TECHNISCHE UNIVERSITÄT WIEN
Vienna University of Technology

DIPLOMARBEIT

Optimierungsmethoden bei mehrstufigen Optimierungsproblemen

Ausgeführt am Institut für
Wirtschaftsmathematik
der Technischen Universität Wien

unter der Anleitung von
Ao. Univ. Prof. Dipl.-Ing. Dr.techn. Alexander Mehlmann
und
Senior Lecturer Dipl.-Ing. Dr. techn. Josef Leopold Haunschmied
als verantwortlich mitwirkenden Assistenten

durch
Sebastian Wörgötter
Grünentorgasse 18/6
1090 Wien

Danksagung

Ich bedanke mich in erster Linie bei meiner Frau und meinen beiden Töchtern, die mir stets die Kraft und die Motivation gegeben haben, diese Arbeit abzuschließen.

Weiters möchte ich meinen Eltern danken, die mich während meines ganzen Studiums unterstützt haben, damit ich den Fokus auf das Studium richten konnte.

Zuletzt möchte ich mich bei Senior Lecturer Dipl.-Ing. Dr. techn. Josef Leopold Haunschmied bedanken, der mir jederzeit bei Fragen und Problemen geholfen und mich hervoragend betreut hat.

Inhaltsverzeichnis

Tabellenverzeichnis

Abbildungsverzeichnis

Abstract Deutsch

Diese Arbeit befasst sich mit einem Güter-Verteilungsproblem in Kombination mit einem Lokations-Allokationsproblem eines Reparaturnetzwerks. Konkret wird ein Netzwerk von Fliegerhorsten betrachtet. Dabei ist die Position der Fliegerhorste gegeben und es wird die kostenminimierende Verteilung von reparierbaren Ersatzturbinen gesucht. Zusätzlich sollen schadhafte Ersatzturbinen in Werkstätten repariert werden und wieder in den Verteilungsprozess zurückgeführt werden. Dabei wird untersucht, ob es günstiger ist, bei jedem Fliegerhorst eine Werkstatt zu errichten und die Reparatur vor Ort durchzuführen, oder ob es eine Kostenersparnis bringen kann, wenn nur ausgewählte Fliegerhorste eine Werkstatt erhalten und die Reparatur zum Teil ausgelagert wird. Demnach liegt neben der Verteilung der Ersatzturbinen ebenfalls die Positionierung von Werkstätten bei den Fliegerhorsten im Fokus der Optimierung. Durch die Aggregation von taktischen und strategischen Entscheidungen und der damit verbundenen Komplexität des Problems (NP-schwer) dazu führt, dass eine Lösung mittels mathematischer Modellierung nur für sehr kleine Referenzmodelle möglich ist. Es wird gezeigt, dass eine Teilung der Optimierung in zwei Schritte die Komplexität des Modells ausreichend verringert, sodass die Problemstellung auch für große Netzwerke gelöst werden können. Es werden verschiedene zweistufige Lösungsmethoden getestet und die Genauigkeit sowie die Rechenlaufzeiten der Lösungen verglichen. Für einen Benchmark werden Ergebnisse aus einer simulierten Abkühlung herangezogen, die zeigen, dass der Informationsverlust aufgrund der Teilung in zwei Stufen zu einem starken Genauigkeitsverlust führt.

Abstract Englisch

This diploma thesis analyzes a combination of an inventory distribution problem with a location allocation problem. The reference model is a network of airfields. The position of the airfields is given and the target is the cost minimizing distribution of repairable turbines for replacement. The defective turbines shall be repaired at repair facilities and redistributed in the network. Therefore, the second target of the optimization is the location-allocation of these repair facilities. It will be compared if every airfield should have its own repair facility or if the aggregation of the reparation to facilities at chosen airfields will save costs. The combination of tactical and strategic decisions raises the complexity of the problem and cause that mathematical programming only solves small reference models (np-hard). It will be shown that the splitting of the model into two echelons reduces the complexity so that large networks can also be solved. Different variations of two-echelon-models will be tested. Moreover, the accuracy and the runtime will be compared. For a benchmark I use the results of a simulated annealing algorithm. The comparison of these results with the two echelon models will contribute to the conclusion that the loss of information caused by the splitting into two echelons leads to an unsatisfying loss of accuracy.

Kapitel 1

Einführung

In der heutigen Zeit, vor allem hinsichtlich der finanziellen Anspannung der Wirtschaft, ist die Suche nach einer optimalen Kostenpolitik und die effiziente Nutzung der Ressourcen im zentralen Fokus eines erfolgreichen Unternehmens. Diese optimierenden Überlegungen beginnen im Idealfall schon bei der Planung von langfristigen Entscheidungen, wie beispielsweise die strategisch günstigste Lokalisierung und Positionierung einer Fabrik oder eines Verteilungszentrums. Kommt es dabei zu groben Fehlentscheidungen kann es möglicherweise langfristig zu höheren Umsatz- und vor allem Gewinneinbußen führen.

Abgesehen davon sollte ein Unternehmen neben diesen wichtigen strategischen Entscheidungen auch die schneller anpassbaren und flexibleren, die sogenannten taktischen Entscheidungen, wie beispielsweise die Anzahl der Mitarbeiter an einem Standort, berücksichtigen. Diese sind zwar grundsätzlich einfacher an die Anforderungen des Unternehmens anzupassen (Mitarbeiteranzahl erhöhen), jedoch bringt diese Flexibilität ebenfalls die Schwierigkeit, dass deren Vorhersehbarkeit von vielen unbekannten Größen sowie Messungenauigkeiten abhängen. Die strukturelle Komplexität kann in vielen Fällen nur durch Schätzer in der Optimierung umgesetzt werden und Bedarf eine Abstraktion in der Modellierung.

Viele Entscheidungen in solchen Modellen gehen ebenfalls über die strategischen sowie die taktischen Entscheidungen. Durch diesen doppelten Einfluss, in einer ersten Stufe in die strategischen Komponenten und in einer weiteren Stufe in die taktischen, existiert eine inhärente Zweistufigkeit. Diese bietet die Basis für die Erwartung, dass eine mehrstufige Modellierung ebenfalls sinnvoll sein könnte.

In dieser Arbeit sollen verschiedene Optimierungsverfahren an einem Beispiel eines mehrstufigen Optimierungsproblems verglichen werden. Zuerst soll überprüft werden, ob sich das Modell trotz der Komplexität für verschieden große Referenzmodelle mit einstufiger mathematischer Programmierung lösen lässt. Sofern dieser Ansatz für zumindest ein Referenzmodell keine Lösung berechnen kann, sollen einige zweistufige Algorithmen mit heuristischer, mathematischer und gemischter Programmierung verglichen und die Qualität der Lösung

beurteilt werden, um für zukünftige Anwendungen eine mögliche Adaption auf äquivalente Modelle zu ermöglichen.

1.1 Problemstellung Allgemein

Um die taktischen und strategischen Entscheidungen von Optimierungsverfahren in einem Modell zu vereinen, sollen zwei in der Literatur oft getrennt behandelte Optimierungsprobleme gemeinsam betrachtet werden. Zum einen gibt es sogenannte Lokations-Allokationsprobleme, die sich mit den strategischen langfristig zu entscheidenden Komponenten, wie beispielsweise der Standortwahl, zu beschäftigen. Dabei sollen z.B. Verteilungszentren oder Werkstätten kostenminimierend bzw. die Nachfrage befriedigend positioniert werden. Zum anderen gibt es die Güter Verteilungsprobleme, die sich mit den taktischen bzw. kurzfristigen Entscheidungen beschäftigen. Dabei soll unter anderem die optimale Verteilung der Güter, zum Beispiel ein Ersatzteil oder mehrere Ersatzteile, in einem Verteilungsnetzwerk bestimmt werden.

1.2 Lokations-Allokationsprobleme

Lokations-Allokationsprobleme beschäftigen sich mit der Aufgabe aus einem Pool von möglichen Standorten aufgrund bestimmter Anforderungen (in den meisten Fällen die Nachfrage nach einem gewissen Gut bzw. einer Dienstleistung) die optimale Menge und Allokation von Geschäften zu finden, damit mit möglichst wenig Aufwand die zugrundeliegenden Anforderungen erfüllt sind.

Somit wird bei Allokations-Optimierungsverfahren die strategische Frage behandelt, wo es für ein Unternehmen am effizientesten ist, Standorte für Ihre Lager, Fabriken bzw. Geschäftsstellen zu eröffnen oder positionieren. Diese grundlegenden Überlegungen sind für ein Unternehmen essentiell und müssen mit besonderem Bedacht überlegt werden. Wenn man bedenkt, dass beispielsweise die Eröffnung einer Fabrikhalle mit großen finanziellen Aufwänden verbunden ist, muss die Entscheidung dafür langfristig rentabel sein. Daher muss bei diesen Problemstellungen genau überlegt werden, welchen Entscheidungsvariablen das System unterliegt und welche Parameter das System in welchem Ausmaß beeinflussen. Zuerst muss die Menge der möglichen Standorte gewählt werden. Je nach Möglichkeit sollten diese für ein optimales Ergebnis von vorne herein die Orte der zu beliefernden Geschäfte berücksichtigen. Letzteres definiert die zweite wichtige Menge. Anhand der Distanz, der Transportkosten, der Kapazitäten der Fabriken sowie der Nachfrage der einzelnen Geschäfte soll das kostenminimierende System berechnet werden. Ein typisches Allokations-Modell wird unter anderem von Qian Zhang, Xiangpei Hu [9] definiert. Sie haben sich für einen heuristischen Lösungsansatz mit vorhergehender Wavelet-Analyse entschieden, um Schwankungen in der Nachfrage besser einfließen zu lassen.

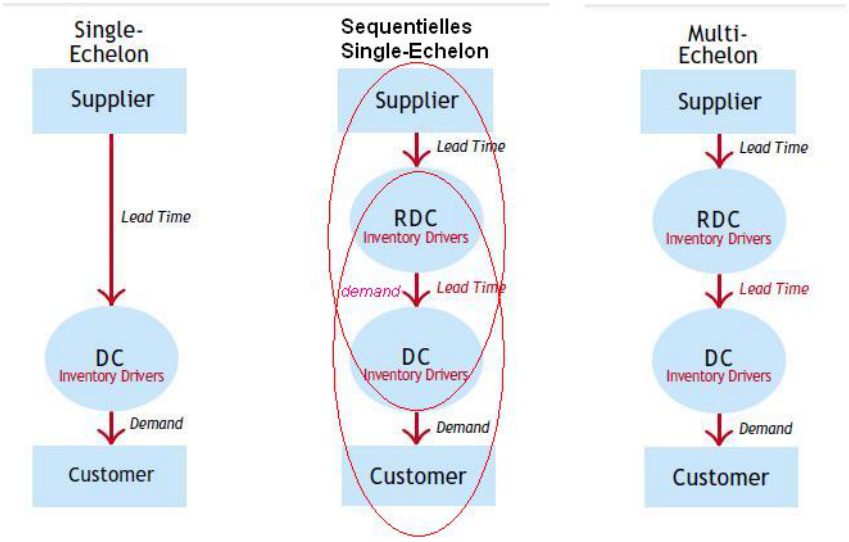

Abbildung 1.1: Single und Multi-Echelon Varianten

1.3 Güter-Verteilungsprobleme

Neben der Allokation von den Verteilungszentren ist die optimale Verteilungs- bzw. Nachbestückungspolitik der von dem Verteilungszentrum vertriebenen Güter ein wichtiger Punkt. In der Literatur findet man viele Abhandlungen von zwei wichtigen Unterkategorien: Den (sequenziell) einstufigen (Single-Echelon) bzw. mehrstufigen (Multi_Echelon) Systemen. Eine grafische Illustration von Lee [15] in Abbildung 1.1 zeigt die drei verschiedenen Echelon Varianten.

1.4 Single-Echelon Modelle

Single-Echelon Modelle besitzen zwischen Anbieter und Konsument lediglich ein Verteilungszentrum (Geschäft). Daher entsteht nur eine zu betrachtende Stufe. Dieses Verfahren lässt sich aber ebenso auf mehrstufige Systeme anwenden. Bei den sequenziellen einstufigen Systemen werden in einem Güter-Verteilungsnetzwerk die einzelnen Stufen (Echelons) jeweils eigenständig wie eine Single-Echelon betrachtet und die Höhe des Inventars gesondert optimiert. Zum Beispiel könnte ein zweistufiges System mit einer Fabrik, die die Güter an mehrere Warenlagerhäuser liefert, die wiederum diese an die Wiederverkäufern verteilen mit diesem Verfahren optimiert werden. Die erste Stufe betrachtet die Nachfrage der Wiederverkäufer, die zweite Stufe die Verteilung der Güter von der Fabrik zu den Warenhäusern.

So betrachten Cesaro und Pacciarelli [12] mit einer gestaffelten Single-Echelon Optimierung die optimale Verteilung von Ersatzteilen an einem Beispiel einer

italienischen Logistikfirma, die diese an Flughäfen verteilt. Ein interessanter Aspekt ist die Mitberücksichtigung von einer schnellen Ersatzlieferung, wenn es bei einem Verteilungszentrum keine Ersatzteile mehr gibt. Optional kann eine schnelle Übernacht-Lieferung von einem anderen Verteilungszentrum den Lieferrückstand kompensieren. Reddy, Narayanan and Pandian [13] nutzen ebenfalls die Möglichkeit, die Güter nicht nur von einer Stufe in die nächste (Warenlager zu Verkäufer), sondern auch in einer Stufe untereinander zu verteilen (Verkäufer zu Verkäufer), sofern der Bedarf gegeben ist. Aufgrund der gesonderten Betrachtung der einzelnen Stufen kann es möglicherweise zu höheren Lagerbeständen kommen, als bei einer Gesamtbetrachtung des Systems (Multi-Echelon).

1.4.1 Multi-Echelon Modelle

Im Unterschied zu den Single-Echelon Modellen betrachten Multi-Echelon Modelle das gesamte Modell mit allen Stufen auf einmal und berücksichtigen. somit die Nachfrage aller Stufen gemeinsam. Köchel und Nieländer [14] betrachten ein System mit 5-Stufen. Somit hat das Produkt von Produktion bis Verkauf 5 Stationen mit jeweils eigenen Nachfragebedürfnissen zu berücksichtigen. Lee [15] vergleicht die sequentielle Single-Echelon Optimierung mit der Multi-Echelon Variante. So betrachtet der sequentielle Test immer nur das optimale Level an Gütern bei der aktuellen Stufe (Warenlager, Verteilungszentrum). Im Gegenzug dazu wird bei der Mulit-Echelon Variante immer die Nachfrage und die Service-Ziele des Endkunden bei minimalem Inventar betrachtet [15]. Ebenso ist es nur im Multi-Echelon Modell möglich, die Kostenauswirkungen einer Stufe mit dem optimiert zugewiesen Inventar auf die nächste zu berücksichtigen, da dass gesamte System auf einmal betrachtet wird. Im sequenziellen ist dies nicht möglich [15]. Als letztes möchte ich noch die Möglichkeit, die Nachbestellungen zwischen den Stufen zu synchronisieren, als Vorteil der Multi-Echelon Optimierung erwähnen. Dadurch werden unnötige Lücken in der Nachlieferung vermieden [15].

1.4.2 Location-Allocation kombiniert mit Güter-Verteilungsproblemen

Die vorweg beschriebenen Modelle haben in der Modellierung immer nur die Allokation oder die Verteilung der Güter berücksichtigt. Dies kann aufgrund der gegebenen Struktur des zu optimierenden Systems sinnvoll und ausreichend sein. Wenn es aber das Ziel ist, ein neues Netzwerk für die Verteilung eines gewissen Gutes zu erzeugen, wäre es interessant, diese beiden Modelle zu aggregieren und in einem System die Zwischenstationen (Verteilungszentren, Lagerhäuser etc.) im Güter-Verteilungsprozesses nicht nur bestücken, sondern auch bestimmen zu können. Wenn zum Beispiel eine Auswahl an möglichen Verteilungszentren variabel bestimmt werden kann, ist es auch möglich in Abhängigkeit von der Nachfrage, den verfügbaren Gütern und der Geschwindigkeit der Nachbestückung die optimal angepasste Menge und optimale Positionierung an Verteilungszentren zu bestimmen. Im Vergleich zu einem System, wo diese schon fix vorgegeben

sind, ist zu erwarten, dass die gleichzeitige Betrachtung der strategischen Positionierung gemeinsam und in gegenseitiger Abhängigkeit mit der Verteilung der Güter, die Ressourcen optimaler genutzt werden können und dadurch die Ergebnisse gewinnbringender sind.

Daskin, Snyder und Berger [16] beschreiben in ihrer Arbeit *Facility Location in Supply Chain Design* einige verschiedene Modellvarianten, von solchen kombinierten Varianten. Das Modell von Rappold und Van Roo [1] beschäftigt sich mit einem Problem für reparable Ersatzteile in einem Reparaturnetzwerk. Dieses soll in einem Multi-Echelon-Problem in Kombination mit einem Allokationsproblem gelöst werden.

1.5 Ziel der Arbeit

In dieser Arbeit möchte ich den Fokus auf ein mehrstufiges Optimierungsproblem setzen, wo ein Allokation-Lokation gemeinsam mit der Güter-Verteilung betrachtet wird. Dafür ziehe ich das Modell von Rappold und Van Roo heran [1]. Es soll das Modell vorgestellt und mit mathematischer Programmierung umgesetzt werden. Anhand verschiedener Referenzmodelle soll die Lösbarkeit und die Performance getestet und mit dem von Rappold und Van Roo definierten mehrstufigen heuristischen Lösungsansatz verglichen werden.

Vorweg werde ich im nächsten Abschnitt das Modell im Detail beschreiben.

Kapitel 2

Das Referenzmodell von Rappold und Van Roo

Ein anschauliches Beispiel für ein mehrstufiges Optimierungsmodell bietet die wissenschaftliche Arbeit *Design multi-echelon sevice parts networks with finite repair capacity* von James A. Rappold und Ben D. Van Roo, zwei mathematischen Wissenschaftlern aus den Vereinigten Staaten. Darin beschreiben Sie ein Netzwerk von Fliegerhorsten der amerikanischen Airforce (USAF) und befassen sich mit der Kostenoptimierung in Hinblick auf den Ersatz von defekten Turbinen, sowie deren Reparatur. Das Modell setzt sich aus einem Lokations-Allokationsproblem und einem Güter-Verteilungsproblem zusammen.

In der ersten Stufe beschäftigt sich die Allokation mit den strategischen Entscheidungen des Modells: Es wird die kostenminimierende Auswahl an Fliegerhorsten, die mit Werkstätten ausgestattet werden, sowie der Zuordnung der werkstattfreien Fliegerhorste zu den Werkstätten bestimmt. Dabei wird jeder Fliegerhorst von exakt einer Werkstätte bedient. Somit erhalten wir für jede Werkstatt ein Subnetzwerk. In der zweiten Stufe werden die taktischen Komponenten gelöst: Es wird die kostenminimierende Anzahl an Reparaturteams, sowie die Lagerbestände der Horste sowie der Werkstätten für jedes Subnetzwerk bestimmt.

Ein wichtiger Aspekt des Modells ist der ausschließliche Fokus auf Vernetzung der Fliegerhorste und den Werkstätten und die Reparatur und den Ersatz von schadhaften Turbinen. Die Optimierung des Flugverkehrs oder der Flugrouten werden in diesem Modell nicht betrachtet, und der Flugverkehr ist nicht zwangsweise auf das Netzwerk beschränkt.

Grundsätzlich gibt es verschiedene Möglichkeiten, wie Organisationen wie die amerikanische Airfoce ihr Reparatursystem organisieren könnte. Zum einen könnten Sie die Reparaturen direkt an den einzelnen Fliegerhorsten durchführen lassen. Das führt zu schnellem Ersatz ohne anfallende Transportkosten. Der Preis dafür sind hohe Erhaltungskosten, da an jedem Standort eigene Reparaturteams sowie eine eigene Werkstatt verfügbar sein müssen und bei unerwar-

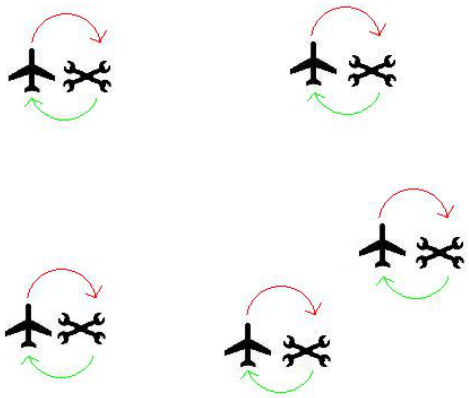

Abbildung 2.1: Reparaturnetzwerk mit Selbstbedienung

tet hohem Ausfall könnte möglicherweise ein langer Lieferrückstand entstehen, wenn es keinen Austausch mit anderen Standorten gibt.

Alternativ könnte die Reparatur ausgelagert werden, indem eine Auswahl an Fliegerhorsten eine Werkstätte erhalten und als Reparaturstätte für ausgewählte, zugordnete Standorte dient. Diese senden Ihre zu reparierenden Maschinen an die Werkstätte. Wenn diese eine Ersatzmaschine auf Lager hat, wird eine Ersatzmaschine von der Reparaturstätte an den Fliegerhorst geschickt. Ansonsten entsteht ein Lieferrückstand und der Fliegerhorst muss warten bis eine Maschine einsatzfähig aus der Werkstätte kommt, um an den Fliegerhorst verschickt zu werden. Währenddessen betritt die zu reparierende Maschine den Reparaturprozess. Ist ein Reparaturteam frei, wird sie sofort repariert, ansonsten reiht sich die Maschine am Ende der Warteschlange ein. Nach der Reparatur wird sie ins Lager gebracht, bis von einem nächsten Fliegerhorst eine Ersatzanfrage kommt. Dadurch verringern sich die Kosten für Reparaturteams. Dafür steigen die Transportkosten und die Lieferverzögerungen durch die Transportzeiten.

In den Abbildungen 2.1, 2.2 und 2.3 sind verschiedene Beispiele für Reparaturnetzwerke dargestellt. Ich beginne mit dem erwartungsgemäß teuersten Modell der Selbstbedienung (Abbildung 2.1). Selbstbedienung heißt, dass jeder Flughafen eine eigene Werkstatt erhält. Somit geht der Ersatz von defekten Turbinen schnellstmöglich und die Kosten für Lieferrückstände sind minimal. Als Preis dafür kommen die Fixkosten einer Werkstatt bei jedem Fliegerhorst, sowie die variablen Kosten, abhängig von der Anzahl an Reparaturteams dazu.

Die Abbildung 2.1 zeigt ein Netzwerk mit einer Werkstatt. Alle Standorte müssen somit von einer Werkstätte bedient werden. Dieses System kann unter Umständen effizient sein, sofern die Distanz zwischen den einzelnen Standorten nicht zu allzu langen Transportzeiten führt und es dadurch zu einem sehr langen verzögerten Ersatz kommen kann (Abbildung 2.2).

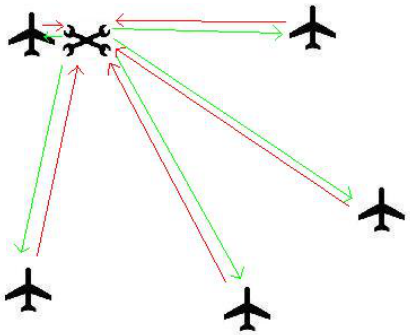

Abbildung 2.2: Reparaturnetzwerk mit einer Werkstatt

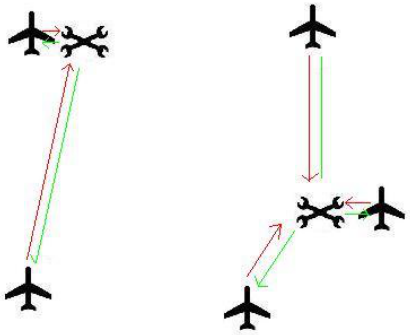

Abbildung 2.3: Reparaturnetzwerk mit zwei Werksttten/Subnetzwerken

Für sehr große Netzwerke, wo die einzelnen Horste weit auseinander liegen, würde es zumindest für einige zu sehr hohen Transportkosten und gleichzeitig sehr langen Lieferverzögerungen kommen. Also wäre es möglicherweise sinnvoll nicht nur eine Werkstätte, sondern mehrere zu bestimmen. Da aber die Öffnung einer Werkstätte mit hohen zusätzlichen Kosten verbunden ist, bedarf es einem System, dass alle diese Faktoren für die Kosten miteinbezieht um ein kosteneffizientes Netzwerk zu generieren.

In der Abbildung 2.3 ist ein Netzwerk mit zwei Werkstätten und entsprechenden zwei Subnetzwerken abgebildet. Jede Werkstatt hat die ihm zugewiesenen Standorte exklusiv für sich. Das bedeutet, dass ein Standort nicht von beiden Werkstätten gemeinsam bedient werden kann. Aufgrund dieser Eigenschaft sind diese beiden Netzwerke unabhängig und haben daher keinen direkten Einfluss auf einander. Nachdem in der ersten Stufe die Standorte fix gewählt und die Zu-

ordnung geschehen ist bleibt noch ein Allokationsproblem der Ersatzturbinen.

Kapitel 3

Das Problem

Wir betrachten hier eine Kombination aus einem Allokationsproblem von Werkstätten mit einem Verteilungsproblem von vorhandenen Ersatzturbinen. Genauer werden wir ein System mit nur einem Maschinentyp, den Flugzeugturbinen, formulieren. Das heißt, dass alle Flugzeuge bei allen Fliegerhorsten mit dem selben Turbinentyp ausgestattet sind. Im Allgemeinen werden oft mehrere verschiedene Ersatzteile betrachtet. Ich werde mich aber auf das eine sehr teure Ersatzteil in dieser Arbeit beschränken und die Reparatur sowie die Ersatzlieferungen nur eines Inventar-Typen betrachten. Die Ausformulierung für mehrere Ersatzteile läuft abgesehen von wenigen Ergänzungen äquivalent, erschwert aufgrund der wachsenden Komplexität die Ausformulierung.

3.1 Variablen

Um die relevanten Entscheidungen zu charakterisieren, sollen vorab einige Fragen helfen

- Wie viele Werkstätten müssen geöffnet werden?
- Wo sollten die Werkstätten positioniert werden?
- Welche Fliegerhorste sollen welchen Werkstätten zugewiesen werden?
- Wie geschieht der Turbinenersatz?
- Wie sollen die Ersatzmaschinen verteilt sein?
- Wie viele Teams werden in den jeweiligen Werkstätten benötigt um die Nachfrage ausreichend zu bedienen?

Daraus ergeben sich folgende Variablen als die Entscheidungselemente des Modells:

- Werkstätte bei Fliegerhost j

- Zuweisung des Fliegerhorsts zu einer geöffneten Werkstätte
- Anzahl an Reparaturteams pro Werkstatt
- Verteilung der Ersatzturbinen auf die Subnetzwerke
- Verteilung der Ersatzturbinen innerhalb jedes Subnetzwerkes

Sowie in der Einleitung erwähnt, zeigen die Fragen sowie die Entscheidungsvariablen schon die Zweistufigkeit des Problems. Der ersten drei Fragen beziehen sich auf die strategischen bzw. längerfristigen Entscheidungen. Besonders die Positionierung der Werkstätten sollte zu Beginn optimal bestimmt werden können, da nachträgliche Änderungen sehr teuer sind und nicht kurzfristig verändert werden können. Die beiden letzten Fragen beziehen sich auf die taktischen, kurzfristigen Entscheidungen. Diese sind im Allgemeinen flexibler und einfacher zu ändern. Diese Flexibilität ist zeitgleich schwieriger zu modellieren, da die Entscheidungen von stochastischen Größen abhängig sind.

3.2 Parameter

Nach den Entscheidungen sind für die Problemstellung noch einige Parameter von Bedeutung. Hier sollen ebenfalls ein paar Fragen für die Modellierung dienen:

- Wie hoch sind die Transportkosten für Ersatzturbinen von fremden Fliegerhorsten?
- Wie viel Kapital muss für das Netzwerk für die Errichtung der Werkstätten aufgebracht werden?
- Kann das Netzwerk auch die Konsumbedürfnisse der zugeordneten Fliegerhorste decken?
- Wie hoch sind die Einbußen durch ausständige Ersatzlieferungen?

Daraus ergeben sich folgende Größen als die Hauptkomponenten der Kosten des Systems:

- Fixkosten jeder Reparaturstätte
- Reparaturteamkosten (inkl. Equipment etc.)
- Transportkosten
- Erhaltungskosten des Inventars
- Kosten durch Lieferrückstände

Ersatzpolitik für schadhafte Turbinen.

Die Ersatzpolitik der Turbinen spielt bei der Modellierung ebenfalls eine sehr wichtige Rolle. In diesem Beispiel wird bei einer schadhaften Turbine eine einsatzfähige, sofern vorhanden, aus dem Lager ersetzt. Die defekte Turbine wird sofort zur Werkstatt geschickt und das Lager sendet eine Bestellung an die Werkstatt. Diese sendet wiederrum eine einsatzfähige Turbine an das Lager. Ist bei einer Anfrage keine Ersatzmaschine im Lager vorhanden, kommt es zu einem Ersatzrückstand und es muss auf eine Turbine aus der Werkstatt gewartet werden. Wenn bei der Werkstatt eine Ersatzturbine vorhanden ist, bestimmt lediglich die Transportzeit die Ersatzverzögerung. Ist das Lager bei der Werkstatt jedoch ebenfalls leer, kommt es zu einem Lieferrückstad und es muss auf eine Turbine aus dem Reparaturprozess gewartet werden.

Das Ziel wird es sein, diese Kosten insgesamt zu minimieren und eine langfristig effizientes System zu erzeugen, in dem die erwarteten Lieferrückstände möglichst klein und kurz gehalten sind, ohne überdurchschnittlich hohe Kosten in den Werkstätten selbst zu haben. Ebenso soll die Lösung des Problems auch in Bezug auf die Rechenleistung und der benötigten Rechenzeit effizient sein. Nach Ausformulierung des Lösungsansatzes von Van Roo und Rappold sollen in weiterer Folge alternative Ansätze unter diesen Aspekten verglichen werden.

3.3 Ziel der Optimierung

Ein weiterer wichtiger Punkt ist die Zielsetzung die Kosten für das Modell zu minimieren. Somit handelt es sich um ein monokriterielles Problem mit einem Ziel. Daher werden die Entscheidungen auf die Kostenminimierung beschränkt.

Es sei noch einmal darauf hingewiesen, dass das Gesamtproblem auf zwei Stufen aufgeteilt ist: Zuerst wird die Allokation der Werkstätten und die Zuordnung der Fliegerhorste zu den möglichen Werkstätten bestimmt. Anhand des daraus resultierenden Ergebnis werden in der zweiten Stufe die Maschinen in dem Netzwerk verteilt, sodass bei jedem Fliegerhorst ausreichend Ersatzmaschinen zur Verfügung stehen, um einen Ausfall schnell zu ersetzen. Da aber die Anzahl der Maschinen beschränkt ist und die Haltungskosten für jede Maschine miteinbezogen wird, soll das Modell eine ausgewogene Verteilung als Ergebnis liefern, sodass die Mischung aus Haltungskosten und Kosten durch Ersatzverzögerung die Gesamtkosten minimiert.

Kapitel 4

Annahmen und Verteilungen der einzelnen Komponenten

4.1 Relevanz der Entscheidungen

Die Flexibilität in der nachträglichen Änderung der Variablen bestimmt in gewisser Weise auch die Relevanz der Entscheidungen. Die schwerfälligste Variable ist die Allokation der Werkstätten. Einmal entschieden. gibt es kaum Möglichkeiten den Standort zu tauschen, da es mit immensen Kosten verbunden wäre. Daher muss besonders diese Entscheidung mit besonderem Bedacht geschehen. Die Zuordnung der Fliegerhorste zu den Werkstätten lies sich unter Extrembedingungen, wie zum Beispiel einem Ausfall von überdurchschnittlich vielen Maschinen, zwar kurzfristig ermöglichen, würde aber das System sehr stark ins Schwanken und möglicherweise zum Kollabieren bringen.

Die Anzahl an Reparaturteams lässt sich zwar mathematisch schnell anpassen, ohne schwerwiegende Eingriffe ins System vorzunehmen, aber in der Praxis sollte vorab eine ausgewogene und funktionierende Teamstruktur berechnet sein. Es ist für kein Unternehmen möglich permanent die Anzahl der Mitarbeiter an die momentan optimale Balance anzupassen. Wenn ein Unternehmen ständig Angestellte entlässt, kann selbstverständlich kein gesundes Arbeitsklima entstehen. Andererseits geht es auch nicht prompt genug neue kompetente Mitarbeiter zu finden. Trotzdem ist diese Variable einfacher anzupassen.

Die Verteilung der Ersatzmaschinen ist die flexibelste Variable und lässt sich mit verhältnismäßig wenig Aufwand ändern.

Verteilungen

Für die Ausfall- bzw. Fehlerrate der Maschinen bei den Flughäfen nehmen wir eine Poissonverteilung an. Rappold und Van Roo [1] haben in ihrer Arbeit an-

hand von Daten der USAF getestet, dass diese Annahme angemessen ist.

Die Reparaturzeiten werden als exponentialverteilt angenommen, wobei sie unabhängig und identisch verteilt sind. Wieder haben Rappold und Van Roo [1] geschrieben, dass diese Verteilung nicht nur praktisch in der Handhabung, sondern auch aus den Daten USAF empirisch evident ist, da die Exponentialverteilung die Daten sehr gut approximiert. Leider liegen die exakten Daten der USAF nicht vor, sodass dieser Annahme kein Beweis beifügt werden können.

Die Transportzeiten zwischen den einzelnen Flughäfen und den Werkstsätten sollen als konstant angenommen werden, da kleine Varianzen im System kaum gewichtig Einfluss haben. Man könnte ggf. überlegen, ob man zur fixen Transportzeit einen kleinen Prozentsatz an Verzögerung als Sicherheitspuffer bzw. Unsicherheitsfaktor hinzuaddiert.

Ersatzpolitk und Warteschlagen

Die Ersetzungspolitik folgt dem „first come - first serve“ Prinzip. Es besagt, dass in der Reihenfolge in der die Fliegerhorste die Maschinen anfordern, sie diese auch erhalten. Das Entspricht einer Bedingung verglichen mit einer Warteschlange vor einem Schalter oder an der Kassa im Supermarkt. Ein Kunde wird abhängig von den vor ihm eingetroffenen Kunden in der Warteschlange bedient. Erst wenn alle, die sich vor ihm eingereiht haben, abgefertigt sind, ist er an der Reihe. Es wäre ggf. auch möglich unter gewissen Aspekten einzelnen Kunden eine gewisse Präferenz zuzuteilen. Wenn zum Beispiel ein Flughafen eine sehr hohe Ausfallrate aufweist, könnte eine gewisse Vorzugskomponente eingebaut werden, die die Belieferung öfter zuteilt. Doch letztere Überlegung soll in dieser Arbeit nicht behandelt werden, da durch die Möglichkeit mehrere Werkstätten zu öffnen, höhere Nachfragen zufriedenstellend abgedeckt werden.

Eine weitere wichtige Restriktion im Modell ist, dass jeder Flughafen genau einer Werkstätte zugewiesen werden soll und dass es in jeder Werkstätte eine endliche Anzahl an (in der Leistung und Ausstattung) identischen Reparaturteams gibt, die in den jeweiligen Werkstätten parallel arbeiten.

Die Anzahl der gesamten Ersatzmaschinen im System wird a-priori bestimmt und wird als Beschränkung und Nebenbedingung definiert. Angenommen dieses System soll für eine bestehende Firma angewendet werden. In diesem Fall ist die Anzahl der Maschinen schon vor Optimierung gegeben und kann nicht als Variable eingesetzt werden, da die bestehenden Maschinen nicht einfach vernichtet werden könnten. Sofern es aber möglich ist, die Anzahl an Ersatzturbinen zu reduzieren (durch Verkauf beispielsweise) könnte die aktuell verfügbare Anzahl als obere Schranke dienen. Im gegenteiligen Szenario, wo die Anzahl der verfügbaren Ersatzmaschinen im Optimum zu gering ist, würde das System statt Maschinen zu „kaufen“ mehr Reparaturteams einsetzten, damit die entstehenden Lieferrückstände und die daraus resultierenden Kosten minimiert werden. Hier wäre die Überlegung interessant, anstelle der fixen Annahme der Anzahl an Ersatzmaschinen, Nachkäufe zu ermöglichen. Wenn es Kosteneffizienter ist, mehr Turbinen als vorhanden zur Verfügung zu haben könnte das System diese hinzufügen. Diese Überlegung würde auch in der Praxis Sinn machen, sofern

alle mit dem Nachkauf verbunden Kosten miteinbezogen würden und das Ersatzteil schnell und einfach nachlieferbar wäre. Da aber die Turbinen sehr teure und große Ersatzteile sind, ist auch eine Nachbestellung mit langen Wartezeiten und hohen Kosten verbunden. Daher werde ich diese Variante nicht weiter verfolgen. Im Unterschied zu Rappold und Van Roo habe ich im Referenzmodell dieser Arbeit die Anzahl an Ersatzturbinen als obere Schranke angenommen und somit Verkäufe von teuren Lagerüberschüssen zu erlaubt. Die Lagerhaltung von den Ersatzturbinen ist sehr teuer und ebenfalls ein elementarer Kostentreiber im Modell. Daher scheint es mir legitim, nicht notwendige Lagerbestände zu verkaufen.

Ersatzpolitik bei alternativen Problemstellungen

Da in diesem Modell an jedem Flughafen dasselbe Turbinenmodell zu ersetzen ist, ist das „first come - first serve“ System ebenfalls sinnvoll, da jede reparierte Maschine an jeden Flughafen geschickt werden kann, unabhängig davon, von welchem die schadhafte Turbine stammt. Hätten wir hingegen ein System mit verschiedenen Maschinentypen könnte wieder eine präferenziernde Politik sinnvoller sein.

Da es sich um sehr teure und große Maschinen handelt, werden die Maschinen auch Stück für Stück und nicht gebündelt ersetzt. Fehlt eine Ersatz-Turbine bei einem Flughafen könnte daraus ein möglicher Ausfall eines Flugzeuges resultierenden und daher immense Kosten verursachen. Dazu sei erwähnt, dass es in einigen Modellen, vor allem bei kleinen und billigen Verkaufsprodukten, eine gebündelte Nachlieferung deutlich günstiger kommen wird als ein sofortiger Stück für Stück Ersatz.

Als Beispiel dafür können wir uns einen Filter einer Maschine vorstellen. Geht einer kaputt beziehungsweise ist zu reinigen, wird er vom Lager ersetzt. Erst wenn ein gewisser Lagerbestand unterschritten wird, werden Blockweise Ersatzfilter bestellt bzw. die Verschmutzten zum Reinigen oder Recycling geschickt.

4.2 Mathematische Modellierung

Sei R die Menge aller möglichen Werkstätten mit Index $j \in R$. F soll die Menge der Fliegerhorste mit Index $i \in F$.

F^j definiert die Menge der Fliegerhorste die der geöffneten Werkstätte j zugewiesen sind. Die Fehlerrate am Fliegerhorst $i \in F^j$ sind poissonverteilt mit Parameter λ_{ij} und λ_{j0} gibt die gesamte Nachfragerate an die Werkstatt j an. Die Reparaturzeiten der fehlerhaften Turbinen sei für Werkstätte j exponentialverteilt und mit Mittelwert μ_j ident verteilt. Somit ergibt sich ein Warteschlangensystem $M/M/T$, wobei T die Anzahl an Reparaturteams darstellt. Die Reihenfolge der Reparaturen folgt dem „first come, first serve“ Prinzip nachdem sich jede neue zu reparierende Maschine am Ende der Warteschlange anstellt. Die Transportkosten vom Fliegerhorst i zur Werkstatt j sollen konstant als c_{ji}

gelten. Daraus ergibt sich folgendes System:

R Menge aller möglichen Werkstätten

F ist die Menge der Fliegerhorste

K ganzzahlige Indexmengen mit Index k und $k = 0, 1, ..., |K|$ entspricht der maximalen Anzahl an Reparaturteams

Mit folgenden Entscheidungsvariablen:

s_{ji} Anzahl an Ersatzturbinen bei den Fliegerhorsten $_i$

s_{j0} Anzahl an Ersatzturbinen bei der Werkstätte j sofern diese geöffnet ist

$$x_j = \begin{cases} 1, & \text{wenn die Werkstatt } j \text{ geöffnet ist} \\ 0, & \text{sonst} \end{cases}$$

$$y_{ji} = \begin{cases} 1, & \text{Flughafen ist der Werkstatt } j \text{ zugewiesen} \\ 0, & \text{sonst} \end{cases}$$

$$z_{jk} = \begin{cases} 1, & \text{Werkstätte } j \text{ hat } k \text{ oder mehr Teams} \\ 0, & \text{sonst} \end{cases}$$

$z_j = \sum_k z_{jk}$ Anzahl der Reparaturteams bei der Werkstatt j und folgenden Parametern:

λ_{ij} Durchschnittliche Fehlerrate pro Tag der Turbinen bei dem Fliegerhorst $i \in F^j$

$\lambda_{j0} = \sum_{i \in F^j} \lambda_{ij}$ die durchschnittliche Nachfrage an die Werkstatt j

μ_j durchschnittliche Reparaturrate pro Team von der Werkstatt j

l_{ji} konstanten Transportzeiten von j nach i

c_{ji} Transportkosten von j nach i

f_j jährliche Fixkosten um die Werkstatt j zu öffnen

a_{jk} jährlichen Kosten um ein k-tes Reparaturteam bei der Werkstatt j hinzuzufügen

h_{ji} Haltungskosten einer Turbine pro Jahr in der Basis

h_{j0} Haltungskosten einer Turbine pro Jahr in der Werkstätte

π_{ji} Kosten bei Fehlen einer Turbine

S Menge an Ersatzturbinen im System - in meinem Modell eine obere Schranke

Für jedes autonome Subnetzwerk aus einer Werkstätte und der ihr zugewiesenen Fliegerhorste gelten folgende Funktionen:

$D_{j0}(\lambda_{j0}, z_j)$ = Zufallsvariable, stehend für die Nachfrage an Turbinen j während der Reparatur (durchschnittliche Wartezeit + Servicezeit) einer defekten Turbine

$\Psi(z_j, s_{j0}) = E[s_{j0} - D_{j0}(\lambda_{j0}, z_j)]^+$ Erwartete Lagerbestände im statistischen Gleichgewicht bei j

$E[D_{j0}(\lambda_{j0}, z_j) - s_{j0}]^+$ = offene Lieferrückstände einer Werkstätte an die zugeordneten Fliegerhorste im statistischen Gleichgewicht

$\tau_j(z_j, s_{j0}) = E[D_{j0}(\lambda_{j0}, z_j) - s_{j0}]^+ / \lambda_{j0}$ Erwartete Lieferverzögerung bei der Ersetzung der fehlerhaften Turbine, falls ein Lieferrückstand auftritt

$D_{ji}(\lambda_{ji}[\tau_j + l_{ji}])^+$ = Zufallsvariable, stehend für die entstehende Nachfrage, die während der Ersatzzeit=Transport+Verzögerungszeit

$\Gamma_i(\lambda_{ji}, z_j, s_{j0}, s_{ji} = E[s_{ji} - D_{ji}(\lambda_{ji}[\tau_j + l_{ji})]^+$ Erwarteter Lagerbestand bei dem Fliegerhorst i im statistischen Gleichgewicht

$\Phi_i(y_{ij}, z_j, s_{j0}, s_{ji}) = E[D_{ji}(\lambda_{ji}[\tau_j + l_{ji}]) - s_{ji})]$ Erwartete Fehlmengen beim Fliegerhorst i

In dem Modell ist die Größe der Fliegerhorste, in Bezug auf die Anzahl der verkehrenden Flugzeuge über die Fehlerrate λ_{ij} abgebildet. Somit ist die Höhe der Fehlerrate unter anderem auch von der Flugfrequenz abhängig. Einen weiteren Einfluss hat die Größe der Fliegerhorste nicht und muss daher nicht gesondert betrachtet werden.

4.3 Das Minimierungsproblem

In diesem Abschnitt soll aus dem beschriebenen Modell ein Minimierungsproblem modelliert werden. Da die Gesamtkosten minimiert werden sollen, muss überlegt werden, wie die Kosten modelliert werden:

- die fixen Kosten der Werkstätten $\sum_{j \in R} x_j f_j$
- die Kosten der Reparaturteams $\sum_{j \in R} \sum_{i \in F} z_{jk} a_{jk}$

  Dabei ist die degressive Entwicklung der Kosten a_{jk} und somit die Konvexität der Funktion ein wichtiger Punkt, da dadurch EOS (Economies of scales), also positive Skalenerträge entstehen.
- die Transportkosten $\sum_{i \in F} 365 * \lambda_{ji} y_{ji} c_{ji}$

  Da die Fehlerrate λ_{ji} auf täglicher Basis definiert ist, muss die Summe mit 365 multipliziert werden um die jährlichen Ersatzbedarf zu erhalten.
- die Lagerkosten für Maschinen bei den Werkstätten $\sum_{j \in R} \sum_{i \in F} h_{j0} \Psi(z_j, s_{j0})$

  Die Lagerkosten sind Abhängig von dem erwarteten durchschnittlichen Lagerbestand bei den Werkstätten. Diese sind einerseits von der gesamten Anzahl an Ersatzturbinen S_j, die der Werkstätte zugewiesen sind, sowie von der Anzahl an Reparaturteams z_j und der erwarteten Nachfrage λ_{j0} an j. Die letzten beiden Größen steuern die Warteschlange, die anhand der erwarteten Ausfälle die Ersatzgeschwindigkeit bestimmt.
- die Lagerkosten für Maschinen bei den Fliegerhorsten $\sum_{j \in R} \sum_{i \in F} h_{ji} \Gamma_i(\lambda_{ji}, z_j, s_{j0}, s_{ji})$
- die Kosten durch Lieferrückstände bei den Fliegerhorsten $\sum_{j \in R} \sum_{i \in F} \pi_{ji} \Phi_i(y_{ij}, z_j, s_{j0}, s_{ji})$

  Die Lieferrückstände sowie die Lagerkosten bei den Fliegerhorsten werden äquivalent zu den Lagerkosten bei den Werkstätten über die Warteschlange gesteuert und sind daher ebenfalls von den Mittelwerten der Zufallsvariable abhängig.

Die Zielfunktion lautet dann:

$$\min_{x_j,y_{ji},z_{jk},s_{ji},s_{j0}} \sum_{j\in R} x_j f_j + \sum_{i\in F} 365 * \lambda_{ji} y_{ji} c_{ji} + \sum_{j\in R}\sum_{i\in F} z_{jk} a_{jk}$$

$$+ \sum_{j\in R}\sum_{i\in F} (h_{j0}\Psi(z_j, s_{j0}) + h_{ji}\Gamma_i(\lambda_{ji}, z_j, s_{j0}, s_{ji}) + \pi_{ji}\Phi_i(y_{ij}, z_j, s_{j0}, s_{ji}))$$

s.t.

$$\sum_{j\in R} y_{ji} = 1 \quad \forall i \in F$$

$$y_{ji} \leq x_j \quad \forall i \in F \quad \forall j \in R$$

$$\sum_{i\in F} y_{ji}\lambda_{ji} \leq \mu_j \sum_{k\in K} z_{jk} \quad \forall j \in R$$

$$\sum_{j\in R} \left(s_{j0} + \sum_{i\in F} s_{ji} \right) \leq S$$

$$z_{j(k+1)} \leq z_{jk} \quad \forall j \in R \quad k = 0, 1, ..., |K|$$

$$z_j = \sum_{k\in K} z_{jk} \quad \forall j \in R$$

$$s_{ji}, s_{j0} \neq 0 \quad \forall i \in F \quad \forall j \in R$$

$$s_{ij}, s_{j0} \in N \quad \forall i \in F \quad \forall j \in R$$

$$x_j, y_{ji}, z_{jk} \quad \text{binär.}$$

- $\sum_{j\in R} y_{ji} = 1 \quad \forall i \in F$ sichert, dass jeder Flughafen nur einer Reparaturstätte zugewiesen wird, durch
- durch $y_{ji} \leq x_j$ werden Flughäfen nur geöffneten Werkstätten j zugewiesen
- $\sum_{i\in F} y_{ji}\lambda_{ji} \leq \mu_j \sum_{k\in K} z_{jk} \quad \forall j \in R$ sichert, dass die Kapazitäten der Werkstätten j die Nachfrage der zugewiesenen Fliegerhorste bedienen kann
- $\sum_{j\in R} \left(s_{j0} + \sum_{i\in F} s_{ji} \right) \leq S$ ist die Summe der genutzten Ersatzturbinen. Hier sei noch einmal erwähnt, dass ich im Gegensatz zu Rappold und Van Roo die Anzahl an maximal verfügbaren Ersatzturbinen als ober Schranke definiere, da die Tests ergeben haben, dass die meisten Netzwerke im Optimun deutlich weniger benötigen würden als vorhanden.
- $z_{j(k+1)} \leq z_{jk}$ sichert dass eine Erhöhung der Reparaturteams schrittweise geschieht und
- $z_j = \sum_{k\in K} z_{jk}$ bestimmt die Anzahl der Reparaturteams bei der Werkstatt j an.

- anhand von $s_{ij}, s_{j0} \in N$ wird definiert, dass die Ersatzmaschinen positive und ganzzahlige Variablen sind.
- und zuletzt werden die Werte der Werkstätten x_j der Zuweisungen y_{ji} und der Teams z_{jk} als binär fixiert

4.3.1 Sicherheitparameter ρ

Ein weiterer wichtiger Parameter soll als Sicherheit für unerwartet hohe Nachfrage eingebaut werden. Bisher sollte die erwartete Nachfrage eins zu eins von den Reparaturstätten bedient werden:

$$\sum_{i \in F} y_{ij} \lambda_{ji} \leq \mu_j \sum_{k \in K} z_{jk} \quad \forall j \in R$$

Für die erwarteten Ereignisse ist dies auch die kostengünstigste Lösung Wenn wir aber davon ausgehen, dass die Ereignisse sich nicht immer nach den Erwartungswerten ereignen werden, erscheint es sinnvoll, einen Paramater einzuführen und so zu adjustieren, dass bei einem unerwartetem Anstieg der Nachfrage die Reserven zumindest bis zu einem gewissen Grad die Nachfrage bedienen kann. Ansonsten könnte sich eine lange Warteschlange bilden und es würde zu langen Lieferrückständen und daraus resultierenden hohen Lieferrückstandskosten kommen.

Bezeichnet wird dieser Parameter im Modell mit $\rho \in [0, 1]$. Dadurch soll nicht nur die Anzahl an Reparaturteams vorhanden sein, die die erwarteten Ausfälle exakt bewältigt, sondern ein zusätzliches Sicherheitsmaß an Überschuss, damit unerwartet hohe Ausfälle und Schwankungen in der Nachfrage an Reparatur abgefangen werden. Wenn dieser, wie in meiner Umsetzung, auf 0,79 gesetzt ist, bedeutet dies, dass die Kapazitäten während des Durchschnittbetriebs zu 79 % ausgelastet sind und dementsprechend 21% für unerwartete Reparaturanfragen in Reserve behalten. Solange die erwartete Nachfrage in etwa den Erwartungen entspricht, sind dadurch die Kosten für die Reparaturteams teuer als notwendig. Dies wiederum ist der Preis für die Sicherheit, kurzfristige starke Schwankungen problemlos zu bedienen. Daraus ergibt sich eine adjustierte Nebenbedingung mit dem Sicherheitsparameter ρ :

$$\sum_{i \in F} y_{ij} \lambda_{ji} \leq \rho \mu_j \sum_{k \in K} z_{jk} \quad \forall j \in R$$

Laut Rappold und Van Roo [1] verwendet die amerikanische Air Force für $\rho = 0,77$. Das entspricht einer durchschnittlichen Auslastung von 77 % der Reparaturteams, bei der erwarteten Beanspruchung.

Bei diesem Wert ist eindeutig zu erkennen, dass es sich bei der Airforce um eine sehr teure Maschine handelt, wo die Kosten eines nicht umgehend ersetzbaren Ausfalls so hoch sein dürften, dass Sie beinahe ein Viertel mehr Teams in den Reparaturstätten platzieren. Einerseits erhöht dies die Kosten für die Teams selbst, andererseits werden generell die Ersatzverzögerungen vor allem bei durchschnittlichem Bedarf wesentlich kürzer sein. Dies wiederum führt zu geringeren Kosten bei Ausfallen einer Turbine, wodurch die Teamkosten zumindest teilweise kompensiert werden.

4.3.2 Beschränkung des Transports

Weiters führen wir noch eine Beschränkung für die sich im Transport befindlichen Turbinen ein. Bei der Überlegung, dass vor allem bei sehr teuren (und großen) Maschinen der Transport sowie die dafür notwendigen Transportmittel sehr teuer sind, scheint auch hier eine Beschränkung sinnvoll, da die Transportmittel auch nicht infinit sind. Wir definieren V als die obere Schranke der sich im Transport befindlichen Maschinen.

4.3.3 Berechnung der Reparaturzeit sowie der Anzahl an Maschinen im Reparaturprozess

Um den genauen Bedarf an Ersatzmaschinen zu bestimmen, ist die Berechnung der Maschinen im Reparaturprozess (Reparaturzeit + Transportzeit) notwendig. Letztere wird als konstant angenommen, daher ist es interessanter die Reparaturzeit zu betrachten. Rappold und Van Roo gehen davon aus, dass der Reparaturprozess eine $M/M/\kappa$ Warteschlange ist. Das bedeutet, dass die Ankunftsrate, die in diesem Modell der Fehlerrate entspricht, poisson- und die Reparaturrate exponentialverteilt sind [6]. κ steht für die Anzahl an Reparaturteams in der Warteschlange ($M/M/\kappa$ =memoryless/memoryless/k). Für ein $M/M/\kappa$ Warteschlangensystem gilt, dass mit steigender Anzahl an Reparaturteams die Anzahl an benötigten Ersatzmaschinen im System monoton abnimmt. Dieser Umstand ist einfach nachvollziehbar, wenn man bedenkt, dass bei gleichbleibender Ausfallwahrscheinlichkeit die Wartezeit sinken muss, wenn mehr Reparaturteams eingesetzt werden.

Dieses System ist mit der Kassa eines Supermarktes vergleichbar. Sind aktuell eine gewisse Anzahl an Kassen geöffnet und es wird eine zusätzliche aufgemacht, kann ab sofort in jeder Zeiteinheit ein Kunde mehr bedient werden.

In unserem System liegt der Unterschied zu einem Supermarkt, dass es trotz mehrerer Teams nur eine Warteschlange gibt. Daher gibt es kein schnelles Überspringen von Plätzen im System, wie es oft im Supermarkt der Fall ist, wo derjenige in der Warteschlange der als erster bemerkt, dass eine neue Kassa geöffnet wird, sich als erstes in der neuen Warteschlange einreihen kann. Dieses System ist oftmals beispielsweise bei der Post bei der Bank oder an Ämtern zu erleben. Es gibt mehrere Schalter, aber nur eine große Warteschlange. Damit kann es nicht zu einer zufälligen Bevorzugung eines Kunden kommen.

Weiters ist es auch logisch, dass bei einem Startwert $z_j = \underline{z_j}$ eine Aufstockung der Reparaturteams den höchsten Gewinn einbringen wird, wenn $\underline{z_j}$, die untere Schranke des j^{ten} Subnetzwerks ist und daher die Wartezeit am höchsten sein muss.

Gehen wir wieder vom Supermarkt aus: Ist nur eine Kasse geöffnet und die Schlange schon recht lang, so ist der Zeitgewinn, eine zweite Kasse zu öffnen am höchsten, da nun die erwartete Wartezeit sich im Durchschnitt halbieren müsste, da ja doppelt so viele Bedienungen in gleicher Zeit vollzogen werden können. Erhöhen wir nun die Anzahl der Kassen wieder um 1 haben wir nun insgesamt drei Kassen. Nun beträgt die Wartezeit, verglichen zur ursprünglichen mit einer

Kassa, nur ein Drittel. Offensichtlich ist die Öffnung jeder weiteren Kassa ein Zeitgewinn, wobei der Gewinn monoton abnimmt und gegen die Durchschnittliche Bedienzeit (oder in unserem Modell die Reparaturzeit) μ_j konvergiert.

Weiters ist zu beachten, dass für wachsendes S_j der Kostennutzen eines weiteren Reparaturteams durch Verkürzung der Reparaturverzögerung abnimmt. Ein höherer Lagerbestand bei den Flughäfen erlaubt einen längeren Lieferrückstand, da vorerst die Maschinen vor Ort ersetzt werden können und die Maschine aus der Reparaturstätte nur das Lager wieder auffüllt. Erhöhen wir also den Lagerbestand, verringern sich die Kosten für Lieferrückstände, aber die Kostenersparnis, durch eine weitere Erhöhung im Lager könnte möglicherweise kleiner sein, als die dadurch entstehenden (Fix-)Kosten für die Lagerung.

4.4 Umsetzung Masterproblem

Im ersten Schritt soll überprüft werden, ob sich das Problem in seiner Komplexität direkt aus dem Masterproblem mit mathematischer Programmierung lösen lässt und bis zu welcher Größe eine vernünftige Anwendung noch möglich ist. Im Unterschied zu Rappold und Van Roo habe ich die Anzahl an Ersatzturbinen nicht als fest angenommen sondern die gegebene Anzahl als obere Schranke festgelegt. Dadurch soll auch ein unnötig verfügbarer Lagerbestand, der nur hohe Lagerkosten mit sich bringt verhindert werden.

Da das gesamte Problem NP-Schwierig ist und es aufgrund der Haupt- und Nebenbedingungen insgesamt als MINLP (Mixed Integer Non Linear Programming) durchlaufen muss, ist zu erwarten, dass es bei umfangreicheren größeren Netzwerken möglicherweise keine Lösungen mehr bringen wird.

4.4.1 Numerische Ergebnisse

Für alle numerischen Tests habe ich einen Intel(R) Core(TM) Duo CPU E7500 2,93 GHZ mit 4,00 GB RAM auf einem Windows 7 Service Pack 1 64-Bit-Betriebssystem benutzt. Nach einigen Durchläufen anhand verschiedener Testinstanzen (siehe Abbildungen 5.2, 5.3 und 5.4 ab 30)scheiterte die Umsetzung bei Systemen mit mehr als 5 Standorten (Abbildung 4.1).

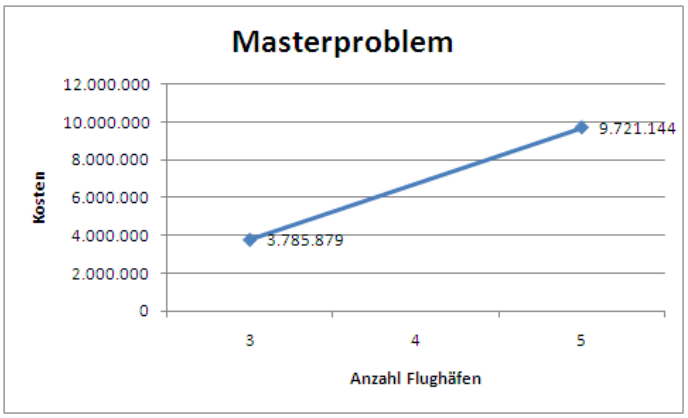

Abbildung 4.1: Umsetzung Masterproblem

Kapitel 5

Lösungsansätze

Die unbefriedigenden Ergebnisse aus der mathematischen Modellierung des Modells führen zu der Motivation alternative Lösungswege zu implementieren und zu überprüfen, ob es eine Umsetzung gibt, die für die Problemstellung adäquat und anwendbar ist. Vorrangig werde ich die Grundidee von Rappold und Van Roo heranziehen und das Modell in zwei Stufen lösen. Dadurch erhoffe ich, dass sich das Modell für alle Testdaten lösen lässt und sinnvolle Ergebnisse liefert. Es sollen verschiedene zweistufige Ansätze modelliert und verglichen zu werden, um den genauesten für die Problemstellung zu finden. Für einen Benchmark modelliere ich ebenfalls einen einstufigen Simulated Annealing Algorithmus, um den Genauigkeitsverlust der Teilung in zwei Stufen zu vergleichen.

5.1 Zweistufige Lösungsansätze

Bei der zweistufigen Modellierung ergibt sich die Möglichkeit die einzelnen Stufen mit verschiedenen Lösungsansätzen umzusetzen. Vor allem sollen die heuristischen Verfahren gegen die zweistufige mathematische getestet werden. In der ersten Stufe sollen die strategischen Entscheidungen des Modells getroffen werden. Zum einen soll die Allokation der Werkstätten stattfinden und zum anderen soll in der ersten Stufe schon die Zuweisung der Fliegerhorste fixiert werden. In der zweiten Stufe sollen die übrigen taktischen Entscheidungen gelöst werden: Die Verteilung der Ersatzturbinen bei den Lagern der Fliegerhorste und den Werkstätten sowie die Anzahl an Reparaturteams bei den Werkstätten.

Da sich das Modell in der ersten Stufe sehr einfach als lineares Problem darstellen lässt, werde ich die erste Stufe in allen Modellen mit mathematischer Modellierung lösen. Dieser Ansatz wird sich später als sinnvoll erweisen. Somit unterscheiden sich die Ansätze nur durch die Lösung der zweiten Stufe und die Betitelung der Ansätze wird sich auf den Ansatz der zweiten Stufe beziehen. Im Folgenden werde ich diese kurz beschreiben. Die Ausformulierung und eine detaillierte Beschreibung folgt im Kapitel der Analyse.

5.1.1 Zweistufiger Lösungsansatz mit mathematischer Modellierung

Nachdem die mathematische Modellierung in einem Schritt scheiterte, werde ich zunächst das Modell in zwei Stufen mathematisch formulieren. Dabei wird die Nicht-Konvexität des Modells eine Hürde bei der Lösung darstellen und lässt vermuten, dass besonders große Netzwerke die Lösung erschweren.

5.1.2 2-stufiger Lösungsansatz heuristisch

Um die Lösbarkeit auch für komplexere Systeme zu erleichtern, haben Van Roo und Rappolt das Problem in zwei Schritten gelöst. Zuerst fixieren Sie anhand mathematischer Modellierung die Verteilung der Werkstätten sowie die Zuordnung der Fliegerhorste zu den Werkstätten. Im zweiten Schritt wird für jedes reduzierte System (Subnetzwerk: eine Werkstatt + zugeordnete Fliegerhorste) die Anzahl an Reparaturteams sowie die Verteilung der Ersatzmaschinen heuristisch gelöst. Ich werde zwei sehr ähnliche heuristische Varianten mit einem kleinen Unterschied in einem Teilabschnitt der zweiten Stufe testen, um zu überprüfen, ob für diese ebenfalls ein mathematischer oder ein heuristischer Ansatz genauere Ergebnisse liefert.

5.1.3 Datenvergleich

Um die Genauigkeit der Ergebnisse aus den numerischen Tests vergleichen zu können, habe ich gehofft, die Ergebnisse von Rappold und Van Roo zu erhalten. Nach Anfrage per E-Mail habe ich leider die Antwort erhalten, dass die Daten leider nicht mehr vorhanden sind und es nur mehr die Idee der Umsetzung gibt. Daher ist es die erste Aufgabe, diese umzusetzen und zu überprüfen. Damit ich für die Ergebnisse trotzdem einen Benchmarkvergleich erhalte, kommt noch ein weiterer Lösungsansatz, das Simulated Annealing, hinzu. Der zentrale Punkt des Vergleichs soll die Zweistufigkeit in Frage stellen und daher soll im Unterschied zu den vorangegangen Alternativen des Simulated Annealing Algorithmus in einer Stufe gelöst werden. Aus den Vergleichen soll hervorgehen, ob die Teilung in zwei Stufen die optimale Lösung findet. Möglicherweise führt die Teilung in den einzelnen Stufen durch den jeweiligen Informationsverlust zwar die optimale Lösung für die jeweilige Stufe, jedoch könnte es sein, dass diese nicht die optimalen Parameter für die folgende liefert.

5.1.4 Simulated Annealing

Bei der simulierten Abkühlung werden ein großer Teil der Variablen vorgegeben und nur mehr ein triviales Restproblem gelöst. Dieses Ergebnis wird mit zufällig gewählten Nachbarlösungen verglichen. Günstigere Ergebnisse werden fix als neue Lösung angenommen, teurere unter gewissen Voraussetzungen. Eine genaue Beschreibung folgt im Bereich der Umsetzung.

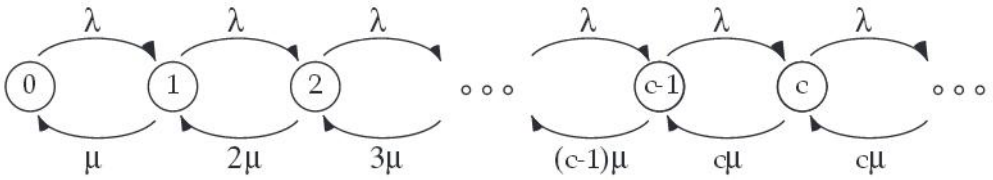

Abbildung 5.1: Wahrscheinlichkeiten der Zustandswechsel

Vorerst möchte ich ein paar Spezifika zu der technischen Umsetzung und dem verwendeten Programm vorstellen.

5.2 Technische Umsetzung

5.2.1 Programmierungsplattform und Computerdaten

Für die Umsetzung habe ich das Modellierungsprogramm GAMS (General Algebraic Modeling System) gewählt. Es eignet sich hervorragend für die Modellierung von Optimierungsverfahren, da es eine sehr benutzerfreundliche und intuitive Programmierplattform ist. Es bietet verschiede vorgefertigte Lösungsverfahren für die unterschiedlichen Modelltypen.

5.2.2 Berechnung der Verzögerungszeit während der Reparatur

Für die Berechnung der Verzögerungszeit benötigen wir eine $M/M/\kappa$ Warteschlange, damit wir die erwartete Reparaturzeit erhalten. Diese $M/M/\kappa$ Warteschlange wiederum wird über einen Geburt und Sterbeprozess dargestellt. Dabei wechselt ein Prozess mit einer gewissen Wahrscheinlichkeit in einen höheren (es kommt ein neue defekte Maschine) bzw. niedrigeren Zustand (eine Maschine ist fertig und verlässt den Reparaturprozess). In unserem Modell entsprechen diese Wahrscheinlichkeiten der Ausfälle λ_{j0} sowie der Reparaturrate μ_j bei der Reparaturstätte j. In Folge werde ich für die Darstellung der Berechnung der Verzögerungszeit die Indizes weglassen und diese neutral mit λ und μ definieren. Bei einer $M/M/\kappa$ Warteschlange steht das κ für die Anzahl an Schaltern oder in unserem Modell für die Anzahl an Reparaturteams. Somit erhöht sich die Anzahl an Maschinen im System in jeder Zeiteinheit mit Wahrscheinlichkeit λ und verringert sich je nach Zustand mit μ multipliziert mit der Anzahl an schon besetzten Reparaturteams, da ja bei jedem eine Maschine repariert wird. Sobald mehr als κ Maschinen repariert werden müssen und somit alle Teams beschäftigt sind, ist die Zustandsänderung nach unten mit Wahrscheinlichkeit $\mu\kappa$.Grafisch kann die Zustandsänderung wie in Abbildung 5.1 dargestellt werden.

Damit eine Warteschlange ein statistisches Gleichgewicht hat, muss folgende Gleichung gelten:

$$\frac{\lambda}{\mu\kappa} < 1$$

Das heißt, dass die Fehlerrate λ kleiner sein muss als die Reparaturrate aller Teams gemeinsam. Wäre $\frac{\lambda}{\mu\kappa} > 1$ würde zwangsläufig die Warteschlange gegen ∞ konvergieren, da mehr Maschinen kaputt als repariert würden.

Sei $b = \frac{\lambda}{\mu}$ und $\sigma = \frac{b}{\kappa} = \frac{\lambda}{\kappa\mu}$ dann gilt für die Wahrscheinlichkeit in den Zuständen p_n, also dass genau n Maschinen im Reparaturprozess sind:

$$\begin{aligned}
p_1 &= bp_0 \\
p_2 &= \frac{b^2}{2 \cdot 1} p_0 \\
p_2 &= \frac{b^3}{3!} p_0 \\
&\cdots \\
p_\kappa &= \frac{b^\kappa}{\kappa!} p_0 \\
p_{\kappa+1} &= \sigma \frac{b^\kappa}{\kappa!} p_0 \\
&\cdots \\
p_n &= \sigma^{n-\kappa} \frac{b^\kappa}{\kappa!} p_0
\end{aligned}$$

Um eine Darstellung von p_0 zu erhalten, nutzen wir die Eigenschaft dass die Summe über alle Wahrscheinlichkeiten 1 ist. Daher erhalten wir:

$$1 = \sum_{k=0}^{\infty} p_k = p_0 \left(\sum_{k=0}^{\kappa-1} \frac{b^k}{k!} + \frac{b^\kappa}{\kappa!} \sum_{k=\kappa}^{\infty} \sigma^{k-\kappa} \right) = p_0 \left(\sum_{k=0}^{\kappa-1} \frac{b^k}{k!} + \frac{b^\kappa}{\kappa!} \frac{1}{1-\rho} \right)$$

daraus folgt

$$p_0 = \left(\sum_{k=0}^{\kappa-1} \frac{b^k}{k!} + \frac{b^\kappa}{\kappa!} \frac{1}{1-\rho} \right)^{-1}$$

Eine Maschine muss sich genau dann in der Warteschlange anstellen, wenn mindestens κ Maschinen schon im Prozess sind. Sei $U(t)$ die Anzahl an Maschinen, dann gilt [17]:

$$\begin{aligned}
\lim_{t\to\infty} P(U(t) &\geq \kappa) = G(b,\kappa) = \sum_{k=\kappa}^{\infty} p_k = 1 - \sum_{k=0}^{\kappa-1} p_k = \\
p_0 \left(\frac{1}{p_0} - \sum_{k=0}^{\kappa-1} \frac{b^k}{k!} \right) &= p_0 \frac{b^\kappa}{\kappa!(1-\sigma)}
\end{aligned}$$

Die Anzahl an Maschinen im Warteprozess L_q im statistischen Gleichgewicht ist gegeben durch:

$$\begin{aligned} L_q &= \sum_{k=\kappa}^{\infty}(k-\kappa)p_k = \sum_{k=\kappa}^{\infty}(k-\kappa)\frac{b^\kappa}{\kappa!\kappa^{k-\kappa}}p_0 = \\ p_0\frac{b^\kappa}{\kappa!}\sum_{k=1}^{\infty}k\sigma^k &= p_0\frac{b^\kappa}{\kappa!}\frac{\sigma}{(1-\sigma)^2} = \\ & \frac{\sigma}{(1-\sigma)}G(b,\kappa) \end{aligned}$$

Die durchschnittliche Wartezeit W_q :

$$W_q = \frac{L_q}{\lambda} = \frac{1}{\lambda}\frac{\sigma}{(1-\sigma)}G(b,\kappa) = \frac{1}{\sigma\mu(1-\sigma)}G(b,\kappa)$$

Und die gesamte Zeit im Reparaturprozess ist dann

$$W = W_q + \frac{1}{\mu}$$

Wenn demnach keine Ersatzmaschinen bei einer Werkstätte vorhanden sind, kann es eine Ersatzverzögerung von bis zu W zuzüglich der Transportzeit geben. Die Formel für die Zeit im Reparaturprozess dient als Basis für die Berechnungen von den Lagerbeständen bzw Lieferrückständen($\Psi, D_{j0}, \tau_j, D_{ji}, \Gamma_i, \Phi_i$).

5.2.3 Daten für die Benchmark-Probleme

Um die Performance der unterschiedlichen Lösungsansätze zu testen, benötige ich verschiedene Testinstanzen. Dafür habe ich vier fiktive Netzwerke von Fliegerhorsten erstellt. Die meisten Parameter sollen, sofern dies möglich und sinnvoll ist, für alle Netzwerke gleich sein. Unter anderem soll die Reparaturrate, die im Durchschnitt für jedes Team für jede Werkstatt konstant und ident angenommen werden und für alle Netzwerke gleich sein. Hingegen muss die Anzahl an Ersatzmaschinen in einer gewissen Weise auch abhängig von der Anzahl an Fliegerhorsten sein.

Als Testbeispiele sollen Netzwerke von der Größe zwischen 3 und 15 Flughäfen getestet und verglichen werden. Folgende zentrale Fragen sollen aus den Ergebnissen beantwortet werden:

- Wie ist die Qualität der 2-stufigen Lösungsansätze im Vergleich zum Simulated-Annealing Algorithmus?
- Wie verhält sich die Laufzeit der Algorithmen bei wachsenden Netzwerken?
- Welche Lösungswege führen effizient zu einem sinnvollen Ergebnis?

Testwerte, für alle Netzwerke gleich	
Max. Anzahl an Reperaturteams/Werkstatt |K|	5
Reperaturrate/Team/Tag μ	0,49
Transportzeit l	4 Tage
Transportkosten h	3.000
Fixkosten, eine Reperaturstätte zu öffnen f	1.000.000
Lagerhaltungskosten/Stück/Jahr h	700.000
Ausfallkosten einer Maschine pro Tag π	1.500
Obergrenze der Auslastung ρ	79%
Kosten eines Reperaturteams a	900.000

Abbildung 5.2: Gemeinsame Parameter der Referenzmodelle

durchschnittliche Fehlerrate λ bei Fliegerhorst Fx	F1	F2	F3	F4	F5	F6	F7	F8
	0.170	0.446	0.326	0.223	0.220	0.192	0.243	0.451
	F9	F10	F11	F12	F13	F14	F15	
	0.128	0.305	0.509	0.337	0.506	0.413	0.154	

Abbildung 5.3: Fehlerrate der Fliegerhorste

Die gemeinsamen Parameter der Referenzmodelle werden in der Abbildung 5.2 dargestellt.

In der Abbildung 5.3 werden die Werte für die Fehlerrate und in Abbildung 5.4 die Testinstanzen sowie die entsprechende Anzahl an Ersatzmaschinen dargestellt.

5.3 Lösungsansatz nach Van Roo und Rappold

Im Abschnitt 4.4 auf Seite 23 kam ich zum Ergebnis, dass eine einfache Lösung des Masterproblems in einem Schritt bei größeren Systemen keine sinnvollen Ergebnisse mehr lieferte. Daher sollen alternative Lösungswege überprüft und verglichen werden. Es ist ebenfalls zu erwarten, dass die Aufteilung der Optimierung in mehrere Stufen und in kleinere Subprobleme, das Lösen, insbesondere von sehr großen Referenzmodellen, erleichtern wird. In Summe wird die Berechnung der einzelnen Stufen durch die geringere Komplexität effizienter und voraussichtlich möglich sein. Daher kann ebenfalls eine höhere Genauigkeit der Ergebnisse erwartet werden. Leider brachte das Masterproblem keine verwendbaren Vergleichswerte um diese Erwartungen zu verifizieren. Daher habe ich auch einen Simulated Annealing Algorithmus als Referenz modelliert, wenn auch dieser eine nicht optimale, aber hinreichend gute Lösung liefern wird.

Testinstanzen und entprechnende Anzahl an Ersatzmaschinen				
Anzahl an Flughäfen	3	5	10	15
Anzahl Ersatzmaschinen	10	15	30	50

Abbildung 5.4: Testinstanzen

5.3.1 Die taktischen und strategischen Entscheidungen

Die mehrstufige Modellierung bedarf vorab einer Aufteilung der Entscheidungen in die einzelnen Stufen. Hier bietet es sich an, diese in die taktischen und strategischen zu teilen und dafür jeweils eine Stufe zu bilden:

Strategische Entscheidungen Die strategischen Entscheidungen sind in erster Linie die Positionierung der einzelnen Werkstätten bei den Fliegerhorsten. Diese ist für das System die wichtigste langfristige Entscheidung, da es mit sehr hohen Kosten verbunden wäre, einen Standort zu ändern. Für die Standortwahl ist ebenfalls die Zuteilung der Fliegerhorste zu den Werkstätten maßgeblich und daher auch eine strategische Entscheidung.

Taktische Entscheidungen Die taktischen Entscheidungen des Modells sind die Aufteilung der Ersatzturbinen auf die Lager der Fliegerhorste und der Werkstätten, sowie die Bestimmung, wie viele Reparaturteams bei den einzelnen Werkstätten eingesetzt werden sollen.

5.3.2 Aufteilung in zwei Stufen

Die Spaltung der Entscheidungsvariablen in die taktischen und strategischen motiviert Rappold und Van Roo das Modell in zwei Stufen zu teilen, wo in der ersten Stufe eine Lösung für die strategischen Entscheidungen getroffen wird. Anhand dieses Ergebnisses werden in der zweiten Stufe die taktischen Entscheidungen getroffen:

Erste Stufe : Zuerst wird das Allokationsproblem gelöst. Dadurch wird die kostenminimale Zusammensetzung von geöffneten Werkstätten, der Zuordnung der Fliegerhorste und die minimale Anzahl an Reparaturteams berechnet.

Zweite Stufe : Im zweiten Schritt werden anhand der Ergebnisse aus Schritt 1 die optimale Anzahl an Reparaturteams gemeinsam mit der Verteilung der Ersatzmaschinen im Netzwerk über die Kostenfunktion minimiert.

5.4 Erste Stufe: Allokation der Werkstätten und Zuordnung der Fliegerhorste

In der ersten Stufe werden nur die strategischen bzw. langfristigen Kostenelemente betrachtet. Es ist leicht verständlich, dass einerseits die Entscheidung, bei welchen Fliegerhorste eine Werkstatt gebaut werden soll, langfristig geplant sein sollte, da es sehr aufwendig und teuer wäre eine Werkstätte zu schließen und ggf. eine andere ersetzend zu öffnen. Die größten Preiskomponenten der Öffnung einer Reparaturstätte hängen von den Flughäfen, die ihr zugewiesen werden und die sich daraus ergebenden Transportkosten ab. Also sind Allokation, Zuweisung und Transportkosten die elementaren Faktoren und daher das Ziel des ersten Schrittes. Im Gegenzug dazu werden in der erste Stufe die Ersatzpolitik, die Verteilung der Ersatzturbinen sowie die Bestimmung der tatsächlichen Anzahl an Teams bei den Werkstätten nicht betrachtet. Dafür wird eine untere Schranke für die Anzahl an Teams $\underline{z_j}$ mitbestimmt, damit das statistische Gleichgewicht für $M/M/\kappa$ Warteschlangen erfüllt sind (Beweis siehe unten). In der ersten Stufe erhalten wir für das reduzierte strategische Teilmodell folgendes lineares Optimierungsproblem:

$$\min_{x_j y_{ji} z_{jk}} \sum_{j \in F} x_j f_j + \sum_{j \in F} \sum_{i \in F} \lambda_{ji} y_{ji} c_{ji} + \sum_{j \in F} \sum_{k \in K} z_{jk} a_{jk}$$

s.t.

$$\sum_{j \in F} y_{ji} = 1 \quad \forall i \in F$$

$$y_{ij} \leq x_j \quad \forall i \in F \forall j \in R$$

$$\sum_{i \in F} \lambda_{ji} y_{ji} \leq \rho \mu_j \sum_{k \in K} z_{jk} \quad \forall j \in R$$

$$\sum_{i \in F} \sum_{j \in R} (l_{ji} \lambda_{ji} y_{ji}) \leq V$$

$$0 \leq z_{j(k+1)} \leq z_{jk} \quad \forall j \in R, \quad k = 0, 1, ..., K-1,$$

$$x_j, y_{ji}, z_{jk} \; bin\ddot{a}r$$

Lemma 1 *Das Minimierungsproblem liefert immer die untere Schranke für* $\underline{z_j} = \sum_{k \in K} z^*_{jk}$. *(dabei sind* z^*_{jk} *die Ergebnisse* z_{jk} *aus der Minimierung)*

Beweis. Die Beweisführung ist folgendermaßen trivial:

1. z_{jk} hat keinen Einfluss auf $\sum_{j \in F} x_j f_j$
2. z_{jk} hat keinen Einfluss auf $\sum_{j \in F} \sum_{i \in F} \lambda_{ji} y_{ji} c_{ji}$

Daher beschränkt sich die Anzahl an Reparaturteams lediglich auf die Erfüllung der Konvergenz des Warteschlangensystems im statistischen Gleichgewicht. Daher darf die Auslastung nicht größer 1 sein. Durch ρ wird dies im Modell formal dargestellt und zusätzlich durch die Gleichung $\sum_{i \in F} \lambda_{ji} y_{ji} \leq \rho \mu_j \sum_{k \in K} z_{jk}$ gewichtet. Da z_{jk} auf die beiden obigen Teile der Zielfunktion keinen weiteren Einfluss hat, wird die kleinstmögliche Summe $\underline{z_j} = \sum_{k \in K} z^*_{jk}$ gewählt, sodass diese Auslastungsrestriktionen erfüllt sind. Somit existiert kein kleineres z_j, das als Lösung in Frage kommt. Bezeichne Z^j die Menge aller möglichen $z_{j\cdot}$ Dann gilt für die Lösungsmenge von $z_j \in Z^j$:

$$\underline{z_j} = \sum_{k \in K} z^*_{jk} \leq z_j \quad \forall \; z_j \in Z^j$$

q.e.d. ■

Als Ergebnis aus der ersten Stufe erhalten wir die Menge der geöffneten Wekstätten, den ihnen zugewiesenen Fliegerhorste und eine untere Schranke für die Anzahl an Reparaturteams. Wenn wir $x^*_j \in \mathbf{M}$ als die Menge der geöffneten Werkstätten definieren, erhalten wir daraus unabhängige Subprobleme Für jedes $x_j = 1$ wird ein Subsystem mit einer Werkstatt und zugewiesenen Fliegerhorsten mit folgenden dazugehörigen Parametern bestimmt:

- x^*_j als die Menge der geöffneten Werkstätten
- $\underline{z_j} = \sum_{k \in K} z^*_{jk}$ als die untere Schranke an Teams bei Fliegerhorst j
- y_{ji} als die Zuweisung der Flughäfen zu den Werkstätten

Somit werden x_j und y_{ji} für die zweite Stufe zu Parametern. Anhand dieser Ergebnisse können wir nun in der zweiten Stufe die kostenminimierende Verteilung der Ersatzmaschinen auf die Fliegerhorste und die Werkstätten bei gleichzeitiger Bestimmung der optimalen Anzahl an Reparaturteams in den jeweiligen Untersystemen modellieren.

5.5 Zweite Stufe: Ablaufoptimierung

In der ersten Stufe haben wir die eigentliche Struktur des Netzwerks bestimmt und die strategischen Entscheidungen der kostenminimalen Zuweisungen eruiert. Für diese Konfiguration sollen nun die taktischen Entscheidungen optimiert werden. Diese geschehen für jedes Subsystem und läuft darauf hinaus, dass die optimale Anzahl an Reparaturteams und die Lagerbestände bestimmt werden. Hier

treten zwei Probleme auf. Die Verteilung der Ersatzturbinen auf die Fliegerhorste und die Werkstätten im Subnetzwerk ist von der Kostenfunktion abhängig. Diese ist wiederum von der Anzahl an für das Subnetzwerk verfügbaren Ersatzmaschinen $S_j \leq S$ abhängig. Die Aufteilung der insgesamt vorhandenen Ersatzmaschinen S muss aber ebenfalls kostenminimierend modelliert werden. Daher gibt es hier ein Minimierungsproblem, dass sich über alle Subnetzwerke zieht und daher eine Abhängigkeit schafft.

Die ebenfalls noch zu treffende Entscheidung, wie viele Teams jeweils positioniert werden sollen, ist vom globalen Netzwerk nur von der Anzahl an zugewiesenen Ersatzmaschinen S_j im Subnetzwerk abhängig, aber sonst vom globalen Problem unabhängig und muss daher nicht gemeinsam gelöst werden.

5.5.1 Kostenfunktionen $SP_j(S_j)$ der Subnetzwerke

Der zentrale Punkt in der zweiten Stufe ist demnach eine Funktion $SP_j(S_j)$ für jedes Subnetzwerk, die für die jeweilige Anzahl an Ersatzmaschinen die kostenminimale Verteilung der Ersatzmaschinen, sowie die optimale Anzahl an Reparaturteams liefert. Folglich sollen daraus die global optimalen Kosten anhand der Funktionen $SP_j(S_j)$ gefunden werden unter der Restriktion, dass maximal S Maschinen verfügbar sind:

$$\min_{S_j} \sum_{j \in F^*} SP_j(S_j)$$
$$\text{s.t.} \sum_{j \in F^*} S_j \leq S.$$

5.5.2 Modellierung von $SP_j(S_j)$

Die Grundidee bei der Modellierung ist, in einem ersten Schritt die minimalen Kosten der einzelnen Subnetzwerke für vorgegebene Werte von S_j und z_j zu berechnen. Diese Kostenfunktion bezeichnen wir mit $C_j(z_j, S_j)$ .Wenn die daraus resultierenden Ergebnisse über z_j minimiert werden, erhalten wir genau die gewünschte Funktion $SP_j(S_j)$:

$$SP_j(S_j) = \min_{z_j \geq \underline{z_j}} C_j(z_j, S_j).$$

Da aber die Berechnung der Kosten von allen Kombinationen von S_j und z_j besonders für große Systeme sehr aufwendig ist, beschreiben Rappold und Van Roo einen Algorithmus der einige strukturelle Eigenschaften dieser Kostenfunktionen ausnutzt und daher nur die Berechnung von ausgewählten Kostenfunktionen benötigt. Ein weiterer kritischer Punkt ist die Nicht-Konvexität der Funktion von $SP_j(S_j)$. Wären diese Funktionen konvex, wäre das lösen verhältnismäßig trivial und ich könnte anhand von diskretem abtasten die optimale Verteilung leicht finden. Daher müssen die Funktionen $SP_j(S_j)$ konvexisiert werden, damit eine zumindest fast optimale Lösung gefunden werden kann. Davor werde ich aber die Berechnung von $SP_j(S_j)$ selbst erklären.

5.5.3 Bestimmung von $SP_j(S_j)$

Die durch S_j und z_j parametrisierte Optimierungsaufgabe $C_j(z_j, S_j)$ wird mittels folgendem Minimierungsproblem gelöst:

$$C_j(z_j, S_j) = \min_{s_{ji}, s_{j0}} \sum_{k=0}^{z_j} a_{jk} + \sum_{i \in F^j} h_{j0}\Psi(z_j, s_{j0}) +$$

$$\sum_{i \in F^j} h_{ji}\Gamma_i(\lambda_{ji}, z_j, s_{j0}, s_{ji}) + \sum_{i \in F^j} \pi_{ji}\Phi_i(y_{ij}, z_j, s_{j0}, s_{ji})$$

$$\text{s.t.} \qquad S_j \geq s_{j0} + \sum_{i \in F^j} s_{ji},$$

$$s_{j0}, s_{ji} \geq 0, \qquad \forall i \in F^j.$$

Durch Gleichung $S_j = s_{j0} + \sum_{i \in F^j} s_{ji}$ wird sichergestellt, dass das Subnetzwerk auch die gesamten zugewiesenen Maschinen verteilt und Ungleichung $s_{j0}, s_{ji} \geq 0$ sichert, dass es zu keinem negativen Lagerbestand kommen kann. In meiner Umsetzung habe ich im Gegensatz zu Rappold und Van Roo S_j als obere Schranke angenommen. Somit wird aus der Beschränkung von Rappold und Van Roo $S_j = s_{j0} + \sum_{i \in F^j} s_{ji}$ die oben dargestellte Abwandlung $S_j \geq s_{j0} + \sum_{i \in F^j} s_{ji}$. Diese Ungleichung soll verhindern, dass der Lagerbestand unnötig groß und dadurch teurer wird, als eigentlich erforderlich. Wenn dieser Fall eintreten sollte, wird es erwartungsgemäß Möglichkeiten geben, Ersatzmaschinen zu verkaufen. Weiters zu beachten ist, dass die Kostenfunktion $C_j(z_j, S_j)$ nur für $z_j \geq \underline{z_j}$ definiert ist. Also verwenden wir in der zweiten Stufe die aus der ersten Stufe bestimmten unteren Schranke $z_j = \underline{z_j}$. Bei der Optimierung mit der unteren Schranke wird in der zweiten Stufe die Inanspruchnahme der maximal verfügbaren Ersatzturbinen am höchsten sein. Dies liegt daran, dass bei wenigen Teams, die Reparaturzeit länger dauert und dadurch die Ersatzlieferung mehr Zeit in Anspruch nimmt. Daher müssen mehr Ersatzmaschinen vor Ort verfügbar sein, da es sonst zu Lieferrückständen kommt.

In der Abbildung 5.5 ist einerseits der Zusammenhang zwischen $C_j(\cdot)$ und $SP_j(\cdot)$ für wachsende Anzahl an Reparaturteams (z_j, z_{j+1}, z_{j+2}) und die nur beinahe aber nicht tatsächliche Konvexität gut ersichtlich.

Offensichtlich hat jede einzelne Kostenkurve $C_j(\cdot)$ für jedes S_j auch eine minimale Anzahl an Reparaturteams $z_j^*(S_j)$. Sei

$$z_j^*(S_j) = arg \min_{z_j \geq \underline{z_j}} C_j(z_j, S_j).$$

Für jedes einzelne Subnetzwerk haben wir nun ein System entwickelt, wodurch die Kosten geschätzt werden können. Das Ziel ist es, die Kostenfunktionen, die wir bestimmt haben, so zusammenzuführen, dass die Gesamtkosten des Netzwerks von Subnetzwerken global minimiert werden. Dafür muss Anhand der Kostenfunktionen $SP_j(\cdot)$ der einzelnen Subnetzwerke die gesamte Anzahl

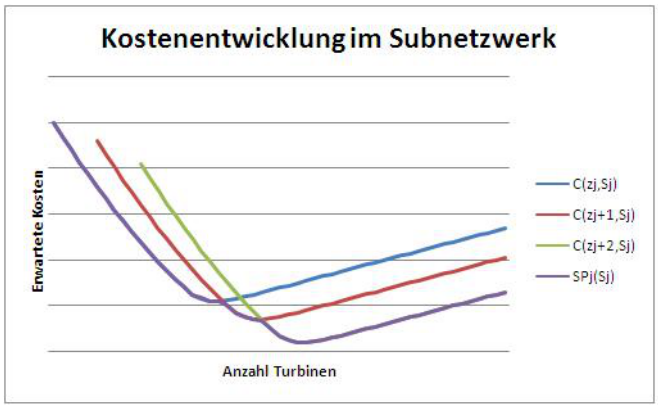

Abbildung 5.5: Kostenkurven der Subnetzwerke in Abhängigkeit von der Anzahl an Ersatzmaschinen [1]

an Maschinen des gesamte Systems so bestimmt werden, dass die Gesamtkosten minimiert werden:

$$\min_{S_j} \sum_{j \in F^*} SP_j(S_j)$$

$$\text{s.t.} \sum_{j \in F^*} S_j \leq S.$$

Wie schon vorher erwähnt, ist zwar jede Kostenfunktion $C_j(z_j, S_j)$ der Subnetzwerke für sich konvex, aber die aggregierte Zielfunktion nur beinahe. Das hindert einen einfachen Lösungsalgorithmus. Aber aus der Abbildung 5.5 lässt sich die Vermutung erstellen, dass eine konvexe Approximation von $SP_j(\cdot) = \overline{SP_j}(\cdot)$ in den meisten Werten der tatsächlichen Funktion $SP_j(\cdot)$ entspricht, oder zumindest ihr sehr nahe sein wird. Wenn wir gleichzeitig bedenken, dass es sich bei den Werten S_j um ganzzahlige Werte handelt, ist die Erwartung, die tatsächlichen Werte zu treffen, noch wahrscheinlicher. Hier sei wieder erwähnt, dass Rappold und Van Roo die Nebenbedingung $\sum_{j \in F^*} S_j = S$ hatten, wodurch sie in jedem Fall das gesamte verfügbare Ersatzkontingent verteilt haben.

5.5.4 Konvexisierung von $SP_j(S_j)$

Im Folgenden soll der Algorithmus nach Van Roo und Rappold für die Entwicklung einer Konvexisierung ausführlich erklärt und im Anschluss kritisch betrachtet und analysiert werden. Die Konvexisierung erfolgt in vier Schritten:

1. Berechnung einer oberen Schranke für die Anzahl an Reparaturteams

2. Berechnung der erwarteten Verteilung der Anzahl an Turbinen im Reparaturprozess in Abhängigkeit von der Anzahl an Teams

3. Optimale Verteilung der Ersatzmaschinen bei vorgegebenen Teams mit entsprechend minimalen Kosten

4. Konstruktion einer konvexen Approximation

Erster Schritt: Berechnung einer oberen Schranke für die Anzahl an Reparaturteams In diesem Schritt setzten wir die Anzahl der Ersatzmaschinen $S_j = 0$. Das bedeutet, dass wir die obere Schranke für die Anzahl an Teams bei den jeweiligen Werkstätten j unter der Annahme, dass es keine einzige Ersatzturbine im System gibt, berechnen. Wenn es keine Ersatzturbine gibt ist die Erhöhung der Teams offensichtlich optimaler, da jeder Ausfall einer Maschine sofort zu Ersatzverzögerung führt und die fehlende Maschine bis zum Ende der Reparatur nicht ersetzt werden kann, da es ja keinen Ersatz gibt.

Lemma 2 *$C_j(z_j, S_j)$ ist konvex für festes S_j*

Beweis. trivial ■

Lemma 3 *Gibt es keine Ersatzmaschinen, also $S_j = 0$, erhalten wir eine obere Schranke für die Teamanzahl $\overline{z}_j \ \forall \ S_j$*

Beweis. $C_j(z_j, S_j)$ ist konvex für festes S_j. Daher gilt:

$$(C_j(z_{j-1}, 0) - C_j(z_j, 0)) \geq (C_j(z_{j-1}, S_j) - C_j(z_j, S_j)),$$

für alle z_j, solange

$$(C_j(z_{j-1}, 0) - C_j(z_j, 0)) \geq 0.$$

■

Das heißt, dass bei $S_j = 0$ die Anzahl an Teams so lange erhöht wird, bis das Minimum erreicht ist und eine weitere Aufstockung kostensteigend ist. Daher ist es sinnvoll, diese Methode für die Berechnung der oberen Schranke zu nutzen, da eine obere Schranke für diese z_j bezüglich $S_j = 0$ sicher auch eine obere Schranke bei wachsenden Lagerbestand ist. Sobald eine Erhöhung der Teams, im Fall, dass ich keine Ersatzmaschinen habe, mehr Kosten verursacht, als einspart, ist die Überlegung trivial, dass es in demselben System mit Ersatzmaschinen sicher schon bei früheren z_j zu dem Punkt kommt, wo $C_j(z_{j-1}, S_j) < C_j(z_j, S_j)$ gilt.

Ein weiterer wichtiger Punkt ist, dass $SP_j(0) = argmin_{z_j} C_j(z_j, 0) = C_j(\overline{z}_j, 0)$ der initiale Punkt der Approximation ist und hier der tatsächliche Wert der Kostenfunktion der konvexen Approximation entspricht. Im Weiteren werden wir die bisher nur nach unten beschränkte Menge der möglichen z_j noch weiter durch eine obere Schranke durch

$$Z^j = [z_j : \underline{z}_j \leq z_j \leq \overline{z}_j]$$

beschränken.

Zweiter Schritt: Berechnung der erwarteten Verteilung der Anzahl an Turbinen im Reparaturprozess in Abhängigkeit von der Anzahl an Teams Dieser Schritt ist eigentlich eine Vorbereitung für den dritten Schritt, wo die Kosten der Subnetzwerke bei sukzessiver Erhöhung berechnet werden. Dafür sind die in diesem Schritt zu berechnenden Verteilungen der Maschinen im Reparatursystem notwendig, da die erwartete Anzahl an sich im Reparatursystem befindlichen Maschinen für die erwartete Reparaturzeit verantwortlich sind, welche wiederrum $D_{j0}(\lambda_{j0}, z_j)$, die erwartete Nachfrage während eines Warteprozesses beeinflusst.

Da wir im Reparaturprozess ein $M/M/\kappa$ Warteschlangensystem haben, berechnen wir also nun für jedes Subnetzwerk und jedes $z_j \in Z^j$ die Reparaturzeit.

3. Schritt: Optimale Verteilung der Ersatzmaschinen bei vorgegebenen Teams mit entsprechend minimalen Kosten Der Umstand, dass in dem Subproblem die Werte S_j und z_j als Parameter miteinfließen erleichtern die Berechnung, werden aber bei sehr großen Netzwerken die Berechnungsdauer aller Möglichkeiten deutlich in die Höhe treiben. In diesem Schritt soll das System für jedes Subnetzwerk sukzessive die Anzahl an verfügbaren Ersatzmaschinen S_j erhöhen und jeweils die optimalen Kosten berechnen. Daraus ergibt sich eine Matrix mit allen optimalen Kostenwerten der Kostenkurven $C_j(z_j, S_j)$ für $z_j \in Z^j$ und S_j zwischen $0 \leq S_j \leq S$. Da die Berechnung für alle möglichen Kombinationen bei sehr großen Systemen sehr ineffizient wäre, nutzen Rappold und Van Roo [1] eine sehr wichtige von der Kapazität abhängige strukturelle Eigenschaft der Kostenkurven:

Zwei Kostenkurven aus $C_j(z_j, S_j)$ schneiden einander, wenn das optimale $z_j^*(S_j) \neq z_j^*(S_{j-1})$ gilt.

Und entlang der optimalen Hülle gilt:

1. Zwei Kostenkurven schneiden einander meistens, wenn z_j um nur eine Einheit erhöht wurde.

2. Die Kurven schneiden sich genau in einem Punkt.

Aufgrund dieser beiden Eigenschaften ist es für jedes S_j ausreichend jeweils die Kostenwerte zwischen z_j und $z_j - 1$ zu vergleichen.

Bezeichnen wir $SP_j(S_j)$ als die Familie der (noch nicht konvexen) optimalen Kostenkurven der Subnetzwerke. Wenn wir nun den Algorithmus bei dem vorher bestimmten $\overline{z}_j$ und bei $S_j = 1$ beginnen bestimmen wir $SP_j(S_j)$ durch:

$$SP_j(S_j) = \min(C_j(z_j, S_j), C_j(z_{j-1}, S_j)$$

gilt

$$C_j(z_{j-1}, S_j) \leq C_j(z_j, S_j)$$

setze

$S_j \backslash C(z_j, S_j)$	$C(2, S_j)$	$C(3, S_j)$	$C(4, S_j)$	$C(5, S_j)$	$C(6, S_j)$	$C(7, S_j)$
0						▼
1				■ >	▼	< ■
2				■ >	▼	
3		■ >	▼	< ■	< ■	
4		■ >	▼			
5		■ >	▼			
6	■ >	▼	< ■			
7	▼	< ■				
8	▼					

Tabelle 5.1: Bestimmung von $SP(S_j)$

$$z_j = z_j - 1$$

und wiederhole die Prozedur. Gilt hingegen

$$C_j(z_j, S_j) < C_j(z_{j-1}, S_j)$$

setze

$$SP_j(S_j) = C_j(z_j, S_j)$$

und

$$S_j = S_j + 1$$

und wiederhole die Prozedur. Diese Schritte sollen bis $S_j = S$ durchgeführt werden. Damit konstruieren wir $SP_j(S_j)$, welches nicht notwendigerweise konvex ist.

In der Tabelle 5.1 wird dies unter der Annahme, dass $\underline{z}_j = 2$ und $\overline{z}_j = 7$ ist, illustriert.

Dabei sind die Dreiecke ▼ die gewählten Werte für $SP_j(S_j)$ und die Vierecke ■ die Vergleichswerte. Die Grafik illustriert noch einmal den Ablauf und die Ersparnis, alle Werte auszurechnen. Für jede Zeile (festes S_j) werden nur die Nachbarlösungen verglichen. Sobald ein Wert höher ist, wird die Anzahl an Ersatzturbinen erhöht. Weiters sieht man auch, dass der Startwert $SP_j(0)$ genau dem Wert $\min_{z_j} C_j(z_j, 0)$ entspricht und das die Funktion $C_j(z_j, S_j)$ zeilenweise (für festes S_j und wachsendes z_j) konvex ist.

Vierter Schritt: Kontsruiere die Konvexe Approximation Sobald nun die optimalen Kostenkurven $SP_j(S_j)$ berechnet wurden, muss daraus die konvexe Approximation $\overline{SP}_j(S_j)$ generiert werden um darauffolgend die Minimierung durchführen zu können. In Abbildung 5.6 ist ein Beispiel für eine nicht konvexe Funktion von $SP_j(S_j)$ illustriert.

Der einfachste Weg wäre, bei dem Beispiel in Abbildung 5.6 eine konvexe Hülle auf trivialem Weg zu generieren. Dies würde zu dem Ergebnis in Abbildung 5.7 führen.

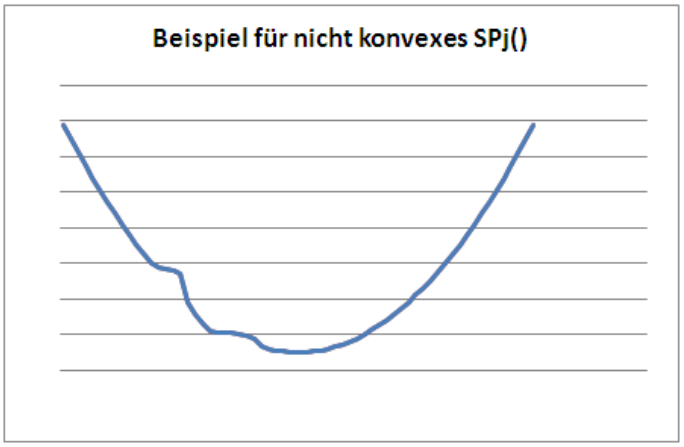

Abbildung 5.6: Nicht konvexes $SP_j(S_j)$

Dieser Weg wäre sehr einfach und ich habe ihn für den Vergleich auch umgesetzt (siehe 48), aber die Abbildung 5.7 zeigt, dass einige Werte in den nicht konvexen Bereichen deutlich zu optimal angenommen werden und sie daher möglicherweise gewählt würden, obwohl es bessere Lösungen gäbe.

Um diese Problematik ein wenig abzuschwächen, bedienen sich Rappold und Van Roo einer sehr interessanten Methode, diese Konvexisierung mit möglichst geringem Genauigkeitsverlust zu generieren. Dafür benutzen Sie ein paar Eigenschaften der Kostenkurven und bilden damit geglättete Anstiege und Differenzen, womit Beulen besser approximiert werden können und Tiefpunkte nicht zu stark überschätzt werden. Um die Herangehensweise von Rappold und Van Roo zu betrachten, werden wir lediglich den nicht-konvexen Teil betrachten (Abbildung 5.8).

Um eine optimalere Approximation zu erreichen werde ich im Folgenden einige Differenzen definieren, die mir die Richtung bzw. den approximierten Anstieg zwischen zwei Werten bestimmen sollen. Dadurch sollen verschiede Szenarien (Sattelpunkte, Schwankungen) geglättet werden, damit die konvexe Approximation einen möglichst geringen Genauigkeitsverlust hat.

5.5.5 Bestimmung der Differenzen Δ_s Δ_g Δ_l

Bevor ich die Differenzen selbst definiere, müssen ein paar notwendige Werte dafür bestimmt und gespeichert werden. Es soll

$$SP_j(S_j^*(z_j)) = \min_{S_j} SP_j(\cdot)$$

die lokal kostenminimierende Anzahl an Ersatzmaschinen der einzelnen Subnetzwerke bezüglich jedem z_j sein. Für jeden Wert S_j speichern wir die zugehörigen Werte z_j, also die Anzahl an Ersatzmaschinen, das lokale Minimum

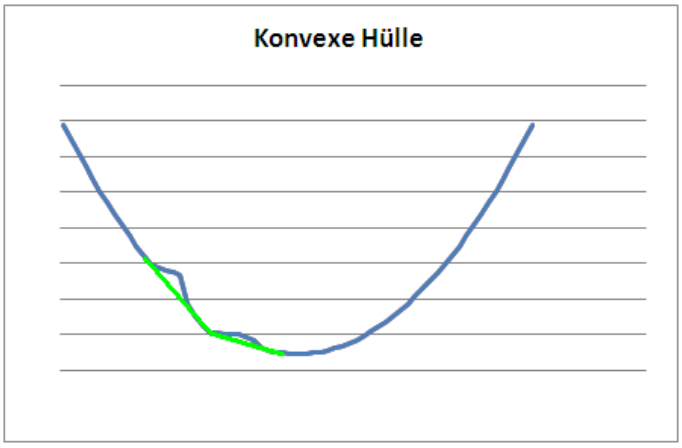

Abbildung 5.7: Konvexe Hülle

$S_j^*(z_j)$ sowie die Kosten im lokalen Minimum $SP_j(S_j^*(z_j))$. Das globale Minimum $SP_j(S_j^{**})$ sowie den zugehörigen global optimalen Wert S_j^{**} wird ebenfalls gespeichert Mithilfe dieser Werte werden folgende verschiedene Steigungswerte mithilfe von Differenzen definiert, um damit die konvexe Approximation zu konstruieren. Es sei

$$\Delta_s(S_j) = |SP_j(S_j) - \overline{SP}_j(S_j - 1)|$$

also der Unterschied der Kosten zwischen der Approximation des lokalen Minimums beim vorhergehenden Wert $S_j - 1$ und den tatsächlichen Kosten $SP_j(S_j)$ (Abbildung 5.9).

Der tatsächliche Anstieg zwischen der Approximation bei $\overline{SP}_j(S_j - 1)$ und den zu approximierenden Wert ist die wünschenswerteste Annahme, da diese die tatsächlichen Anstiege von $SP(S_j - 1)$ zu $SP(S_j)$ am ehesten entsprechen und sofern sich die Funktion in einem konvexen Teilbereich befindet auch für folgende Werte die Genauigkeit der Approximation nicht gefährdet. Sobald aber die Funktion nicht-konvex wird, würde diese Differenz alleine die Folgewerte mit hohem Genauigkeitsverlust approximieren.

Weiters sei

$$\Delta_g(S_j) = \begin{cases} \frac{|SP_j(S_j) - SP_j(S_j^{**})|}{|S_j - S_j^{**}|}, & wenn\, S_j \neq S_j^{**}; \\ 0 & sonst; \end{cases}$$

die Steigung zwischen S_j und dem globalen Minimum. Damit soll sichergestellt sein, dass die Approximation zumindest in Richtung globalen Minimum geht und in den nicht-konvexen Teilbereichen die Richtung nicht deutlich darüber liegen wird (Abbildung 5.10).

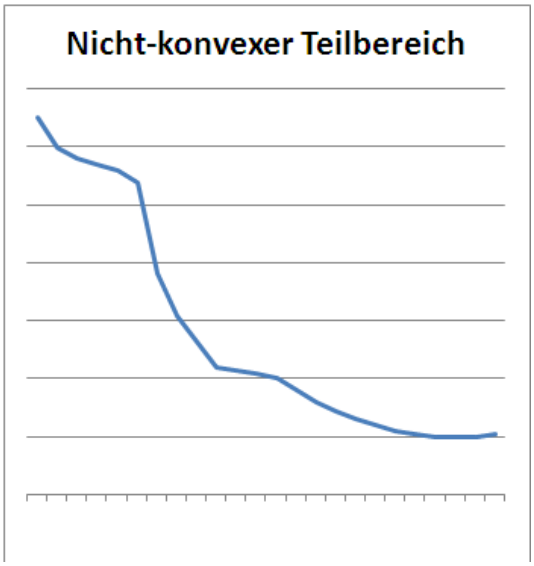

Abbildung 5.8: Nicht Konvexer Teilbereich

Sei

$$\Delta_l(S_j) = \begin{cases} \frac{|SP_j(S_j) - SP_j(S_j^*(z_j-1))|}{|S_j - S_j^*(z_j-1)|}, & wenn\, S_j \neq S_j^*(z_j-1); \\ 0 & sonst; \end{cases}$$

die Steigung zwischen S_j und dem optimalen Wert $S_j^*(z_j - 1)$ bei einem Reparaturteam weniger $(z_j - 1)$ (Abbildung 5.11).

Die Differenz zwischen dem lokalen Minimum bei einem Team weniger und dem aktuell zu approximierenden Wert war am schwierigsten zu legitimieren. Es war vorab die Überlegung, dass es möglicherweise sogar einen Genauigkeitsverlust bringen könnte, wenn $\Delta_l(S_j)$ sehr groß ist und $S_j < S_j^{**}$ gilt. Ich habe erwartet, dass in diesem Fall die Werte zu tief approximiert werden und daher zu "günstig"geschätzt sein könnten. Weiters nahm ich an, dass es sehr rechenintensiv sein könnte, wenn ich alle Kostenwerte $C_j(z_j, S_j)$ bei festem z und wachsenden S_j für den Erhalt von $SP_j(S_j^*(z_j - 1))$ berechnen muss. Da kommt aber die hilfreiche Eigenschaft der Konvexität der Funktionen $C_j(z_j, S_j)$ bei festem z_j dem Algorithmus zu Gute. Dadurch ist es simpel, die Werte zu berechnen, da abgesehen von der Verteilung der Turbinen alle Entscheidungsvariablen fest sind. Somit ist die Berechnung kein großer Aufwand und spricht daher nicht gegen die Nutzung. Damit musste natürlich noch die Sinnhaftigkeit von $\Delta_l(S_j)$ überprüft werden. Damit Δ_l gewählt wird, muss gelten, dass $\Delta_g < \Delta_l$ und $\Delta_s < \Delta_l$. In diesem Fall gibt es zwei mögliche Szenarien:

Entweder gilt zusätzlich $\Delta_s < \Delta_g$. In diesem Fall muss sich $SP(S_j)$ selbst auf einem Sattel befinden. Somit wäre an sich auch Δ_g eine konvexe Approximation. Ab diesem Moment aber würde sich die konvexe Approximation linear

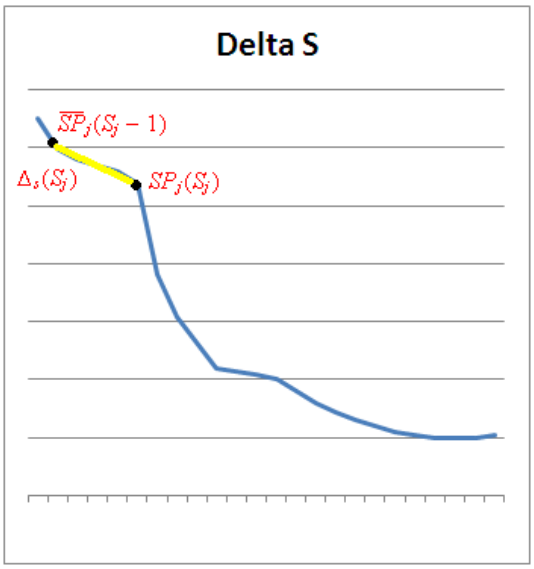

Abbildung 5.9: Darstellung Δ_s

auf das globale Minimum bewegen und wird die meisten der Werte $SP(S_j)$, die dazwischen liegen, zu teuer approximieren, da:

$$\Delta_g(S_j) = \min(\max\{\Delta_s(S_j), \Delta_g(S_j), \Delta_l(S_j)\}, \Delta_p(S_j)), wenn\ \Delta_p(S_j) = \Delta_g(S_j - 1)$$

gilt. Damit dieser ungünstige Fall nicht (zu früh) eintritt, wird $\Delta_l(S_j)$ genutzt. Dafür muss eine wichtige Eigenschaft von $SP(S_j)$ genutzt werden. Es gilt:

$$SP_j(S_j^*(z_j - 1)) \geq SP(S_j) \text{ für } S_j^*(z_j - 1) = S_j$$

Das bedeutet, dass die Funktion $SP(S_j)$ zumindest so tief wie das lokale Minimum verläuft. Somit gibt es einen Sattel und die Funktion macht eine Kurve tiefer fallend als $\Delta_g(S_j)$ es approximieren würde. Hier könnte man die Überlegung aufstellen, dass es vielleicht sinnvoller wäre, direkt die Differenz vom aktuellen $SP(S_j)$ zu diesem $SP'(S_j)$, wo $SP_j(S_j^*(z_j - 1)) \geq SP(S_j)$ gilt, zu nehmen. Dadurch würde aber die tatsächliche geringe Kostenersparnis im Sattel bei $SP(S_j)$ nicht berücksichtigt. Um also einerseits den Verlauf der Funktion in Richtung zu $SP'(S_j)$ nicht gänzlich zu verlieren und trotzdem $SP(S_j)$ nicht zu schlecht zu approximieren, stellt Δ_l besonders für diese Sattel eine ausgewogene Approximation dar.

Die zweite mögliche Variante ist, dass $\Delta_g < \Delta_s$ gilt. Es könnte sogar sein, dass die Funktion in diesem Gebiet lokal konvex ist. Wenn es aber einen Wert Δ_l gibt für den in diesem Fall auch $\Delta_g < \Delta_s < \Delta_l$ gilt, dann muss es zwangsweise einen Sattelpunkt, wie oben schon beschrieben, geben, auch wenn dieser erst bei höheren z_j ist. Dann gilt zwar, dass die aktuellen Werte etwas schlechter

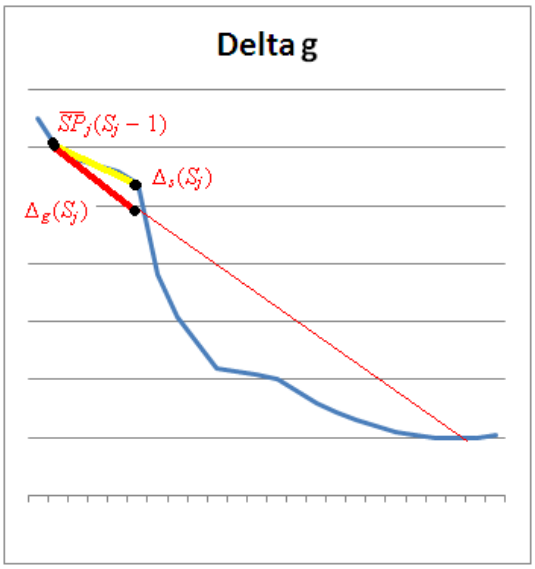

Abbildung 5.10: Darstellung Δ_g

dargestellt, aber der tiefer fallende Bereich nach dem Sattel deutlich besser approximiert werden.

Wenn es keinen Sattelpunkt gibt und die Funktion konvex ist, werden ohnedies die tatsächlichen Werte der Funktion angenommen. Somit ist die Nutzung von Δ_l nicht nur legitimiert, sondern zeigt auch, dass die Approximation dadurch genauer verlaufen wird.

Zuletzt wird

$$\Delta_p(S_j) = \Delta^*(S_{j-1})$$

als die Steigung der gewählten Approximation bei $S_{j-1} = \Delta^*(S_{j-1})$ definiert (Abbildung 5.12).

$\Delta_p(S_j)$ sichert die Konvexität, da der Anstieg im aktuellen Schritt die vorherige nicht übersteigen darf.

Mithilfe dieser Differenzen Δ_* soll die konvexe Hülle konstruiert werden. Es sei

$$\overline{SP}_j(S_j) = \overline{SP}_j(S_j - 1) \begin{cases} -\min(\max\{\Delta_s(S_j), \Delta_g(S_j), \Delta_l(S_j)\}, \Delta_p(S_j)), \\ \quad \text{wenn } S_j \leq S_j^{**} \text{ und } S_j \leq S_j^*(z_j - 1); \\ -\min(\max\{\Delta_s(S_j), \Delta_g(S_j)\}, \Delta_p(S_j)), \\ \quad \text{wenn } S_j \leq S_j^{**} \text{ und } S_j \geq S_j^*(z_j - 1); \\ +\max(\min\{\Delta_s(S_j), \Delta_g(S_j)\}, \Delta_p(S_j)), \\ \quad \text{wenn } S_j \geq S_j^{**} \text{ und } S_j \leq S_j^*(z_j - 1); \\ +\max(\min\{\Delta_s(S_j), \Delta_g(S_j), \Delta_l(S_j)\}, \Delta_p(S_j)), \\ \quad \text{wenn } S_j \geq S_j^{**} \text{ und } S_j \geq S_j^*(z_j - 1); \end{cases}$$

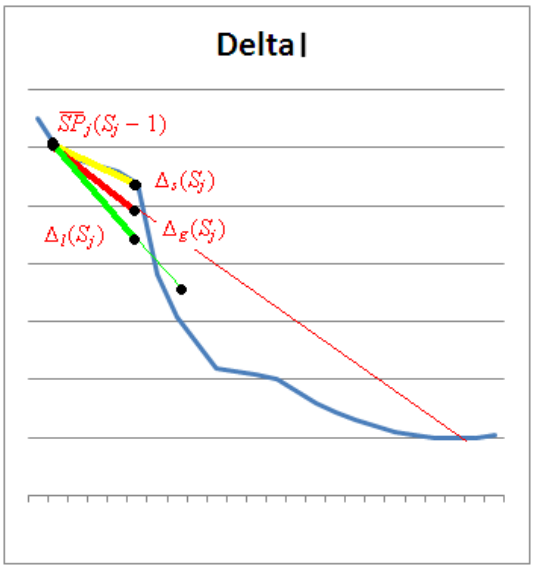

Abbildung 5.11: Darstellung von $\Delta_l(S_j)$ im Vergleich mit $\Delta_g(S_j)$ und $\Delta_s(S_j)$

Dadurch soll die Konvexität zwischen Erhöhungen von S_j überall gewährleistet und in nicht konvexen Bereichen der Funktion geschaffen werden. Trotzdem sollen dadurch jene Werte in den nicht konvexen Bereichen mit möglichst geringem Genauigkeitsverlust approximiert werden. Nebenbei sei erwähnt, dass es zu Beginn keinen Wert für Δ_p gibt und daher im ersten Schritt als ∞ angenommen wird, damit er aus der Auswahl ausfällt. Ein weiterer Vorteil dieser Approximation ist die einfache und schnelle Berechnung. Damit wir daraus die minimalen Kosten für das gesamte Netzwerk berechnen können, muss das Restproblem:

$$\min_{S_j} \sum_{j j \epsilon F^j} \overline{SP}_j(S_j)$$

$$s.t. \quad \sum_{j j \epsilon F^j} S_j \leq S.$$

gelöst werden.

5.5.6 Ergebnisse aus der Umsetzung der konvexen Approximation mit Gams

Bevor ich die ersten Ergebnisse grafisch darstelle, möchte ich erwähnen, dass bei allen Grafiken die Kosten des gesamten Problems, also auch die Kosten aus dem ersten Schritt der mehrstufigen Lösungsvarianten, als Vergleichswerte herangezogen werden. Dies ist vor allem im Vergleich mit dem Simulated Annealing Algorithmus notwendig, da es nicht ausgeschlossen sein kann, dass das optimale

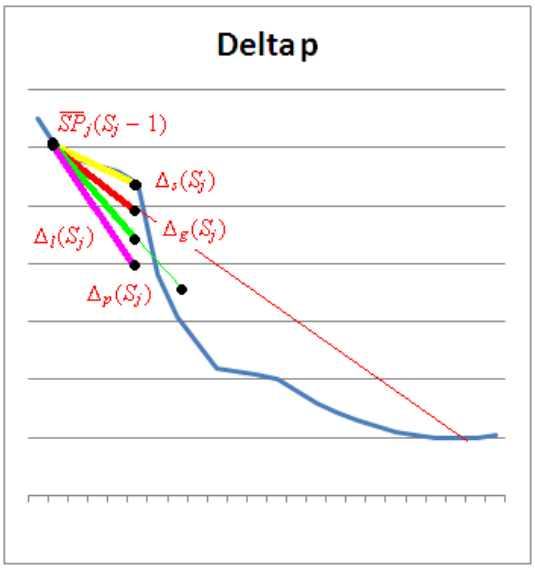

Abbildung 5.12: Darstellung von $\Delta_p(S_j)$ im Vergleich mit $\Delta_g(S_j)$, $\Delta_s(S_j)$ und $\Delta_l(S_j)$

Netzwerk im Simulated Annealing Algorithmus nicht der mehrstufigen Lösung entspricht.

Als Referenz habe ich vorweg überprüft, welche Kosten ein Selbstbedienungsmodell erzeugen würde. In dieser Variante hat jeder Flughafen seine eigene Werkstatt und bedient sich somit selbst. Die Vorteile dieser Netzwerkgestaltung liegen in dem schnellen Ersatz von schadhaften Turbinen (keine Transportzeit) und dem Wegfallen der Transportkosten. Für die Berechnung fällt hier der erste Teil des Modells weg, da die Zuordnung und die Öffnung der Werkstätte klarerweise schon gegeben ist. Den einzigen Wert, den ich benötige, ist das statistische Gleichgewicht für die Warteschlange des Reparaturprozesses. In Abbildung 5.13 ist die Entwicklung der Kosten für das Selbstbedienungsmodell dargestellt.

In Abbildung 5.13 ist ersichtlich, dass sich die Kosten exakt linear entwickeln. Das liegt primär daran, dass die Fehlerrate bei den einzelnen Flughäfen geringer als die Reparaturrate eines Teams ausfällt, wodurch bei den Beispielnetzwerken ein Team pro Werkstatt immer ausreichend ist und es dadurch nie zu Fehlmengen kommen kann. In den meisten Fällen, sollte die Selbstbedienung ebenfalls als Obergrenze der möglichen Kosten für die Modelle dienen.

In Abbildung 5.14 sind die Ergebnisse bei ausgelagerter Reparatur ersichtlich.

Die Kosten bei ausgelagerter Reparatur entwickeln sich in etwa linear mit leichtem Anstieg. Letzterer ist mit der verhältnismäßig höheren Nachfrage der Flughäfen 11-15 zu erklären. Abbildung 5.15 zeigt den Vergleich zwischen dem Selbstbedienungsmodell und der Variante mit ausgelagerter Reparatur.

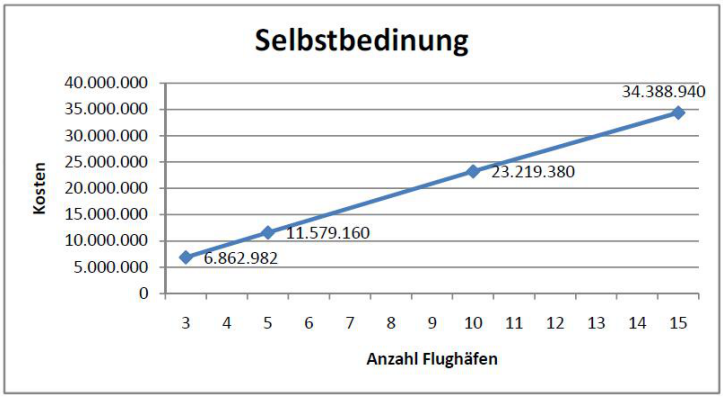

Abbildung 5.13: Kostenentwicklung, wenn jeder Flughafen eine eigene Wekstatt hat

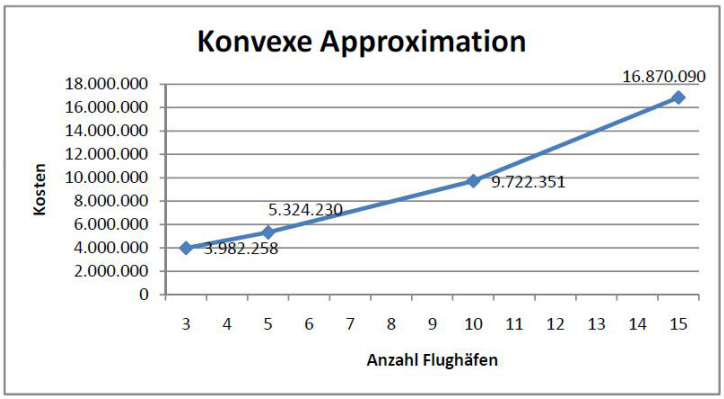

Abbildung 5.14: Kostenentwicklung bei wachsenden Netzwerken der konvexen Approximation

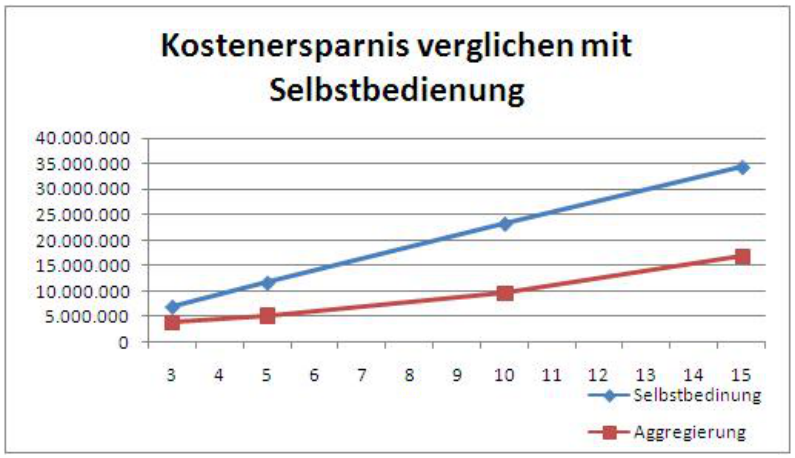

Abbildung 5.15: Kostenvergleich Selbstbedienung vs. Aggregation

Die Abbildung 5.15 zeigt, dass die Aggregation der Reparatur eine deutliche Ersparnis um die 50% bei allen Systemen bringt.

Die Laufzeit des gesamten Algorithmus (Stufe 1 & 2) bei Aggregation der Reparatur ist in Abbildung 5.16 zu sehen.

Verglichen mit der Laufzeit bei Selbstbedienung läuft der Algorithmus deutlich schneller durch (Abbildung 5.17).

Dies liegt unter anderem daran, das der Algorithmus mit der konvexen Approximation für die Berechnung aller notwendigen Werte $C_j(z_j, S_j)$ für jedes geöffnetes Subnetzwerk (bei Selbstbedienung alle Fliegerhorste) viel Rechenzeit in Anspruch nimmt. Da es durch die Aggregation der Reparaturen deutlich weniger Netzwerke gibt, reduziert sich die Rechenzeit entsprechend.

5.5.7 Konvexe Hülle

Im Bereich der konvexen Approximation habe ich in der Abbildung 5.7 auf Seite 41 schon eine grafische Darstellung einer einfachen konvexen Hülle präsentiert. Um zu überprüfen, ob die konvexe Approximation genauere Ergebnisse als die konvexe Hülle liefert, habe ich die konvexe Hülle umgesetzt. Der Algorithmus ist folgendermaßen:

Für jedes Subnetzwerk j gilt, dass solange $S_j \leq S_j^{**}$, also kleiner als das globale Minimum im Subnetzwerk j ist und daher das globalen Minimum noch nicht erreicht ist, folgendes(Hinweis: $\overline{SP}_j$ definiert hier die konvexe Hülle):

1. Starte bei $S_j^s = 1$ mit $\overline{SP}_j(1) = SP_j(1)$(ich starte hier bewusst bei $S_j = 1$, die Erklärung folgt in der Auswertung der konvexen Hülle)

2. Berechne $\Delta_k(S_j^s) = \max_{S_j^+}(\frac{|\overline{SP}_j(S_j^s) - SP(S_j^+)|}{S_j^+ - S_j^s})$ für $S_j^s < S_j^+ < S_j^{**}$ also die größte Differenz (negative Steigung) zwischen S_j^s und S_j^{**}.

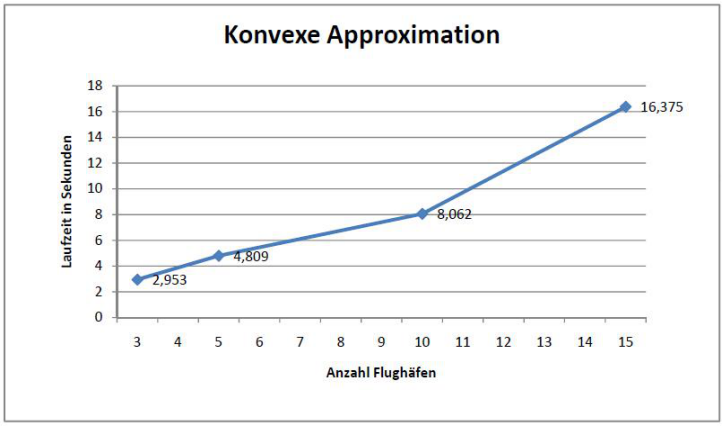

Abbildung 5.16: Laufzeitentwicklung bei wachsendem Netzwerk

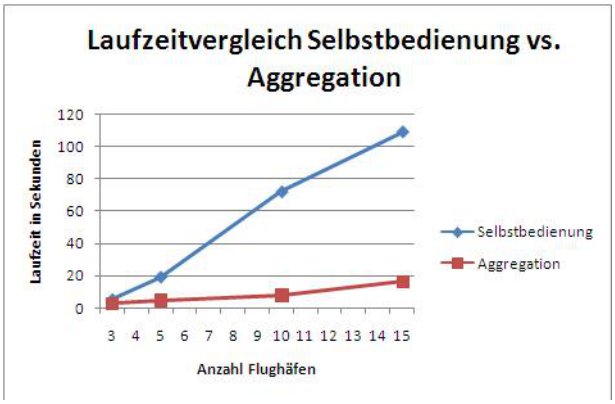

Abbildung 5.17: Laufzeitvergleich Selbstbedienung vs. Aggregation

3. Setze $\overline{SP}_j(S_j^s) = SP_j(S_j^s - 1) - \Delta_k(S_j^s)$ für $S_j^s \leq S_j < S_j^{++} = \arg\max(\Delta_k(S_j^s))$
4. Setze $S_j^s = S_j^{++}$und $SP_j(S_j^+) = \overline{SP}_j(S_j^{++})$
5. wiederhole Schritt 1-4. bis $S_j^+ = S_j^{**}$
6. Setze $SP_j(S_j^{**}) = \overline{SP}_j(S_j^{**})$, $S_j^s = S_j^{**}$
7. Berechne $\Delta_k(S_j^s) = \min_{jS_j^+}(\frac{|\overline{SP}_j(S_j^s) - SP(S_j^+)|}{S_j^+ - S_j^s})$ für $S_j^{**} \leq S_j^s < S_j^+$ also die kleinste Differenz (positive Steigung) zwischen S_j^s und höheren Werten von S_j
8. Setze $\overline{SP}_j(S_j^s) = SP_j(S_j^s - 1) + \Delta_k(S_j^s)$ für $S_j^s \leq S_j < S_j^+$
9. Setze $S_j^s = S_j^+$ und $SP_j(S_j^+) = \overline{SP}_j(S_j^+)$
10. wiederhole Schritt 7-10. bis $\overline{SP}_j(S_j)$ für alle Werte S_j berechnet ist.

Im ersten Teil (Schritt 1-6) wird die konvexe Hülle bis zum globalen Minimum berechnet. Da die Funktion fallend ist, wird immer der größtmögliche Abstieg $\Delta_k(S_j^s)$ genommen, bis der jeweilige Punkt $SP_j(S_j^{++})$ erreicht ist. Dadurch ist gesichert, dass die Funktion SP_j des Subnetzwerks j oberhalb oder auf der konvexen Hülle liegt. Weiters wird das globale Minimum dadurch sicher erreicht und nie unterschritten. Ab S_j^{**} geht die Funktion wieder aufwärts und muss daher langsam wieder steigen. Daher werden in den Schritten 7-10 immer die kleinsten Steigungen zuerst gewählt, sodass wieder alle Unregelmäßigkeiten von $SP_j(S_j)$ konvexisiert werden.

Damit erhalten wir zwar eine konvexe Hülle, aber wenn es eine Beule in der Kostenfunktion gibt werden die umliegenden Werte besonders schlecht approximiert und die wahre Kostentwicklung wird bei der Nutzung der konvexen Hülle verfälscht. Dadurch könnte es leicht passieren, dass beispielsweise ein Sattelpunkt oder ein lokales Maximum als Lösung angenommen wird, wodurch die tatsächlichen Kosten deutlich über den geschätzten liegen werden. Trotz dieser Gründe gegen die konvexe Hülle, habe ich den Algorithmus von Van Roo mit diesem verglichen, um zu sehen, ob die Berechnungsersparnis (da die Konvexisierung deutlich weniger Aufwendig ist) dem Genauigkeitsverlust überwiegt.

5.5.8 Ergebnisse aus der Umsetzung der konvexen Hülle

Bei der Erstellung der konvexen Hülle kam gleich zu Beginn ein überraschendes Ergebnis, dass den Weg in dieser Form erschwert. Die Kosten $C(z_j, 0)$ lagen bei den Testdaten im System mit 15 Flughäfen bei einer der geöffneten Werkstätten deutlich unter den Kosten von $C(z_j, 1)$ wodurch die konvexe Hülle extrem unter den tatsächlichen Werten lag. Daher habe ich den Algorithmus erst bei $S_j = 1$ gestartet, um diesen ersten Sprung zu eliminieren und dadurch den Einfluss von späteren nicht-konvexen Teilbereichen besser zu erkennen. Wenn ich erst bei

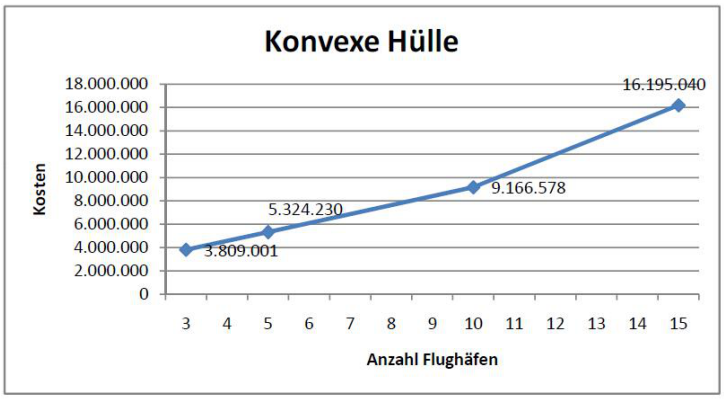

Abbildung 5.18: Kostenentwicklung der Konvexen Hülle

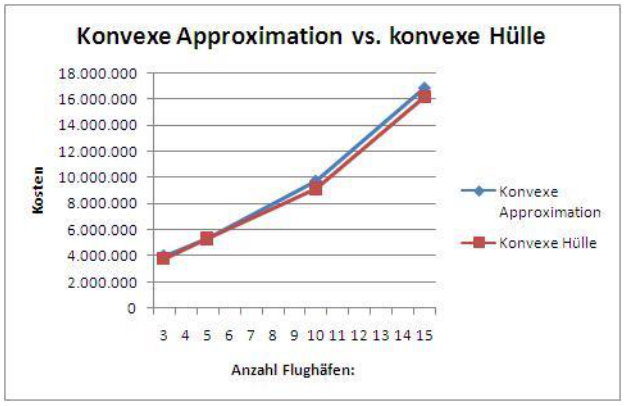

Abbildung 5.19: Kostenvergleich konvexe Approximation vs. konvexe Hülle

$S_j = 1$ beginne, kommt lediglich die Restriktion zustande, dass ich mindestens eine Ersatzmaschine in jedem Netzwerk zur Verfügung haben muss. Die Tests haben gezeigt, dass die Verteilung der Ersatzmaschinen zwischen den einzelnen Netzwerken ähnlich verteilt ist. Damit ist es legitim, bei $S_j = 1$ anzufangen.

Die Ergebnisse der konvexen Hülle werden in der Abbildung 5.18 dargestellt.

Abbildung 5.19 zeigt einen Vergleich der Ergebnisse der konvexen Approximation und der konvexen Hülle.

Die konvexe Hülle antizipiert die Kosten bei allen Netzwerken um ca. 4% niedriger. Diese gleiche Abwertung liegt voraussichtlich an den teilweise gleichwertigen Daten der Fehlerrate und eine aufgrund dieser gleichen Daten ähnlichem Beule in der Kostenfunktion wodurch einmal zumindest Δ_l bei der Konvexisierung gewählt wird. Je ausgeprägter die Dellen in der Funktion SP_j sind, umso größer wäre die Abweichung.

Die Laufzeit (Abbildung 5.20) entwickelt sich ähnlich wie die der konvexen

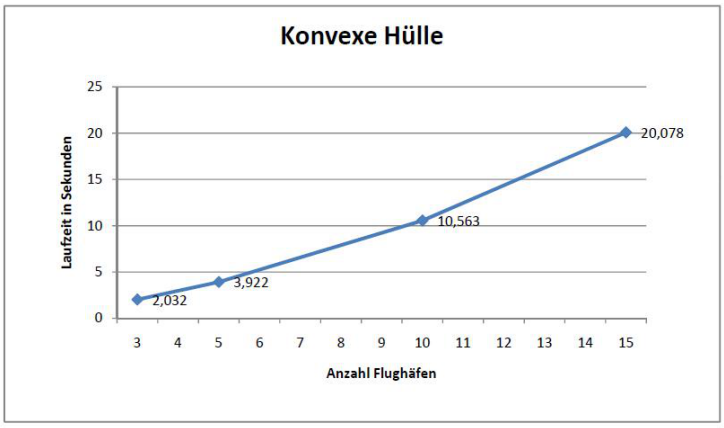

Abbildung 5.20: Rechenlaufzeit Konvexe Hülle

Approximation. Auch wenn Sie bei einem Netzwerk von 15 Flughäfen um ca 25% länger benötigt, ist das nicht relevant, da wir uns noch im Sekundenbereich befinden.

5.5.9 Heuristischer Lösungsansatz für den Schritt 2.4

Im letzte Schritt im zweiten Teil des Lösungsansatzes von Rappold und Van Roo habe ich die kostengünstigste Verteilung der Ersatzmaschinen mit mathematischer Programmierung gelöst. Da aber in diesem Schritt das schon konvexe Optimierungsproblem sehr Umfangreich sein kann, habe ich eine sehr effiziente Heuristik überlegt. Da ich im Schritt 2.3 die konvexe Funktion (entweder die konvexe Hülle, oder die konvexe Approximation) $\overline{SP}_j(S_j)$ für jedes Subnetzwerk berechnet habe und diese Werte für alle möglichen S_j kenne, starte ich bei den optimalen Kosten $K(S^*)$:

$$K(S^*) = \sum_j \min_{S_j} \overline{SP}_j(S_j)$$

und berechne

$$S^* = \sum_j S_j^{**}$$

wobei S_j^{**} jene S_j repräsentieren für die $S_j = \arg\min_{S_j}(\overline{SP}_j(S_j))$ gilt. Hier ergeben sich zwei Fälle:

1. Die Summe $\sum_j S_j^{**} \leq S$. Wenn dies der Fall ist, sind wir fertig und die Optimierung abgeschlossen.
2. Die Summe $\sum_j S_j^{**} > S$ Dann muss die Anzahl an Ersatzmaschinen reduziert werden.

Meine Idee war, die Anzahl an Ersatzmaschinen sukzessive zu reduzieren, bis die maximal verfügbaren Ersatzmaschinen S erreicht ist und dabei die Konvexität der Kostenfunktionen $\overline{SP}_j(S_j)$ auszunutzen. Dabei überprüft das System für jede Reduktion um eine Ersatzmaschine, bei welcher Werkstätte diese am günstigsten ist:

$$K(S^* - 1) = K(S^*) + \min_j(\overline{SP}_j(S_j^{**} - 1) - \overline{SP}_j(S_j^{**}))$$

Für jenes $\overline{SP}_j(S_j^{**})$ das ersetzt wurde, wird das alte S_j^{**} durch den neuen Wert $S_j^{**} - 1$ ersetzt: $S_j^{**} = S_j^{**} - 1$. Danach überprüft der Algorithmus, ob nun die Summe $\sum_j S_j^{**} = S$ ergibt. Wenn dies gilt, sind wir fertig, ansonsten muss der obige Schritt solange wiederholt werden, bis $\sum_j S_j = S$ gilt. Nach dem Ansatz von Rappold und Van Roo gilt für die Optimierung,dass alle verfügbaren Ersatzmaschinen verteilt werden müssen. In diesem Fall könnte es ebenfalls zu dem Szenario kommen, dass $\sum_j S_j^{**} < S$ gilt und somit im globalen Optimum weniger Maschinen benötigt werden, als verfügbar sind. In diesem Fall würde im Ansatz nach Rappold und Van Roo der Algorithmus die Anzahl an Maschinen erhöhen, sonst aber äquivalent vorgehen.

Also wird bei Fall 2 die Anzahl der Ersatzmaschinen Schritt für Schritt reduziert und zwar genau bei jenen Subnetzwerken, wo die Kostenerhöhung am niedrigsten ist. Da die Grenzkosten für kleinere S_j immer größer werden (da $\overline{SP}_j(S)$ konvex ist) führt dieser Algorithmus zur kostenoptimalen Verteilung der Maschinen.

5.5.10 Beispiel für die Umsetzung des alternativen Lösungsansatzes

Nachdem die alternative heuristische Lösung für den Schritt 2.4 mathematisch beschrieben ist, soll der Lösungsansatz noch einmal grafisch verdeutlicht werden. Das Beispiel besteht aus fünf Subnetzwerken j und für das gesamte Netzwerk sollen insgesamt 23 Ersatzmaschinen verfügbar sein ($S = 23$). Für jedes Subnetzwerk gilt die Restriktion, dass maximal 10 Ersatzmaschinen S_j erlaubt sind. Die Kosten sind in diesem Beispiel trivialerweise ganzzahlig. Für den ungewöhnlichen Fall, dass die Preiserhöhung aufgrund der Reduzierung (oder Erhöhung) der Ersatzmaschinen in zwei Subnetzwerken mit tatsächlichen Daten gleich sein sollten, ändert der Algorithmus die Anzahl bei dem Netzwerk, dass nach der Ordnung zuerst kommt (Indifferenz). Abbildung 5.21 zeigt ein Netzwerk mit fünf Subnetzwerken und maximal 10 Ersatzturbinen pro Werkstatt.

Die kostengünstigste Lösung, unabhängig von der tatsächlichen Anzahl an Ersatzmaschinen wird in Abbildung 5.21 dargestellt.

Das Netzwerk in Abbildung 5.21 würde demnach 59 Einheiten kosten. Um diese Kosten zu erreichen müssten aber 28 Maschinen verfügbar sein. Jetzt soll schrittweise die Anzahl auf 23 Stück reduziert werden. Hier sei nochmals

J \ S	1	2	3	4	5	6	7	8	9	10
1	20	17	15	13	12	11	12	13	15	17
2	25	21	18	16	15	14	13	14	15	16
3	24	20	17	14	17	20	24	30	36	43
4	18	15	13	12	11	12	13	15	18	23
5	33	27	22	17	13	10	13	17	22	27

Abbildung 5.21: Beispiel für Schritt 2.4, 5 Subnetzwerke, max. 10 Ersatzmaschinen

S ist:	28	S soll:	23	Kosten:		59				
J \ S	1	2	3	4	5	6	7	8	9	10
1	20	17	15	13	12	11	12	13	15	17
2	25	21	18	16	15	14	13	14	15	16
3	24	20	17	14	17	20	24	30	36	43
4	18	15	13	12	11	12	13	15	18	23
5	33	27	22	17	13	10	13	17	22	27

Abbildung 5.22: Beispiel: Optimale Lösung

erwähnt, dass es sich hier um ein einfaches ersichtliches Beispiel handelt und es daher einige gleichwertige Kostensteigerungen gibt.

In Abbildung 5.23 ändert der Algorithmus die Anzahl an Maschinen im ersten Subnetzwerk, da die Kostensteigerung gemeinsam mit Subnetzwerk 2 und 4 am geringsten ist und es in der Ordnung als erstes kommt.

Da die Kostensteigerung in unserem Beispiel (Abbildung 5.24)im ersten Subnetzwerk immer noch am günstigsten ist, wird wieder die Anzahl verringert.

Als Ergebnis erhalten wir ein System mit der Verteilung der vorhandenen 23 Maschinen mit dem kleinstmöglichen Kostenanstieg von 59 auf 64 Preiseinheiten.

S ist:	28	S soll:	23	Kosten:		59				
J \ S	1	2	3	4	5	6	7	8	9	10
1	20	17	15	13	12 ←	11	12	13	15	17
2	25	21	18	16	15	14	13	14	15	16
3	24	20	17	14	17	20	24	30	36	43
4	18	15	13	12	11	12	13	15	18	23
5	33	27	22	17	13	10	13	17	22	27

Abbildung 5.23: Beispiel: erste Reduktion

S ist:	27	S soll:	23	Kosten:		60				
J \ S	1	2	3	4	5	6	7	8	9	10
1	20	17	15	13	←12	11	12	13	15	17
2	25	21	18	16	15	14	13	14	15	16
3	24	20	17	14	17	20	24	30	36	43
4	18	15	13	12	11	12	13	15	18	23
5	33	27	22	17	13	10	13	17	22	27

Abbildung 5.24: Beispiel: zweite Reduktion

S ist:	26	S soll:	23	Kosten:		61				
J \ S	1	2	3	4	5	6	7	8	9	10
1	20	17	15	13	12	11	12	13	15	17
2	25	21	18	16	15	14	←13	14	15	16
3	24	20	17	14	17	20	24	30	36	43
4	18	15	13	12	11	12	13	15	18	23
5	33	27	22	17	13	10	13	17	22	27

→

S ist:	25	S soll:	23	Kosten:		62				
J \ S	1	2	3	4	5	6	7	8	9	10
1	20	17	15	13	12	11	12	13	15	17
2	25	21	18	16	15	←14	13	14	15	16
3	24	20	17	14	17	20	24	30	36	43
4	18	15	13	12	11	12	13	15	18	23
5	33	27	22	17	13	10	13	17	22	27

Abbildung 5.25: Beispiel: dritte und vierte Reduktion

5.5.11 Ergebnisse aus der Umsetzung des heuristischen Lösungsansatzes für den Schritt 2.4

Die Ergebnisse der heuristischen Variante für den Schritt 2.4 sind in Abbildung 5.27 dargestellt

Da nur das Auswahlverfahren im letzten Schritt geändert wurde, ist es nicht überraschend, dass die Werte gleich berechnet werden und die Ergebnisse übereinstimmen. Daher ist es möglicherweise von Bedeutung, ob die Laufzeit deutlich kürzer ist und somit bei extrem großen Systemen eine Zeitersparnis bei der Optimierungsdauer einbringt.

In der Abbildung 5.29 ist ein Vergleich zwischen der Laufzeit des mathematischen und des heuristischen Ansatzes für den Schritt 2.4.. Dabei zeigt sich tendenziell eine kleine Ersparnis bei dem heuristischen Ansatz.

Wie in Abbildung 5.29 zu sehen, fällt die Ersparnis der Laufzeit bei heuristischer Lösung für den Schritt 2.4 nicht besonders spürbar aus. Die liegt unter anderem daran, dass hier die Laufzeiten der gesamten Optimierung verglichen werden. Da davor alle Schritte gleich ausfallen, ist der Unterschied ausschließlich auf den Unterschied im letzten Schritt zurückzuführen.

5.6 Zwei-stufige mathematische Programmierung

Um die Genauigkeit bzw. den Genauigkeitsverlust der konvexen Approximation messen zu können, habe ich für den Vergleich, anstelle der zweiten Stufe nach Rappold und Van Roo, die zweite Stufe ebenfalls mit mathematischer Programmierung gelöst. Der erste Schritt zur Lokation der Werkstätten, der günstigsten Zuweisung der Flughäfen zu den Werkstätten, sowie der minimalen Anzahl an

S ist:	24	S soll:	23	Kosten:		63				
J \ S	1	2	3	4	5	6	7	8	9	10
1	20	17	15	13	12	11	12	13	15	17
2	25	21	18	16	15	14	13	14	15	16
3	24	20	17	14	17	20	24	30	36	43
4	18	15	13	12	11	12	13	15	18	23
5	33	27	22	17	13	10	13	17	22	27

→

S ist:	23	S soll:	23	Kosten:		64				
J \ S	1	2	3	4	5	6	7	8	9	10
1	20	17	15	13	12	11	12	13	15	17
2	25	21	18	16	15	14	13	14	15	16
3	24	20	17	14	17	20	24	30	36	43
4	18	15	13	12	11	12	13	15	18	23
5	33	27	22	17	13	10	13	17	22	27

Abbildung 5.26: Beispiel: fünfte und sechste Reduktion

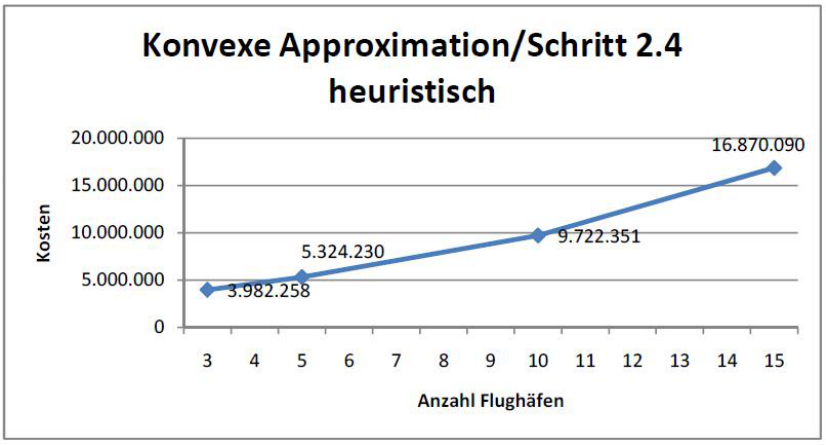

Abbildung 5.27: Konvexe Approximation mit heuristischem Schritt 2.4

Reparaturteams entspricht dem Lösungsansatz von Rapplod und Van Roo. Der zweite Teil der Optimierung, also die Verteilung der Ersatzmaschinen sowie die optimale Anzahl an Reparaturteams bei den einzelnen Werkstätten wird wie im Masterproblem gelöst, wobei in diesem Schritt die vorhergehenden Variablen x_j und y_{ji} als Parameter aus der ersten Stufe genutzt werden.

5.6.1 Ergebnisse aus der Umsetzung

Die Berechnung des Modells war für alle Netzwerke bis zu einer Größe von zehn Fliegerhorsten möglich. Daher eignet sich der zweistufige mathematische Algorithmus lediglich für kleine Netzwerke (Abbildung 5.30 S. 58).

Die Ergebnisse verglichen mit der konvexen Approximation sind in Abbildung 5.31 auf Seite 58.

Interessanterweise ist der Genauigkeitsverlust genau bei dem kleinsten Netzwerk mit drei Flughäfen am größten. Offensichtlich dürfte hier ein stark nichtkonvexer Teilbereich sein, der durch die Konvexisierung die Genauigkeit der konvexen Approximation verschlechtert. Bei höheren Netzwerken entsprechen die Ergebnisse der konvexen Approximation mit einer Schwankung von ca. 1%

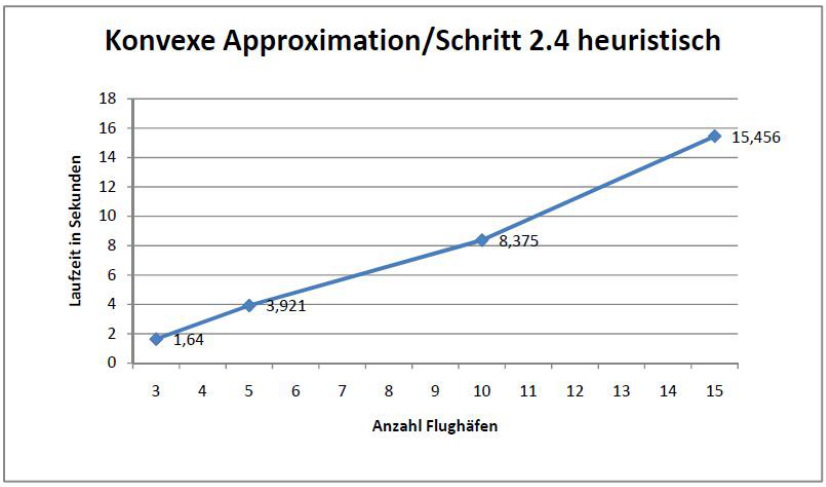

Abbildung 5.28: Laufzeit bei konvexer Approximation mit heuristischem Schritt 2.4

den Ergebnissen der zweistufigen mathematischen Lösung. Somit ist die konvexe Approximation, die auch für höhere Netzwerke lauffähig ist, insgesamt eine sehr gute Alternative. Wenn man jedoch sehr kleine Modelle betrachtet, ist die mathematische Methode vorzuziehen, da der Genauigkeitsverlust sich bei stark nicht konvexen Modellen größer ausprägen könnte. Sobald die Modelle größer sind, muss aber in jedem Fall eine andere Lösung gewählt werden und dafür ist die konvexe Approximation sehr geeignet.

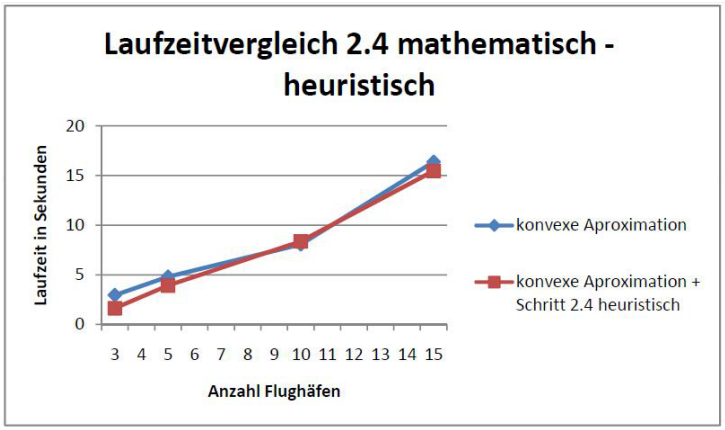

Abbildung 5.29: Laufzeitvergleich Schritt 2.4. mathematisch vs. heuristisch

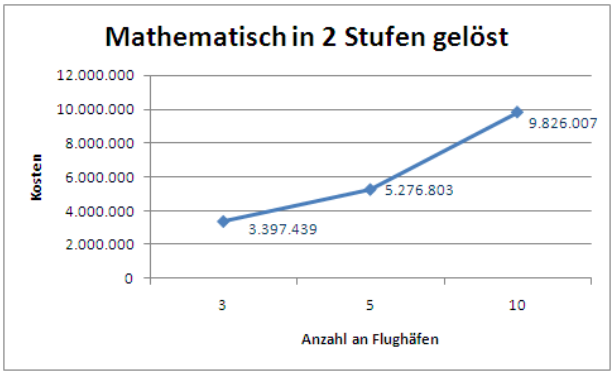

Abbildung 5.30: 2-stufige mathematische Programmierung

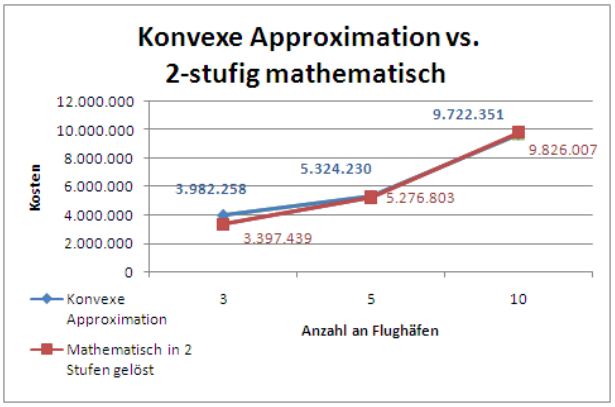

Abbildung 5.31: konvexe Approximation vs. 2-stufige mathematische Programmierung

Kapitel 6

Simulated Annealing

Ich habe einen einstufigen Lösungsweg implementiert um Vergleichswerte zu erhalten. Wie schon früher gezeigt, lässt sich das Problem nicht in einem Schritt mit mathematischer Programmierung lösen. Daher habe ich mich dafür entschieden, das Problem mit einem Simulated Annealing Algorithmus zu lösen. In diesem Abschnitt soll vorab das Prinzip von Simulated Annealing, also der simulierten Abkühlung erklärt und danach die Adaption des Lösungsansatzes auf unser Problem definiert und ausgewertet werden.

6.1 Die Idee des Simulated Annealing

Bei der simulierten Abkühlung (Simulated Annealing) greift die Mathematik auf das physikalisches Prinzip eines Abkühlungsprozesses und dem Energieniveau bei dem Wechsel vom flüssigen Aggregatzustand in den Festen zurück. Dieses besagt, dass eine Schmelze bei langsamer Abkühlung mit höherer Wahrscheinlichkeit ein niedrigeres Energieniveau erreichen wird als bei Schockabkühlung, da die Moleküle mehr Zeit haben eine schöne ausgewogene Struktur anzunehmen. Wenn nun zum Beispiel ein Stück Metall zuerst bis zur Schmelze erhitzt wird und danach zu schnell abgekühlt wird, werden einzelne Teile im nicht optimal niedrigsten Energiezustand steckenbleiben. Wenn hingegen die Abkühlung langsam stattfindet, haben die Teilchen genügend Zeit sich anzuordnen und es wird mit höherer Wahrscheinlichkeit ein nahezu optimal niedriger Energiezustand erreicht. Befindet sich die Schmelze in einer sehr hohen Temperatur, ändert sich die Struktur der Moleküle leichter, wodurch mit höherer Wahrscheinlichkeit auch gering schlechtere Energiezustände angenommen werden. Diese Schwankungen ermöglichen der Schmelze lokal optimale Energiezustände zu verlassen und dadurch ein steckenbleiben zu verhindern. Je kälter die Schmelze umso langsamer sind die Moleküle und ein Wechsel zu einem schlechteren Energieniveau wird unwahrscheinlicher [11]. Diesem Prinzip folgt auch der Lösungsalgorithmus der simulierten Abkühlung. Zu Beginn wird bei einem vorgegebenen Temperaturniveau von einer Startlösung eine Nachbarlösung gesucht, die entweder optimaler

ist oder zumindest gut genug, wenn auch schlechter als die Startlösung. Dabei werden schlechtere Lösungen eher angenommen, wenn die Temperatur noch höher ist. Für jedes Temperaturniveau werden entweder eine fixe Anzahl an Nachbarlösungen oder an vorgegebene Abbruchbedingungen geknüpfte Mengen an Nachbarlösungen verglichen. Danach wird die Temperatur gesenkt und die Nachbarlösungen werden wieder abgesucht.

Der Algorithmus wird wahlweise nach einer fix vorgegeben Anzahl an Abkühlungsschritten beendet, oder nach einer bestimmten Anzahl an Abkühlungen mit nur sehr geringer bis keiner Energieveränderung abgebrochen.

6.2 Der Algorithmus

Sei $f(x)$ die Zielfunktion und x^* eine gewählte Startlösung mit Energieniveau $f^*(x^*)$. Weiters wird eine Starttemperatur T_0 festgelegt, sowie eine Temperaturabnahmefunktion, sodass $T_t = \alpha T_{t-1}$ mit $\alpha \in]0, 1[$, meist mit $\alpha \in [0.80, 0, 995]$ beschränkt.

Die Anzahl n der Vergleiche mit Nachbarlösungen pro Temperaturniveau wird festgelegt und damit für jeden Abkühlungsschritt n-Mal eine Nachbarlösung y gewählt, die Zielfunktion $f(y)$ berechnet und mit dem aktuellen Wert $f^*(x^*)$ verglichen. Wenn eine Nachbarlösung $f(y)$ ein besseres Ergebnis liefert, so wird $f(y)$ fix als neue aktuell beste Lösung $f^*(x^*) = f(y)$ mit den Werten $x^* = y$ angenommen. Ist hingegen die Lösung $f(y)$ schlechter als die Lösung des derzeitig beste $x^* = x$ wird die Lösung y nur unter gewissen Voraussetzungen als neue Lösung akzeptiert. Diese sind einerseits von der Verschlechterung der Lösung und anderseits von der Temperatur abhängig:

- Umso "heißer"das Verfahren ist, umso höher ist die Wahrscheinlichkeit, dass die schlechtere Lösung angenommen wird.
- Umso schlechter die Nachbarlösung ist, umso niedriger ist die Wahrscheinlichkeit, dass sie sie angenommen wird.

Aufgrund der Abkühlung wird es immer unwahrscheinlicher, dass schlechtere Nachbarlösungen angenommen werden und wenn die Temperatur gegen Null geht, pendelt sich die aktuelle Lösung mit hoher Wahrscheinlichkeit im bzw. sehr nahe dem globalen Minimum ein.

6.2.1 Bestimmung der Starttemperatur

Damit der Algorithmus der simulierten Abkühlung mit hoher Sicherheit lokale Minima verlassen wird und gegen das globale Minimum konvergiert, ist es sinnvoll bei der Bestimmung der Starttemperatur zu testen, wie ein gewähltes Temperaturniveau einen Sprung von der aktuellen Lösung x^* zu einem Nachbarn erlaubt. Nach Kirkpatrick [11] gibt es eine praktikable Regel für die Bestimmung einer geeigneten Starttemperatur:

Zuerst starte ich bei einer nach eigenem Ermessen gewählten Temperatur und führe 50-200 Schritte in diesem Temperaturniveau durch. Wenn die Sprunghäufigkeit unter 80% liegt, verdoppelt ich die Temperatur und führe die Überprüfung erneut durch. Sobald die Häufigkeit über 80% liegt, habe ich eine geeignete Starttemperatur gefunden.

6.2.2 Bestimmung der Anzahl an Schritten in einer Temperatur

Grundsätzlich gibt es zwei Möglichkeiten die Anzahl de Schritte in einer Temperatur zu wählen. Entweder ist die Anzahl der Schritte fix vorgegeben, oder es gibt eine Abbruchbedingung, die solange eine bestimmte Menge an Schritten (z.B.: 50-100) wiederholt indiziert bis ein gewisser Zustand erreicht ist und die Nachbarlösungen nur mehr wenig variieren. Dann wird die Temperatur verringert.

6.2.3 Bestimmung der Wahrscheinlichkeit, Nachbarlösugen anzunehmen

Sei $\Delta ZF = f(y) - f^*(x^*)$, also die Differenz zwischen der derzeitigen besten Wertes und der gewählten Nachbarlösung. Die Wahrscheinlichkeit $P(\Delta ZF)$, dass die Nachbarlösung y mit $f(y)$ angenommen wird, wird wie folgt bestimmt:

$$P(\Delta ZF) = \begin{cases} e^{\frac{-\Delta ZF}{T_t}} & \Delta ZF > 0 \\ 1 & \Delta ZF < 0 \end{cases}$$

Dabei definiert T_t die Temperatur im Iterationsschritt t.

6.2.4 Hybride Algorithmen

Es ist ebenfalls möglich eine simulierte Abkühlung gemeinsam mit einer mathematischen Programmierung zu lösen. Hierfür bestimmt die simulierte Abkühlung nicht alle Variablen (wie im reinen Simulated Annealing), sondern nur eine Auswahl. Übrig bleibt ein deutlich einfacheres Restproblem, welches mittels mathematischer Programmierung gelöst wird. Diese Variante eignet sich vor allem für Modelle in denen die Entscheidungsvariablen einen sehr stark variierenden Einfluss auf das Modell haben, da sonst die Gefahr besteht, dass die reine simulierte Abkühlung bei zu vielen unterschiedlichen Variablen in den Lösungsräumen hin und her springt.

6.3 Anwendung an das Problem

Aufgrund der vorangehenden Überlegungen habe ich das Modell mit einem hybriden Algorithmus aus Simulated Annealing und mathematischer Programmierung gelöst. Der Grund für die Wahl des hybriden Ansatzes ist der unterschiedlich starke Einfluss der Variablen auf das System. So ist es einerseits aus

unternehmerischer Sicht verhältnismäßig eine Kleinigkeit die Anzahl an Reparaturteams zu ändern. Hingegen hat es einen sehr starken Einfluss, eine Werkstatt zu schließen oder zu öffnen, da diese Änderung ebenfalls in das übrige System stark eingreift, sodass sich anderen Variablen zwangsweise mitändern müssen. Insgesamt ergeben sich daher für die Umsetzung folgende essenzielle Fragen:

- Wo starte ich den Algorithmus - Startlösung
- Welche Werte sollen über den Simulated Annealing Algorithmus gelöst werden
- Wie wähle ich Nachbarlösungen
- Wie wähle ich die Starttemperatur
- Wie schnell soll die Temperatur abkühlen
- Wie oft wiederhole ich die Prozedur in einer Temperatur

6.3.1 Bestimmung der Startlösung

Für die Startlösung habe ich trivialerweise bei allen Flughäfen eine Werkstätte geöffnet und somit eine Selbstbedienung erstellt. Vorangehende Ergebnisse haben gezeigt, dass dieses System eindeutig teurer ist als ein System, mit wenigen Werkstätten und ausgelagerter Reparatur. Es wird genau die kleinste Anzahl an Reparaturteams angenommen, die das statistische Gleichgewicht für die Warteschlangen erfüllt. Mit diesen Werten berechnet der Algorithmus den Startwert.

6.3.2 Bestimmung der Variablen, die über Simulated Annealing gelöst werden

Ich habe mich dafür entschieden die strategischen Entscheidungsvariablen x_j, also die Bestimmung, ob eine Werkstatt bei dem Fliegerhorst j geöffnet wird und die Zuordnung der Fliegerhorste zu den Werkstätten y_{ji}, sowie die Anzahl an Teams z_{jk} mittels der simulierten Abkühlung zu lösen. Das Restproblem besteht daher nur noch aus der Zuordnung der Ersatzturbinen auf die einzelnen Subnetzwerke S_j und der Verteilung der Ersatzmaschinen in den einzelnen Subnetzwerken auf die Lager der Werkstätten s_{j0} und auf der Fliegerhorste s_{ji}.

Der Grund für diese Aufteilung liegt an dem sehr stark variierenden Einfluss der einzelnen Variablen, die es erschweren, das Modell mittels einem reinen Simulated Annealing Algorithmus zu lösen. Damit der Algorithmus nicht zu stark hin und her springt habe ich die Gruppe der Ersatzturbinen herausgezogen und diese mit den Ergebnissen aus der simulierten Abkühlung mittels einem einfachen mathematischen Restproblems berechnet.

6.3.3 Wählen von Nachbarlösungen bei der simulierten Abkühlung

Aufgrund des stark variierenden Einfluss der einzelnen Variablen habe ich ebenfalls einen Weg gewählt, bei dem die algorithmisch einfacher änderbaren Variablen mit höherer Wahrscheinlichkeit gewählt werden. Vorab möchte ich aber meine Überlegungen erörtern: Die in diesem Modell einfachste und flexibelste änderbare Variable ist die Anzahl an Reparaturteams z_{ji}, da diese keinen unmittelbaren Einfluss auf die anderen Entscheidungsvariablen in diesem Schritt haben, solange das statistische Gleichgewicht gesichert ist. Die Zuordnung der Flughäfen y_{ji} ist ebenfalls eher flexibel, wobei es bei einer Änderung auch ggf. zu einer Änderung von z_{ji} kommen muss, sofern dadurch die Bedingung des statistischen Gleichgewichts bei der "begünstigten"Werkstatt nicht mehr erfüllt sein sollte. Die Öffnung bzw. die Schließung einer Werkstätte x_j hat, wie schon erwähnt, zeitgleich Einfluss auf die Zuordnung y_{ji} und die Anzahl der Teams z_{ji}.

Damit der Einfluss der Variable auch die Häufigkeit der Auswahl beeinflusst, wird über eine Zufallsvariable bestimmt, welche Variable geändert wird um eine Nachbarlösung zu berechnen. Dabei ist die Änderung der Anzahl an Teams am Wahrscheinlichsten, die Zuordnung an zweiter Stelle und die Öffnung oder Schließung einer Werkstätte mit der kleinsten Wahrscheinlichkeit, damit der daraus entstehende, starke Unterschied nicht zu häufig geschieht.

Die Auswahlwahrscheinlichkeiten, welche Variable geändert wird, wird folgendermaßen bestimmt:

1. Erhöhe die Anzahl an Reparaturteams bei einer Werkstätte

   Bedingung: Die maximal mögliche Anzahl am Teams ist noch nicht erreicht

   Auswahlwahrscheinlichkeit: 30%

2. Reduziere die Anzahl an Reparaturteams bei einer Werkstätte

   Auswahlwahrscheinlichkeit: 30%

   Bedingung: Es sind mindestens zwei Teams verfügbar

3. Weise einen Fliegerhorst einer anderen Werkstätte zu:

   Der Algorithmus:

   - Wähle eine Zuordnung
   - Entferne die alte Zuordnung
   - Weise den Flughafen einer zufällig gewählten, nicht der alten entsprechenden Werkstätte
   - Überprüfe, ob die Anzahl an Reparaturteams bei der neuen Werkstätte trotz der hinzugekommenen Zuordnung noch immer das statistische Gleichgewicht für Warteschlangen erfüllt.
   - Erhöhe ggf. die Anzahl an Teams, damit die Bedingung erfüllt ist.

Auswahlwahrscheinlichkeit: 25%

Bedingung: Es sind mehr als eine Werkstatt vorhanden, der Fliegerhorst hat selbst keine Werkstatt

4. Öffne eine neue Werkstätte:

   Der Algorithmus:

   - Wähle zufällig einen Flughafen ohne Werkstätte und öffne die Werstätte
   - Der Flughafen mit der neuen Werkstatt wird dieser zugewiesen
   - Die alte Zuordnung wird gelöscht
   - Die Mindestanzahl an Teams bei der neuen Werkstatt wird ermittelt und gewählt

   Auswahlwahrscheinlichkeit: 7,5%

   Bedingung: Es herrscht noch nicht der Zustand der Selbstbedienung

5. Schließe eine Werkstätte

   Der Algorithmus:

   - Wähle zufällig eine Werkstätte
   - Schließe die Werkstätte
   - Alle Flughäfen, die dieser Werkstätte zugeordnet waren, werden den nächstgelegenen (geringste Distanz) zugeordnet, sofern dort damit das statistische Gleichgewicht noch erfüllt werden kann. Bei distanzieller Indifferenz wird jener Flughafen gewählt, wo die vorgegebene Auslastung niedriger ist.
   - Erhöhe ggf. die Anzahl an Teams bei den Werkstätten denen die Flughäfen zugeordnet wurden, damit die Bedingung erfüllt ist.

   Auswahlwahrscheinlichkeit: 7.5%

   Bedingung: es sind mindestens zwei Werkstätten vorhanden.

Nachdem mittels der simulierten Abkühlung ein Teil der Entscheidungsvariablen ermittelt wurden, wird das übrige Restproblem mittels mathematischer Programmierung gelöst:

$$\min_{S_j, s_{ji}, s_{j0}} \sum_{j \in R} \sum_{i \in F} (h_{j0}\Psi(z_j, s_{j0}) + h_{ji}\Gamma_i(\lambda_{ji}, z_j, s_{j0}, s_{ji}) + \pi_{ji}\Phi_i(y_{ij}, z_j, s_{j0}, s_{ji}))$$

s.t.

$$\sum_{j \in R} S_j \leq S$$

$$s_{j0} + \sum_{i \in F} s_{ji} = S_j$$

$$s_{ji}, s_{j0} \neq 0 \quad \forall i \in F : y_{ji} = 1 \quad \forall j \in R : x_j = 1;$$

$$s_{ij}, s_{j0} \in N \quad \forall i \in F : y_{ji} = 1 \quad \forall j \in R : x_j = 1;$$

6.3.4 Bestimmung der Temperatur und der Abkühlungsgeschwindigkeit sowie der Schritte

Nach einigen Testläufen hat sich ein Temperaturniveau von $T_0 = 200$ als praktikabel herausgestellt. Pro Temperaturniveau habe ich 25 Schritte festgelegt. Die Anzahl der Schritte ist vergleichsweise sehr niedrig gehalten (manche Modelle haben 100-200 Schritte mit Abbruchbedingung), dafür habe ich die Abkühlung auf ein Grad pro Zeiteinheit festgelegt und diese Variante ist für den Benchmarkvergleich ausreichend. Dadurch ergeben sich insgesamt 5000 Vergleiche mit Nachbarlösungen.

6.3.5 Annahme und Ablehnung von Nachbarn

Das Ergebnis wird, wie früher schon erklärt, im Abkühlungsniveau T_t zum Zeitpunkt t mittels der Differenz $\Delta ZF = f(y) - f^*(x^*)$, also die Differenz zwischen der derzeitigen besten Wertes und der gewählten Nachbarlösung und der Annahmewahrscheinlichkeit $P(\Delta ZF)$, dass die Nachbarlösung y mit $f(y)$ angenommen wird, wie folgt bestimmt:

$$P(\Delta ZF) = \begin{cases} e^{\frac{-\Delta ZF}{T_t}} & \Delta ZF > 0 \\ 1 & \Delta ZF < 0 \end{cases}$$

6.3.6 Ergebnisse aus der Umsetzung des Simulated Annealing

Bei der simulierten Abkühlung ist es gebräuchlich, während des Durchlaufs das insgesamt beste erreichte Ergebnis ebenfalls zu speichern. In Abbildung 6.1 (S. 66) habe ich einen Vergleich zwischen dem Endergebnis und dem erreichten Optimum dargestellt.

In der Abbildung 6.1 ist ebenfalls ersichtlich, dass der Algorithmus die beste erreichte Lösung gestriffen, aber meist wieder verlassen hat. Das ist ein Qualitätsproblem des Lösungsansatzes der simulierten Abkühlung. Trotzdem ist es interessant, dass beide Ergebnisse zwar bei den kleineren Referenzmodellen den Ergebnissen aus dem Lösungsansatz von Rappold und Van Roo sehr nahe sind, aber bei größeren Referenzmodellen deutlich optimalere Ergebnisse liefern (Abbildung 6.2 S. 67).

Aus dem Ergebnis aus der Abbildung 6.2 Seite 67 lässt sich eindeutig schließen, dass die Aufteilung in zwei Stufen zu starken Einbußen in der Optimierung führen. Das liegt vor allem daran, dass es durch die Teilung in zwei Stufen bei der Optimierung der ersten Stufe einen starken Informationsverlust gibt, da die Verteilung der Ersatzturbinen, sowie die Anzahl an Reparaturteams gänzlich aus der Entscheidung ausgeschlossen sind. Offensichtlich haben diese beiden taktischen Entscheidungen einen starken Einfluss auf die strategischen Entscheidungen. Damit ist in diesem Fall die Aufteilung in zwei Stufen wenig praktikabel.

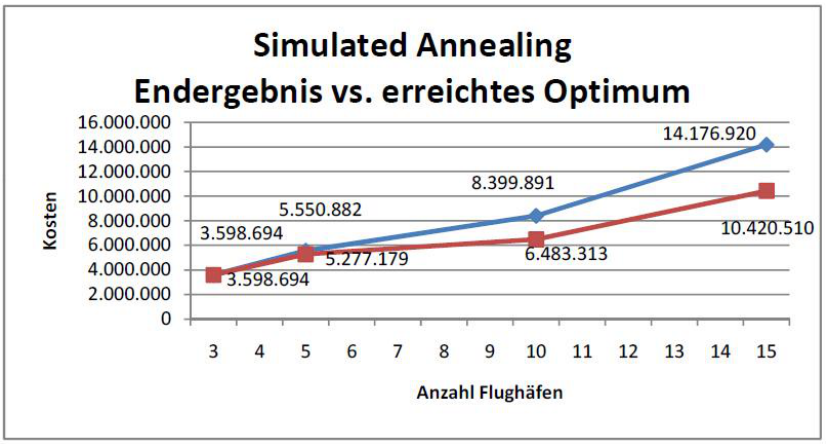

Abbildung 6.1: Ergebnisvergleich Simualted Annealing

Laufzeit der hybriden Simulierten Abkühlung

Die Laufzeit der simulierten Abkühlung bewegt, wie in Abbildung 6.3 (S. 67) sich in einem Zeitrahmen von ca. 20-40 Minuten.

Die Laufzeit ist zwar um ein vielfaches größer als die der anderen getesteten Algorithmen, jedoch sind diese immer noch in einer vertretbaren Dauer. Der Vergleich der Laufzeiten der simulierten Abkühlung in Abbildung 6.3 zeigt, dass sich die Laufzeit auch bei sehr großen Netzwerken kaum verändert. Dadurch ist gezeigt, dass das Restproblem nach Bestimmung der Teillösung des Nachbarn mittels simulierter Abkühlung trivial und schnell gelöst werden kann und die Laufzeit lediglich an der Anzahl der Schritte und der Abkühlungsgeschwindigkeit liegt.

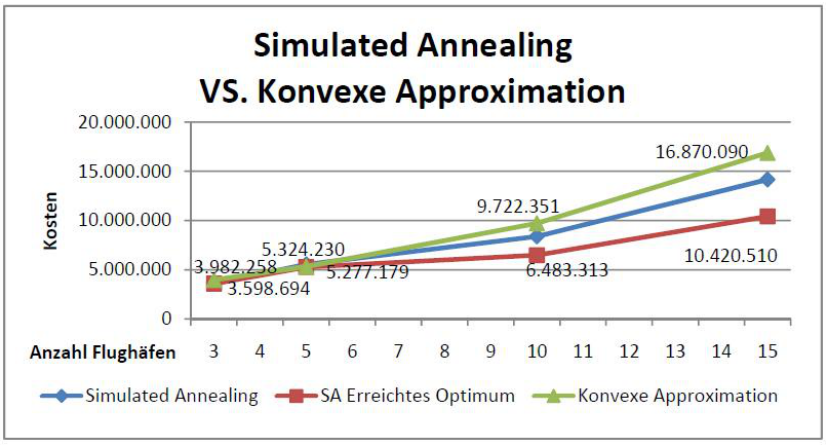

Abbildung 6.2: Ergebnisvergleich Simulated Annealing - Konvexe Approximation

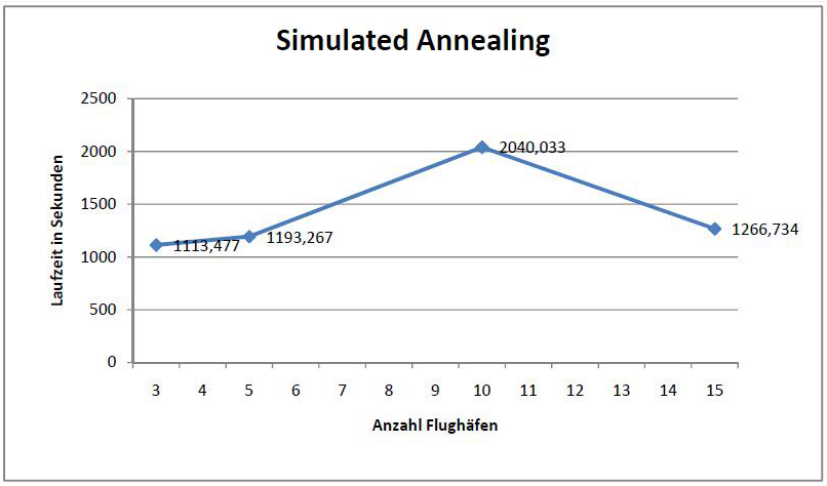

Abbildung 6.3: Laufzeit des Simulated Annealing Algorithmus

Kapitel 7

Conclusio

Abschließend kann ich feststellen, dass zum einen eine Aggregation der Reparaturen der Fliegerhorste auf ausgewählte Werkstätten immer eine deutliche Einsparung bringt und dabei trotzdem die Leistungsbedingungen erfüllt werden können. Die Tests haben gezeigt, dass schon bei kleineren Systemen das Problem nicht in einem Schritt mittels mathematischer Programmierung gelöst werden konnten. Das liegt daran, dass das Problem NP-schwierig ist. Die zweistufige mathematische Programmierung hat zwar für kleinere Referenzmodelle sinnvolle Ergebnisse geliefert, aber bei Referenzmodellen mit mehr als 10 Fliegerhorsten wurde das Problem zu komplex.

Der Lösungsansatz von Rappold und Van Roo [1] hat sehr schnell das Problem gelöst. Die Varianten der mathematischen oder heuristischen Lösung des vierten Unterschrittes des zweiten Teils der Optimierung (Schritt 2.4) haben bei ähnlich schnellen Laufzeiten die selben Ergebnisse geliefert und sind somit beide auch für große Netzwerke zulässig.

Der Vergleich mit der konvexen Hülle hat gezeigt, dass die konvexe Approximation nach Rappold und Van Roo der konvexen Hülle sehr nahe kommt und daher durch den geringeren Genauigkeitsverlusts gegenüber der konvexen Hülle die Zielfunktion nicht im selben Ausmaß verfälscht. Dadurch erhält man mit der konvexen Approximation eine genauere Lösung, da die Kosten nicht, wie bei der konvexen Hülle, durchgehend unterschätzt werden sondern eine konvexe Funktion eingepasst wird.

Der Vergleich mit den Ergebnissen aus dem hybriden Simulated Annealing Algorithmus haben gezeigt, dass besonders bei großen Referenzmodellen durch die Aufteilung in zwei Stufen und den dadurch verlorenen Einfluss der taktischen Entscheidungen in der ersten Stufe die Zweistufigen Modelle ein deutlich schlechteres Ergebnis liefern. Damit hat sich gezeigt, dass die Berücksichtigung aller Entscheidungsvariablen für die Bestimmung der Netzwerkstruktur eine deutliche Kostenersparnis bringen würde.

Alle Ergebnisse sind noch einmal in der Abbildung 7.1 illustriert.

In der Abbildung 7.1 ist noch einmal ersichtlich, dass alle Algorithmen kostengünstigere Varianten lieferten als die Selbstbedienung. Bezüglich der Lauf-

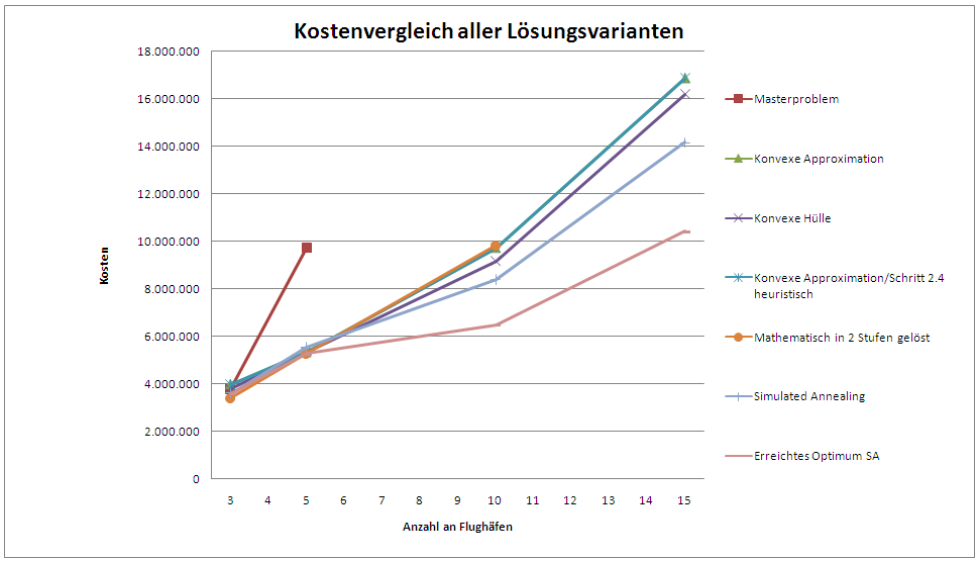

Abbildung 7.1: Ergebnisvergleich der verschiedenen Algorithmen

zeiten ist ein Gesamtvergleich wenig aussagekräftig, da Aufgrund der inhärenten Modellierung des Simulated Annealing Algorithmus dessen Laufzeit unvergleichbar länger ist.

Literaturverzeichnis

[1] Rappold, J.A., Van Hoo, B.D., 2008. Designing multi-echelon service parts networks with finite repair capacity. European Journal of Operational Research 199, 706-722.

[2] Neumann, K., 1977. Operations Research Verfahren Band II. Carl Hanser Verlag, München Wien.

[3] Prabhu, N.U., 1965. Queues and Inventories: A Study Of Their Basic Stochastic Processes. John Wiley and Sons, New York.

[4] Muckstadt, J.A., 2005. Analysis and Algorithms for Service Parts Supply Chains. Springer Verlag New York.

[5] Richard E. Rosenthal, GAMS - A User's Guide;Tutorial

[6] Christian Dornbacher, 1994. Warteschlangen

[7] Sherbrooke, C.C.1968, Metric, A multi-echelon technique for recoverable item control

[8] Schietzelt, T. H. and Densham, P.J. 2003. Location-allocation in GIS

[9] Qian Zhang, Xiangpei Hu, Heuristic Algorithm for Location-Allocation Problem Based on Wavelet Analysis in Integrated Logistics Distribution

[10] Karthik Sourirajan, Leyla Ozsen, Reha Uzsoy, 2009, A genetic algorithm for a single product network design model with lead time and safety stock considerations

[11] S. Kirkpatrick; C. D. Gelatt; M. P. Vecchi, Optimization by Simulated Annealing

[12] Annalisa Cesaro, Dario Pacciarelli, 2009, Optimal stock allocation in single echelon inventory systems subject to a service constraint

[13] K. Balaji Reddy, S. Narayanan, P. Pandian, 2011, Single-Echelon Supply Chain Two Stage Distribution Inventory Optimization Model for the Confectionery Industry

[14] P. Köchel, U. Nieländer, Simulation-based optimisation of multi-echelon inventory systems

[15] Calvin B. Lee, Multi-Echelon Inventory OptimizationCalvin B. Lee, Ph.D.

[16] Mark S. Daskin, Lawrence V.Snyder, Rosemary T. Berger, 2003, Facility Location in Supply Chain Design

[17] Wen Zhou, Department of Statistics Iowa State University, 2011, The M/M/c queue