt:

- *$x_j^*, ..., x_n^*$ ist eine optimale Politik, die das System vom Zustand z_{j-1}^* in den Endzustand überführt. Sie ist somit eine Lösung von $P_{j-1}(z_{j-1}^*)$.*
- *$x_1^*, ..., x_{j-1}^*$ist eine optimale Politik, die das System vom Anfangszustand in den Zustand z_{j-1}^* überführt.*

Algorithmus 1 Dynamische Optimierung in Form der Rückwärtsrekursion

1: Löse $P_{n-1}(z_{n-1})$ für alle $z_{n-1} \in Z_{n-1}$.
2: Setze $x_n^*(z_{n-1})$ gleich der Lösung von $P_{n-1}(z_{n-1})$.
3: **for** $k = n-1$ **to** 1 **do**
4: Löse $\mathbf{P_{k-1}(z_{k-1})}$ für alle $z_{k-1} \in Z_{k-1}$ durch Lösen der Gleichung:
$F_{k-1}^*(z_{k-1}) = \min_{x_k \in X_k(z_{k-1})} \{f_k(z_{k-1}, x_k) + F^*(z_k = \mathrm{tr}_k(z_{k-1}, x_k))\}$
5: Setze $x_k^*(z_{k-1}) = \arg\min_{x_k \in X_k(z_{k-1})} \{f_k(z_{k-1}, x_k) + F^*(z_k = \mathrm{tr}_k(z_{k-1}, x_k))\}$
6: **end for**
7: Sei x_1^* die Entscheidung, die $F_0^*(a)$ minimiert und $z_1^* = \mathrm{tr}_1(a, x_1^*)$
8: **for** $k = 2$ **to** n **do**
9: Setze $x_k^* = x_t(z_{k-1}^*)$ und $z_k^* = \mathrm{tr}_k(z_{k-1}, x_k^*)$.
10: **end for**
11: $x^* = (x_1^*, ..., x_n^*)$ ist die Lösung von $P_0(a)$.

Auf Grund des Bellman'schen Optimalitätsprinzips ist es möglich, durch Rückwärtsrekursion die Optimallösung zu finden.

Definition 5.6. *(Bellman'sche Gleichung)[6]*
Die Bellman'sche Gleichung ist gegeben durch:

$$F_{k-1}^*(z_{k-1}) = \min_{x_k \in X_k(z_{k-1})} \{f_k(z_{k-1}, x_k) + F^*(z_k = t_k(z_{k-1}, x_k))\}.$$

Beispiel 5.1. *Ein Beispiel für ein solches Optimierungsmodell ist das* ***Postkutschen-Problem****[14]: Zur Zeit des Goldrausches möchte ein Kaufmann von Osten nach Westen durch die USA reisen, wofür ihm verschiedene Routen zur Verfügung stehen. Auf allen Strecken besteht die Gefahr, dass er von Banditen überfallen wird. Für jeden Teilabschnitt der Routen (vgl. Abb. 5.1) bieten die Postkutscher eine unterschiedlich hohe Lebensversicherung an. Da es sich bei dem Reisenden um einen ängstlichen Mann handelt, beschließt er, die Strecke zu wählen, bei der die Summe aller zu zahlenden Lebensversicherungspolicen am geringsten ist. Denn er geht davon aus, dass die Höhe der Versicherung ein Maß für die Gefahr ist, auf der Strecke überfallen zu werden.*

Das Auffinden der kostengünstigsten Route lässt sich mit Hilfe der Dynamischer Optimierung lösen. Dabei ist der vorgegebene Anfangszustand $\mathbf{z_0}$ *die Stadt Start und der Endzustand* $\mathbf{z_n}$ *die Stadt End. Aufgrund der Anzahl an Zwischenstopps beträgt die Anzahl* $\mathbf{n}$ *an Stufen für das Problem drei. In der ersten Stufe fährt der Reisende in die Stadt A, B oder C. Somit ist die* ***Zustandsmenge*** *der ersten Stufe* $Z_1 = \{$*„Reisender ist in A“, „Reisender ist in B“, „Reisender ist in C“*$\}$*. In der zweiten Stufe fährt der Reisende dann in die Stadt D oder E. Somit ist* $Z_2 = \{$*„Reisender ist in D“, „Reisender ist in E“*$\}$*. Für die* ***Entscheidungsmengen*** *ergibt sich aus Abbildung 5.1:*

- X_1(*„Reisender ist in Start“*) = {*Start* → *A, Start* → *B, Start* → *C*}
- X_2(*„Reisender ist in A“*) = {*A* → *D, A* → *E*}
- X_2(*„Reisender ist in B“*) = {*B* → *D, B* → *E*}
- X_2(*„Reisender ist in C“*) = {*C* → *D, C* → *E*}
- X_3(*„Reisender ist in D“*) = {*D* → *End*}
- X_3(*„Reisender ist in E“*) = {*E* → *End*}

Die ***Transformationsfunktion*** *gibt an, in welcher Stadt sich der Reisende befindet, nachdem er die Entscheidung* x_k *getroffen hat. Der Beitrag einer Entscheidung zu der Zielfunktion ist die Versicherungspolice, die*

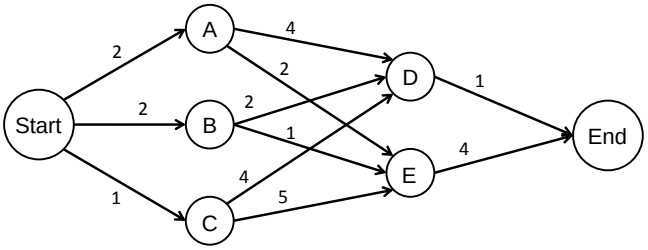

Abbildung 5.1: Alle Strecken, die dem Kaufmann für seine Routenplanung zur Verfügung stehen. Die Gewichte an den Kanten sind die zu zahlenden Policen in Hundert Dollar

auf der gewählten Strecke anfällt. Die Police ist auf den Kanten als Gewicht c_{ij} angegeben. Somit gilt für die ***stufenbezogenen Zielfunktion*** *$f_k(z_{k-1}, x_k) = c_{ij}$, wobei i die Stadt des Zustandes z_{k-1} ist und j die Stadt, in der der Reisende ankommt, wenn er die Entscheidung x_k trifft.*

Der Idee der DO folgend kann das Problem wie folgt gelöst werden: Angenommen, der Reisende ist schon in einer der beiden Städte der letzten, also der zweiten Stufe angekommen, so ist der günstigste Weg zum Ziel eindeutig vorgegeben. Für jede Stadt der zweiten Stufe ist somit bekannt, welcher der günstigste Weg nach End ist.

z_2	x_3	$f_3(z_2, x_3)$	$F_2^*(z_2)$
„Reisender ist in Stadt D“	*$D \rightarrow End$*	*1*	*1*
„Reisender ist in Stadt E“	*$E \rightarrow End$*	*4*	*4*

Wird anschließend angenommen, dass der Reisende sich in einer Stadt der ersten Stufe befindet, so kann auch für jede dieser Städte der Stufe 1 berechnet werden, wie teuer der Weg bis zu jeder Stadt der zweiten Stufe ist. Mit Hilfe der vorherigen Berechnung von $F_2^*(z_2)$ kann auch bestimmt werden, wie teuer der Weg von dort aus nach End ist. Somit kann auch hier festgestellt werden, über welche der beiden Städte D oder E der Weg nach End am günstigsten ist. Damit ist auch für die Städte der ersten Stufe bekannt, welcher der günstigste Weg nach End ist.

z_1	x_2	$f_2(z_1, x_2)$	$f_2(z_1, x_2) + F_2^*(z_2)$	$F_1^*(z_1)$
„Reisender ist in Stadt A“	*$A \rightarrow D$*	*4*	*$5 \leftarrow min$*	*5*
	$A \rightarrow E$	*2*	*6*	
„Reisender ist in Stadt B“	*$B \rightarrow D$*	*2*	*$3 \leftarrow min$*	*3*
	$B \rightarrow E$	*1*	*5*	
„Reisender ist in Stadt C“	*$C \rightarrow D$*	*4*	*$5 \leftarrow min$*	*5*
	$C \rightarrow E$	*5*	*9*	

Für die Stadt Start kann nun einfach bestimmt werden, über welche Stadt der ersten Stufe der kürzeste Weg nach End führt.

z_0	x_1	$f_1(z_0, x_1)$	$f_1(z_0, x_1) + F_1^*(z_1)$	$F_0^*(z_0)$
„Reisender ist in Stadt Start“	*Start → A*	*2*	*7*	
	Start → B	*2*	*5 ← min*	*5*
	Start → C	*1*	*6*	

Somit ist die optimale Politik mit $x_0^*(z_0)$ =(Start → B, B → D, D→ End) gefunden.

5.2 Lösen des Aussteuerungsproblems

In diesem Abschnitt wird gezeigt, dass das Optimierungsproblem bei fester Kaufentscheidung und diskreten Speicherfüllständen gemäß Kapitel 3 mittels Dynamischer Optimierung lösbar ist. Zur Vereinfachung soll hier ein Netzwerk mit einem Speicher betrachtet werden. Die Übertragung auf Netzwerke mit mehreren Speichern wird in Bemerkung 5.2 erläutert.

Definition 5.7.

- *Sei $\mathscr{P} = \min\{c^T x : Ax \leq b, x \in \mathbb{R}^m \times \mathbb{Z}^k\}$ das Optimierungsmodell mit festgelegter Kaufentscheidung.*
- *Sei $\mathscr{P}_t = \min\{c^T x_t : A_t x_t \leq b_t : x_t \in \mathbb{R}_t^m \times \mathbb{Z}_t^k\}$ das Optimierungsmodell mit festgelegter Kaufentscheidung für einen gegeben Zeitschritt t.*
- *Sei $\mathscr{P}_t(z_{t-1}) = \min\{c^T x_t : A(z_{t-1})_t x_t \leq b_t : x_t \in \mathbb{R}_t^m \times \mathbb{Z}_t^k\}$ das Optimierungsmodell für einen ausgewählten Zeitschritt mit festgelegter Kaufentscheidung und einem festen Anfangsfüllstand z_{t-1}.*
- *Sei $\mathscr{P}_t(z_{t-1}, z_t) = \min\{c^T x_t : A(z_{t-1}, z_t)_t x_t \leq b_t : x_t \in \mathbb{R}_t^m \times \mathbb{Z}_t^k\}$ das Optimierungsmodell für einen ausgewählten Zeitschritt mit festgelegter Kaufentscheidung und einem festen Anfangs- und Endfüllstand z_{t-1} und z_t.*

Das Aussteuerungsproblem lässt sich in der Form aus AG-DO2 darstellen:

$\mathbf{n}$	**Anzahl an Stufen** entspricht der Anzahl an Zeitschritten aus dem MIP bzw. Lastfall plus eins. Im Modell gemäß Kapitel 3 ist kein fester Endfüllstand für den Speicher gegeben. Aus diesem Grund wird eine zusätzliche Stufe angehängt und gefordert, dass der Speicher am Ende dieser Stufe leer sein muss.
$\mathbf{z_t}$	**Zustandsvariable**, die angibt, wie hoch das Wasser am Ende des Zeitschrittes t im Speicher steht.
$\mathbf{Z}$	**Zustandsmenge**: Menge aller möglichen Wasserstände des Speichers.
$\mathbf{z_0 = 0}$	vorgegebener **Anfangsfüllstand** des Speichers.
$\mathbf{z_n = 0}$	vorgegebener **Endfüllstand** des Speichers.
$\mathbf{x_t}$	**Entscheidungsvariable**, die im Zeitschritt t getroffen wird. Die Entscheidung, die im Zeitschritt n getroffen wird, ist, den Speicher zu leeren.
$\mathbf{X_t(z_{t-1})}$	**Entscheidungsmenge**, die der Lösung des Problems $\mathscr{P}_t(z_{t-1})$ entspricht[1].
$\mathbf{e_t(z_{t-1}, z_t)}$	**Entscheidungsfunktion**, die sich zu $e_t(z_{t-1}, z_t) = \arg\min\{c^T x_t : A(z_{t-1}, z_t)_t x_t \leq b_t : x_t \in \mathbb{R}^{m_t} \times \mathbb{Z}^{k_t}\}$ ergibt.
$\mathbf{f_t(z_{t-1}, x_t)}$	**stufenbezogene Zielfunktion**, die sich zu $\mathbf{f_t(z_{t-1}, x_t)} = c_t^T x_t$ ergibt. Der Übergang vom Zustand z_{n-1} nach z_n erzeugt keine Kosten.

[1] Hat das Problem mehrere Optimallösungen, so wird eine ausgewählt.

Bemerkung 5.2. *Werden Netzwerke mit mehreren Speichern betrachtet, so ist ein Systemzustand eine Kombination aus Füllständen. Somit enthält die Zustandsmenge alle möglichen Kombinationen an Füllständen.*

Das Aussteuerungsproblem lässt sich somit durch Dynamische Optimierung lösen. Die sich in diesem Fall ergebende Rückwärtsrekursion ist im nachfolgenden Algorithmus 2 abgebildet.

Da die Entscheidungs- und Zustandsmengen endlich sind, lässt sich das Problem auch als Graph darstellen.

Beispiel 5.2. *Hat das zu lösende Aussteuerungsproblem drei Zeitschritte und jeweils vier mögliche Füllstände, dann ist die Graphendarstellung in Abbildung 5.2 zu sehen.*

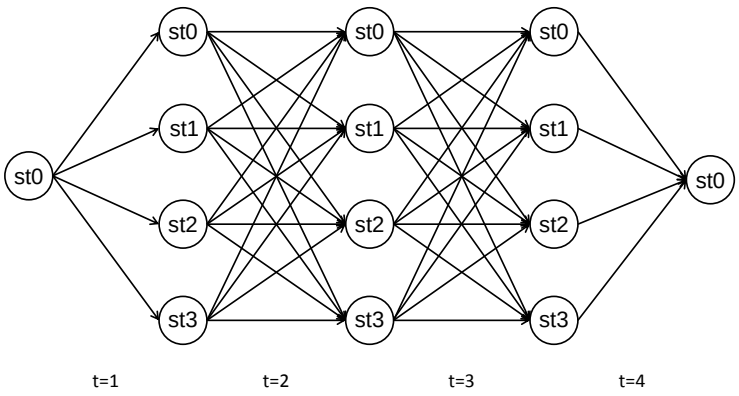

Abbildung 5.2: Austeuerungsproblem mit drei Zeitschritten und jeweils vier möglichen Füllständen

Algorithmus 2 Lösen des Aussteuerungsproblems mit Rückwärtsrekursion

1: Setze $F^*_{n-1}(z_{n-1})=0$.
2: **for** $t = n-1$ **to** 1 **do**
3: Löse $\mathbf{P_{t-1}}(\mathbf{z_{t-1}})$ für alle $z_{t-1} \in Z$ durch Lösen der Gleichung:
$F^*_{t-1}(z_{t-1}) = \min_{z_t \in Z}\{f_t(z_{t-1}, x_t) + F^*_t(z_t) : x_t = e_t(z_{t-1}, z_t)\}$
4: Setze $x^*_{t-1}(z_{t-1}) = \underset{x_t = e_t(z_{t-1}, z_t)}{\arg\min} \{f_t(z_{t-1}, x_t) + F^*_t(z_t)\}$
5: **end for**
6: Sei x^*_1 die Entscheidung, die $F^*_0(a)$ minimiert und $z^*_1 = \mathrm{tr}_1(z_0, x^*_1)$
7: **for** $t = 2$ **to** n **do**
8: Setze $x^*_t = x_t(z^*_{t-1})$ und $z^*_t = \mathrm{tr}_t(z_{t-1}, x^*_t)$.
9: **end for**
10: **return** $x^* = (x^*_1, ..., x^*_n)$ ist die Lösung von $P_0(z_0)$.

Bei Problemen mit vielen Zeitschritten kann die vorgestellte Dynamische Optimierung erhebliche Verkürzungen der Rechenzeit gegenüber einer Lösung des gesamten MIP mit diskreten oder kontinuierlichen Füllständen bewirken. Dabei ist zu beachten, dass bei zu grober Diskretisierung, das Problem AG-DO2 evtl. keine zulässige Politik hat. Dies kann beispielsweise auftreten, wenn das Wasserangebot aus der Quelle oder die Pumpenleistung nicht ausreicht um den Füllstand des Speichers um einen Schritt zu

erhöhen. Des Weiteren kann die benötigte Diskretisierung der Füllstände zum Verlust der Optimalität führen. Die gefundene Lösung ist zwar zulässig für das MIP mit kontinuierlichen Füllständen, jedoch durch die eingeschränkte Auswahl an Füllständen nicht zwangsläufig optimal.

Der größte Rechenaufwand in der hier vorgestellten DO entsteht bei der Lösung der Teilprobleme $\mathscr{P}_t(z_{t-1}, z_t)$, da hierfür MIPs gelöst werden müssen. Deren Anzahl hängt von der Problembeschaffenheit ab. Im Worst-Case müssen insgesamt #Zeitschritte · #Füllstände · #Füllstände viele MIPs gelöst werden. Um den Rechenaufwand des vorgestellten Algorithmus 2 zu verringern, werden daher die folgenden zwei Anpassungen eingeführt.

Zunächst wird der Algorithmus um ein Abbruchkriterium ergänzt, das den Algorithmus beendet, sobald für einen Zeitschritt t keines der Probleme $\mathscr{P}_{\mathbf{t}}(\mathbf{z_{t-1}})$ lösbar ist. In diesem Fall enthält die Graphdarstellung des Problems keinen Weg von z_0 nach z_n und das Gesamtproblem ist unlösbar.

Da die Zeit so eingeteilt wurde, dass in jedem Zeitschritt die Nachfrage konstant ist, kann in mehreren unterschiedlichen Zeitschritten die gleiche Menge an Wasser nachgefragt werden. Haben diese Zeitschritte dieselbe Dauer, so sind die entsprechenden zu lösenden MIPs identisch. Zur Reduktion des Rechenaufwandes werden die MIPs somit lediglich einmalig gelöst. Die Lösung wird abgespeichert und für die nachfolgenden identischen Probleme ausgelesen. Die so angepasste Version des Algorithmus 2 ist im weiter unten aufgeführten Algorithmus 3 abgebildet.

$\mathbf{d}$	Paar aus Nachfrage und Dauer
$\mathbf{D}$	Menge an auftretenden Paaren d
$\mathbf{t_d}$	erster Zeitschritt t, in dem d auftritt
$\mathbf{d_t}$	Paar, welches in Zeitschritt t auftritt
$\mathbf{x_d}$	die Entscheidung, die im Fall von d getroffen wird
$\mathbf{X_d(z_{j-1})}$	Menge an Entscheidungen im Fall d, die ausgehend von Zustand z_{j-1} getroffen werden können
$\mathrm{e_d(z_i, z_j)}$	**Entscheidungsfunktion**, die sich zu $\mathrm{e}_d(z_i, z_j) = \arg\min\{c^T x_t : A(z_i, z_j)_{t_d} x_{t_d} \leq b_{t_d} : x_{t_d} \in \mathbb{R}^m_{t_d} \times \mathbb{Z}^k_{t_d}\}$ ergibt. Dieses Minimum muss entweder berechnet werden oder kann ausgelesen werden
$\mathbf{f_d(z_j, x_d)}$	Beitrag zur Zielfunktion, wenn im Fall d ausgehend von Zustand z_j die Entscheidung x_d getroffen wird.

In den Fällen, in denen zwar die Nachfrage von zwei Zeitschritten übereinstimmt, die Dauer aber abweicht, werden im Algorithmus 3 zwei verschiedene MIPs gelöst. Um auch in diesem Fall nur ein MIP zu lösen, werden alle Zeitschritte mit gleicher Nachfrage so unterteilt, dass sie die gleiche Dauer haben. Dafür kann beispielsweise der größte gemeinsame Teiler verwendet werden. In diesem Fall werden für jede auftretende Nachfrage #Füllstände · #Füllstände viele MIPs gelöst. Insgesamt sinkt die Anzahl an zu lösenden MIPs auf #Füllstände · #Füllstände · #Nachfragen. Auch in diesem Fall muss beachtet werden, dass die Reduktion der Rechenzeit zu einem schlechteren Zielfunktionswert führen kann. Ist zum Beispiel für einen zu unterteilenden Zeitschritt in der optimalen Politik vorgesehen, dass der Füllstand um eine Einheit gesenkt wird, um die Nachfrage zu bedienen, kann nach der Einteilung des Zeitschrittes diese optimale Politik nicht mehr verfolgt werden. Auf Grund der Diskretisierung kann der Füllstand nur in den vorgegebenen Schritten geändert werden und müsste somit um zwei Level gesenkt werden, oder die Nachfrage in beiden Teilschritten wird nicht mehr mit Wasser aus dem Speicher gedeckt.

Ist die Anzahl an auftretenden Nachfragen nicht wesentlich geringer als die Anzahl an betrachteten Zeitabschnitten, gibt es eine andere Möglichkeit, die Anzahl an zu lösenden MIPs zu verringern. Dabei wird

Algorithmus 3 Lösen des Aussteuerungsproblems mit Rückwärtsrekursion (angepasste Version)

Setze $F^*_{n-1}(z_{n-1})$=0.
for $t = n-1$ **to** 1 **do**
    Löse $\mathbf{P_{t-1}}(\mathbf{z_{t-1}})$ für alle $z_{t-1} \in Z$ durch Lösen der Gleichung:
    $F^*_{t-1}(z_{t-1}) = \min_{z_t \in Z}\{f_t(z_{t-1}, x_t) + F^*_t(z_t) : x_t = e_{d_t}(z_{t-1}, z_t)\}$
    Setze $x^*_{t-1}(z_{t-1}) = \underset{x_t \in X_t(z_{t-1})}{\arg\min}\ \{f_t(z_{t-1}, x_t) + F^*_t(z_t) : z_t \in Z\}$
    if $\mathbf{P_{t-1}}(\mathbf{z_{t-1}})$ unlösbar für alle $z_{t-1} \in Z$ **then**
        return Das Problem ist nicht lösbar.
    end if
end for
Sei x^*_1 die Entscheidung, die $F^*_0(a)$ minimiert und $z^*_1 = \mathrm{tr}_1(z_0, x^*_1)$
for $t = 2$ **to** n **do**
    Setze $x^*_t = x_t(z^*_{t-1})$ und $z^*_t = \mathrm{tr}_t(z_{t-1}, x^*_t)$.
end for
return $x^* = (x^*_1, ..., x^*_n)$ ist die Lösung von $P_0(z_0)$.

für die einzelnen Zeitschritte nicht jedes Problem $\mathscr{P}_t(z_{t-1}, z_t)$ mit festgelegtem Anfangs- und Endstand gelöst, sondern lediglich $\mathscr{P}_t(z_{t-1})$ mit festgelegtem Anfangsstand z_{t-1}. Der Endstand z_t des Speichers bleibt eine Entscheidungsvariable. Die Binärvariablen aus 3.2.5, die die Auswahl des Endstandes z_t angeben, werden mit dem Koeffizienten $F^*_t(z_t)$, in die Zielfunktion des Problems $\mathscr{P}_t(z_{t-1})$ aufgenommen. Dadurch ersetzt das Lösen des Optimierungsproblems das Lösen der Bellman'schen Gleichung.

Sei $\hat{\mathscr{P}}_t(z_{t-1})$ das Problem mit angepasster Zielfunktion. Die sich in diesem Fall ergebende Rückwärtsrekursion ist im nachfolgenden Algorithmus 4 dargestellt. Die Anzahl an zu lösenden MIPs beträgt #Füllstände · #Zeitschritte. Dabei ist aber zu beachten, dass die zu lösenden MIPs #Füllstände zusätzliche Binärvariablen und eine zusätzliche, ganzzahlige Variable beinhalten. Es besteht die Möglichkeit, dass mache Füllstände im „Branch&Bound" [18] abgeschnitten werden. Geschieht dies nicht, so müssen alle Füllstände enumeriert werden.

Algorithmus 4 Lösen des Aussteuerungsproblems mit Rückwärtsrekursion (variable Endpufferstände)

Setze $F^*_n(z_n)$=0.
for $t = n-1$ **to** 1 **do**
    Löse $\hat{\mathscr{P}}_{\mathbf{t}}(\mathbf{z_{t-1}})$ für alle $z_{t-1} \in Z$.Sei x^* die Optimallösung.
    Setze $x_t(z_{t-1}) = x^*$.
    Passe die Zielfunktion von $\mathscr{P}_{\mathbf{t-1}}(\mathbf{z_{t-2}})$ an.
    if $\hat{\mathscr{P}}_{\mathbf{t}}(\mathbf{z_{t-1}})$ unlösbar für alle $z_{t-1} \in Z$ **then**
        return Das Problem ist nicht lösbar.
    end if
end for
Sei x^*_1 die Entscheidung, die $F^*_0(a)$ minimiert und $z^*_1 = \mathrm{e}_1(z_0, x^*_1)$
for $t = 2$ **to** n **do**
    Setze $x^*_t = x_t(z^*_{t-1})$ und $z^*_t = \mathrm{e}_t(z_{t-1}, x^*_t)$.
end for
$x^* = (x^*_1, ...x^*_n)$ ist die Lösung von $P_0(z_0)$.

Aufgrund ihrer Effizienz wurden im Rahmen dieser Arbeit die Algorithmen 3 und 4 in C++ implementiert. Die einzelnen MIPs werden mit der ILOG CPLEX Callable Library C API [12] aufgestellt und gelöst. In Kapitel 8 können einige Laufzeitresultate nachgelesen werden.

6 Primale Heuristiken

In diesem Kapitel werden zwei Heuristiken vorgestellt, welche für das Optimierungsproblem aus Abschnitt 3.2 „gute“ zulässige Lösungen finden. Heuristiken werden verwendet, da viele kombinatorische Optimierungsprobleme auf Grund ihrer Größe oder Komplexität nicht in akzeptabler Zeit durch exakte Optimierungsverfahren zu lösen sind. Um für solche Probleme dennoch „gute“ Lösungen zu finden, wurden in den letzten Jahren viele Heuristiken entwickelt.

Die erste vorgestellte Heuristik ist eine lokale Suche. Diese betrachtet Systemaufbauten, welche zulässig im Sinne der Nebenbedingungen aus 3.2.4 sind. Ausgehend von einer zulässigen Lösung werden Nachbarschaftslösungen betrachtet, die durch das Ersetzen, Tauschen, Hinzufügen und Löschen von verbauten Komponenten erzeugt werden.

Die zweite Heuristik ist ein sogenannter Ameisenalgorithmus, der unter Berücksichtigung einer Wahrscheinlichkeitsverteilung zufällig erzeugte Aufbauten betrachtet, die aus seriell und parallel verschalteten Pumpen und Speichern bestehen. Diese Aufbauten können als Startlösung für die lokale Suche verwendet werden.

In beiden Heuristiken werden die gefundenen Lösungen durch Dynamische Optimierung bewertet.

6.1 Tabu-Suche

Die lokale Suche findet seit vielen Jahren Einsatz im Bereich der kombinatorischen Optimierung [1]. Im Rahmen dieser Arbeit wurde eine Tabu-Suche zum Finden von guten Kaufentscheidungen für das Problem aus Kapitel 3 entwickelt und implementiert. Bevor diese Heuristik näher vorgestellt wird, werden zuerst einige mathematische Grundlagen der lokalen Suche eingeführt.

6.1.1 Mathematische Grundlagen

Die lokale Suche kann zum Lösen von kombinatorischen Optimierungsproblemen angewendet werden. Die in diesem Abschnitt gegebenen Definitionen stammen aus „Theoretical Aspects of Local Search“ [17].

Definition 6.1. *(Instanz eines kombinatorischen Optimierungsproblems)*
*Eine **Instanz** eines kombinatorischen Optimierungsproblems ist ein Paar (S, f), wobei S die Lösungsmenge (endlich oder abzählbar) und $f : S \to \mathbb{R}$ die Kostenfunktion des Problems ist.*

Definition 6.2. *(kombinatorisches Optimierungsproblem)*
*Ein **kombinatorisches Optimierungsproblem** Π ist eine Menge aus Instanzen (S, f) und ist entweder ein Minimierungs- oder ein Maximierungsproblem.*

In dieser Arbeit werden nur Minimierungsprobleme betrachtet. Alle Aussagen und Definitionen gelten jedoch analog für Maximierungsprobleme und können durch eine Multiplikation der Zielfunktion mit -1 erzeugt werden.

Definition 6.3. *(Global Optimal)*
*Handelt es sich bei Π um ein Minimierungsproblem, so gilt für jede Instanz (S, f), dass $s^* \in S$ genau dann ein **globales Optimum** für die Instanz ist, wenn gilt $f(s^*) \leq f(s) \forall s \in S$. Die Lösung s^* wird auch optimale Lösung von (S, f) genannt.*

Das Prinzip der lokalen Suche besteht darin, bei einer beliebigen Lösung zu starten und deren Nachbarschaft nach neuen Lösungen zu durchsuchen. Ist eine der neuen Lösungen ausgewählt, wird von ihr ausgehend eine neue Nachbarschaft durchsucht. Dieses Vorgehen wird wiederholt, bis ein Abbruchkriterium erfüllt ist sei hierfür die Nachbarschaft wie folgt definiert:

Definition 6.4. *(Nachbarschaft)*

- *Für eine Instanz eines Kombinatorischen Optimierungsproblems (S, f) wird die Abbildung*

$$N : S \to \mathscr{P}(S)$$

  ***Nachbarschaftsfunktion** genannt*
- *Für eine Lösung $s \in S$ heißt $N(s)$ **Nachbarschaft***
- *$\#N(s)$ ist die **Größe** der Nachbarschaft*
- *Eine Nachbarschaftsfunktion ist **symmetrisch**, falls gilt*

$$s' \in N(s) \Leftrightarrow s \in N(s').$$

Das Durchsuchen von Nachbarschaften und das Wechseln zu anderen Lösungen kann auch als Durchlaufen eines Graphen dargestellt werden. Dieser Graph heißt Nachbarschaftsgraph und ist wie folgt definiert:

Definition 6.5. *(Nachbarschaftsgraph)*
*Sei $G_N = (V, E)$ der **Nachbarschaftsgraph** für die Instanz (S, f) und die Nachbarschaftsfunktion N, dann gibt es für jede Lösung $s \in S$ einen Knoten v_s, d.h. $V = \{v_s : s \in S\}$. Zwei Knoten v_{s_i} und v_{s_j} sind durch eine Kante (v_{s_i}, v_{s_j}) verbunden, wenn s_j in der Nachbarschaft von s_i liegt, d.h. $E = \{(v_{s_i}, v_{s_j}) : s_j \in N(s_i)\}$.*

Bemerkung 6.1. *Für symmetrische Nachbarschaftsfunktionen N können die Kanten (v_{s_i}, v_{s_j}) und (v_{s_j}, v_{s_i}) durch eine ungerichtete Kante $\{v_{s_i}, v_{s_j}\}$ ersetzt werden.*

Definition 6.6. *(Erreichbar)*
*Eine Lösung s_j ist von der Lösung s_i aus **erreichbar**, falls es einen Weg von v_{s_i} nach v_{s_j} in G_N gibt. Es gibt somit eine Folge von Lösungen $s_1, ..., s_n$ mit $s_1 = s_i$, $s_n = s_j$ und $s_{k+1} \in N(s_k)$ $k = 1...n$.*

Definition 6.7. *(Stark zusammenhängend)*
*Der Nachbarschaftsgraph G_N heißt **stark zusammenhängend**, wenn für jedes Lösungspaar (s_i, s_j) gilt, dass der Knoten v_{s_j} von v_{s_i} aus erreichbar ist.*

Definition 6.8. *(Lokal optimal)*
*Eine Lösung $\hat{s}$ wird **lokal optimal** bezüglich N genannt, wenn gilt:*

$$f(\hat{s}) \leq f(s) \quad \forall s \in N(\hat{s}).$$

Wird ein Wechsel zu einer Nachbarschaftslösung nur dann vorgenommen, wenn sie einen besseren Zielfunktionswert hat als die aktuelle Lösung, endet die lokale Suche im ersten gefundenen lokalen Optimum. Soll dies verhindert werden, gibt es verschiedene denkbare Strategien wie in [17] Kapitel 7 vorgestellt. Zwei Klassen von Strategien können unterschieden werden: Die Strategien, die zu einem langen Suchweg durch den Nachbarschaftsgraph führen und solche, die mehrere kurze Wege im Nachbarschaftsgraphen durchsuchen. In dieser Arbeit wird die Tabu-Suche betrachtet, die ausgehend von einer Lösung zum besten Nachbarn wechselt und durch Verbote von Lösungen zu vermeiden versucht, dass zu einer bereits bewerteten Lösung zurückgekehrt wird. Die Tabu-Suche verfolgt somit die Strategie eines langen Suchweges.

6.1.2 Finden von Kaufentscheidungen

In diesem Kapitel wird folgendes kombinatorisches Optimierungsproblem betrachtet:

„Finde eine zulässige Kaufentscheidung, sodass das Betreiben und Anschaffen des Systems kostenminimal ist."

Eine Kaufentscheidung ist zulässig, wenn sie die Nebenbedingungen für Kaufentscheidungen aus Abschnitt 3.2.4 erfüllt[1]. Somit ist die Lösungsmenge S gegeben durch die Menge der ganzzahligen Punkte im Polyeder, das durch diese Nebenbedingungen erzeugt wird. Die Kosten für eine Lösung werden mit Hilfe der Dynamischen Optimierung aus Kapitel 5 ermittelt. Somit entspricht das Auswerten der Kostenfunktion f dem Lösen des Problems $\mathbf{P_0}(\mathbf{z_0} = \mathbf{a})$, d.h. dem Finden einer optimalen Politik für AG-DO1. Hat $\mathbf{P_0}(\mathbf{z_0} = \mathbf{a})$ keine Lösung, so betragen die Kosten der Lösung M, wobei M eine genügend große Zahl ist.

Nachbarschaftsfunktion

Für die Nachbarschaftsfunktion werden zuerst vier einzelne Nachbarschaften definiert und diese dann zu einer Nachbarschaft zusammengefasst.

Ersetzen von Komponenten:
Ausgehend von einem Netzwerk wird eine verbaute Komponente k_1 mit einer nicht verbauten Komponente k_2 ausgetauscht. Die neue Komponente wird an der Stelle von k_1 eingefügt. Diese Nachbarschaft wird *Replace-Nachbarschaft* oder kurz N_{Rep} genannt.

Tauschen von Komponenten:
Ausgehend von einem Netzwerk werden zwei verbaute Komponenten k_1 und k_2 vertauscht. Dabei werden die Komponenten an der Stelle der jeweils anderen eingefügt. Diese Nachbarschaft wird *Swap-Nachbarschaft* oder kurz N_{Swap} genannt.

Löschen von Komponenten:
Ausgehend von einem Netzwerk wird eine verbaute Komponente k_1 gelöscht. Für das Löschen eines Knotens aus dem Graphen werden zwei Möglichkeiten berücksichtigt:

1. Die erste Möglichkeit besteht darin den Knoten und seine ausgehenden und eingehenden Kanten aus dem Graphen zu löschen. In diesem Fall kann es passieren, dass Knoten entstehen, deren Eingangs- oder Ausgangsgrad null ist. In diesem Fall ist das erzeugte Netzwerk unzulässig und das Löschen auf diese Art kein gültiger Nachbarschaftstausch.
2. Der Rückgriff auf die andere Möglichkeit führt zur Entfernung des Knotens und der Verbindung der Vorgänger-Knoten mit Nachfolger-Knoten. Im Fall, dass ein Nachfolger-Knoten von k_1 der Vorgänger-Knoten des Vorgänger-Knotens von k_1 ist, entsteht eine Doppelkante (Hin- und Rückkante) und das Netzwerk wird unzulässig. Dann ist das Löschen kein gültiger Nachbarschaftstausch.

Diese Nachbarschaft, die durch eine der beiden Arten des Löschens entsteht, wird als *Delete-Nachbarschaft* bzw. kurz N_{Del} bezeichnet.

Hinzufügen von Komponenten:
Ausgehend von einem Netzwerk wird eine Komponente k_1 in dieses Netzwerk eingefügt. Für das Einfügen eines Knotens in einen Graphen werden drei Möglichkeiten berücksichtigt:

1. Der neue Knoten k_1 wird hinter einem bestehenden Knoten k_2 eingefügt. Dabei wird eine Teilmenge der Nachfolger des Knotens k_2 zu den Nachfolgern des Knotens k_1. Der neue Knoten k_1 wird in die Menge der Nachfolger des bestehenden Knotens k_2 aufgenommen.

[1] Die Nebenbedingungen sind für serien-parallele Netzwerke erfüllt. Ein Netzwerk muss diese Eigenschaft aber nicht besitzen, um zulässig zu sein.

2. Der Knoten k_1 wird vor einem bestehenden Knoten k_2 eingefügt. Hier wird eine Teilmenge der Vorgänger des bestehenden Knotens k_2 zu den Vorgängern des neuen Knotens k_1 und der neue Knoten k_1 wird in die Menge der Vorgänger des bestehenden Knotens eingefügt.
3. Ein Knoten k_1 wird eingefügt, indem eine beliebige Auswahl an Knoten zu Vorgängern und eine beliebige Auswahl an Knoten zu Nachfolgern von k_1 gemacht wird. Hierbei ist zu beachten, dass ein Knoten nicht sowohl zum Vorgänger als auch zum Nachfolger gemacht werden darf, um die Nebenbedingungen für die Kaufentscheidungen aus Abschnitt 3.2.4 nicht zu verletzen. Es werden zwei Mengen $V_1, V_2 \in \mathcal{P}(V)$ mit $V_1 \cap V_2 \neq \emptyset$ gewählt, wobei V die Menge aller Knoten im aktuell betrachteten Netzwerk ist. Die Knoten aus V_1 werden die Vorgänger des neu eingefügten Knotens k_1 und die Knoten aus V_2 die Nachfolger. Damit können insbesondere Komponenten parallel zu anderen Komponenten oder Teilnetzen geschaltet werden.

Die Nachbarschaft, die aus den drei Arten des Einfügens entsteht, wird **Add-Nachbarschaft**, kurz N_{Add}, genannt.

Aus diesen vier Definitionen ergibt sich die Nachbarschaftsfunktion N durch

$$N = N_{Swap} \cup N_{Rep} \cup N_{Del} \cup N_{Add}.$$

Ist die Lösung s zulässig im Sinne der Nebenbedingungen aus Abschnitt 3.2.4, dann erfüllen auch alle Lösungen aus $N(s)$ die Nebenbedingungen. Die Lösungen in den Nachbarschaften N_{Swap} und N_{Rep} entsprechen bis auf die Umbenennung der Knoten dem Graph aus der Lösung s. Beim Erstellen der Nachbarschaften N_{Add} und N_{Del} wird darauf geachtet, dass die neuen Netzwerke zulässig bleiben und somit wechselt die Tabu-Suche nie zu einer unzulässigen Kaufentscheidung.

Für die Nachbarschaftsfunktion N gelten die folgenden Aussagen.

Satz 6.1. *Für jede Lösung $s \in S$ sind die Nachbarschaften $N_{Swap}(s)$, $N_{Rep}(s)$, $N_{Del}(s)$ und $N_{Add}(s)$ paarweise disjunkt.*

Beweis. *Sei n die Anzahl an Knoten im Graphen zur Lösung s. Dann gilt für die Lösungen*

- *in $N_{Del}(s)$, dass die Graphen $n-1$ Knoten haben*
- *in $N_{Add}(s)$, dass die Graphen $n+1$ Knoten haben*
- *in $N_{Rep}(s)$, dass die Graphen n Knoten haben*
- *in $N_{Swap}(s)$, dass die Graphen n Knoten haben.*

Somit gilt N_{Swap}, N_{Del} und N_{Add} bzw. N_{Rep}, N_{Del} und N_{Add} sind paarweise disjunkt.

Die Menge an verbauten Bauteilen in $N_{Swap}(s)$ entspricht der Menge an verbauten Bauteilen in s. Dahingegen unterscheiden sich die Menge an verbauten Bauteilen in $N_{Rep}(s)$ und die Menge an verbauten Bauteilen in s genau um ein Bauteil. Somit gilt also auch, dass N_{Rep} und N_{Swap} sind paarweise disjunkt.

Satz 6.2. *Die Nachbarschaftsfunktion N ist symmetrisch, es gilt also*

$$s' \in N(s) \Leftrightarrow s \in N(s').$$

Beweis. *Die Nachbarschaftsfunktion N_{Swap} ist symmetrisch, d.h.*

$$s' \in N_{Swap}(s) \Leftrightarrow s \in N_{Swap}(s').$$

Entsteht s' aus s durch das Vertauschen der Komponenten k_1 und k_2, so lässt sich s aus s' erzeugen, indem diese beiden Komponenten wieder zurück getauscht werden.

Die Nachbarschaftsfunktion N_{Rep} ist symmetrisch, d.h.

$$s' \in N_{Rep}(s) \Leftrightarrow s \in N_{Rep}(s').$$

Entsteht s' aus s durch das Ersetzen der Komponenten k_1 durch k_2, so lässt sich s aus s' erzeugen, indem k_2 wieder durch k_1 ersetzt wird.

Die Nachbarschaftsfunktion $N_{Add} \cup N_{Del}$ ist symmetrisch, d.h.

$$s' \in N_{Add}(s) \Leftrightarrow s \in N_{Del}(s').$$

Entsteht s' aus s durch das Einfügen einer Komponenten k_1 mit V_1 und V_2 so dass die Knoten aus V_1 zu Vorgängern des neuen Knotens werden und die Knoten aus V_2 zu den Nachfolgern werden, lässt sich s aus s' erzeugen, indem k_1 mit allen inzidenten Kanten gelöscht wird.

Entsteht s' aus s, indem eine Komponente k_1 vor eine bestehende Komponente k_2 eingefügt wird und eine Teilmenge aus Vorgängern von k_2 zu Vorgängern von k_1 wird und k_1 in die Menge aus Vorgängern von k_2 aufgenommen wird, so lässt sich s aus s' erzeugen, indem die Komponente k_1 wieder gelöscht wird und alle Vorgänger von k_1 zu Vorgängern von k_2 gemacht werden.

Entsteht s' aus s, indem eine Komponente k_1 nach einer bestehenden Komponente k_2 eingefügt wird und eine Teilmenge aus Nachfolgern von k_2 zu Nachfolgern von k_1 wird und k_1 in die Menge der Nachfolger von k_2 aufgenommen wird, so lässt sich s aus s' erzeugen, indem die Komponente k_1 wieder gelöscht wird und alle Nachfolger von k_1 zu Nachfolgern von k_2 gemacht werden.

Die Argumentation, dass jede Löschoperation durch eine Einfügeoperation rückgängig gemacht werden kann, ergibt sich analog.

Satz 6.3. *Der Grad $\#N(s)$ eines Knotens s im Nachbarschaftsgraph ist abhängig von der Anzahl n an verbauten Bauteilen und erfüllt die folgenden Ungleichungen:*

$$\#N(s) \leq 2 \cdot \left(\sum_{k=0}^{\lfloor \frac{n}{2} \rfloor} \binom{n}{k} 2^k\right) \cdot \left(\sum_{k=0}^{l} \binom{l}{k}\right) + \binom{n}{2} + n(m-n) + \sum_{k=1}^{(n-1)+l} \binom{l}{k} + 2(n-1) + \sum_{k=1}^{n} \binom{n}{k}$$

$$\#N(s) \geq 2 \cdot \left(\sum_{k=0}^{\lfloor \frac{n}{2} \rfloor} \binom{n}{k} 2^k\right) \cdot \left(\sum_{k=0}^{l} \binom{l}{k}\right) + \binom{n}{2} + n(m-n) + 2.$$

wobei m die Anzahl an potentiell verbaubaren Bauteilen und l die Anzahl an Senken im Netzwerk ist.

Beweis. *Da für die einzelnen Nachbarschaften gilt, dass $N_{Swap}, N_{Exchg}, N_{Del}, N_{Add}$ paarweise disjunkt sind, gilt auch*

$$\#N(s) = \#N_{Swap}(s) + \#N_{Rep}(s) + \#N_{Del}(s) + \#N_{Add}(s).$$

Die Anzahl an Nachbarn, die durch N_{Swap} entstehen, entspricht der Anzahl an Paaren, die ohne Zurücklegen und ohne Beachtung der Reihenfolge aus der Menge an verbauten Bauteilen gezogen werden können:

$$\#N_{Swap}(s) = \binom{n}{2}.$$

Die Anzahl an Nachbarn, die durch N_{Rep} entstehen, ergibt sich aus der Anzahl an nicht verbauten Bauteilen multipliziert mit der Anzahl an verbauten Bauteilen:

$$\#N_{Swap}(s) = (m-n) \cdot n.$$

Die Anzahl an Nachbarn, die durch das Hinzufügen einer Komponente aus der Vorgängermenge V_1 und Nachfolgermenge V_2 entsteht, ergibt sich aus der Anzahl an paarweise verschiedener Teilmengen der Menge an verbauten Bauteil multipliziert mit 2 und mit $\sum_{k=0}^{l} \binom{l}{k}$. Dies ist der Fall, da jede Vorgänger Menge aus einer Teilmenge aus Komponenten und gegebenenfalls der Quelle besteht und jede Nachfolgermenge aus einer Auswahl an Bauteilen und evtl. einer Auswahl an Senken besteht.

$$2 \cdot (\sum_{k=0}^{\lfloor \frac{n}{2} \rfloor} \binom{n}{k} 2^k) \cdot (\sum_{k=0}^{l} \binom{l}{k})$$

Die Anzahl an Nachbarn, die durch das Einfügen einer Komponente k_1 vor eine bestehende Komponente k_2 entsteht, hängt vom Eingangsgrad i von k_2 ab und ist durch

$$\sum_{k=1}^{i} \binom{i}{k}$$

gegeben. Es gilt $1 \leq i \leq (n-1)+1$, da jeder Knoten mit mindestens einer eingehenden Kante inzident sein muss und die Menge an Vorgängern maximal der Menge an verbauten Komponenten außer k_2 und der Quelle entsprechen kann.

Die Anzahl an Nachbarn, die durch das Einfügen einer Komponente k_1 nach einer bestehenden Komponente k_2 entsteht, hängt vom Ausgangsgrad o von k_2 ab und ist durch

$$\sum_{k=1}^{o} \binom{o}{k}$$

gegeben. Es gilt $1 \leq o \leq (n-1)+l$, da jeder Knoten mit mindestens einer eingehenden Kante inzident sein muss und die Menge an Nachfolgern maximal der Menge an verbauten Komponenten außer k_2 und allen l Senken entsprechen kann.

Somit leiten sich folgende Ungleichungen für die Größe von N_{Add} ab:

$$\#N_{Add}(s) \leq 2 \cdot (\sum_{k=0}^{\lfloor \frac{n}{2} \rfloor} \binom{n}{k} 2^k) \cdot (\sum_{k=0}^{l} \binom{l}{k}) + \sum_{k=1}^{n} \binom{n}{k} + \sum_{k=1}^{(n-1)+1} \binom{l}{k}$$

$$\#N_{Add}(s) \geq 2 \cdot (\sum_{k=0}^{\lfloor \frac{n}{2} \rfloor} \binom{n}{k} 2^k) \cdot (\sum_{k=0}^{l} \binom{l}{k}) + 2$$

Die Anzahl an Nachbarn, die durch N_{Del} entstehen, entspricht der Anzahl an Bauteilen, die jeweils aus dem Netzwerk gelöscht werden können, ohne dass dieses unzulässig wird:

$$0 \leq \#N_{Del}(s) \leq 2(n-1).$$

Somit ergibt sich für die Größe der Nachbarschaft $N(s)$:

$$\#N(s) \leq 2 \cdot (\sum_{k=0}^{\lfloor \frac{n}{2} \rfloor} \binom{n}{k} 2^k) \cdot (\sum_{k=0}^{l} \binom{l}{k}) + \binom{n}{2} + n(m-n) + \sum_{k=1}^{(n-1)+l} \binom{l}{k} + 2(n-1) + \sum_{k=1}^{n} \binom{n}{k}$$

$$\#N(s) \geq 2 \cdot (\sum_{k=0}^{\lfloor \frac{n}{2} \rfloor} \binom{n}{k} 2^k) \cdot (\sum_{k=0}^{l} \binom{l}{k}) + \binom{n}{2} + n(m-n) + 2.$$

Satz 6.4. *Jede Lösung mit $e \leq m-1$ verbauten Bauteilen ist von jeder Lösung mit $d \leq m-1$ verbauten Bauteilen aus erreichbar.*

Beweis. *Für jedes gegebene Netzwerk werden durch N_{Swap} und N_{Rep} lediglich die Knoten umbenannt. An der Struktur des Netzwerks ändert sich nichts. Für eine gegebene Graphenstruktur lässt sich durch die beiden Tauschoperationen von N_{Swap} und N_{Rep} jede, durch die Menge an verbaubaren Komponenten gegebene, mögliche Benennung der Knoten erzeugen.*

Um zu zeigen, dass jede Lösung mit $e \leq m-1$ verbauten Bauteilen von jeder Lösung mit $d \leq m-1$ verbauten Bauteilen aus erreichbar ist, genügt es, die Menge an zulässigen Netzwerken mit $n \leq m-1$ inneren Knoten, mit ausgezeichneter Quelle und ausgezeichneten Senken und sonst unbenannten Knoten zu betrachten. In diesem Zusammenhang werden alle Knoten, die weder Quelle noch Senke sind, als innere Knoten bezeichnet.

Es muss gezeigt werden, dass sich durch die Tauschoperationen aus N_{Add} und N_{Del} jedes zulässige Netzwerk mit $e \leq m-1$ inneren Knoten in jedes zulässige Netzwerk mit $d \leq m-1$ inneren Knoten überführen lässt.

Dies wird bewiesen indem gezeigt wird, dass sich jedes Netzwerk mit l inneren Knoten, durch Löschen und Einfügen von Knoten, in ein Netzwerk mit nur einem inneren Knoten überführen lässt. Durch die Symmetrie von $N_{Add} \cup N_{Del}$ gilt dann auch, dass sich jedes Netzwerk mit e inneren Knoten, durch Löschen und Einfügen von Knoten, aus einem Netzwerk mit einem inneren Knoten erzeugen lässt. Somit lässt sich jedes Netzwerk der Ordnung d in jedes Netzwerk der Ordnung e überführen.

Bevor dies gezeigt wird, wird erläutert, wie sich durch Operationen aus $N_{Add} \cup N_{Del}$ eine Kante in einem Graph Einfügen und Löschen lässt.

Eine Kante kann zwischen zwei Knoten k_1 und k_2 eingefügt werden, indem ein Knoten k in das Netzwerk eingefügt wird und dabei k_1 zu seinem Vorgänger und k_2 zu seinem Nachfolger des neuen Knotens wird.

Anschließend wird k wieder gelöscht, indem seine Vorgänger zu den Vorgängern seiner Nachfolger werden und umgekehrt.

Eine Kante zwischen Knoten k_1 und k_2 kann gelöscht werden, indem ein Knoten k vor k_2 in das Netzwerk eingefügt wird und dabei der Knoten k zum Nachfolger von k_1 wird und k in die Menge aller Vorgänger von k_2 aufgenommen wird. Anschließend wird k mit allen inzidenten Kanten gelöscht.

Wird im Folgenden vom Einfügen und Löschen gesprochen, ist das eben beschriebene Vorgehen gemeint.

Ein Netzwerk mit d inneren Knoten lässt sich in ein Netzwerk mit einem inneren Knoten überführen indem die folgenden vier Schritte befolgt werden:

1. *Schritt: Füge zwischen die Quelle und jeden inneren Knoten eine Kante ein, falls diese nicht bereits im Netzwerk enthalten ist.*
2. *Schritt: Füge zwischen jeden inneren Knoten und jede Senke eine Kante ein, falls diese nicht bereits im Netzwerk enthalten ist.*
3. *Schritt: Lösche alle Kanten, die weder mit einer Quelle noch mit einer Senke inzident sind.*
4. *Schritt: Lösche $d-1$ beliebige Knoten mit allen inzidenten Kanten.*

Bemerkung 6.2. *Ein Netzwerke mit m verbauten Bauteilen ist von einem Netzwerk mit $e \leq m-1$ inneren Knoten aus erreichbar, falls mindestens ein Bauteil aus diesem Netzwerk gelöscht werden kann, ohne dass das Netzwerk unzulässig wird. Ein Knoten kann aus dem Netzwerk gelöscht werden, falls eine der drei folgenden Bedingungen erfüllt ist.*

1. *Fall: Der Knoten k, hat keinen inneren Knoten als Vorgänger oder Nachfolger mit Eingangs- bzw. Ausgangsgrad eins , dann kann k mit allen inzidenten Kanten gelöscht werden.*
2. *Fall: Der Knoten k, hat einen Vorgängerknoten k_1 mit Ausgangsgrad eins, dessen Vorgänger alle keine Vorgänger von k sind, dann lässt sich k_1 löschen, indem seine Vorgänger zu Vorgängern von k werden.*
3. *Fall: Der Knoten k, hat einen Nachfolgerknoten k_2 mit Eingangsgrad eins, dessen Nachfolger alle keine Nachfolger von k sind, dann lässt sich k_2 löschen, indem seine Nachfolger zu Nachfolgern von k werden.*

6.2 Ameisenalgorithmus

In diesem Abschnitt wird ein Ameisenalgorithmus vorgestellt, der Systemaufbauten erzeugt, die aus einer seriellen und parallelen Verschaltung von Speichern und Pumpen bestehen.

Der Ameisenalgorithmus imitiert das Verhalten einer Ameisenkolonie auf der Futtersuche. Die ersten Ameisen durchsuchen die Umgebung des Baus noch zufällig. Findet eine Ameise eine Futterquelle, markiert sie den Weg zurück zum Bau mit einer Pheromonspur. Nachfolgende Ameisen lassen sich bei ihrer Suche von den Pheromonspuren leiten. Je intensiver eine Pheromonspur auf einem Weg ist, desto höher ist die Wahrscheinlichkeit, dass eine Ameise diesem Weg folgt.

6.2.1 Lösungssuche

Sind zwei Pumpen parallel zueinander geschaltet, so ergibt sich die Fördermenge des Systems aus der Summe der Fördermengen beider Komponenten. Die Druckerhöhung dieses Systems entspricht der Druckerhöhung der einzelnen Komponenten. In einer Reihenschaltung (seriell) hingegen ist die Druckerhöhung des Systems die Summe aus beiden Druckerhöhungen. Die Fördermenge dieses Systems entspricht der Fördermenge der einzelnen Komponenten. Das Gleiche ist zu beobachten, wenn Systeme aus Komponenten parallel oder seriell verschaltet werden. Da sich beide Effekte in serien-parallelen

Systemen kombinieren lassen und somit gleichzeitig der Druck als auch die Fördermenge des Systems erhöht werden können, ist ein serien-paralleler Systemaufbau eine intuitive Lösung.

Ein Wasserversorgungssystem mit seriell und parallel verschalteten Komponenten und Teilnetzen lässt sich als serien-paralleles Netzwerk $G = (V, E)$ darstellen. In diesem Fall entsprechen - wie in Abschnitt 3.1 eingeführt - die Hauptknoten des Netzwerkes Speicher- oder Pumpenknoten. Der Grad des Netzwerkes entspricht somit der Anzahl an verbauten Speichern und Pumpen. Sind zwei Hauptknoten v_1 und v_2 über einen Kopplungsknoten k verbunden, d.h. es existieren Kanten (v_1, k) und (k, v_2) in E, bedeutet dies im realen System, dass die Rohre der zugehörigen Komponenten miteinander verschaltet sind (vgl. Bedeutung von v_1, v_2 aus Kapitel 3.1). Die Quelle und Senke des Netzwerkes stellen - wie in 3.1 - den Wasserzu- bzw. Ablauf des Systems dar. Werden Wasserversorgungsysteme mit mehreren Senken betrachtet, z.B. mit mehreren räumlich verteilten Verbrauchern, muss die Senke des Netzwerkes aufgespalten werden.

In Kapitel 4 wurde gezeigt, dass sich sp-Netzwerke als Baum darstellen lassen. Nach der Vorentscheidung, ob ein serielles oder paralleles Netzwerk erzeugt werden soll, „suchen“ die Ameisen iterativ und von einer Wahrscheinlichkeitsverteilung geleitet solche Bäume. Daraufhin werden den Blättern des gefundenen Baumes Pumpen bzw. Speicher zugeordnet. Zum Schluss wird dann die Senke aufgespaltet, um einen Systemaufbau für das zu planende System zu erhalten.

Festlegen der Netzwerkart

Die erste Entscheidung die getroffen werden muss, ist ob der Baum seriell oder parallel interpretiert wird. Die Wahrscheinlichkeiten für die nächsten Schritte der Suche hängen von dieser Entscheidung ab.

Sei $X_{ser} \in \{0, 1\}$ die Variable, die die Art der Interpretation angibt. Es gilt, ist $X_{ser} = 1$, dann wird seriell interpretiert. Ist dagegen $X_{ser} = 0$, wird parallel interpretiert.

Für jede Interpretationsweise gibt es einen Pheromonwert w_{ser} bzw. w_{par}, aus dem sich ableiten lässt, mit welcher Wahrscheinlichkeit eine Ameise sich für diese Interpretation entscheidet. Die Wahrscheinlichkeiten sind durch folgende Gleichungen gegeben [7]:

$$p_{ser} = \frac{w_{ser}}{w_{ser} + w_{par}}$$
$$p_{par} = \frac{w_{par}}{w_{ser} + w_{par}}.$$

Iterativer Aufbau eines Baums

Nachdem die Art des Netzwerkes bestimmt ist, wird der Baum aufgebaut. Die Anzahl an Blättern im Baum ist maximal so groß wie die Anzahl an verbaubaren Komponenten, die durch m gegeben ist. Jedem Knoten im Baum wird eine eindeutige ID *kID* zugeordnet, die sich aus der Stufe s ergibt, auf dem sich der Knoten befindet, der ID des Vaterknotens *vID* und der Position j, in der sich der Knoten in der Nachfolgerliste des Vaterknotens befindet.

$$kID(s, vID, j) = \sum_{i=1}^{s} m^i + m \cdot (vID - ((s-1) \cdot m)) + j$$

Die ID des Wurzelknotens ist 0. Die IDs der Knoten im Netzwerk entsprechen den IDs der Blätter im Baum.

Jede Ameise beginnt mit einem Baum, der nur aus der Wurzel besteht, und würfelt die Anzahl an Nachfolgern. Es ist zu beachten, dass jeder Knoten entweder keine oder mindestens zwei Nachfolger hat, da in Kapitel 4 die Interpretation von Bäumen mit Knoten von einem Grad größer als 2 als serien-parallele Netzwerke vorgestellt wurde. Für den Wurzelknoten ist, abhängig von der Netzwerkinterpretation, ein Pheromonwert $w_n(X_{ser})$ für jede mögliche Anzahl n an Nachfolgern gegeben, aus dem sich die Wahrscheinlichkeit für diese Anzahl errechnen lässt.

$$p_n(X_{ser}) = \frac{w_n(X_{ser})}{\sum_{j=1}^{m} w_j(X_{ser})} \quad \forall i = 1...m$$

Der gewürfelten Anzahl entsprechend werden Knoten in die Nachfolgerliste der Wurzel eingefügt. Ihre IDs ergeben sich zu $kID(1,0,i)\ \forall i = 1....n$. Entspricht n der Anzahl an maximal verbaubaren Komponenten m, dann können keine weiteren Knoten mehr in den Baum eingefügt werden und die Ameise beendet diesen Schritt der Suche. Im Fall, dass die ausgewählte Anzahl an Nachfolgern kleiner ist als die Anzahl m an zu verbauenden Komponenten, wird für jeden neuen Knoten die Anzahl an Nachfolgern bestimmt. Wie viele Nachfolger maximal für einen Knoten i ausgewählt werden, hängt ab von der Anzahl h_i an Nachfolgern, die den Knoten vor i bereits zugeordnet wurden, der Anzahl l_i an Knoten, denen vor dem Knoten i eine positive Anzahl an Nachfolgern zugeordnet wurde, und schließlich von n.

$$m_{Rest}(i) = m - n + l_i - h_i \quad \forall i = 1...n$$

Für die Auswahl an Nachfolgerknoten für die Kinder der Wurzel hängen auch die Pheromonwerte w_i, die zur Anzahl i gehören, von X_{ser} ab. Die Wahrscheinlichkeit ergibt sich aus den Pheromonwerten:

$$p_i(X_{ser}) = \frac{w_i(X_{ser})}{\sum_{j=1}^{m_{Rest(i)}} w_j(X_{ser})} \forall i = 1...m_{Rest}(i).$$

Nachdem für alle neuen Knoten die Anzahl an Nachfolgern bestimmt wurde, werden je Knoten der Anzahl entsprechend neue Knoten in den Baum eingeführt. Knoten, die keine Nachfolger haben, werden zu Blättern im Baum. Für die neuen Knoten wird wieder die Anzahl an Nachfolgern bestimmt. Die maximale Anzahl an Nachfolgern berechnet sich wie oben. Für jeden Knoten *kID* existiert für jede Anzahl an Nachfolgern ein Pheromonwert $w_i^{kID}, (X_{ser})$ aus dem sich die Wahrscheinlichkeit für eine bestimmte Anzahl an Nachfolgern bestimmen lässt.

$$p_i^{kID}(X_{ser}) = \frac{w_i^{kID}(X_{ser})}{\sum_{j=1}^{m_{Rest(i)}} w_j^{kID}(X_{ser})} \forall i = 1...m_{Rest}(i)$$

Die neuen Knoten werden wieder in den Baum eingeführt. Auch für diese wird wieder die Anzahl an Nachfolgern bestimmt. Diese Vorgehensweise wird so lange wiederholt, bis $m_{Rest} \leq 1$.

Zuordnung von Knoten zu Komponenten

Den Blättern der gefundenen Baumstruktur werden Pumpen und Speicher zugeordnet. Dabei wird jedem Blatt *kID* genau eine Pumpe oder ein Speicher zugeordnet. Alle Knoten besitzen für jede Pumpe bzw. jeden

Speicher b einen Pheromonwert w_b^{kID}, der die Wahrscheinlichkeit bestimmt, mit der eine Komponente dem Knoten zugeordnet wird. Auch hier gilt wieder, dass die Wahrscheinlichkeit von X_{ser} abhängig ist.

$$p_f^{kID}(X_{ser}) = \frac{w_b^{kID}(X_{ser})}{\sum_{j=1}^{m} w_j^{kID}(X_{ser})} \quad \forall b = 1, ..., m.$$

Zuordnung von Komponenten zu Senken

Wird der Baum und dessen Bauteilzuordnung als realer Systemaufbau interpretiert, hat das entstandene Wasserversorgungssystem nur eine Senke. Im Fall von Versorgungssystemen mit r Senken, muss die Senke des sp-Netzwerkes geteilt werden. Auch in diesem Fall wird auf ein Zufallsexperiment zurückgegriffen.

Für alle Knoten, die im sp-Netzwerk mit der Senke verbunden sind, wird ausgewürfelt, mit welcher realen Senke s dieser Knoten verbunden ist. Die Wahrscheinlichkeit für eine Zuordnung ist wie alle bisherigen Wahrscheinlichkeiten von X_{ser} abhängig. Zusätzlich wird die Wahrscheinlichkeit des Knotens von der zugeordneten Komponente beeinflusst. Die Pheromonwerte für eine Knoten-Senke-Zuordnung sind durch $w_s^{kID}(X_{ser}, k)$ gegeben. Somit ergibt sich die Wahrscheinlichkeit für eine Zuordnung:

$$p_s^{kID}(X_{ser}) = \frac{w_s^{kID}(X_{ser}, b)}{\sum_{j=1}^{m} w_s^{kID}(X_{ser}, b)} \forall s = 1...r \text{ , wobei } b \text{ } kID \text{ zugeordnet ist.}$$

6.2.2 Pheromonupdate

Zu Beginn des Ameisenalgorithmus werden alle oben eingeführten Pheromonwerte mit einem Startwert initialisiert. Nachdem eine gegebene Anzahl an Ameisen (Population P) ausgewertet wurde, werden diese Werte aktualisiert (Pheromonupdate). Hierfür werden zuerst alle Pheromonwerte mit $(1 - \rho)$ multipliziert, um das Verdunsten der Pheromonspur in der Realität nachzubilden. In diesem Kontext wird ρ der Verdunstungsfaktor genannt.

$$w \leftarrow w \cdot (1 - \rho)$$

Anschließend sondern alle Ameisen a, die ein Netzwerk gefunden haben, mit dem die Deckung der Nachfrage möglich war, Pheromone ab. Wie viel Pheromon abgesondert wird, hängt vom Zielfunktionswert f_a der Ameise ab.

Sei hierfür:

B_a	der Baum der Ameise
E_a	die Blätter im Baum bzw. Knoten im Netzwerk
$n_a(k)$	die Anzahl an Nachfolgern, die dem Knoten k zugeordnet sind
$n_a(0)$	die Anzahl an Nachfolgern, die der Wurzel zugeordnet sind
$b_a(k)$	das Bauteil, das dem Knoten k zugeordnet ist
$S_a(k)$	die Menge aus Senken, die dem Knoten k zugeordnet sind. Ist k nicht mit der Netzwerksenke verbunden, so gilt $S_a(k) = \emptyset$.

Wie in [7] vorgeschlagen wird eine Funktion $F(a)$ eingeführt, für die gilt:

$$F(a_1) < F(a_2) \Leftrightarrow f_{a_1} > f_{a_2}.$$

Die Pheromonwerte der einzelnen Entscheidungen, die zum Finden der Lösung von a geführt haben, werden um $F(a)$ erhöht. Es gilt somit, je niedriger der Zielfunktionswert ist, desto stärker wird der Pheromonwert erhöht.

Mit $F(a) = \frac{c}{f_a}$ für ein $c > 0$ ergeben sich die folgenden neuen Pheromonwerte:

$$\begin{aligned}
w_{ser} &\leftarrow w_{ser} + X_{ser}\frac{c}{f_a} \\
w_{par} &\leftarrow w_{par} + (1 - X_{ser})\frac{c}{f_a} \\
w_{n_a(0)}(X_{ser}) &\leftarrow w_{n(0)}(X_{ser}) + \frac{c}{f_a} \\
w^k_{n_a(k)}(X_{ser}) &\leftarrow w_{n(k)}(X_{ser}) + \frac{c}{f_a} \forall k \in E_a \\
w^k_{f_a(k)}(X_{ser}) &\leftarrow w^k_{f_a(k)}(X_{ser}) + \frac{c}{f_a} \forall k \in E_a \\
w^k_s(X_{ser}, f_a(k)) &\leftarrow w^k_s(X_{ser}, f_a(k)) + \frac{c}{f_a} \forall k \in E_a,\ s \in S_a(k).
\end{aligned}$$

Aus der in diesem Abschnitt beschriebenen Vorgehensweise ergibt sich der nachfolgende Ablauf für den Ameisenalgorithmus.

Algorithmus 5 Ameisenalgorithmus zum Finden einer Kaufentscheidung

```
1:  Setze initiale Pheromonwerte
2:  while Abbruchkriterium nicht erfüllt do
3:      for a ∈ P do
4:          X_ser ← Würfel(w_ser, w_par)
5:          B_a ← SucheBaum(X_ser)
6:          b_a ← SucheBauteilZuordnung(X_ser)
7:          S_a ← SucheSenkenZuordnung(X_ser)
8:          f_a ← Bewerte(a)
9:      end for
10:     AktualisierePheromonwerte(P)
11: end while
```

7 Duale Schranken

In diesem Kapitel werden zwei duale Schranken für die Lösung des Problems aus Abschnitt 3.2 vorgestellt, welche zum einen die zeitliche Kopplung und zum anderen die Pumpenkennfelder relaxieren. Duale Schranken können zum einen dazu verwendet werden, Teile des Lösungsraumes auszuschließen und zum anderen dazu, die Güte einer gefundenen Lösung anzugeben. In dieser Arbeit werden die dualen Schranken verwendet um die Güte der vom Ameisenalgorithmus und der Tabu-Suche gefundenen Lösungen abzuschätzen.

Die Idee der dualen Schranken ist Variablenbelegungen zu finden, die nicht zwangsläufig zulässig sind, aber einen besseren Zielfunktionswert als die optimale Lösung liefern. Hierzu können einige Nebenbedingungen der Probleme gelockert werden. Das Ziel bei der Lockerung von Nebenbedingungen ist, ein einfacherer zu lösendes Problem zu erhalten. Bei der Vereinfachung ist zu beachten, dass die Lösungsmenge des Originalproblems eine Teilmenge der Lösungsmenge des relaxierten Problems ist. Dies bedeutet, dass alle zulässigen Lösungen im relaxierten Problem noch zulässig sind. Der Zielfunktionswert des relaxierten Problems wird *duale Schranke* des Originalproblems genannt.

Die Tatsache, dass der Lösungsraum des Originalproblems im Lösungsraum der Relaxierung enthalten ist, führt dazu, dass die duale Schranke besser als der Zielfunktionswert des unrelaxierten Problems ist. Für Minimierungsprobleme gilt, dass die Zielfunktionswerte jeder zulässigen Lösung größer gleich dem optimalen Zielfunktionswert des Originalproblems sind und dieser wiederum größer gleich der dualen Schranke ist.

Ist eine duale Schranke für ein Optimierungsproblem gegeben, kann für jeden zulässigen Punkt angegeben werden, wie stark er maximal vom Optimum abweicht. Die prozentuale Abweichung wird auch *Optimalitätslücke* genannt. Je näher die duale Schranke am Optimum des Originalproblems liegt, desto näher liegt die Optimalitätslücke an der tatsächlichen Abweichung vom Optimum. Aus diesem Grund wird nach möglichst hohen dualen Schranken gesucht.

7.1 Relaxierung der Zeitkopplung

Im MIP aus Abschnitt 3.2 sind die einzelnen Zeitschritte durch die Füllstände der Speicher gekoppelt. So ist der Anfangsfüllstand eines Zeitschritts der Endfüllstand des vorherigen Zeitschritts. Die Kopplung der Zeitschritte wird gelockert, indem für jeden Zeitschritt ein eigener Anfangs- und Endfüllstand eingeführt wird. Für diskrete Füllstände werden zusätzliche Binärvariablen für die Auswahl des jeweiligen Füllstandes benötigt:

$st^t_{Start,i} \geq 0$	Anfangsfüllhöhe des Speichers i im Zeitschritt t
$st^t_{End,i} \geq 0$	Endfüllhöhe des Speichers i im Zeitschritt t
$s^t_{Start,i,n} \in \{0,1\}$	ausgewählter Anfangsfüllstand n für den Speicher i im Zeitschritt t
$s^t_{End,i,n} \in \{0,1\}$	ausgewählter Endfüllstand n für den Speicher i im Zeitschritt t.

In allen Nebenbedingungen des Modells werden die Variablen st_i^{t-1} und st_i^t durch die neu eingeführten Variablen $st_{Start,i}$ bzw. $st_{End,i}$ ersetzt. Somit ergeben sich die folgenden neuen Nebenbedingungen:

$$\Delta_T \cdot (Q^t_{In,i} - Q^t_{Out,i}) + G_i st^t_{Start,i} = \quad G_i st_{End,i} \quad \forall i \in Sp,\ t = 1...T$$

und im Fall mit diskreten Füllständen:

$$\sum_{n=1...N_i} s^t_{Start,i,n} \cdot p_{i,n} = st^t_{Start,i} \quad \forall i \in P,\ t = 1...T \tag{7.1}$$

$$\sum_{n=1...N_i} s^t_{Start,i,n} = x_i \quad \forall i \in P,\ t = 1...T \tag{7.2}$$

$$\sum_{n=1...N_i} s^t_{End,i,n} \cdot p_{i,n} = st^t_{End,i} \quad \forall i \in P,\ t = 1...T \tag{7.3}$$

$$\sum_{n=1...N_i} s^t_{End,i,n} = x_i \quad \forall i \in P,\ t = 1...T. \tag{7.4}$$

Das so entstandene MIP ist eine Relaxierung des Originalproblems. Allerdings kann nun Wasser aus dem Speicher entnommen werden, das nie in den Speicher gefüllt wurde. Es ist somit möglich, trotz begrenzten Wasserangebotes beliebig viel Wasser zu verwenden. Des Weiteren gelangt Wasser in den Speicher, ohne dass eine Pumpe dafür betrieben werden muss, d.h. ohne Stromkosten zu verursachen. Die aus diesem Problem entstehende duale Schranke ist unter Umständen wesentlich kleiner als der optimale Zielfunktionswert. Aus diesem Grund wird eine zusätzliche Nebenbedingung eingeführt, die die Zeitschritte lose koppelt:

$$\sum_{t=1}^{T} \sum_{(j,i) \in V} Q^t_{In,i} \geq \sum_{t=1}^{T} \sum_{(i,j) \in V} Q^t_{Out,i} \quad \forall i \in Sp. \tag{7.5}$$

Die Nebenbedingung fordert, dass die Summe des über den ganzen Zeitraum in den Speicher strömenden Volumenstroms größer gleich dem über den ganzen Zeitraum aus dem Speicher strömenden Volumenstrom sein muss. Sie stellt somit sicher, dass nur Wasser verbraucht wird, das durch die Quelle in das System gelangt ist und außerdem, dass das gesamte aus dem Speicher entnommene Wasser auch in einem beliebigen Zeitschritt in den Speicher geflossen ist.

7.2 Relaxierung der Pumpen-Kennfelder

Ein großer Anteil an Binärvariablen im MIP aus 3.2 stammt aus der Linearisierung der Pumpenkennfelder. Eine Vereinfachung des Problems aus 3.2 kann somit durch die Relaxierung der Linearisierung erreicht werden. Dazu werden zwei Hyperebenen eingeführt; die eine überschätzt die Druckerhöhung, die von der Pumpe erzeugt wird und die andere unterschätzt die Leistungsaufnahme. Die Pumpen können folglich eine größere Druckerhöhung erzeugen als im Kennfeld angegeben ist und verbrauchen dabei weniger Strom.

Überschätzen der Druckerhöhung

Die Hyperebene, die die Druckerhöhung einer Pumpe i überschätzt, sei durch $H_i(Q, n) = a_i Q + b_i n + c_i$ gegeben. Für alle Stützstellen der Linearisierung gilt, dass sie unter der Ebene liegen. Die Bedingung, die den Zusammenhang zwischen dem Volumenstrom, der Drehzahl und der Druckerhöhung relaxiert, ist dann

$$H_i^t \geq a_i Q_i^t + b_i n_i^t + c_i \quad \forall i \in B, t = 1...T. \tag{7.6}$$

Zusätzlich soll sichergestellt werden, dass für verwendete Pumpen die Volumenströme Q und Drehzahlen n nur Werte annehmen, die sie auch in der Linearisierung angenommen hätten und dass die Druckerhöhung nicht kleiner ist als der minimale Wert, den sie durch die Linearisierung annehmen kann, d.h.

$$\bar{n}_{min,i} \leq n \leq \bar{n}_{max,i} \tag{7.7}$$
$$Q_i^t \leq \bar{Q}_{max,i} \tag{7.8}$$
$$\bar{H}_{min,i} \leq H_i^t \quad , \tag{7.9}$$

wobei

- $\bar{Q}_{max,i}$ den maximalen Volumenstrom angibt, der durch die Linearisierung entstehen kann,
- $\bar{n}_{max,i}$ die maximalen Drehzahl angibt, die durch die Linearisierung erreicht werden kann,
- $\bar{n}_{min,i}$ die minimale Drehzahl angibt, die durch die Linearisierung erreicht werden kann,
- $\bar{H}_{min,i}$ die minimale Druckerhöhung angibt, die durch die Linearisierung erreicht werden kann.

Es ist zu beachten, dass die Drehzahl die Grenzen aus der Linearisierung nur einhalten muss, falls die Pumpe verwendet wird. Somit werden die Schranken der Drehzahl durch die folgenden Bedingungen ersetzt:

$$n_i^t \leq \bar{n}_{max,i} + (1 - y_i^t)M \tag{7.10}$$
$$n_i^t \geq \bar{n}_{min,i} + (y_i^t - 1)M. \tag{7.11}$$

Für die Druckerhöhung der Pumpe muss gelten, dass sie größer als die minimale Druckerhöhung aus der Linearisierung ist, wenn die Pumpe selbst verwendet ist. Somit wird die untere Grenze der Druckerhöhung durch die Nebenbedingung

$$H_i^t \geq \bar{H}_{min,i} + (y_i^t - 1)M \tag{7.12}$$

ersetzt.

Werden Kreiselpumpen betrachtet, so gilt laut den Skalierungsgesetzen (vgl Kapitel **??**), dass die Druckerhöhung H mit der Drehzahl n steigt und sich dabei wie n^2 verhält. Zusätzlich gilt, dass die Druckerhöhung bei fester Drehzahl eine konkave Funktion des Volumenstroms ist. Somit lässt sich eine überschätzende Hyperebene durch die folgenden drei Punkte bestimmen: Die beiden Stützpunkte der maximalen Drehzahl mit den zwei größten Druckerhöhungen und den Stützpunkt der minimalen Drehzahl mit der größten Druckerhöhung.

Unterschätzen der Leistungsaufnahme

Die Hyperebene, die die Leistungsaufnahme der Pumpe i unterschätzt, sei durch $P(Q,n) = d_i Q + e_i n + f_i$ gegeben. Alle Stützstellen der Linearisierung müssen über dieser Ebene liegen. Die Bedingung, die den Zusammenhang zwischen dem Volumenstrom, der Drehzahl und der Leistungsaufnahme relaxiert, ist dann

$$P_i^t \leq d_i Q_i^t + e_i n_i^t + f_i \quad \forall i \in B, t = 1...T. \tag{7.13}$$

Wird eine Pumpe nicht verwendet, so ist der Volumenstrom null. Das Gleiche muss auch für die Leistungsaufnahme gelten. Da P in der Zielfunktion liegt und nur durch die Nebenbedingung der Relaxierung bestimmt wird, ist es ausreichend, für den Fall einer nicht verwendeten Pumpe zu ermöglichen, dass P den Wert null annimmt. Die Nebenbedingung 7.13 stellt dies bereits sicher, da für nicht verwendete Pumpen $Q_i^t = 0$ gilt und n_i^t entsprechend gewählt werden kann.

Werden erneut Kreiselpumpen betrachtet, so gilt laut den Skalierungsgesetzen, dass die Leistungsaufnahme P mit der Drehzahl n steigt und sich dabei wie n^3 verhält. Die Leistungsaufnahme bei fester Drehzahl beschreibt eine konkave Funktion des Volumenstroms. Somit lässt sich eine unterschätzende Hyperebene aus den folgenden drei Punkten erzeugen: Die äußeren beiden Stützstellen der niedrigsten Drehzahl und die Stützstelle der zweit-kleinsten Drehzahl mit geringster Leistungsaufnahme.

8 Auswertung

In diesem Kapitel wird das Verhalten und die Lösungsgüte der in dieser Arbeit vorgestellten Heuristiken untersucht. Hierfür wird zuerst die Datengrundlage der Testinstanzen beschrieben. Aufbauend auf diese Grundlage wird untersucht, wie sich die Beschaffenheit der Daten auf die Laufzeit der Dynamischen Optimierung auswirkt. Anschließend wird die Lösungssuche der lokalen Suche mit der des Solvers cplex [12] verglichen.

8.1 Testinstanzen

Ein Bürger in Deutschland verbraucht am Tag durchschnittlich 121 Liter Wasser [19], womit er auf einen durchschnittlichen Jahresverbrauch von 44165 Litern kommt. Die Zusammensetzung des durchschnittlichen Tagesverbrauchs ist in der Tabelle 8.1 abgebildet.

Als Testinstanzen für die vorgestellten Heuristiken werden Siedlungen mit privaten Haushalten angenommen. Hierfür werden vier verschiedene beispielhafte Haushalte konstruiert: Eine Familie mit zwei Kindern und mit zwei berufstätigen Eltern; Eine Familie mit zwei Kindern und nur einem berufstätigen Elternteil; Ein Einpersonen-Haushalt und ein Zweipersonen-Haushalt, in denen jeweils alle berufstätig sind. Für jeden dieser Haushalte werden die Ganglinien des Wasserverbrauchs aufgrund des durchschnittlichen Wasserverbrauchs eines Einwohners Deutschlands und einem angenommenen Verhalten geschätzt. Dabei wird das folgende angenommen:

- Es gibt 122 Sommertage und 243 sonstige Tage.
- Im Sommer wird jeden Tag der Garten bewässert. Hierfür werden jeweils 7.2 l Wasser benötigt.
- Im Sommer duscht jede Person einmal am Tag, sonst nur alle eineinhalb Tage; Beim Duschen werden 46 l Wasser verbraucht.
- Jede Person benötigt pro Tag 9.6 l Wasser für sonstige Körperpflege.
- Jede Person geht dreimal am Tag auf die Toilette und verbraucht bei jeder Spülung 12 l Wasser.
- Jede Person benötigt zum Trinken und zum Zubereiten des Essens 4.4 l Wasser pro Tag. Diese verteilen sich auf Frühstück (25 %) und eine bzw. zwei weitere Mahlzeiten (75 % bzw. 25 %, 50 %).
- Das Betreiben der Waschmaschine verbraucht 74 ℓ Wasser. Die Spülmaschine verbraucht 32 l.
- Für das Putzen eines Haushaltes werden pro Person 39 l Wasser benötigt.

Tabelle 8.1: Prozentuale Zusammensetzung des durchschnittlichen Tagesverbrauches

Tätigkeit	durchschnittlicher Verbrauch(%)
Toilettenspülung	29.7
Duschen/Körperpflege	39.6
Essen/Trinken	4.4
Spülmaschine	6.6
Waschmaschine	13.2
Putzen/Garten	6.6

Familie mit zwei Kindern und zwei berufstätigen Elternteilen

Bei einer Familie mit zwei Kindern und zwei berufstätigen Eltern werden die Morgenstunden eines Tages für Körperpflege und Duschen verwendet. Die Toilette wird über den ganzen Tag verteilt benutzt. An den Abenden wird gekocht und anschließend die Spülmaschine angeschaltet. An den Wochenenden wird auch mittags gekocht, sowie die Hausarbeiten erledigt. Dabei werden Freitag, Samstag und Sonntag jeweils zwei Maschinen Wäsche gewaschen und samstags wird geputzt. Daraus ergeben sich die geschätzten Wochenganglinien für einen Sommermonat und einen sonstigen Monat, die auf der beigelegten CD zu finden sind.

Familie mit zwei Kindern und einem berufstätigen Elternteil

Bei einer Familie mit zwei Kindern und einem berufstätigen Elternteil werden die Morgenstunden jedes Tages für Duschen und Körperpflege verwendet. Die Toilette wird über den ganzen Tag verteilt genutzt. An den Abenden wird gekocht und die Spülmaschine anschaltet. An den Wochenenden wird auch mittags gekocht. Die Hausarbeit wird auf die Woche verteilt erledigt. Daraus ergeben sich die geschätzten Wochenganglinien für einen Sommermonat und einen sonstigen Monat, die auf der beigelegten CD zu finden sind.

Zweipersonen-Haushalt

In einem Haushalt mit zwei berufstätigen Erwachsenen werden auch die Morgenstunden jedes Tages für Duschen und Körperpflege verwendet. Die Toilette wird tagsüber nicht benutzt. An den Abenden wird gekocht. Am Wochenende wird zusätzlich auch mittags gekocht. Die Spülmaschine wird alle zwei Tage verwendet. Die anfallende Hausarbeit wird am Wochenende erledigt. Freitags wird eine Maschine und samstags zwei Maschinen Wäsche gewaschen. An diesem Tag wird auch geputzt. Daraus ergeben sich die geschätzten Wochenganglinien für einen Sommermonat und einen sonstigen Monat, die auf der beigelegten CD zu finden sind.

Einpersonen-Haushalt

In einem Haushalt mit einem berufstätigen Erwachsenen werden erneut die Morgenstunden jedes Tages für Duschen und Körperpflege verwendet. Die Toilette wird tagsüber nicht benutzt. An den Abenden wird gekocht und am Wochenende wird zusätzlich auch mittags gekocht. Die Spülmaschine wird alle vier Tage angeschaltet. Die anfallende Hausarbeit wird am Wochenende erledigt. Samstags wird abwechselnd eine bzw. zwei Maschine Wäsche gewaschen und geputzt. Daraus ergeben sich die geschätzten Wochenganglinien für einen Sommermonat und einen sonstigen Monat, die auf der beigelegten CD zu finden sind.

Druckzonen

Die Haushalte werden zu Druckzonen zusammengefasst, welche die Abnehmer für das zu planende System darstellen. Es werden drei beispielhafte Druckzonen betrachtet.

- Druckzone 1:
  - eine Familie mit zwei berufstätigen Elternteilen
  - eine Familie mit nur einem berufstätigen Elternteil
  - zwei Zwei-Personen Haushalte
  - zwei Ein-Personen Haushalte
- Druckzone 2:

  - zwei Familien mit zwei berufstätigen Elternteilen
  - zwei Familien mit nur einem berufstätigen Elternteil
  - ein Zwei-Personen Haushalt
  - ein Ein-Personen Haushalt
- Druckzone 3:
  - zwei Familien mit zwei berufstätigen Elternteilen
  - zwei Familien mit nur einem berufstätigen Elternteil
  - zwei Zwei-Personen Haushalte
  - zwei Ein-Personen Haushalte

Für jede dieser Druckzonen wird angegeben wie weit oberhalb sie vom Pumpenhaus liegen und wie lang das Rohr ist, das sie mit dem Pumpenhaus verbindet. Aus diesen Angaben lässt sich mit Hilfe der Bernoullischen Gleichung (vgl. 2.1(2.3)) bestimmen, wie hoch der Druck H_1 am Anfang des Rohres vom Pumpenhaus zum Abnehmer sein muss, damit der gewünschte Volumenstrom beim Abnehmer ankommt. Das Wasser hat beim Abnehmer Atmosphärendruck, welcher dem Druck von 10 mWS entspricht. Dabei werden Rohre betrachtet, deren Rohrreibungszahl $\lambda = 0.001$ und deren Durchmesser 15cm ist.

Die Druckzone 1 liegt 37 m oberhalb des Pumpenhauses und ist mit einem 200 m langen Rohr an das Pumpenhaus angeschlossen. Daraus ergibt sich für den Druck H^1_{D1}:

$$H_1^{D1}(Q) = 47 + 0.2Q^2.$$

Die Druckzone 2 liegt 37 m oberhalb des Pumpenhauses und ist mit einem 130 m langen Rohr an das Pumpenhaus angeschlossen. Daraus ergibt sich für den Druck H_1^{D2}:

$$H_1^{D2}(Q) = 47 + 0.13Q^2.$$

Die Druckzone 3 liegt 32 m oberhalb des Pumpenhauses und ist mit einem 120 m langen Rohr an das Pumpenhaus angeschlossen. Daraus ergibt sich für den Druck H_1^{D3}:

$$H_1^{D3}(Q) = 42 + 0.12Q^2.$$

Speicherstandorte

Es werden nur Systeme mit einem Speicher betrachtet. Dieser Speicher wird außerhalb des Pumpenhauses aufgestellt. Es sind drei unterschiedliche Speicherstandorte gegeben. Aus den Standorten und der Länge der Rohre, die mit den Speicher mit dem Pumpenhaus verbinden, lässt sich berechnen, wie hoch der Druck H_1 sein muss, damit das Wasser bis zum Speicher fließt. Die Begründung für diese Wahl wird später geliefert.

Der Speicherstandort 1 liegt 35 m oberhalb des Pumpenhauses und ist mit einem 120 m langen Rohr an das Pumpenhaus angeschlossen. Daraus ergibt sich für den Druck H^1_{S1}:

$$H_1^{S1} = 45 + 0,12Q^2.$$

Der Speicherstandort 2 liegt 40 m oberhalb des Pumpenhauses und ist mit einem 200 m langen Rohr an das Pumpenhaus angeschlossen. Daraus ergibt sich für den Druck H^1_{S2}:

Tabelle 8.2: Menge an verbaubaren Pumpen mit fiktiven Kaufpreisen

Pumpe	fiktiver Kaufpreis (in Euro)
Wilo-Multivert MVIE 204 M1	200
Wilo-Economy MHIE 406N-2G	400
Wilo-Economy MHIE 205N-2G	600
Wilo-Helix EXCEL 208-1/16/E/KS	800
Wilo-Multivert MVIE 806	1000
Wilo-Economy MHIE 203N	1200

$$H_1^{S2} = 50 + 0,2Q^2.$$

Der Speicherstandort 3 liegt 36 m oberhalb des Pumpenhauses und ist mit einem 200 m langen Rohr an das Pumpenhaus angeschlossen. Daraus ergibt sich für den Druck H_{S3}^1:

$$H_1^{S3} = 46 + 0,2Q^2.$$

Verbaubare Komponenten

Für das zu planende System steht eine Auswahl an Kreiselpumpen des Herstellers Wilo [22] zur Verfügung. Für jede Pumpe wurde ein fiktiver Kaufpreis angesetzt.

Zur Linearisierung der Pumpenkennfelder wurden 16 Stützpunkte gewählt. Die Wertetabelle der Linearisierung ist auf der beigefügten CD zu finden.

8.2 Laufzeitverhalten der Dynamischen Optimierung

In diesem Abschnitt wird untersucht, wie sich die Beschaffenheit der Datengrundlage auf die Laufzeit der Dynamischen Optimierung auswirkt. Die Laufzeitergebnisse stammen aus dem Ameisenalgorithmus bzw. den Einzellaufzeiten der Ameisen mit zulässiger Lösung. Es werden drei unterschiedliche Variationen der Datengrundlage betrachtet. Der erste Variationsfaktor ist die Anzahl der auftretenden Nachfrageniveaus, der zweite die Diskretisierung der Füllstände und der dritte die Anzahl an Senken. Die hier betrachteten Lastprofile bilden die Nachfrage einer Woche ab, wobei die Zeit in 15 min Intervalle unterteilt ist. In allen Variationen werden die oben aufgezählten Pumpen zur Verfügung gestellt.

Anzahl an Nachfrageniveaus

Für die Untersuchung des Einflusses der Anzahl an auftretenden Nachfrageniveaus werden Systeme betrachtet, die jeweils eine der Druckzonen versorgen müssen, d.h. Netzwerke mit einer Senke. Für jedes Netzwerk ist der Speicherstandort so gewählt, dass das Wasser, welches aus dem Speicher kommt, ausreichend Druck hat um die Abnehmer zu versorgen. Diese Annahme wird getroffen, damit der Ameisenalgorithmus viele zulässige Lösungen findet und somit viele Laufzeitergebnisse zur Verfügung stehen. Für das Netzwerk mit Druckzone 1 wird der Speicherstandort 1 gewählt, für das Netzwerk mit Druckzone 2 wird der Speicherstandort 2 gewählt und für das Netzwerk mit Druckzone 3 wird der Speicherstandort 3 gewählt. Der Speicher hat eine Grundfläche von 0.25m^2 und eine maximalen Füllhöhe von 3m. Der Füllstand wurde in 31 äquidistante Füllstände unterteilt. Im Netzwerk mit Druckzone 1 und 2 kann über den gesamten Zeitraum maximal ein Volumenstrom von $0.4\text{m}^3\,\text{h}^{-1}$ entnommen werden. Zur

Versorgung von Druckzone 3 steht über den gesamten Zeitraum ein Volumenstrom von $0.5 \mathrm{m}^3\,\mathrm{h}^{-1}$ zu Verfügung.

Um die Anzahl an auftretenden Nachfrageniveaus zu variieren, werden die nachgefragten Volumenströme in einem Lastprofil auf eine Nachkommastelle oder auf ein Vielfaches von 0.05 gerundet. Für jede der Druckzonen wurde für jede der vier Wochen des Sommer- bzw. sonstigen Monats mit auf 0.1 oder 0.05 gerundeten oder ungerundeten Volumenströmen ein Ameisenalgorithmus mit 1000 Iterationen durchgeführt.

Die durchschnittliche Laufzeit ergibt sich aus dem Mittelwert der jeweiligen Lastprofile, die eine gleiche Anzahl an auftretenden Nachfrageniveaus haben. Die in der Tabelle 8.3 angegebene Häufigkeit, entspricht dabei der Anzahl an Problemen mit entsprechender Anzahl an Nachfrageniveaus, die durch Dynamische Optimierung gelöst wurden. Im folgenden wird anstatt von Nachfrageniveaus von Nachfragen gesprochen.

Tabelle 8.3: Durchschnittliche Laufzeiten bei auftretenden Nachfrageanzahlen.

Anzahl an Nachfragen	Häufigkeit	durchschnittlich Laufzeit in Sekunden
8	3170	31.369118947
9	1085	35.2247417613
11	983	45.0481266195
12	978	47.6789083098
14	3000	52.1294349903
15	1050	69.42968434
17	246	80.1666125976
19	738	78.7083936274
38	290	200.3274721828
39	2049	158.1091560386
40	745	147.7407331034
41	1278	148.7580205383
42	554	170.7825208538
43	263	172.5434101483

In Abbildung 8.1 sind die beobachteten Werte gegeneinander aufgetragen und zusätzlich durch eine Trendgerade versehen. Die beobachteten Werte lassen auf einen linearen Zusammenhang zwischen der Laufzeit der Dynamischen Optimierung und der Anzahl an Nachfragen schließen.

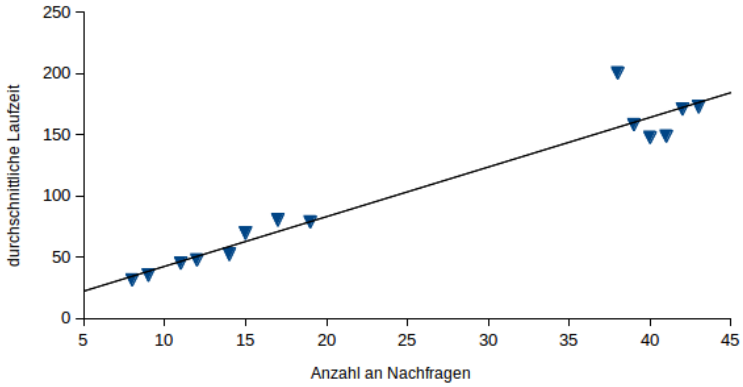

Abbildung 8.1: Beobachtete durchschnittliche Laufzeiten der Dynamischen Optimierung in Abhängigkeit der Nachfrageanzahl mit zugehöriger Trendgeraden.

Tabelle 8.4: Durchschnittliche Laufzeiten bei auftretenden Nachfrageanzahlen und gegebener Einteilung der Füllstände.

Füllstände	Anzahl an Nachfragen	Häufigkeit	durchschnittliche Laufzeit in Sekunden
31	8	3170	31.369118947
31	9	1085	35.2247417613
31	11	983	45.0481266195
31	12	978	47.6789083098
41	8	2793	57.2499104905
41	9	639	59.7982785603
41	11	821	85.5805115713
41	12	821	85.0772228989
61	8	2661	139.9063885757
61	9	809	151.5467243511
61	11	814	183.8953316953
61	12	824	198.6847087379

Anzahl an Füllständen

Zur Untersuchung des Einflusses der Anzahl an Füllständen werden erneut Netzwerke mit nur einer Senke betrachtet. Auch hier sind die Speicherstandorte so gewählt, dass Abnehmer direkt aus dem Speicher mit Wasser versorgt werden können. Die Zuordnung von Druckzonen zu Speicherstandorten ist wie oben gewählt.

Der Speicher hat auch hier eine Grundfläche von 0.25m^2 und eine Höhe von 3m. Es werden drei unterschiedliche Diskretisierungen betrachtet. Dabei wird der Füllstand in 31, 41 und 61 äquidistante Füllstände unterteilt. Im Fall mit 31 Ständen lässt sich der Füllstand um Vielfache von 0.1 m verändern; im Fall mit 41 Ständen lässt sich der Füllstand um Vielfache vom 0.075 m verändern und im Fall von 61 Füllständen sind Änderungen des Standes um Vielfache von 0.05 m möglich. Für jede der drei Druckzonen wurde für jede der vier Wochen des Sommer- bzw. sonstigen Monats mit auf 0.1 gerundeten Volumenströmen für jede der drei Diskretisierungen ein Ameisenalgorithmus mit 1000 Iterationen durchgeführt.

Auch in diesem Fall wird die mittlere Laufzeit für jede Diskretisierung, abhängig von der Anzahl mit der eine Nachfrage auftritt, berechnet. Die beobachteten Werte sind in der Tabelle 8.4 abgebildet.

Die Anzahl an in der Dynamischen Optimierung zu lösenden MIPs steigt bei gleicher Nachfrageanzahl quadratisch mit den Speicherständen. Aus diesem Grund ist zu vermuten, dass die durchschnittliche Rechenzeit ebenfalls quadratisch steigt. In Abbildung 8.2 sind die beobachteten Laufzeiten für jede Nachfrageanzahl über den Pufferständen aufgetragen. Die beobachteten Werte wurden mit Geraden verbunden, welche einen quadratischen Verlauf mit hohem Streckungsfaktor erahnen lassen.

Anzahl an Senken

Zur Analyse des Einflusses der Senkenanzahl auf die Laufzeit wurden zusätzlich je zwei Systeme mit zwei und vier Abnehmern betrachtet. Dazu wird wieder auf die Druckzonen und die auf eine Nachkommastelle gerundeten Wochennachfragen zurückgegriffen. Die Druckzonen werden zu Siedlungen zusammengefasst, wobei jede Druckzone einen zu versorgenden Abnehmer darstellt.

- Siedlung 1 besteht aus der Druckzone 3, der Druckzone 1 und dem Speicherstandort 3.
- Siedlung 2 besteht aus der Druckzone 3, der Druckzone 2 und dem Speicherstandort 2.

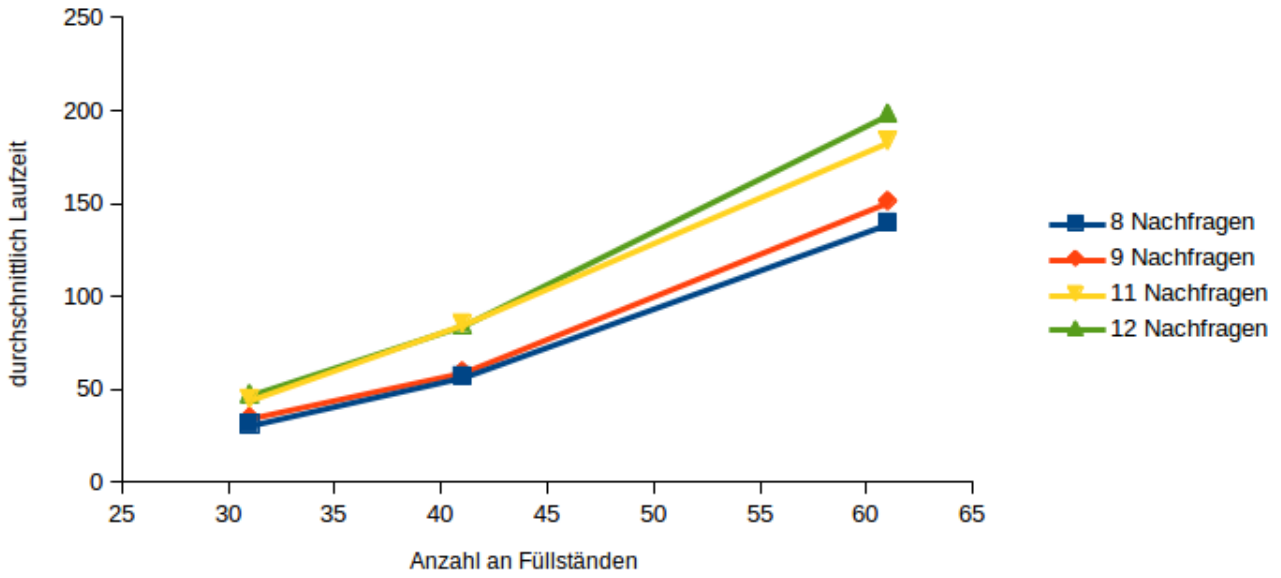

Abbildung 8.2: Beobachtete durchschnittlich Laufzeiten der Dynamsichen Optimierung für vier Nachfragezahlen in Abhängigkeit der Füllstände

- Siedlung 3 besteht aus der Druckzone 3, zwei Druckzonen 2, einer Druckzone 1 und dem Speicherstandort 2.
- Siedlung 4 besteht aus der Druckzone 3, der Druckzone 2, zwei Druckzonen 1 und dem Speicherstandort 2.

Für die Siedlungen mit zwei Abnehmern bzw. vier Abnehmern steht dauerhaft ein Angebot von 0.9 $m^3\,h^{-1}$ bzw. 1.9 $m^3\,h^{-1}$ zur Verfügung. Die Speicherhöhe ist auch hier 3m und in 31 Füllstände unterteilt. Es werden nur die Grundflächen der Speicher angepasst, indem sie bei zwei Abnehmern verdoppelt und bei vier Abnehmern vervierfacht wird.

Die resultierenden durchschnittlichen Laufzeiten sind in der Tabelle 8.5 nachzulesen.

Würde die Laufzeit der Dynamischen Optimierung hauptsächlich von der Anzahl an Nachfragen und Füllständen abhängen, so müssten sich die beobachteten Werte ähnlich wie die Trendgerade aus Abbildung 8.1 verhalten. In Abbildung 8.3 sind die neuen Werte in die bereits vorgestellten durchschnittlichen Laufzeiten in Abhängigkeit der Nachfrageanzahlen eingeordnet. Es ist gut zu erkennen, dass sich die Werte von dem Verlauf der Trendgeraden unterscheiden. Daraus lässt sich schließen, dass auch die Anzahl an Abnehmern möglicherweise die Rechenzeit beeinflusst.

8.3 Verhalten der Tabu-Suche

In diesem Abschnitt soll das Verhalten des Solvers cplex bei der Lösungssuche mit dem Verhalten der Tabu-Suche verglichen werden. Die im Rahmen dieser Arbeit implementierte Tabu-Suche weicht leicht von der in Kapitel 6.1 beschriebenen ab. Das Einfügen von Bauteilen vor bzw. nach ein bestehendes Bauteil ist nur möglich, indem die ganze Menge der Vorgänger bzw. Nachfolger zu Vorgängern bzw. Nachfolgern des neuen Knotens wird und der neu eingefügte Knoten der einzige Vorgänger bzw. Nachfolger des bestehenden Bauteils wird. Zusätzlich werden Löschoperationen immer durchgeführt. Falls dies zu einer unzulässigen Lösung führt, wird die entstandene Lösung, ebenso wie eine Lösung ohne zulässige Pumpen- und Ventileinstellung, mit einem genügend großen Zielfunktionswert bewertet.

Tabelle 8.5: Durchschnittliche Laufzeiten bei auftretenden Nachfrageanzahl und gegebener Abnehmeranzahl.

Abnehmeranzahl	Anzahl an Nachfragen	Häufigkeit	durchschnittliche Laufzeit
4	20	904	149.553539823
4	22	319	156.3539184953
4	23	977	184.2574206755
4	24	159	203.2025157233
4	25	319	193.0786833856
2	14	708	74.9062146893
2	16	477	85.5457023061
2	17	1368	90.580994152
2	18	220	85.8
2	19	209	97.4588516746
2	20	633	112.8532385466

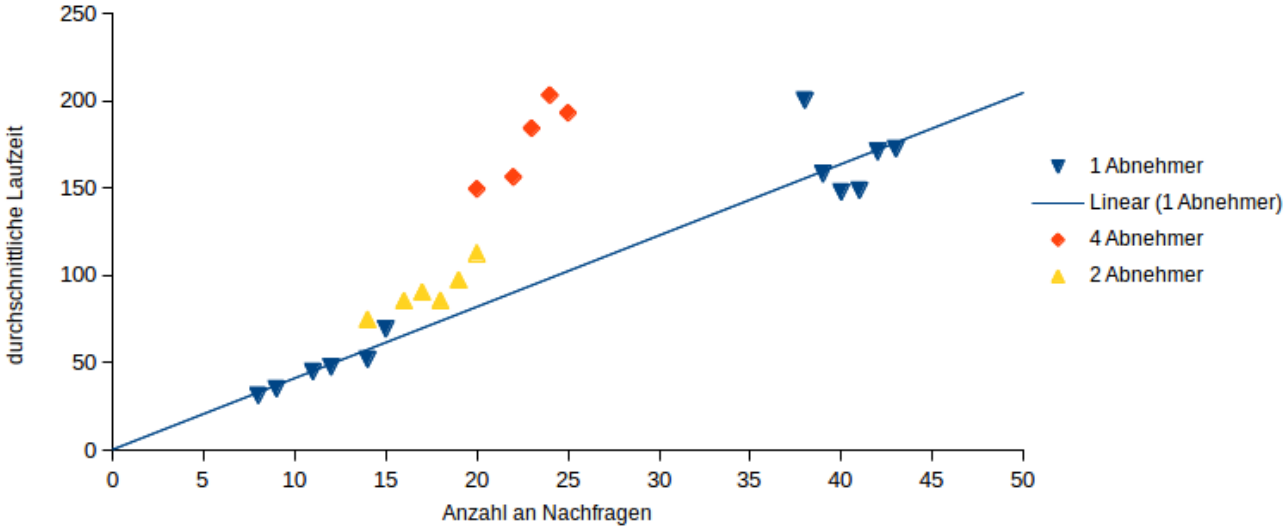

Abbildung 8.3: Beobachtete durchschnittliche Laufzeiten der Dynamsichen Optimierung für zwei und vier Abnehmer im Vergleich zu den durchschnittlichen Laufzeiten aus Abbildung 8.1.

Es werden die Nachfrageprofile für einen Tag mit auf eine Nachkommastelle gerundeten Werten verwendet und Instanzen mit zwei und vier Abnehmern betrachtet. Hierzu wird auf Siedlung 2, 3 und 4 zurückgegriffen. Für Siedlung 2 werden die ersten drei Tage des Sommermonats betrachtet; für Siedlung 3 der erste Tag des Sommermonat und für Siedlung 4 der vierte Tag des sonstigen Monats. Auch hier werden wieder nur Systeme mit einem Speicher betrachtet.

Als Startlösung für die Tabu-Suche wird willkürlich die zweite gefundene Lösung für die Siedlung mit zwei Abnehmern des Ameisenalgorthmus verwendet. Für die Siedlungen mit vier Abnehmern wird die dritte Lösung verwendet. Dabei wurde nur darauf geachtet, dass die gewählten Startlösungen *nicht* die beste von den Ameise gefundene Kaufentscheidung ist (damit sichtbares Verbesserungspotential für die Tabu-Suche vorhanden ist).

Für die Siedlungen mit zwei Abnehmern wurde die Tabu-Suche über rund 50 Stunden beobachtet. Bei den Instanzen mit vier Abnehmern wurde die Suche sogar über rund 60 Stunden beobachtet. In den

Tabellen 1-5 im Anhang ist zu sehen, wie sich der Zielfunktionswert der Tabu-Suche im Laufe der Zeit ändert und durch welchen Nachbarschaftstausch ein Wechsel erzeugt wurde.

Es ist zu erkennen, dass in den Beispielen mit zwei Abnehmern die meisten Nachbarschaftswechsel durch Tauschen von Bauteilen entstehen. Zusätzlich fällt auf, das diese Tauschoperationen keine Verbesserung der Zielfunktionswerte nach sich ziehen. Somit sollte betrachtet werden, ob statt eines Nachbarschaftswechsels durch Tauschen das folgende Vorgehen vorzuziehen ist. In diesem wird für eine gegebene Menge an verbauten Bauteilen die beste Bauteilpositionierung bestimmt und von dort aus nur noch Nachbarn aus N_{Rep}, N_{Add} und N_{Del} untersucht.

Die Verläufe der Zielfunktionswerte des Solvers cplex und der aktuellen Lösung aus der Tabu-Suche sind in den folgenden Abbildungen 8.4-8.8 zu sehen. Dabei ist die beste Lösung der Tabu-Suche, die mit dem niedrigsten Zielfunktionswert. Zusätzlich ist eine duale Schranke für das jeweilige Problem eingezeichnet, um die Güte der gefundenen Lösungen zu verdeutlichen. Diese Werte ergeben sich aus einer Kombination beider in Kapitel 7 vorgestellter Relaxierungen. Die Berechnung der Schranken liefert von Beginn an einen besseren Dualwert als cplex und terminiert nach weniger als einem Tag. In den Graphiken ist der Endwert dargestellt.

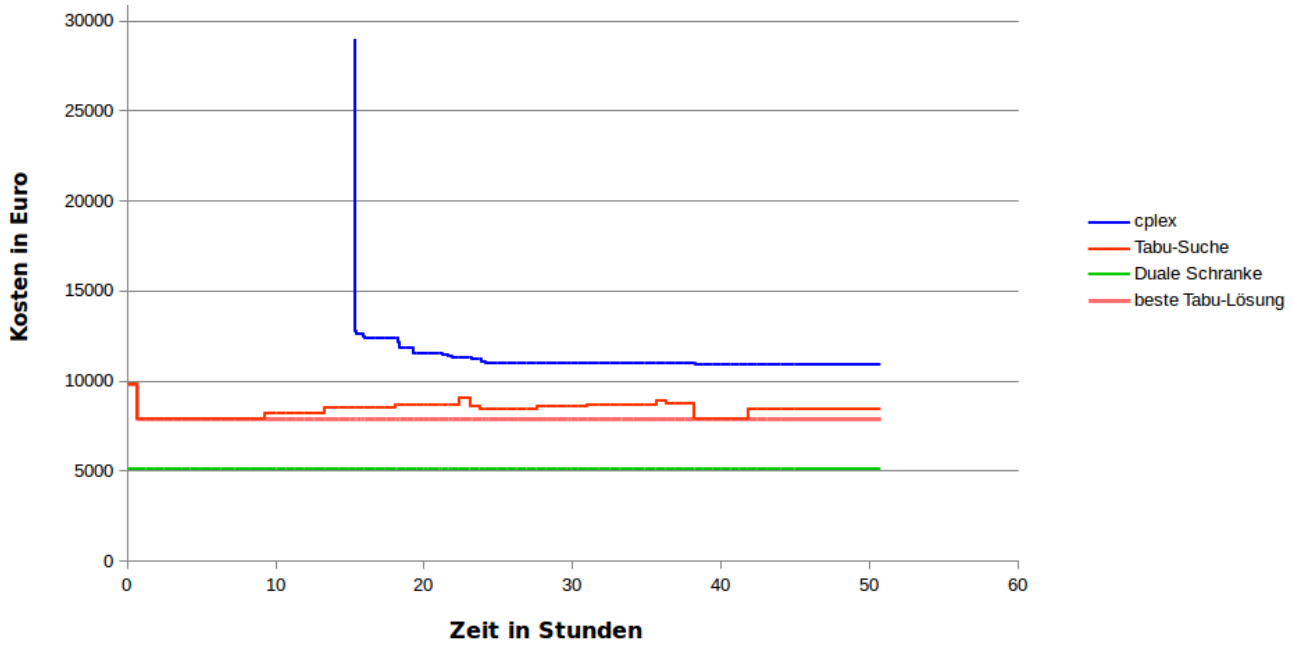

Abbildung 8.4: Optimierungsproblem für Siedlung 2 mit dem ersten Tag des Sommermonats.

Es ist zu erkennen, dass der Ameisenalgorithmus bei diesen Beispielen schon nach kürzester Zeit eine zulässige Lösung findet, welche besser als die erste Lösung von cplex ist. In den betrachteten Beispielen benötigte der Ameisenalgorithmus dabei maximal 7 Minuten, was 0,3% der Zeit entspricht, die cplex zum Finden der ersten zulässigen Lösung benötigt. In allen Beispielen liegt während der gesamten betrachteten Laufzeit der Zielfunktionswert der besten bisher gefunden Lösung der Tabu-Suche stets unter dem Zielfunktionswert der besten Lösung, die von cplex in der selben Zeit gefunden wurde. Dabei auffallend, dass die Tabu-Suche relativ schnell eine Lösung finden, die sie danach nicht mehr verbessert. Es gibt jedoch bestimmte Zeitpunkte in denen die aktuell von der Tabu-Suche betrachtete Lösung einen schlechteren Zielfunktionswert liefert als die aktuelle cplex Lösung. Bei einer Verlängerung der Laufzeit würde jedoch erwartet, dass sich die cplex Lösung weiter der Optimallösung annähert und somit besser werden könnte als die Lösung der Tabu-Suche. Das kann unter anderem daran liegen, dass die Tabu-Suche schon nahe am Optimum lag oder dass die optimale Kaufentscheidung stark von den gefundenen Lösungen abweicht. Um genaue Aussagen hierüber treffen zu können empfiehlt sich eine Langzeitstudie

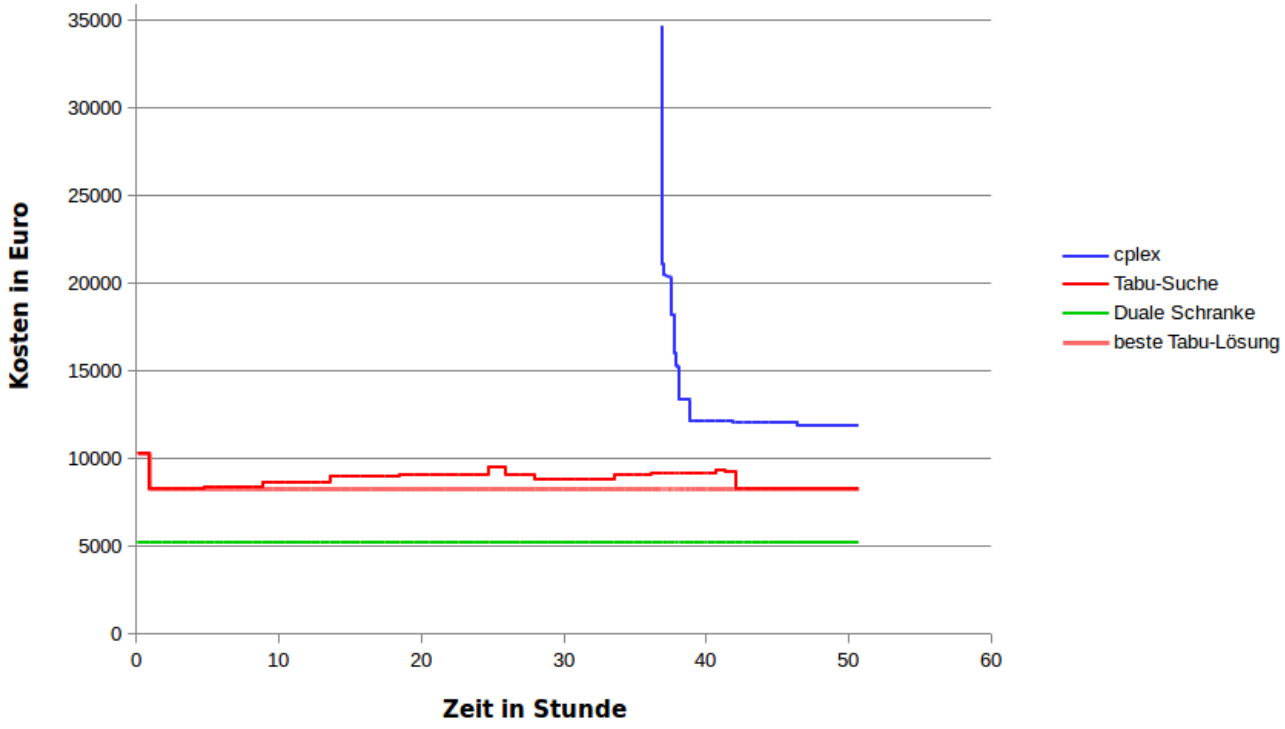

Abbildung 8.5: Optimierungsproblem für Siedlung 2 mit dem zweiten Tag des Sommermonats.

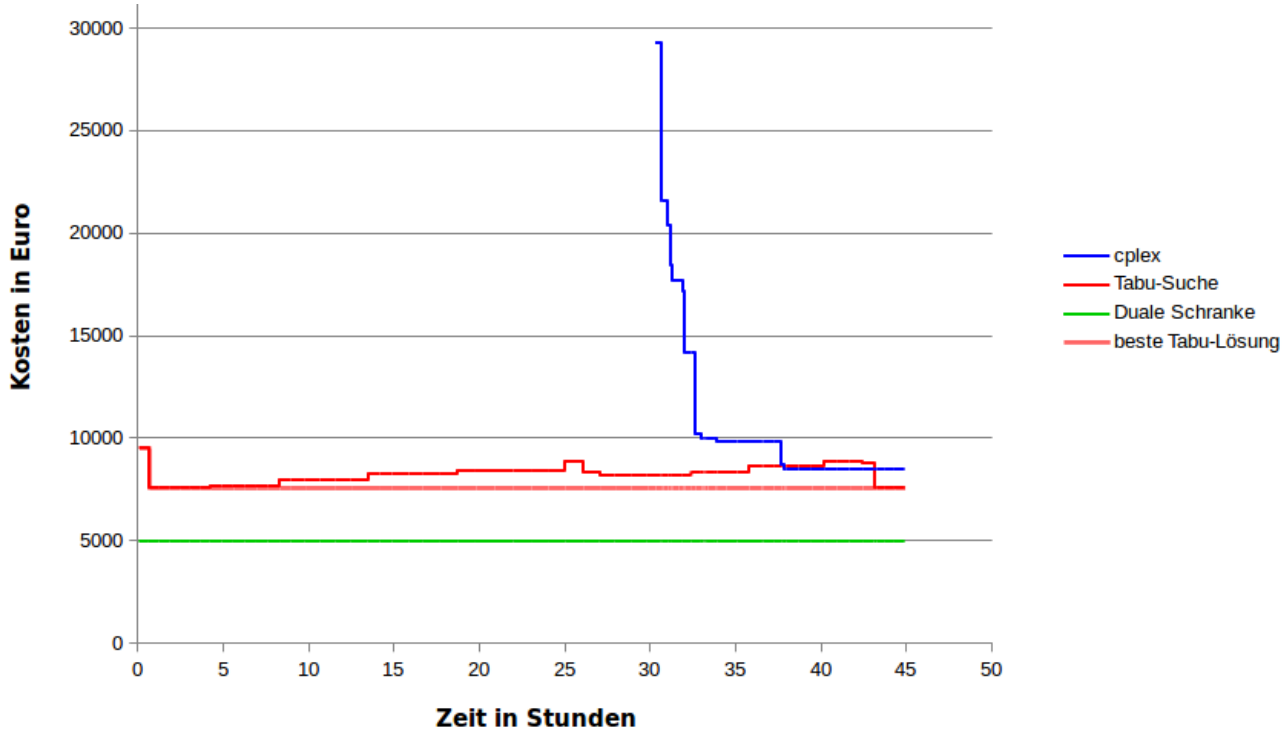

Abbildung 8.6: Optimierungsproblem für Siedlung 2 mit dem dritten Tag des Sommermonats.

oder vereinfachte Testinstanzen die Lösungssuchen so lange zu verlogen bis cplex mit Optimallösung terminiert.

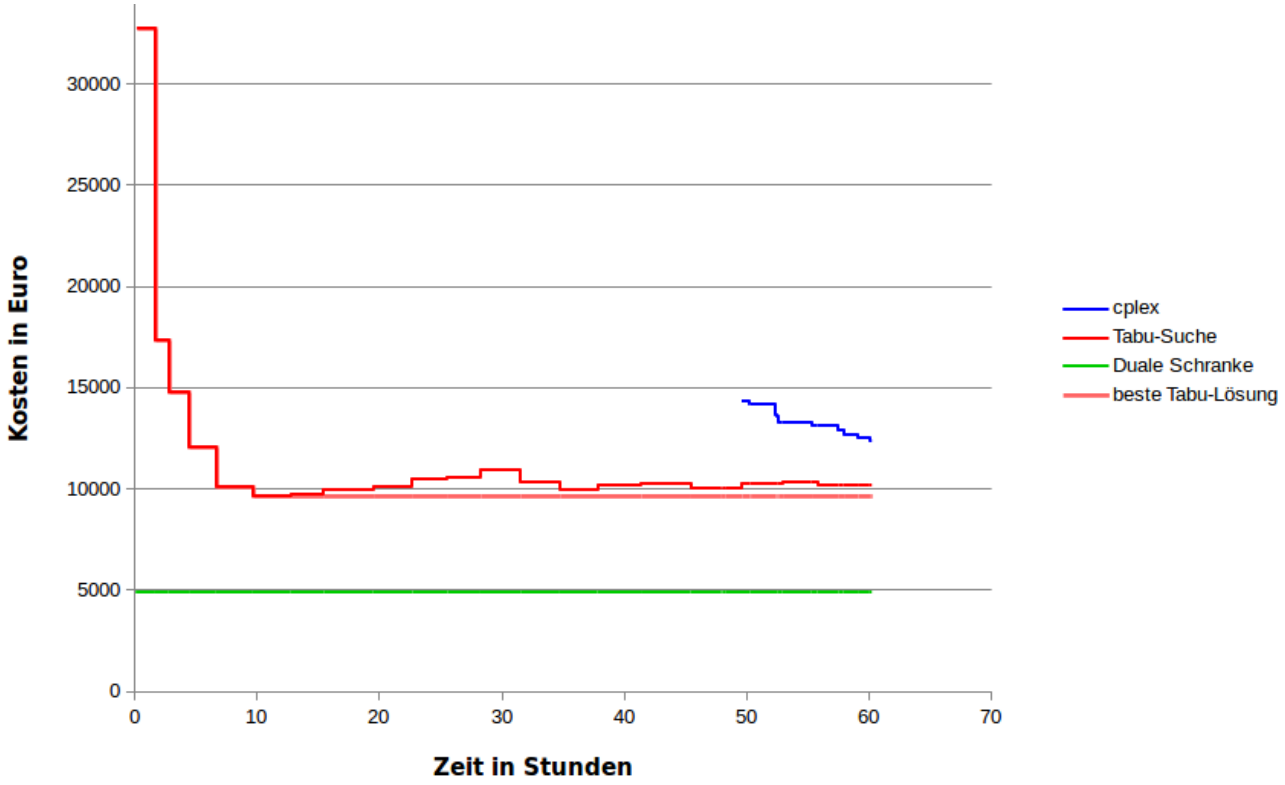

Abbildung 8.7: Optimierungsproblem für Siedlung 3 mit dem ersten Tag des Sommermonats.

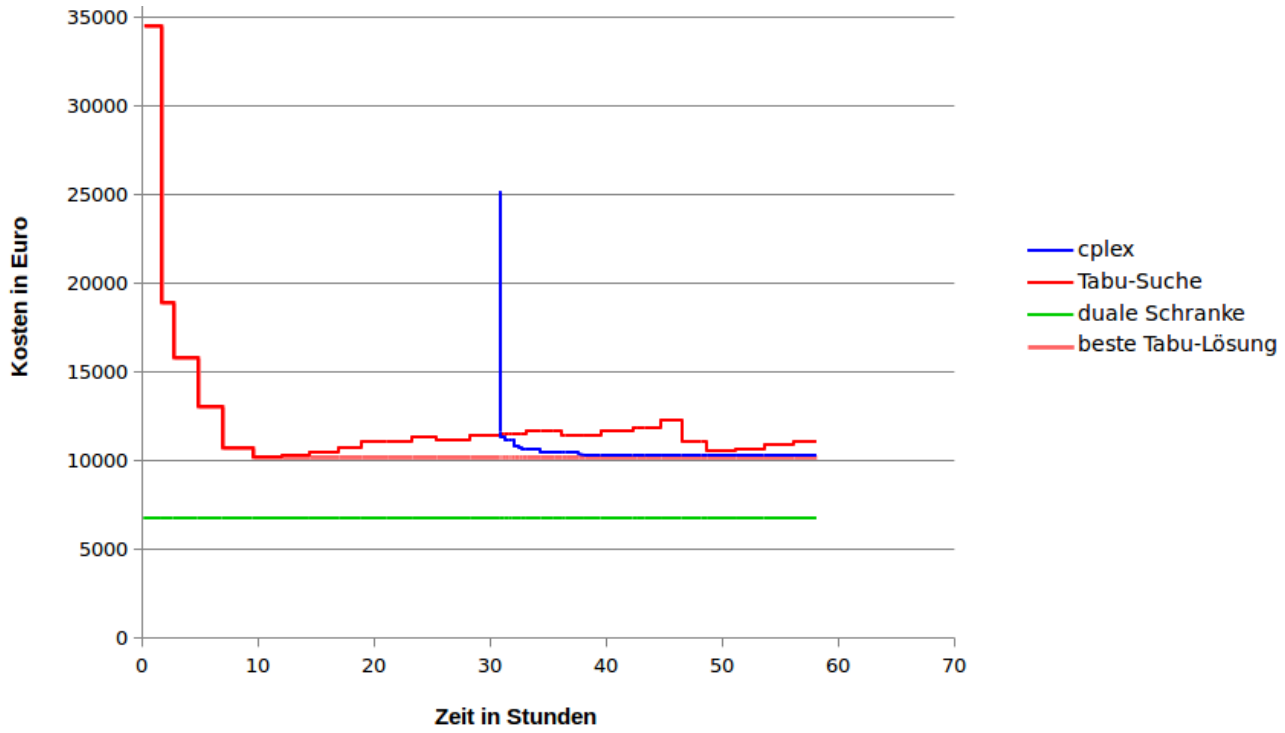

Abbildung 8.8: Optimierungsproblem für Siedlung 4 mit dem vierten Tag des Sommermonats.

Zusätzlich war zu beobachten, dass die duale Schranke von cplex, nie wesentlich höher als bei 1000 Euro lag und in den meisten Fällen sogar unter 800 Euro war. Es sollte untersucht werden, ob sich die

Lösungssuche von cplex durch die Angabe der dualen Schranke und evtl. der zulässigen Lösungen aus der Tabu-Suche beschleunigen lässt.

Die Untersuchung zeigt, dass der Ameisenalgorithmus besonders schnell zulässige Lösungen findet, die durch einige Änderungen der Bauteilzuordnung durch die Tabu-Suche stark verbessert werden können. Deshalb könnte es sich lohnen ein kombiniertes Verfahren statt zweier nacheinander ausgeführter Algorithmen zu betrachten. Beispielsweise könnte durch einen angepassten Ameisenalgorithmus mit einer in den Knoten-Bauteil-Zuordnungsschritt integrierter Tabu-Suche eine weitere Laufzeitverbesserung erzielt werden.

9 Fazit

Die grundlegende Problemstellung dieser Arbeit war die Planung eines Wasserversorgungsystems für Siedlungen, denen aufgrund von Versorgungsengpässen das Wasser nicht jederzeit in ausreichender Menge und eventuell nicht mit ausreichendem Druck zur Verfügung steht.

Hierfür wurde ein MIP aufgestellt, das für ein gegebenes Nachfrageprofil einen Systemaufbau mit zeitabhängigen Pumpen- und Ventileinstellungen liefert, dessen Anschaffungs- und Stromkosten minimal sind.

Mit Standartmethoden der diskreten Optimierung ist dieses MIP schon ab wenigen betrachteten Zeitschritten nicht mehr in akzeptabler Zeit lösbar. Aus diesem Grund wurden in dieser Arbeit zwei Verfahren betrachtet, welche in kürzerer Zeit „gute“ zulässige Lösungen finden. Diese beruhen darauf, dass die zu treffenden Entscheidungen in die zwei Teilentscheidungen des Findens eines Systemaufbaus und des Findens einer Pumpen- und Ventileinstellung aufgeteilt wurden.

Für die Dynamische Optimierung der Pumpen- und Ventileinstellung konnten drei Zusammenhänge bezüglich der Laufzeit festgestellt werden: (1) Bei fester Anzahl an Abnehmern und Füllständen in der Diskretisierung steigt die Laufzeit der Dynamischen Optimierung linear mit der Anzahl an zu berücksichtigenden Nachfragen. (2) Wird die Anzahl an Füllständen in der Diskretisierung bei konstanter Anzahl an Abnehmern und Nachfragen variiert, so zeichnet sich ein quadratischer Zusammenhang zwischen der Anzahl an Füllständen und der Laufzeit ab. (3) Zusätzlich deutet sich an, dass die Laufzeit mit der Anzahl an Abnehmern steigt.

Das Finden eines Systemaufbaus konnte durch die Verwendung einer Tabu-Suche deutlich verkürzt werden. Hierbei wurde die Startlösung vom Ameisenalgorithmus geliefert, die in den betrachteten Beispielen schon innerhalb von maximal 0.3% der Zeit gefunden wurde, die cplex bis zur ersten zulässigen Lösung benötigte. Über die gesamte Laufzeit war zu beobachten, dass die Zielfunktionswerte der Tabu-Suche stets unter den Werten von cplex lagen, sich die Differenz im Verlauf des Beobachtungszeitraums jedoch verringerte.

Insgesamt lässt sich somit aus den Ergebnissen erkennen, dass die entwickelten Verfahren schon für Nachfrageprofile eines einzigen Tages schneller bessere Lösungen finden als ein Standardsolver (vgl. 8.3).

Um diese erreichten Verbesserungen noch weiter zu steigern, empfiehlt sich anhand der gewonnenen Erkenntnisse die Betrachtung eines Ameisenalgorithmus in Kombination mit einer auf Tabu-Suche beruhenden Bauteil-Knoten-Zuordnung als eigenständige Primalheuristik und einer abgeänderten Form der Tabu-Suche, in der die Nachbarschaftsfunktion in zwei Teile zerlegt wird. Ausgehend von einem Netzwerk werden in einem Teilschritt zunächst die gesamten Swap-Nachbarschaften (d.h. alle Netzwerke, die durch (mehrfaches) Tauschen zweier Bauteile erzeugt werden können) betrachtet; daraus wird für diese Netzwerkstruktur und die entsprechende Menge an verwendeten Bauteilen die optimale Bauteilpositionierung bestimmt. Im nächsten Schritt werden nun nur die Delete-, Add- und Replace-Nachbarschaften betrachtet.

Anhang

Ergebnisse Tabu-Suche

Zeitpunkt	Zielfunktionswert	Nachbarschaftstausch
72.00	9886.59	Startlösung
2140.93	7923.21	Rep
4313.37	7923.21	Swap
6508.08	7923.21	Swap
8715.09	7923.21	Swap
10908.07	7923.21	Swap
12998.59	7923.21	Swap
15091.99	7991.54	Rep
17609.40	7991.54	Swap
20049.57	7991.54	Swap
22497.11	7991.54	Swap
25841.50	7991.54	Swap
29473.10	7991.54	Swap
33168.95	8288.80	Rep
36039.78	8288.80	Swap
38306.83	8288.80	Swap
40482.31	8288.80	Swap
42641.63	8288.80	Swap
44908.92	8288.80	Swap
47707.00	8562.56	Rep
51067.83	8562.56	Swap
53632.41	8562.56	Swap
56482.00	8562.56	Swap
59821.23	8562.56	Swap
62372.76	8562.56	Swap
64877.00	8741.75	Rep
67417.69	8741.75	Swap
70075.98	8741.75	Swap
72787.02	8741.75	Swap
75347.13	8741.75	Swap
77760.10	8741.75	Swap
80214.36	9141.75	Add
82829.43	8688.80	Rep
85158.19	8502.27	Swap
87369.77	8502.27	Swap
89587.16	8502.27	Swap
91938.15	8502.27	Swap
94303.09	8502.27	Swap
96631.13	8502.27	Swap

98956.07	8688.80	Swap
101402.49	8688.80	Swap
103921.67	8688.80	Swap
106478.55	8688.80	Swap
108957.10	8688.80	Swap
111281.53	8758.31	Swap
114064.28	8758.31	Swap
116863.67	8758.31	Swap
119620.67	8758.31	Swap
122402.62	8758.31	Swap
125215.37	8758.31	Swap
127975.45	8985.16	Del
130341.35	8785.16	Swap
132537.73	8785.16	Swap
134763.22	8785.16	Swap
137275.35	7923.21	Rep
139472.69	7923.21	Swap
141647.93	7923.21	Swap
143837.99	7923.21	Swap
146052.67	7923.21	Swap
148145.61	7923.21	Swap
150237.95	8523.21	Add
152432.33	8523.21	Swap
154750.76	8523.21	Swap
158154.85	8523.21	Swap
161572.95	8523.21	Swap

Tabelle 1: Verlauf der Tabu-Suche für den ersten Tag des Sommermonats für Siedlung 2

Zeitpunkt	Zielfunktionswert	Nachbarschaftstausch
76.00	10419.50	Startlösung
3092.57	8356.97	Rep
6286.86	8356.97	Swap
8543.81	8356.97	Swap
10726.80	8356.97	Swap
12899.99	8356.97	Swap
14989.67	8356.97	Swap
17146.70	8460.47	Rep
19532.02	8460.47	Swap
21903.92	8460.47	Swap
24296.78	8460.47	Swap
26692.23	8460.47	Swap
29134.01	8460.47	Swap
31564.45	8759.86	Rep
34089.34	8759.86	Swap
37429.27	8759.86	Swap
40635.63	8759.86	Swap
43812.63	8759.86	Swap
46294.56	8759.86	Swap
48660.29	9026.20	Rep
51253.77	9026.20	Swap
53820.86	9026.20	Swap
56355.15	9026.20	Swap
58879.46	9026.20	Swap
62615.44	9026.20	Swap
66441.62	9183.66	Rep
70011.64	9183.66	Swap
73888.97	9183.66	Swap
77613.51	9183.66	Swap
81243.34	9183.66	Swap
85185.56	9183.66	Swap
88914.51	9583.66	Add
93023.46	9159.86	Rep
96669.49	9159.86	Swap
100406.19	8934.21	Swap
103880.62	8934.21	Swap
107374.51	8934.21	Swap
110841.25	8934.21	Swap
114145.73	8934.21	Swap
117480.43	8934.21	Swap
120495.60	9159.86	Swap
122990.77	9159.86	Swap
125427.71	9159.86	Swap
127751.44	9159.86	Swap
130066.23	9218.29	Swap
132734.56	9218.21	Swap
135424.13	9218.29	Swap
138066.73	9218.29	Swap

140726.65	9218.29	Swap
143422.23	9218.29	Swap
146061.92	9460.47	Rep
148726.12	9356.97	Rep
151150.86	8356.97	Del
153296.61	8356.97	Swap
155376.24	8356.97	Swap
157553.22	8356.97	Swap
159718.25	8356.97	Swap

Tabelle 2: Verlauf der Tabu-Suche für den zweiten Tag des Sommermonats für Siedlung 2

Zeitpunkt	Zielfunktionswert	Nachbarschaftstausch
77.00	9580.48	Startlösung
2107.79	7665.74	Rep
4264.23	7665.74	Swap
6444.29	7665.74	Swap
8635.07	7665.74	Swap
10800.99	7665.74	Swap
12907.49	7665.74	Swap
14981.95	7721.07	Rep
17475.22	7721.07	Swap
19886.55	7721.07	Swap
22311.33	7721.07	Swap
24746.31	7721.07	Swap
27174.09	7721.07	Swap
29652.55	8022.02	Rep
32011.36	8022.02	Swap
35012.09	8022.02	Swap
38209.40	8022.02	Swap
41377.10	8022.02	Swap
44717.56	8022.02	Swap
48234.27	8300.81	Rep
50796.62	8300.81	Swap
53348.88	8300.81	Swap
56923.96	8300.81	Swap
59766.89	8300.81	Swap
63525.50	8300.81	Swap
67225.96	8482.13	Swap
70971.65	8482.13	Rep
74890.35	8482.13	Swap
78873.75	8482.13	Swap
82651.43	8482.13	Swap
86219.24	8482.13	Swap
89843.47	8882.13	Add
93674.35	8422.02	Rep
97083.08	8244.80	Swap
100322.69	8244.80	Swap
103713.22	8244.80	Swap
107152.65	8244.80	Swap
110401.53	8244.80	Swap
113804.12	8244.80	Swap
116428.26	8422.02	Swap
118943.11	8422.02	Swap
121429.97	8422.02	Swap
123713.80	8422.02	Swap
126150.60	8422.02	Swap
128549.56	8700.81	Rep
131186.59	8700.81	Swap
133791.03	8700.81	Swap
136411.01	8700.81	Swap

139084.69	8700.81	Swap
141783.10	8700.81	Swap
144461.26	8882.13	Rep
147040.74	8882.13	Swap
149801.90	8882.13	Swap
152511.87	8865.74	Swap
154902.29	7665.74	Del
157057.67	7665.74	Swap
159234.78	7665.74	Swap
161421.21	7665.74	Swap

Tabelle 3: Verlauf der Tabu-Suche für den dritten Tag des Sommermonats für Siedlung 2

Zeitpunkt	Zielfunktionswert	Nachbarschaftstausch
352.23	34593.60	Startlösung
5614.93	18972.70	Add
9258.18	15869.10	Del
16827.79	13107.20	Del
24364.34	10741.90	Del
33906.54	10239.00	Del
43088.13	10355.20	Add
51555.06	10510.10	Rep
60754.03	10755.20	Add
67658.58	11141.90	Rep
75661.99	11155.20	Rep
83428.00	11355.20	Rep
90948.42	11239.00	Del
101468.32	11439.00	Rep
111426.55	11555.20	Add
118845.33	11710.10	Rep
129876.33	11510.10	Rep
141903.63	11741.90	Rep
151881.52	11941.90	Rep
160654.14	12341.90	Add
167050.75	11141.90	Del
174936.56	10597.60	Del
183894.32	10721.10	Rep
192623.96	10940.40	Rep
201880.05	11155.20	Add

Tabelle 4: Verlauf der Tabu-Suche für den ersten Tag des Sommermonats für Siedlung 3

Zeitpunkt	Zielfunktionswert	Nachbarschaftstausch
359.50	32790.10	Startlösung
5790.73	17372.10	Add
9748.62	14806.40	Del
15754.45	12090.90	Del
23705.67	10179.90	Del
34496.37	9725.99	Del
45663.47	9780.67	Add
55390.56	9988.53	Del
69900.86	10180.70	Rep
81475.86	10570.30	Rep
91811.66	10579.90	Rep
101454.17	10964.20	Rep
113268.67	10354.90	Del
124771.37	10029.40	Del
135861.27	10202.90	Add
148662.17	10304.00	Rep
163243.97	10119.60	Rep
178285.77	10311.20	Del
190285.27	10402.30	Add
200543.17	10212.50	Rep

Tabelle 5: Verlauf der Tabu-Suche für den vierte Tag des sonstigen Monats für Siedlung 4

Literaturverzeichnis

[1] Aarts, Emile H. ; Lenstra, Jan K.: *Local search in combinatorial optimization*. Princeton University Press, 2003

[2] Bazaraa, Mokhtar S. ; Jarvis, John J. ; Sherali, Hanif D.: *Linear programming and network flows*. John Wiley & Sons, 2011

[3] Biggs, Norman ; Lloyd, E K. ; Wilson, Robin J.: *Graph Theory, 1736-1936*. Clarendon Press, 1986

[4] Braun, Jost: *Strömungsmaschinen als Kraft-und Arbeitsmaschinen: Teil I: Hydraulische Maschinen*. BoD–Books on Demand, 2014

[5] Cayley, Arthur: XXVIII. On the theory of the analytical forms called trees. In: *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* 13 (1857), Nr. 85, S. 172–176

[6] Domschke, Wolfgang ; Drexl, Andreas: *Einführung in operations research*. Bd. 5. Springer, 2005

[7] Dorigo, Marco ; Caro, Gianni ; Gambardella, Luca: Ant algorithms for discrete optimization. In: *Artificial life* 5 (1999), Nr. 2, S. 137–172

[8] Eppstein, David: Parallel recognition of series-parallel graphs. In: *Information and Computation* 98 (1992), Nr. 1, S. 41–55

[9] Gujer, Willi: *Siedlungswasserwirtschaft*. Springer, 2007

[10] Gupta, Rajat: *Branch-and-cut for piecewise linear optimization*, Texas Tech University, Dissertation, 2012

[11] Hahne, Erich: *Technische Thermodynamik: Einführung und Anwendung*. Oldenbourg Verlag, 2010

[12] IBM: *IBM ILOG CPLEX Optimizer*. – URL `http://www-01.ibm.com/software/commerce/optimization/cplex-optimizer/index.html`

[13] Kauffmann, Ernst ; Bastke, Rainer ; Mayer, Kurt: *Hydraulische Steuerungen*. Springer, 1980

[14] Lew, Art ; Mauch, Holger: *Dynamic programming: A computational tool*. Bd. 38. Springer, 2006

[15] MacMahon, Percy A.: Yoke-chains and multipartite compositions in connexion with the analytical forms called trees. In: *Proceedings of the London Mathematical Society* 1 (1890), Nr. 1, S. 330–346

[16] Martin, Alexander ; Möller, Markus ; Moritz, Susanne: Mixed integer models for the stationary case of gas network optimization. In: *Mathematical programming* 105 (2006), Nr. 2-3, S. 563–582

[17] Michiels, Wil ; Aarts, Emile ; Korst, Jan: *Theoretical aspects of local search*. Springer, 2007

[18] Schrijver, Alexander: *Theory of linear and integer programming*. John Wiley & Sons, 1998

[19] Statistischenbundesamt: *Wasserverwendung*. – URL http://www.umweltbundesamt.de/sites/default/files/medien/384/bilder/2_abb_wasserverwendung-hh_2013-09-09_neu_0.png

[20] Ulrich, Traugott: *Kosten senken durch effizienten Einsatz von Pumpen – Systemauslegung und technischen Innovationen - entscheidend für die Nutzung von Potenziale*. – URL http://www.sam-rlp.de/fileadmin/pdf/seminarbeitraege_pius_2013/8_Ulrich_Effizienzpotenziale-Pumpen.pdf

[21] Vielma, Juan P. ; Ahmed, Shabbir ; Nemhauser, George: Mixed-integer models for nonseparable piecewise-linear optimization: Unifying framework and extensions. In: *Operations research* 58 (2010), Nr. 2, S. 303–315

[22] Wilo, SE: *Wilo Online Katalog*. – URL http://productfinder.wilo.com/de/DE/start