Verknüpfung mathematischer Modelle zur zieloptimierten Produktionsplanung in Rohöl verarbeitenden Industrien

Dissertation

Schriftliche Arbeit zur Erlangung des akademischen Grades eines Doktors der Wirtschaftswissenschaften

Vorgelegt am Fachbereich für Wirtschaftswissenschaften der Universität GH Paderborn

von Dipl. Wirt.-Ing. Franz Haking

Paderborn 2006

Inhalt: **Seite:**

1 Zielsetzung der Arbeit und Abgrenzung des betrachteten Themenbereiches

Zielsetzung dieser Arbeit ist die Entwicklung eines Algorithmus zur zieloptimierten Produktionsplanung in der Rohöl verarbeitenden Industrie. Im Vordergrund steht dabei die Verknüpfung mathematischer Modelle zur Ermittlung des globalen, alle beteiligten mathematischen Modelle umfassenden Optimums auf Basis der verteilten Optimierung.

Neben einer Einführung zu mathematischen Modellen als Hilfsmittel der Produktionsplanung in Rohöl verarbeitenden Raffinerien gliedert sich diese Arbeit im Wesentlichen in zwei Hauptgebiete.

Gegenstand des ersten Teils ist die Ermittlung einer geeigneten Kombination von Planungssystemen und deren mathematischen Modelle zur Umsetzung des Systems der verteilten Optimierung. Zu diesem Zweck werden in Kapitel 3 verbreitete Planungssysteme unter Berücksichtigung des zeitlichen Planungshorizonts vorgestellt und hinsichtlich des Automatisierungsgrades der Systemschnittstellen sowie des Verbreitungsgrads bewertet. In Kapitel 4 werden direkte und indirekte Optimierungsverfahren vorgestellt und in Bezug auf eine Implementierung des Systems der verteilten Optimierung klassifiziert. Unter Berücksichtigung der in den Kapiteln 3 und 4 erarbeiteten Ergebnisse werden in Kapitel 5 die systemseitigen Mindestanforderungen für die Implementierung des Systems der verteilten Optimierung definiert und eine zu verknüpfende Systemkombination festgelegt.

Im zweiten und wesentlichen Teil dieser Arbeit wird schrittweise eine mathematische Beweisführung sowie eine algorithmische Umsetzung des Ansatzes der verteilten Optimierung erarbeitet, durch empirische Testreihen

untersucht und bewertet. Zu diesem Zweck wird in Kapitel 7 der mathematische Beweis der theoretischen Richtigkeit der Grundidee der verteilten Optimierung erbracht und dokumentiert. Die Darstellung einer geeigneten Struktur verbundener Planungssysteme folgt. Basierend auf dem vorgestellten mathematischen Zusammenhang wird im gleichen Kapitel der im Rahmen dieser Arbeit entwickelte Algorithmus zur verteilten Optimierung dargestellt, die Funktionsweise beispielhaft beschrieben sowie die Implementierung in ein bestehendes System der linearen Programmierung (LP-System) erörtert. Schließlich werden in Kapitel 8 die Ergebnisse durchgeführter Untersuchungen bei der Umsetzung des Algorithmus der verteilten Optimierung und den gewonnenen Erkenntnissen hinsichtlich Möglichkeiten und Grenzen vorgestellt. Im Detail werden die Untersuchungsergebnisse in Bezug auf den praktischen Einsatz des entwickelten Ansatzes dokumentiert und eine Aufwandsabschätzung hinsichtlich der praktischen Umsetzung des Ansatzes der verteilten Optimierung gegeben. Zur Ermittlung der Möglichkeiten und Grenzen des implementierten Systems werden die im Rahmen dieser Arbeit durchgeführten Testreihen beschrieben und die erarbeiteten Ergebnisse vorgestellt.

Da sich in den Testreihen die Bestimmung des globalem Optimums im Rahmen der Verknüpfung mathematischer Modelle durch das System der verteilten Optimierung als zentrales Problem zeigte, wird abschließend ergänzend auf bestehende Ansätze zur Bestimmung des globalen Optimums in der Literatur eingegangen.

2 Mathematische Modelle als Hilfsmittel zur Produktionsplanung in Rohöl verarbeitenden Raffinerien

Die Verarbeitung von Rohölen zu Motorkraftstoffen gewann in den vergangenen Jahrzehnten parallel zur steigenden Verbreitung des Automobils zunehmend an Umfang und Komplexität. Ebenso entwickelte sich sowohl für Zukaufs- als auch für Verkaufsprodukte ein stark expandierender Markt. Steigende Komplexität und allgemeine wirtschaftliche Mechanismen förderten die Entwicklung von Hilfsmitteln zur Optimierung des Prozesses. Immer komplexere technische Strukturen und steigende Anforderungen an die Produktqualität treiben bis heute die Entwicklung mathematisch basierter Optimierung voran.

Erste Ansätze in der Historie zur Optimierung komplexer Verteilungsprobleme gehen auf G.B. Dantzig[1] zurück. Bereits 1948 entwickelte dieser bei der US Air Force Algorithmen zur Lösung mathematischer Modelle. Eine bedeutende Entwicklung daraus ist die Simplex-Methode, auf die im weiteren Verlauf dieser Arbeit noch detailliert eingegangen wird.

Eine der ersten größeren Untersuchungen über die Lösung von Entscheidungsproblemen in einer Raffinerie mit Hilfe dieser Form der Mathematischen Programmierung stellte G.H. Symonds[2] 1955 vor. Speziell der Einsatz der Linearen Programmierung in der Raffinerie wurde 1957 von Garvin, Crandall, John und Spellmann in ihrem Artikel "Applications of Linear Programming in the Oil Industry"[3] beschrieben. Über die Optimierung der Versorgung von Raffinerien mit Rohstoffen mittels LP-Modellen wurde von Catchpole[4] 1962 berichtet. Aber auch prozesstechnische Optimierungs-

[1] Dantzig, G.B. (1948)
[2] Symonds, G.H. (1955)
[3] Garvin, W., Crandall, H., John, J., Spellmann, R. (1957), S. 407 ff.
[4] Catchpole, A.R. (1962), S. 163-169

probleme in einer Raffinerie wurden zum Inhalt linearer Optimierungen, wie z.B. die technische Auslegung von Destillationskolonnen[5]. Schließlich zeigte sich das Blending, im Sinne des Mischens fließ- und schüttfähiger Komponenten in Raffinerien als technisch und wirtschaftlich sinnvolles Einsatzgebiet linearer Programmierung, wie Magoulas, Marinos-Kouris, Lygeros[6] sowie Ramsey und Truesdale[7] zeigen.

Parallel zur Entwicklung der Rechnerkapazität von Computern und der stetigen Verbesserung der Optimierungsprogramme stieg der Verbreitungsgrad praktischer Anwendungen mathematischer Modelle im Sinne einer linearen Programmierung sowohl in der Entscheidungsvorbereitung als auch in der Produktionsplanung der Mineralölindustrie. So berichteten die Autoren Baker und Lasdon[8] über die erfolgreiche Einführung der linearen Programmierung bei EXXON bereits 1985.

Fasst man die Einsatzgebiete linearer Programmierung im Bereich der Mineralölindustrie in Anlehnung an die zuvor genannten Literaturhinweise zusammen, so lassen sich daraus im Wesentlichen drei Kerngebiete ableiten:

1. Optimierung der Logistik im Zusammenhang mit der Förderung von Rohölen und der Versorgung von Raffinerien
2. Lösung von Blendingproblemen in einer Raffinerie
3. Optimierung der Raffineriefahrweise in Bezug auf Wirtschaftlichkeit

Das unter Punkt 1 beschriebene Feld der Logistik speziell im Zusammenhang mit der Förderung von Rohölen und die Optimierung des damit verbundenen Transports stellt im Rahmen der Rohölförderung und -verarbeitung eines der

[5] Diwekar, U.M., Madhavan, K.P. (1989), S. 1011-1017
[6] Magoulas, K., Marinos-Kouris, D., Lygeros, A. (1988), S. 44-48
[7] Ramsey, J.R., Truesdale, P.B. (1990), S. 40-44
[8] Baker, T.E., Lasdon, L.S. (1985)

frühesten Einsatzgebiete der Optimierung dar. Allerdings soll im Rahmen dieser Arbeit die Betrachtung logistischer Probleme nicht erfolgen. Vielmehr soll ein wesentlicher Schwerpunkt auf die Optimierung der Verarbeitung von Rohölen, Feedstocks und Zumischstoffen und deren Austausch zwischen zwei oder mehreren Standorten gelegt werden, was im Wesentlichen den Punkten 2 und 3 entspricht.

Die im Punkt 2 zusammengefasste Problematik der Optimierung des Blendings in der Raffinerie zeigt sich in der genannten Literatur als eines der typischen Einsatzgebiete linearer Programmierung. Grund dafür ist die gute Abbildbarkeit dieses Optimierungsproblems. So kann bei der Abbildung eines Blendingvorgangs der Anzahl verfügbarer Komponenten und deren Eigenschaften in einem mathematischen Modell in der Regel ausreichend Rechnung getragen werden[9]. Neben den technischen Merkmalen eines Blendingvorgangs können aufgrund der überschaubaren Komplexität des Optimierungsproblems auch wirtschaftliche und organisatorische Aspekte problemlos berücksichtigt werden.

So fließen bei den derzeit am Markt verfügbaren Optimierungsprogrammen im Bereich Blending Eigenschaften wie Produktverfügbarkeit am Markt, Preisentwicklungen sowie Lagerbestände der Blendkomponenten in die Gesamtoptimierung ein. Mit Hilfe eines derartigen LP-Modells können so die wirtschaftlichen Auswirkungen jeder einzelnen Blendingmöglichkeit ermittelt und bewertet werden. Die Zielfunktion des Modells ist dabei in den meisten Fällen darauf ausgelegt, die vom Produktionsplan vorgegebenen Mengen und Qualitäten an Fertigprodukten, z.B. Ottokraftstoffe, unter Maximierung des Gesamtdeckungsbeitrages zu produzieren.

Allerdings zeigen sich bei der Abbildung der Realität in mathematische Funktionen einer linearen Form dort Grenzen, wo Merkmale nicht linear

abzubilden sind. Beispielweise verhalten sich Kraftstoffeigenschaften wie Cloudpoint, Ansprechverhalten und Dampfdruck nicht linear. Durch die Einführung von Ersatzgrößen, so genannten Indizes, kann unter Reduzierung der Genauigkeit ein Teil der Eigenschaften in eine lineare Form überführt werden.

Neben der Lösung von Blendingproblemen gehört die im Punkt 3 erwähnte Ermittlung wirtschaftlich optimaler Produktionspläne für die Fahrweise einer Raffinerie zu den zentralen Einsatzmöglichkeiten von LP-Modellen.

Die raffineriebezogene Planungsaufgabe im Hinblick einer LP-technischen Optimierung lässt sich grob in drei Planungsbereiche einteilen und in den folgenden Fragen zusammenfassen:

1. Was und in welcher Menge und Qualität soll produziert werden?

2. Auf welche Weise - mit welchen Anlagen - soll produziert werden?

3. Mit welchen Einsatzstoffen und Einsatzmengen soll produziert werden?

Setzt man die zu produzierende Menge und deren Qualität als vom Markt vorgegebene Eingangsgröße voraus, so ist dem Modell eine möglichst breit gefächerte Anzahl an Eingangsstoffen anzubieten. Die Eingangsmaterialien können grob in Rohöle, Feedstocks und Zumischkomponenten unterschieden werden. Die Gruppe der Rohöle gibt dabei die Menge und Qualität der Öle wieder, die im Planungszeitraum verfügbar oder zu beschaffen sind. Unter der Gruppe der Feedstocks sind all die Materialien zu verstehen, die während des

[9] Magous, D., Marinos-Kouris, D. , Lygeros, A. (1988), S. 44 ff.

Produktionsprozesses einem Anlagenteil zugeführt werden können, um freie Anlagenkapazitäten auszunutzen oder Qualitätsdefizite auszugleichen. Zumischkomponenten, die beim Blending der Endprodukte Anwendung finden, können zum Erreichen der Spezifikation dem Endprodukt zugemischt werden. Aber auch Qualitätsüberhänge bei Zwischenprodukten, die aus den Anlagen der Raffinerie stammen und zum Blending des Endproduktes vorgesehen sind, können durch Hinzufügen einer qualitativ schlechteren (und damit oftmals auch preiswerteren) Zumischkomponente wirtschaftlich genutzt werden.

So wie die Rohöle können auch die Feedstocks und Zumischkomponenten am Markt zugekauft werden. Am Markt verfügbare Qualitäten, Mengen und Preise sind bei Planungsbeginn in das Modell einzuarbeiten. Auf Basis dieser Eingangsdaten ergibt sich durch die Optimierung des Modells die für die Produktion bzw. das Blending notwendigen Eingangsstoffe und der - bei Umsetzung des ermittelten Produktionsplans - zu erwartende Gesamt-deckungsbeitrag.

Neben der Möglichkeit einen hinsichtlich des Gesamtdeckungsbeitrages optimalen Produktionsplan zu ermitteln, zeigen sich bei der Abbildung einer Raffinerie in einem mathematischen Modell und dessen Optimierung mittels linearer Programmierung auch die Grenzen des Systems. Eine definitive Bestimmung des Deckungsbeitrages für jede eingesetzte Komponente ist im Rahmen der Modelloptimierung nicht unmittelbar möglich. So wird zwar der zu erwartende Gesamtdeckungsbeitrag ausgewiesen, nicht aber die Einzel-deckungsbeiträge der unterschiedlichen Einsatzstoffe. Die Einzel-deckungsbeiträge können nur eingeschränkt mit Hilfe von so genannten Sensitivitätsrechnungen ermittelt werden, wobei das so erzielte Ergebnis aufgrund des Charakters einer Kuppelproduktion nur als Näherungswert zu sehen ist. Im Rahmen der Sensitivitätsrechnungen werden die mit der betrachteten Eigenschaft in Verbindung stehenden Merkmale schrittweise

verändert. Nach jeder Merkmalsänderung wird die Veränderung des Gesamtdeckungsbeitrages betrachtet. Die Größe der relativen Deckungsbeitragsänderung ist ein Maß für die Sensitivität des Systems bei Variaion der betrachteten Eigenschaft. In den Optimierungsmodellen wird die so bestimmte relative Deckungsbeitragsänderung pro betrachteter Einheit (z.B. Tonne, m³ etc.) i.d.R. als „Marginal Value“ oder auch „Schattenpreis“ bezeichnet.

In Bezug auf mathematische Modelle als Hilfsmittel zur Produktionsplanung in rohölverarbeitenden Raffinerien lässt sich neben der Optimierung im Bereich Logistik und Blending die LP-basierte Optimierung der Gesamtfahrweise unter Maximierung des Deckungsbeitrages als zentrale Anwendung definieren. Darauf aufbauend steht in dieser Arbeit die Ermittlung des globalen Optimums im Sinne eines maximalen Gesamtdeckungsbeitrages unter Austausch eines Produktenstroms (nachfolgend als Komponente bezeichnet) im Vordergrund.

3 Untersuchung von Planungssystemen unter Berücksichtigung der Implementierung des Systems der verteilten Optimierung

Im vorangegangenen Kapitel wurden verschiedene in der rohölverarbeitenden Mineralölindustrie etablierte Planungssysteme vorgestellt. Nachfolgend sollen diese Systeme vor dem zeitlichen Planungshorizont betrachtet werden. Im weiteren werden die Grundlagen geschaffen, um die in Kapitel 5 durchgeführte Bewertung der Kriterien zur Implementierung des Systems der verteilten Optimierung zu ermöglichen.

Ferner wird neben der Einordnung bestehender Planungssysteme der Grad der Kompatibilität zwischen den betrachteten Systemen untersucht. Die Kompatibilität beim Datentransfer stellt für die verteilte Optimierung eine wesentliche Rolle dar, da es notwendig ist, Daten schnell, vollständig und ohne Formatierungsverlust zwischen den Systemen auszutauschen. Um die Neuentwicklung einer derartigen Schnittstelle zwischen den Planungssystemen im Rahmen dieser Arbeit zu vermeiden, wird eine Systemkombination gesucht, die bereits einen hohen Kompatibilitätsgrad aufweist.

Trotz der hohen Flexibilität des Systems der verteilten Optimierung reduziert eine hohe Kompatibilität der unterschiedlichen Planungssysteme das Auftreten von Problemen beim Datenaustausch. Das bedeutet, dass nicht nur die Daten allein, sondern auch ihre Struktur einen reibungslosen Austausch begünstigen. Somit erhöht eine Gleichheit der Planungssysteme, d.h. ein hoher Verbreitungsgrad, die Akzeptanz des Systems der verteilten Optimierung. Aus diesen Gründen wird schließlich der Verbreitungsgrad der betrachteten Planungssysteme untersucht.

3.1 Darstellung gängiger Planungssysteme in rohölverarbeitenden Raffinerien vor dem Hintergrund des Planungszeitraums

Im Folgenden wird der gegenwärtige technische Entwicklungsstand von mathematischen Modellen in der Optimierung hinsichtlich des zeitlichen Planungshorizontes und des Umfangs des zu optimierenden Bereiches dargestellt. In diesem Zusammenhang wird im Wesentlichen die Definition von autarken und integrierten Modellen erarbeitet. Die mit den Modellformen einhergehenden Optimierungsalgorithmen inklusive deren Anwendungsgebiete werden aus Gründen der Übersichtlichkeit in Kapitel 4 behandelt.

Die in einem Modell zusammengefassten Gleichungen sind ein abstrahierendes Abbild eines definierten und damit begrenzten Teils der Realität. In der Praxis kann das ein Planungssystem über die Produktion einer Raffinerie sein. Dieses Planungssystem kann mit anderen computergestützten Planungsformen in Verbindung stehen. Die oftmals mit definierten Schnittstellen verbundenen einzelnen Planungssysteme zeichnen sich in erster Linie durch den unterschiedlichen zeitlichen Planungshorizont und der damit verbundenen technischen Form der Optimierung aus (siehe dazu Abbildung 1).

Den größten Planungszeitraum umfasst in der Regel die sogenannte operative Planung mit bis zu 10 Jahren. Nach Hungenberg[10] ist die Aufgabe der operativen Planung innerhalb des von der strategischen Planung vorgegebenen Handlungsrahmens konkrete Ziele und Maßnahmen in einzelnen Funktionsbereichen zu bestimmen. Schäffer[11] definiert analog die operative Planung als stark formalisierten Planungsprozess im Bereich der strategischen Planung. Eine Optimierung mit Hilfe eines LP-Systems ist zwar theoretisch denkbar, allerdings in der Regel nicht sinnvoll, da aufgrund der Mittelung der

[10] Hungenberg, H. (2001), S. 44 ff
[11] Schäffer, U., Hoffmann, D. (1999), S. 368

Jahresereignisse Opportunitäten entstehen, die in der Praxis nicht bestehen. Man spricht in diesem Fall von einer Überoptimierung.

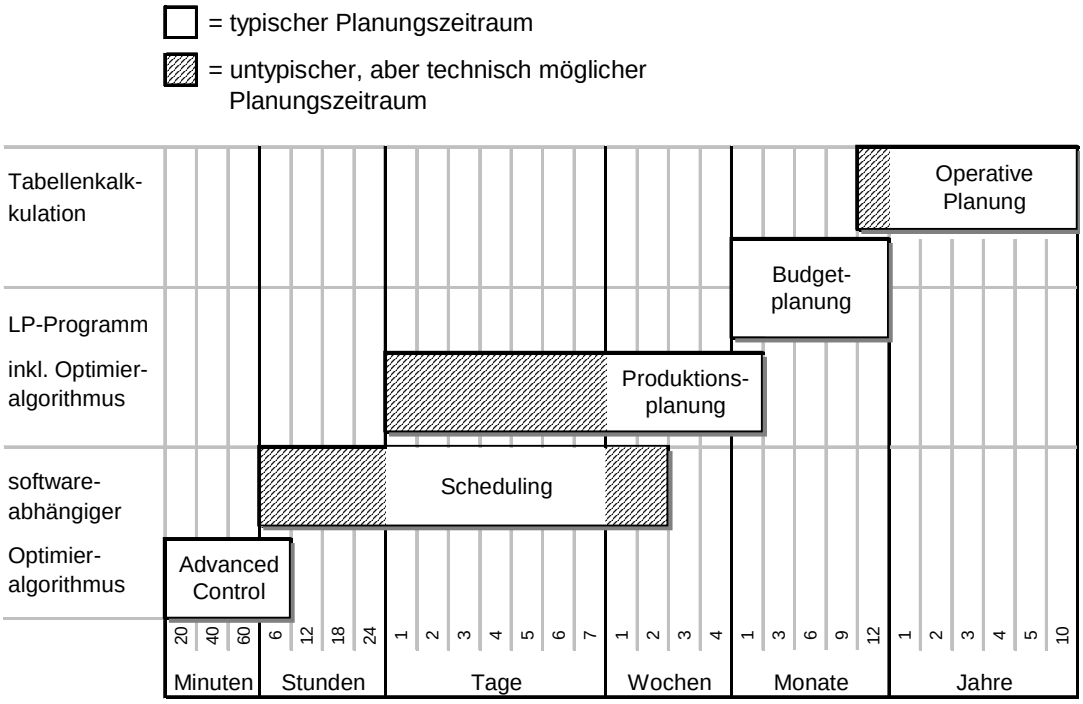


Abbildung 1 – Einordnung von Planungssystemen in Bezug auf Planungszeitraum und Optimierungstechnik

So kann im Rahmen eines Jahres-LP ein Überschuss am Jahresende mengenbilanzseitig an den Jahresanfang transferiert werden, was in der Realität nicht möglich ist. Die zeitlich enger getaktete Planungsform kann als Budgetplanung bezeichnet werden. Diese Planungsform umfasst in den meisten Unternehmen die zwölf Monate des kommenden Kalender- oder Buchungsjahres. Je nach Auslegung des LP-Modells kann diese Planungsform

zum Teil mit monatsgenauen LP-Rechnungen unterstützt und mit Tabellenkalkulationen weiterverarbeitet werden.

Die nächste Stufe der zeitlichen Verfeinerung stellt die Monatsplanung dar. Sie umfasst in erster Linie einen Monat der Produktion und kann auf mehrere Monate ausgeweitet werden. Eine untermonatliche Optimierung der Produktion mittels LP-Modell ist mit Hilfe einer Mehr-Perioden-Rechnung möglich. Diese kann theoretisch einen Tag als kleinste Planungseinheit umfassen, allerdings nimmt der zeitliche Aufwand zur Pflege der Modelldaten und die Auswertung der sehr detaillierten Lösungen stark zu. Eine LP-unterstützte Monatsplanung umfasst somit in den meisten Fällen den Zeitraum eines Monats.

Die aus der Monatsplanung resultierenden Plandaten können mit einem Scheduling-Modul auf die tagesaktuellen Problemstellungen einer Produktion heruntergebrochen werden. Zwar sind die errechneten Ergebnisse minutengenau, allerdings ist in den meisten Schedulingsystemen das Planungsraster auf Stundenbasis ausgelegt. Der maximale Schedulingzeitraum hängt stark von dem Charakter der Produktion ab, ist aber zeitlich kürzer als die dazugehörige Produktionsplanung. Die Grundlage des Schedulings stellt in den meisten Fällen eine Mischung aus Simulation und softwarespezifischen Optimierungsalgorithmen dar.

Das bedeutet, dass im ersten Schritt ein Simulationsalgorithmus versucht, unter Berücksichtigung der aus dem Produktionsplan und dem Tagesschäft anfallenden Restriktionen eine zulässige Lösung zu finden. Da die vom Simulationsteil ermittelte Lösung keinen Anspruch auf eine im LP-Sinne optimale Lösung hat, kommt in einigen Scheduling-Anwendungen ein Optimieralgorithmus zur Anwendung.

Die prozessnahe Umsetzung der Plandaten in die Produktion übernimmt schließlich ein Prozessleitsystem (PLS)[12]. Bei dieser Anwendung steht die Regelung und Einhaltung definierter Parameter im Vordergrund. Eine Optimierung findet nur in sehr geringem Umfang statt, so dass das PLS als Werkzeug zur Optimierung der Produktion zwar genannt, aber nicht als Planungssystem im klassischen Sinne dargestellt werden soll.

Die vorgestellten Planungsformen sollen nur einen Überblick über die am Markt erhältlichen und in der Praxis eingesetzten Softwarelösungen zur Unterstützung der Produktionsplanung in einem kontinuierlichen Produktionsprozess geben. Neben den vorgestellten Planungssystemen finden eine Reihe von Lösungen in der Praxis Anwendung, die hier aufgrund des geringen Verbreitungsgrades nicht betrachtet werden sollen. Lediglich auf folgende sich abzeichnende Neuentwicklung ist hinzuweisen: die Simulation in Verbindung mit einem LP-Modell zur Produktionsplanung.

Die Simulation von technischen Prozessen ist ein seit Jahren erfolgreich eingesetzter und kontinuierlich weiterentwickelter Ansatz. Als Unterstützung zur Tagesplanung einer Produktion allerdings fehlten in der Vergangenheit zwei wesentliche Dinge: Zum einen konnten die zum Betreiben einer Simulation notwendigen Daten nicht im erforderlichen Maße erfasst und datenbanktechnisch verarbeitet werden und zum anderen fehlte eine leistungsstarke Optimierung.

Das Problem der Datenerfassung und -verwaltung konnte im Zuge der Ausweitung der Online-Analytik und der Verbesserung der Datenbankenanwendungen in den meisten Fällen gelöst werden. Neuere Entwicklungen im Bereich der Simulationssoftware erfassen und simulieren nicht nur die im Produktionsprozess anfallenden Daten, sondern wandeln diese

[12] auch Digital-Control-System (DCS) oder Advanced-Control-System (ACS)

mit Hilfe eines so genannten LP Data Generators[13] in lineare Funktionen inklusive der entsprechenden Prozessparameter um. Diese können vom LP-Modell der Produktionsplanung gelesen und optimiert werden. Das optimierte Ergebnis kann schließlich wieder an die Simulation zurückgegeben und auf Plausibilität geprüft werden. Da in einer Simulation in der Regel erheblich zeitnähere Produktionsparameter als ein LP-Modell verarbeitet werden, kann durch die Verknüpfung von Simulation und LP-Optimierung eine sehr praxisnahe Optimierung auf Tagesbasis erfolgen. So ist eine tagesgetaktete Planung lediglich mittels LP-Modell ohne Einbindung einer Simulation zwar theoretisch möglich, aber aufgrund des hohen Rechenaufwandes nicht stark verbreitet und nur in Einzelfällen sinnvoll.

Diese beschriebene Neuentwicklung der Verknüpfung von Simulation und LP-Optimierung befindet sich noch im Aufbau. Zum Zeitpunkt der Entstehung dieser Arbeit waren noch keine Veröffentlichungen über Erfahrungen mit den beschriebenen Kombinationen verfügbar.

3.2 Beschreibung der Schnittstellen sowie des Grades der Kompatibilität zwischen Planungssystemen innerhalb und außerhalb eines Planungssegmentes

Die in Kapitel 3.1 aufgezeigten Planungssysteme lösen Planungsprobleme für unterschiedliche Zeiträume in rohölverarbeitenden Raffinerien. Wie in Abbildung 1 dargestellt, wird der gesamte Planungszeitraum von bis zu zehn Jahren durch verschiedene Planungswerkzeuge mit unterschiedlicher Auflösung der Zeit abgedeckt. Eine Verbindung zwischen den beschriebenen Planungswerkzeugen besteht in der Weise, dass nach erfolgter Optimierung des größeren Zeitfensters die errechneten Ergebnisse gleichzeitig die Vorgabe für

[13] Hyprotech Ltd. (siehe www.hyprotech.com) und KBC Advanced Technologies (siehe www.kbcat.com) haben 2000 in Zusammenarbeit mit Fuji Oil z.B. die beschriebene

die Optimierung der Planungsprobleme im nächst kleineren Zeitfenster darstellen. Die Form und der Grad der Kompatibilität beim Datenaustausch sind abhängig vom Systemlieferant und stellen, wie im weiteren Verlauf noch detailliert erörtert wird, eine wesentliche Grundvoraussetzung für eine Implementierung des Systems der verteilten Optimierung dar. Im Folgenden sollen die Schnittstellen zwischen den genannten Planungssystemen hinsichtlich des Kompatibilitätsgrades untersucht und eine präparierte Systemkombination ermittelt werden.

Als Kompatibilität von Schnittstellen soll in diesem Zusammenhang das Maß der Formatierung verstanden werden, welches notwendig ist, um Datensätze ohne Verlust von Inhalt und Format zwischen Planungssystemen und –segmenten transferieren zu können.

Nachfolgend soll überprüft werden, wie hoch der Kompatibilitätsgrad zwischen den vorgestellten Planungssystemen ist, um hoch kompatible Schnittstellen zur späteren Implementierung des Systems der verteilten Optimierung identifizieren zu können.

Beginnend mit der Operativen Planung werden die Ergebnisse des ersten Gesamtjahres an die Budgetplanung weitergegeben. Diese Ergebnisse stellen die Vorgabe für die Budgetplanung dar. Eine Kompatibilität ist an dieser Stelle weder sinnvoll noch verbreitet. Gleiches gilt für die Schnittstelle von der Budgetplanung zur Produktionsplanung. Auch an dieser Schnittstelle ist ein automatischer Datentransfer in der Praxis nicht etabliert. Grund für die mangelnde Notwendigkeit einer Kompatibilität liegt in erster Linie darin begründet, dass die Budgetplanung – ähnlich wie die Operative Planung – in der Regel nur einmal jährlich durchgeführt wird, um so die Grundlage für die wirtschaftliche Beurteilung des Planungszeitraums zwischen einem Monat und einem Jahr zu schaffen. Darüber hinaus stellen die Ergebnisse der

Verbindung von Simulation HYSIS.Refinery und LP-Modell entwickelt.

Budgetplanung auch die Basis für die Zielwerte dieser Planungsperiode dar. Erwartete Betriebsergebnisse, Etatgrenzen, Ausfallzeiten usw. werden so als Zielgröße festgelegt. Die Produktionsplanung hingegen verfolgt die Umsetzung und Einhaltung der in der Budgetplanung vorgegebenen Zielgrößen. Das Ergebnis der Produktionsplanung ist in der Regel näher an der Praxis orientiert und detaillierter als das der Budgetplanung.

Der im Scheduling auf einen kleineren Zeitraum als in der Produktionsplanung heruntergebrochene Produktionsplan wird aufgrund des hohen Datenvolumens in den meisten Anwendungen automatisiert erstellt. Das bedeutet, dass der mittels eines LP-Modells errechnete Produktionsplan in einer Datenform erzeugt wird, die das nachgeschaltete Schedulingsystem einlesen und verarbeiten kann. Die Form des Datentransfers kann ein Liste von Daten, eine Datei einer gängigen Tabellenkalkulation oder aber auch eine Datenbank sein. Diese in den meisten Anwendungen automatisierte Schnittstelle ist oftmals bidirektional ausgelegt. Das Scheduling simuliert in erster Instanz lediglich die aus dem Produktionsplan eingelesenen Vorgaben. Kommt es im Scheduling nicht zu einer Lösung, in der alle Vorgabeparameter eingehalten wurden, so kann entschieden werden, ob eine Gewichtung oder Vernachlässigung der Vorgaben aus dem Produktionsplan vorgenommen werden kann oder ob eine Information an die Produktionsplanung zurückgehen muss, um einen neuen Produktionsplan mit geänderten Randbedingungen zu erstellen. Im Gegensatz zu LP-Systemen sind die meisten Schedulingsysteme so ausgelegt, dass diese *immer* zu einer Lösung kommen und sie die nicht eingehaltenen Parameter ausweisen. Das ist für das Tagesgeschäft des Scheduling-Anwenders sinnvoll, da dieser so selbständig entscheiden kann, wie wichtig diese Restriktion für das aktuelle Problem ist.

Das aus dem Scheduling resultierende Planungsergebnis könnte ohne größeren technischen Aufwand an das PLS weitergegeben werden, allerdings hat sich hier in der Praxis gezeigt, dass das Volumen der Vorgabedaten in einem PLS

zu umfangreich und zu detailliert ist, um eine Kompatibilität des Datentransfers zu ermöglichen. Im Folgenden soll das PLS nicht weiter betrachtet werden, da es als Steuerungs- und Regelungssystem und weniger als Planungssystem zu sehen ist und hier lediglich zur vollständigen Darstellung der Schnittstellen dient.

Bei der Betrachtung der Schnittstellen zwischen den unterschiedlichen Planungssystemen wurden die Schnittstellen *innerhalb* eines Planungssegmentes dargestellt (siehe dazu Abbildung 2). Als Planungssegment ist in diesem Zusammenhang ein Bereich zu verstehen, der mit je einem Planungssystem je Planungsform ausgestattet ist. Das bedeutet, jedes Planungssystem kommt pro Planungssegment maximal einmal vor. Ein solches Planungssegment kann zum Beispiel einen eigenständig planenden Unternehmenszweig darstellen. Wichtige, bisher noch nicht betrachtete Schnittstellen können sich jedoch auch durch das Verbinden zweier Planungssegmente ergeben.

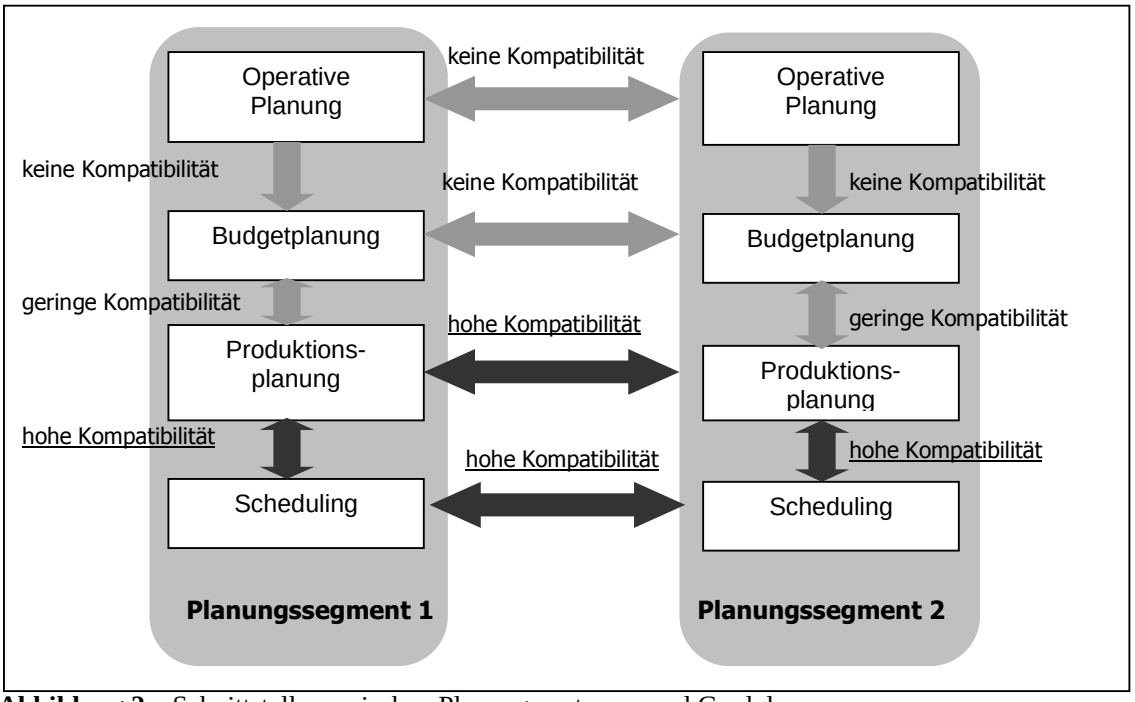


Abbildung 2 – Schnittstellen zwischen Planungssystemen und Grad der Kompatibilität

Innerhalb eines Planungssegmentes wird unter Berücksichtigung der für das jeweilige Planungsszenario geltenden Randbedingungen ein der Zielfunktion entsprechend optimaler Plan erstellt. Die zu berücksichtigenden Randbedingungen resultieren im Wesentlichen aus drei sich zum Teil überschneidenden Bereichen. Zum einen sind die Vorgaben des zeitlich vorgelagerten Planungssystems einzuhalten, zum anderen müssen die für den jeweiligen Zeitraum gültigen Randbedingungen (meist aus der Praxis vorgegeben) berücksichtigt werden. Darüber hinaus können aber auch Vorgaben aus angrenzenden Planungssegmenten resultieren. Die zuletzt genannte Schnittstelle hat für die auf mehrere Planungssegmente verteilte Optimierung wesentlichen Einfluss.

Organisatorisch können zwei oder mehr Planungssegmente auf allen zeitlichen Ebenen der Planung über automatisierte Schnittstellen verbunden sein. Ein derartig vernetztes Planen ist z.B. bei der Abstimmung selbständig planender Produktionsstandorte möglich. Sinnvoll wird ein vernetztes Planen dann, wenn zwischen den Planungssegmenten Wechselwirkungen entstehen und Synergien genutzt werden. So können Planungsergebnisse der unterschiedlichen Planungssegmente abgestimmt werden. Das kann z.B. bedeuten, dass Vorgaben hinsichtlich ausgetauschter Zwischenprodukte wie Menge, Qualität, Preis etc. von einem Planungssegment in ein anderes übermittelt werden.

Wie eingangs beschrieben ist eine hohe Kompatibilität zwischen Planungssystemen die Grundvoraussetzung für eine mögliche Implementierung des im Rahmen dieser Arbeit entwickelten Algorithmus. Basierend auf den betrachteten Schnittstellen zwischen Planungssegmenten und –systemen zeigten sich lediglich die Schnittestellen zwischen Produktionsplanung und Scheduling, sowie zwischen den Segmenten der Produktionsplanung bzw. Systemen des Schedulings als hoch kompatibel (siehe Abbildung 2) und somit geeignet für die Implementierung des Ansatzes der verteilten Optimierung.

4 Untersuchung bestehender Optimierungsalgorithmen und Klassifizierung in Bezug auf die Implementierung des Systems der verteilten Optimierung

Im Rahmen der mathematischen Optimierung im Sinne der Optimierungstheorie[14], werden in der Literatur eine Fülle von Optimierungsformen und deren Weiterentwicklungen vorgestellt. Nachfolgend findet eine Betrachtung geschlossener und nicht geschlossener Lösungen im Sinne der direkten und indirekten Optimierungsverfahren statt. Die Ergebnisse der Klassifizierung dienen dem Ziel, ein Verfahren zu ermitteln, das für die Implementierung des Systems der verteilten Optimierung geeignet ist. Sie bilden entsprechend die Grundlage für die nachfolgenden Ausführungen.

4.1 Klassifizierung verbreiteter Optimierungsverfahren in der Literatur

Ein wesentliches Merkmal der dargestellten Planungssysteme ist, dass alle Systeme in sich geschlossen und autark optimieren. Das bedeutet, dass aus den angrenzenden Schnittstellen von Planungssystemen und -segmenten zwar die für die Optimierung geltende Rand- und Nebenbedingungen resultieren, der Optimieralgorithmus selbst jedoch selbständig und ohne Kommunikation zu anderen Systemen während der Optimierung durchgeführt wird. So kommt es im Vorfeld einer LP-Rechnung im Rahmen der Produktionsplanung zu einem Austausch der beschriebenen Randbedingungen; auch die Planungsergebnisse bilden wieder eine Grundlage für einen Datenaustausch, allerdings werden während des Optimiervorganges keine Daten ausgetauscht. Der Optimieralgorithmus der Planungssysteme arbeitet autark, so dass im Folgenden die gegenwärtig bestehenden Planungssysteme als *autark*

[14] Göpfert, A (1986), S168 ff.

optimierende Planungssysteme verstanden werden. Das so ermittelte Optimum stellt jedoch trotz der berücksichtigten Vorgaben und Randbedingungen angrenzender Planungssysteme lediglich das Optimum des jeweiligen Planungssystems dar.

Nachdem in den vorangegangenen Kapiteln die Auswahl eines geeigneten Planungssystems inklusive der damit verbundenen Schnittstellen erfolgt ist, sollen nun die Lösungsalgorithmen autark optimierender Planungssysteme dahingehend betrachtet werden, ob es möglich ist, den Algorithmus der verteilten Optimierung zu implementieren.

Autark optimierende Modelle lassen sich im Hinblick auf den angewandten Lösungsalgorithmus in „geschlossene“ und „nicht geschlossene“ Lösungen unterscheiden (Abbildung 3).

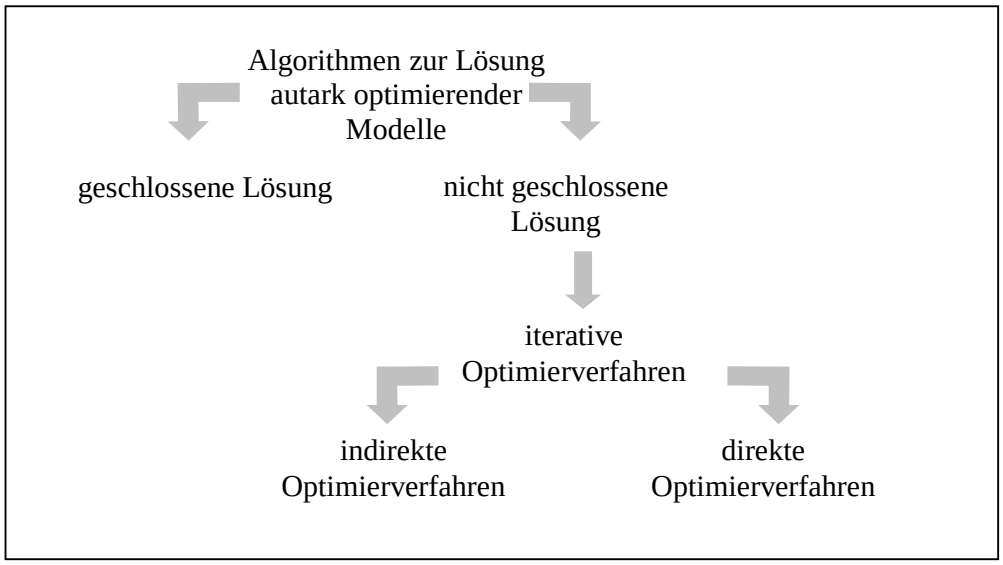


Abbildung 3 – Klassifizierung verbreiteter Optimierverfahren in der Literatur

Dabei ist dann von einer geschlossenen Lösung die Rede, wenn die Lösungsmenge mit Hilfe von Elementarfunktionen ermittelt werden kann[15]. Das damit verbundene Normalgleichungssystem ist somit algebraisch zu lösen, so dass eine geschlossene Formel entsteht. Der Anwendungsbereich dieser Optimierungsform wird durch folgende Nachteile stark eingeschränkt: Es muss das Normalgleichungssystem in einem System mit polynominalen Gleichungen vorliegen oder zumindest in eine solche Form gebracht werden können, um geschlossen lösbar zu sein. So weist Stubenvoll[18] anhand der Lösung einer dreidimensionalen Helmertransformation nach, dass sich das Problem in eine 4 x 4 Matrix formulieren und lösen lässt. Eine gravierende Einschränkung, die durch die Forderung der geschlossenen Lösbarkeit entsteht, ist die begrenzte Möglichkeit, mathematische Zusammenhänge durch Variablen eines Polynoms auszudrücken. So löst Stubenvoll mit sehr hohem Aufwand maximal eine polynominale Gleichung 4. Grades. Zur Abbildung von produktions-technischen Zusammenhängen zum Zwecke der Produktionsplanung reichen die aufgezeigten Möglichkeiten nicht aus. Auch die Verfahren zur Vereinfachung komplizierter polynominaler Gleichungssysteme, wie z.B. diejenigen von Buchberger[16], die so genannte Gröbner-Basen zur Polynomreduktion, erweitern die Lösungsmethodik der geschlossenen Lösungen nicht hinreichend, so dass er in der Produktionsplanung und damit in dieser Arbeit auch keine Berücksichtigung findet. Das bedeutet ebenfalls, dass aufgrund der sehr begrenzten Anzahl an Variablen der noch darzustellende Algorithmus der verteilten Optimierung mit seiner Vielzahl zusätzlicher Gleichungen bei dem Ansatz geschlossener Lösungen technisch nicht einsetzbar ist.

Der Bereich der nicht geschlossenen Lösungen, d.h. die modellmäßige Abbildung in nicht geschlossene Formeln, ist weitestgehend der iterativen Optimierung vorbehalten. Dabei ist unter der iterativen Optimierung die

[15] Stubenvoll, R. (1995)
[16] Buchenberger, B. (1965)

schrittweise Maximierung/Minimierung der Zielfunktion unter Zuhilfenahme eines Optimieralgorithmus zu verstehen. Als iterative Verfahren lassen im sich im Wesentlichen die indirekten Verfahren im Sinne der Gradientenverfahren und die direkten Verfahren nennen.

Da Gradientenverfahren in der Produktionsplanung der rohölverarbeitenden Industrie theoretisch eine Rolle spielen können, sollen an dieser Stelle die drei etabliertesten Gradientenmethoden dargestellt werden. Ferner soll geprüft werden, ob Modelle, die mit Hilfe von Gradientenverfahren lösbar sind, im Sinne dieser Arbeit verknüpft und in der verteilten Optimierung sinnvoll eingesetzt werden können.

4.2 Indirekte Optimierungsverfahren in der verteilten Optimierung am Beispiel der Gradientenverfahren

Lokale Optimierverfahren erzeugen mit einem Startwert u_0 beginnend eine Folge $u_1, u_2, u_3, ..., u_k$. Diese Folge konvergiert gegen ein Minimum/Maximum u von $f(u)$. Geht man von einem angestrebten Minimum aus, spricht man auch von einem Abstiegsverfahren, da im Idealfall jeder Punkt der Folge einen kleineren Funktionswert aufweist.

Nach Teunissen[17] lassen sich Iterationsverfahren durch folgendes Schema klassifizieren:

$$u_{k+1} = u_k + t_k d_k, \text{ mit } k = 0, 1, 2, ... \qquad (1)$$

Dabei gilt:
1. u_0: Punkt, der dem vorhandenen Startwert entspricht
2. d_k: Richtung des Folgepunktes
3. t_k: Festlegung der positiven Schrittweite;
   unter der Bedingung $f(u_{k+1}) < f(u_k)$

[17] Teunissen, P. (1990), S. 138

In der Vergangenheit sind eine Reihe von Gradientenverfahren entwickelt worden, die sich im Wesentlichen nur in der Wahl des Richtungsvektors d_k und des positiven Skalars t_k unterscheiden.

Grundsätzlich lassen sich in der Literatur unterschiedliche Algorithmen finden. Das Grundprinzip des Iterationsverfahren bleibt jedoch gleich, es besteht darin, das funktionale Modell durch Funktionsableitungen in ein einfaches Modell zu überführen. Die drei in der Produktionsplanung etabliertesten Verfahren sollen nachfolgend kurz vorgestellt und hinsichtlich der Möglichkeit zur Implementierung der verteilten Optimierung analysiert werden. Der vollständige Umfang an Gradientenmethoden kann und soll hier nicht wiedergegeben werden.

4.2.1 Klassisches Gradientenverfahren bzw. die Methode des steilsten Abstiegs

Von einem klassischen Gradientenverfahren oder von der Methode des steilsten Abstiegs ist dann die Rede, wenn die Matrix als Einheitsmatrix gewählt wird. Die Iterationsformel lautet:

$$u_{k+1} = u_k - t_k \,.\, \Delta f(u_k) \qquad (2)$$

Wesentlicher Einfluss auf das Ergebnis der Gradientenverfahren hat die Wahl der Schrittweite t_k . So werden sehr viele Iterationen benötigt, wenn die Schrittweite zu klein gewählt wird. Bei einer zu großen Schrittweite hingegen ist die Wahrscheinlichkeit sehr hoch, dass das Verfahren nicht konvergiert. Aus diesem Grund findet die Schrittweitenminimierungsregel Anwendung: Danach wird auf der Halbgraden in Richtung des steilsten Abstiegs nach dem gewünschten Funktionsminimum/-maximum gesucht. Ein Algorithmus

ermittelt dann die optimale Schrittweite für den Nachfolger. Durch die Orthogonalität aufeinander folgender Suchrichtungen kann allerdings ein „Zickzack-Kurs“ entstehen, so dass bei höherdimensionalen Problemen sehr viele Schritte erforderlich sind. Damit sind die klassischen Gradientenverfahren für die Abbildung von Praxisproblemen nur eingeschränkt einsetzbar. Selbst bei einem einfachen Modell kommt es durch die Verknüpfung dieser Modelle untereinander in der Regel zu höherdimensionalen Problemen, so dass der Einsatz klassischer Gradientenverfahren im Zusammenhang mit der verteilten Optimierung ausgeschlossen werden kann.

4.2.2 Newton-Raphson-Verfahren

Der auch als Newton-Verfahren bezeichnete Algorithmus war zunächst für den Einsatz der Nullstellensuche einer Funktion entwickelt worden. Durch Lösen des Normalgleichungssystems $\Delta f(u)=0$, d.h. durch Bestimmung der Nullstellen der ersten Ableitung, lassen sich mit Hilfe des Newton-Raphson-Verfahrens die Minima/Maxima einer Funktion bestimmen und somit Optimierprobleme lösen.

Ein nichtlineares Normalgleichungssystem ist in den meisten Fällen nicht geschlossen lösbar. In diesem Falle ist es gängige Praxis, die Normalgleichung durch eine Taylorreihe 1. Ordnung zu approximieren, d.h. zu linearisieren. Auf dieses Verfahren soll jedoch nicht weiter eingegangen, sondern auf Björk und Germund[18] verwiesen werden. Inwieweit dieses lineare Normalgleichungs-system das ursprüngliche Optimierproblem ersetzen kann, bleibt in der Literatur weitgehend unklar.

[18] Björk A.; Germund, D, 1972, S. 164 ff.

Im Gegensatz zum beschriebenen klassischen Gradientenverfahren wird beim Newton-Rapsohn-Verfahren die Zielfunktion bei jedem Iterationsschritt quadratisch approximiert. Damit konvergiert dieses Verfahren schnell. Bei quadratischen Funktionen ist das sogar in einem Schritt der Fall.

Als Nachteil des Newton-Raphson-Verfahren ist zu nennen, dass das Aufstellen der zweiten Ableitungen je nach Struktur des funktionalen Zusammenhangs sehr kompliziert sein kann und unter Umständen zum Abbruch führt. Aus diesem Grunde erweiterte Levenberg[19] bereits 1944 dieses Verfahren. Ohne an dieser Stelle auf den Inhalt der Weiterentwicklung einzugehen, bleibt festzuhalten, dass das erweiterte Newton-Raphson-Verfahren – auch Trust-Region-Method[20] genannt – ein Kompromiss zwischen der klassischen Gradientenmethode und dem ursprünglichen Newton-Raphson-Verfahren darstellt.

Dieses Verfahren, inklusive seiner Erweiterung durch Levenberg, ist in der Lage, auch höherdimensionale Probleme zu lösen und das gewünschte Maximum bzw. Minimum zu finden. Allerdings nimmt die Rechenzeit bei komplizierteren Funktionen stark zu, der Einsatz in der EDV-gestützten Produktionsplanung und in der verteilten Optimierung ist somit zwar technisch möglich, aber aus Gründen der zu erwartenden langen Rechenzeit nur bedingt sinnvoll.

4.2.3 Gauß-Newton-Verfahren

Einleitend lässt sich sagen, dass im Gegensatz zu dem Newton-Raphson-Verfahren hier die Linearisierung nicht außerhalb, sondern innerhalb einer Norm, d.h. auf der Ebene Normalgleichungen und Zielfunktionen,

[19] Levenberg, K. 1944
[20] Plantenga, T.D., 1999, S. 282 ff.

vorgenommen wird. Die Funktionen der einzelnen Residuen werden an einem Taylorpunkt entwickelt und dann erst einer Norm unterworfen.

Das Gauß-Newton-Verfahren konvergiert schnell, wenn die folgenden Bedingungen erfüllt sind:

- die Zielfunktion sollte nur leicht gekrümmt sein
- die Residuen sollten klein sein
- und die Näherungswerte müssen in der Nähe des Minimums liegen.

Das Gauß-Newton-Verfahren findet in der computergestützten Produktionsplanung nur in Individualentwicklungen Anwendung, nicht aber in den verbreiteten Standardsoftware-Lösungen. Aufgrund der genannten Bedingungen ist das Gauß-Newton-Verfahren nicht in der Lage, die Zielfunktion der in der Regel komplexeren Modellierungen eines Produktionsbetriebes zu optimieren. Die durch eine verteilte Optimierung verstärkt auftretende Komplexität kann somit nur schwer mit dem Gauß-Newton-Verfahren bewältigt werden.

4.3 Direkte Optimierungsverfahren in der lokalen und verteilten Optimierung

Die in Anlehnung an Abbildung 3 nachfolgend betrachtete Gruppe der Verfahren sind die direkten Optimierungsverfahren. Diese beinhalten ausschließlich die Funktionswertberechnung der Zielfunktion. Hilfsfunktionen wie Steigung, Krümmung etc. sind nicht notwendig. Charakteristisch für direkte Optimierungsverfahren ist ein Entscheidungsbaum, der während des Lösungsprozesses in Teillösungen zergliedert wird.

Direkte Optimierverfahren können für eine große Anzahl von Praxisproblemen eingesetzt werden, da im Gegensatz zu den Gradientenmethoden in begrenztem Maße auch Zielfunktionen untersucht werden, die mehrere lokale Optima besitzen. Welches der gefundenen Optima jedoch das globale darstellt, kann auch mit den direkten Optimierungsverfahren nur bedingt gelöst werden. Bezüglich der Erweiterung dieser Verfahren zur Ermittlung des globalen Optimums wird hier auf Kapitel 9 verwiesen.

Als Vorteil der direkten gegenüber den indirekten Optimierungsverfahren wird in der Literatur die Lösbarkeit von komplizierten Zielfunktionen hervorgehoben[21]. Die mathematischen Modelle der Produktionsplanung werden aufgrund der komplexen Modellstruktur mit Hilfe direkter Verfahren gelöst. Die direkten Verfahren bilden somit eine geeignete Grundlage bei der Implementierung der verteilten Optimierung. Nachfolgend soll beispielhaft auf zwei Verfahren der direkten Optimierung unter der Maßgabe einer möglichen Verknüpfung des betrachteten Algorithmus mit dem System der verteilten Optimierung eingegangen werden.

4.3.1 Verfahren der Intervallschachtelung

Nach Kosmol[22] wird eine Strecke AB, die das Minimum/Maximum einschließt, durch einen neuen Punkt C im Verhältnis des Goldenen Schnittes geteilt. Den Goldenen Schnitt formuliert KOSMOL so, dass sich eine längere Teilstrecke AC_2 zu einer kürzeren Teilstrecke AC_1 so verhält wie die Teilstrecke AB zur Teilstrecke AC_2. Dieses Verhältnis lässt sich durch die Punkte C_1 und C_2 einhalten. Anschließend werden die Funktionswerte $f(C_1)$ und $f(C_2)$ verglichen. Der Punkt C_i mit dem kleineren/größeren Funktionswert

[21] Press, W.H., 1992, Kap. 10
[22] Kosmol, P., 1989, S. 64 ff.

wird neuer Endpunkt. Der zweite Endpunkt wird durch das Teilen der verbleibenden Strecke in folgendem Verhältnis ermittelt:

Verhältnis: $2/(\sqrt{5}-1)$ bzw. $2/(3-\sqrt{5})$ **(3)**

Jedes andere Verhältnis ist theoretisch möglich. Diese Vorgehensweise wird solange wiederholt, bis das gewünschte Minimum/Maximum in der gewünschten Genauigkeit erreicht ist.

Das genannte Teilungsverhältnis kann bei einfachen Funktionsverläufen schnell zu einer Lösung führen. Für den Einsatz bei komplizierten Funktionsverläufen ist das Verfahren der Intervallschachtelung nicht geeignet. Der Einsatz dieses Verfahrens im Zusammenhang mit der verteilten Optimierung wird aufgrund der Einfachheit diese Verfahren als nicht sinnvoll angesehen.

4.3.2 Simplex-Verfahren

Eiselt/Pederzoli/Sanblom[23] beschreiben das Simplex-Verfahren als essentielles algebraisches Verfahren zur Lösung jeglicher linearer Probleme durch progressive Annäherung an eine optimale Lösung im Rahmen eines definierten iterativen Prozesses, bis das Optimum in festgelegter Genauigkeit ermittelt worden ist.

Ohne zu detailliert auf das Downhill-Simplex-Verfahren an dieser Stelle einzugehen (näheres zum Verfahren bei Press[24], Wolfe[25] und Göpfert[26]), sollen

[23] Eiselt/Pederzoli/Sanblom, 1987, S. 29 ff.
[24] Press, W.H., 1992, Kap. 10
[25] Wolfe, P., 1965
[26] Göpfert, A. , 1986, S. 204 ff.

hier kurz die wesentlichen Merkmale beschrieben werden, die im Zusammenhang mit der verteilten Optimierung stehen.

Als Simplex wird zunächst ein geometrischer Körper mit *n+1* Punkten im *n*-dimensionalen Raum bezeichnet. Entsprechend der Anzahl der Dimensionen sind für alle *m* Unbekannten jeweils *m+1* Näherungswerte zu definieren, um den Algorithmus zu initialisieren. Ziel des Downhill-Simplex-Verfahrens ist es, für eine konvexe Funktion - das Simplex - so klein werden zu lassen, dass es auf eine durch die Toleranzgrenze ε definierte Minimalgröße zusammengeschrumpft werden kann. Der Schwerpunkt des verbleibenden Simplexes stellt das gesuchte Minimum dar. Das Schrumpfen des Ausgangssimplexes erfolgt durch vier Regeln, die in folgender Reihenfolge anzuwenden sind:

- Regel 1:
  Man sucht den Punkt des Simplexes mit dem ungünstigsten Funktionswert. Dieser Punkt wird an dem Schwerpunkt der übrigen Punkte gespiegelt. Wenn der neue gespiegelte Punkt günstiger ist als alle anderen Punkte des Simplexes, so bleibt zu prüfen, ob eine Spiegelung mit gleichzeitiger Streckung eine Verbesserung des Funktionswertes ergibt.

- Regel 2:
  Ist der Funktionswert des gespiegelten und gestreckten Punktes ungünstiger als der ausschließlich gespiegelte, so wird nur der gespiegelte Punkt als Ersatz des zunächst ungünstigen Funktionswertes genommen.

- Regel 3:
  Bringt auch Regel 2 keine Verbesserung, so kann entlang der in Regel 1 und 2 beschriebenen Geraden eine Stauchung des Simplexes erfolgen.

- Regel 4:
  Haben alle drei Regeln keine Verbesserung des Ausgangssimplexes gebracht, wird der Simplex mit Ausnahme des Punktes mit dem kleinsten Funktionswert entlang aller Dimensionen gestaucht.

Eine sehr gute grafische Darstellung unter Umsetzung der vier aufgezeigten Regeln zeigen Eiselt / Pederzoli / Sanblom[27], wobei eine grafische Lösung nicht mehr als drei Variablen zulässt.

Wie in Kapitel 7.2.1 näher beschrieben wird, kann der Algorithmus der verteilten Optimierung nach jeder abgeschlossenen Simplex-Iteration, die entsprechend der dargestellten vier Regeln durchgeführt wird, eingreifen. Zur Reduzierung der Gesamtrechenzeit kann dies aber auch nach einer definierten Anzahl an Iterationen erfolgen. Prinzipiell sind aufgrund der hohen Leistungsfähigkeit des Downhill-Simplex-Verfahrens höher dimensionale Probleme mit vielen Variablen sowie nicht-geschlossen-lösbare Systeme lösbar.
Eine Verknüpfung dieses Algorithmus mit dem Algorithmus der verteilten Optimierung wird aufgrund des iterativen Charakters und der hohen Leistungsfähigkeit somit als möglich angesehen.

Als Alternative zum Simplex-Algorithmus stellen Mitra, Levkovitz und Tamiz[28] ein Verfahren mit der Bezeichnung „Interior Point Method (IPM)“ vor. IPM ist stabiler und aufgrund der geringeren Anzahl an notwendigen Iterationsschritten effizienter als der Simplex-Algorithmus. Andersen[29] sieht sogar eine derart hohe Effizienz von IPM, dass er den Einsatz von IPM im Rahmen der massiven Parallelität empfiehlt. Aber auch die Kombination von

[27] Eiselt/Pederzoli/Sanblom, 1987, S. 32 ff.
[28] Mitra,G.; Levkovitz, R, Tamiz, M (1991)
[29] AndersonJ.H.; Mitra, G; Parksinson, D. (1991)

IPM und dem Simplex Algorithmus wird von BIXBY[30] als Möglichkeit aufgezeigt, die Kapazität des Simplex-Algorithmus zu erweitern.

Da durch die Erweiterung um IPM relevante Punkte für eine Implementierung des Algorithmus der verteilten Optimierung nicht betroffen sind, wird im weiteren der Simplex-Algorithmus ohne Berücksichtigung von IPM betrachtet.

Abschließend läßt sich zusammenfassen, dass im Rahmen der mathematischen Optimierung im Sinne der Optimierungstheorie[31] in der Literatur eine Reihe von Verfahren bekannt sind, deren Inhalt in Auszügen beschrieben und auf deren Weiterentwicklungen eingegangen wurde. Wesentliche Unterteilungen der Algorithmen zur Lösung autark optimierender Modelle wurden ausgehend von der Abbildung 3 erörtert. Ebenso wurden relevante Optimierverfahren vorgestellt.

Eine abschließende Bewertung der in diesem Kapitel vorgestellten Algorithmen zur Lösung autark optimierender Modelle hinsichtlich des Einsatzes des Systems der verteilten Optimierung wird zusammen mit der Bewertung der Erkenntnisse aus dem Kapitel 3, welches sich mit der Untersuchung verschiedener Formen von Planungssystemen auseinandersetzte, zusammen in Kapitel 5 vorgenommen.

[30] Bixby, R.E.; Gregory, J.W.; Lustig, I.J; Marstend, R.E.; Shanno (1991)
[31] Göpfert, A (1986), S168 ff.

5 Auswertung der Untersuchungen von bestehenden Planungssystemen und Optimierungsalgorithmen im Hinblick auf die Implementierung des Algorithmus der verteilten Optimierung

In den vorangegangenen Kaptiteln 3 und 4 wurden sowohl Planungssysteme aus dem Anwendungsbereich der rohölverarbeitenden Industrie als auch die darin zur Anwendung kommenden Optimierungsalgorithmen im Sinne der Optimierungstheorie erörtert.

Nachfolgend sollen in einem ersten Schritt Kriterien erarbeitet werden, anhand derer dargestellte Planungssysteme und Optimierungsalgorithmen hinsichtlich der Implementierung des Algorithmus der verteilten Optimierung überprüft werden können. Darauf aufbauend werden in einem zweiten Schritt anhand der festgelegten Kriterien die vorgestellten Planungssysteme und Optimierungsalgorithmen im Hinblick auf die Implementierung des Ansatzes der verteilten Optimierung untersucht.

Der in Kapitel 7 vorzustellende Algorithmus wird schließlich in Kapitel 8 in das nachfolgend zu bestimmende Planungssystem und in den dazugehörigen Optimierungsalgorithmus implementiert.

Beginnend mit der Bestimmung der Kriterien zur Bewertung der Planungssysteme und Optimierungsalgorithmen lassen sich im Hinblick auf den Algorithmus der verteilten Optimierung drei zentrale Elemente nennen:

- Hohe Kompatibilität von Schnittstellen zwischen Planungssystemen
- Iterativer Optimierungsalgorithmus
- Hoher Verbreitungsgrad des Planungssystems

In Kapitel 3.2 wurde basierend auf den Definitionen von Planungssystemen und Planungssegmenten die Kompatibilität zwischen diesen Elementen betrachtet. Als Kompatibilität von Schnittstellen wurde in diesem Zusammenhang das Ausmaß der Formatierung verstanden, welches notwendig ist, um Datensätze ohne inhaltliche Verluste zwischen Planungssystemen und –segmenten zu transferieren. Der Begriff der Kompatibilität wurde dabei im umgekehrt proportionalen Sinne zum Ausmaß des Formatierungsaufwandes der zu transferierenden Daten gesehen.

Das Ergebnis dieser Betrachtung wurde in Abbildung 2 dargestellt. Danach eignen sich die Systeme der **Produktionsplanung** und des **Schedulings** hinsichtlich der Kompatibilität für die Implementierung des Algorithmus der verteilten Optimierung. Andere System- und Segmentschnittstellen zeigen diesen Vorteil nicht oder nur eingeschränkt. Beim Datentransfer zwischen den übrigen Systemen und Segmenten muss in der Regel ein Mindestmaß an Formatierung eingebracht werden, um eine vollständige und korrekte Übertragung gewährleisten zu können. Diese individuell anzupassende Formatierung ist nicht im Sinne eines hochkompatiblen Systems.

Das zweite Kriterium zu Implementierung des Algorithmus der verteilten Optimierung ist die Form des Lösungsalgorithmus eines Systems. Dieser muß einen iterativen Charakter haben, um eine Implementierung des in dieser Arbeit entstandenen Algorithmus zu ermöglichen. Grund für die Notwendigkeit des iterativen Charakters ist, wie in Kapitel 7.2.1 beschrieben, die Voraussetzung, Zwischenergebnise innerhalb des Lösungsprozesses zu ex- und importieren, während der Lösungsalgorithmus des jeweiligen Systems temporär unterbrochen und nach dem Datenaustausch weitergeführt wird.

In Kapitel 4 wurden dazu verbreitete Verfahren zur Modelloptimierung dargestellt und es wurde untersucht, in welchem Maß eine Einsetzbarkeit des

jeweiligen Algorithmus in Verbindung mit dem Ansatz der verteilten Optimierung realistisch umgesetzt werden kann.

So konnte festgestellt werden, dass Modelle, die mit Hilfe einer geschlossenen Lösung optimiert werden, zum einen nur bedingt für den Einsatz in der Produktionsplanung bzw. für den des Schedulings geeignet sind, zum anderen nicht die technischen Voraussetzungen für Implementierung des Systems der verteilten Optimierung haben. Dies ist im Wesentlichen auf die Tatsache zurückzuführen, dass mit dem Ansatz der geschlossenen Lösungen maximal polynominale Gleichungen 4. Grades gelöst werden können. Ferner kann diese Form der Optimierung aufgrund des fehlenden iterativen Charakters für das System der verteilten Optimierung ausgeschlossen werden.

Wie in Kapitel 4.2 gezeigt, kann von den untersuchten Gradientenverfahren mit indirekter Optimierung nur das Newton-Raphson-Verfahren als bedingt sinnvoll für den Einsatz im Rahmen der verteilten Optimierung angesehen werden. Andere indirekt optimierende Verfahren mussten aufgrund der mangelnden Leistungsfähigkeit ausgeschlossen werden. Wegen der zur Komplexität der Funktion überproportional ansteigenden Rechenzeit im Rahmen dieser Arbeit soll auch das Newton-Raphson-Verfahren nicht weiter betrachtet werden.

Ausschließlich das Downhill-Simplex-Verfahren wurden von den Verfahren der direkten Optimierung in Kapitel 4.3 als möglicher Algorithmus eingestuft. Er beherrscht die Komplexität einer praxisnahen Planung und macht aufgrund des iterativen Charakters eine Implementierung des Ansatzes der verteilten Optimierung möglich.

Verbindet man diese Erkenntnis mit den in Kapitel 3.1 vorgestellten Planungssystemen der rohölverarbeitenden Industrie, so stellt sich die Frage, in welchen der dargestellten Planungssystemen das Downhill-Simplex-Verfahren zur Anwendung kommt. Als Ergebnis lassen sich in diesem Zusammenhang

ausschließlich LP-basierende Systeme wie die der Produktionsplanung bzw. der Operativen Planung nennen.

Abbildung 4 werden zur Verdeutlichung die Ergebnisse der vorgestellten Optimieralgorithmen hinsichtlich der Einsetzbarkeit im Rahmen der verteilten Optimierung nochmals zusammengefasst:

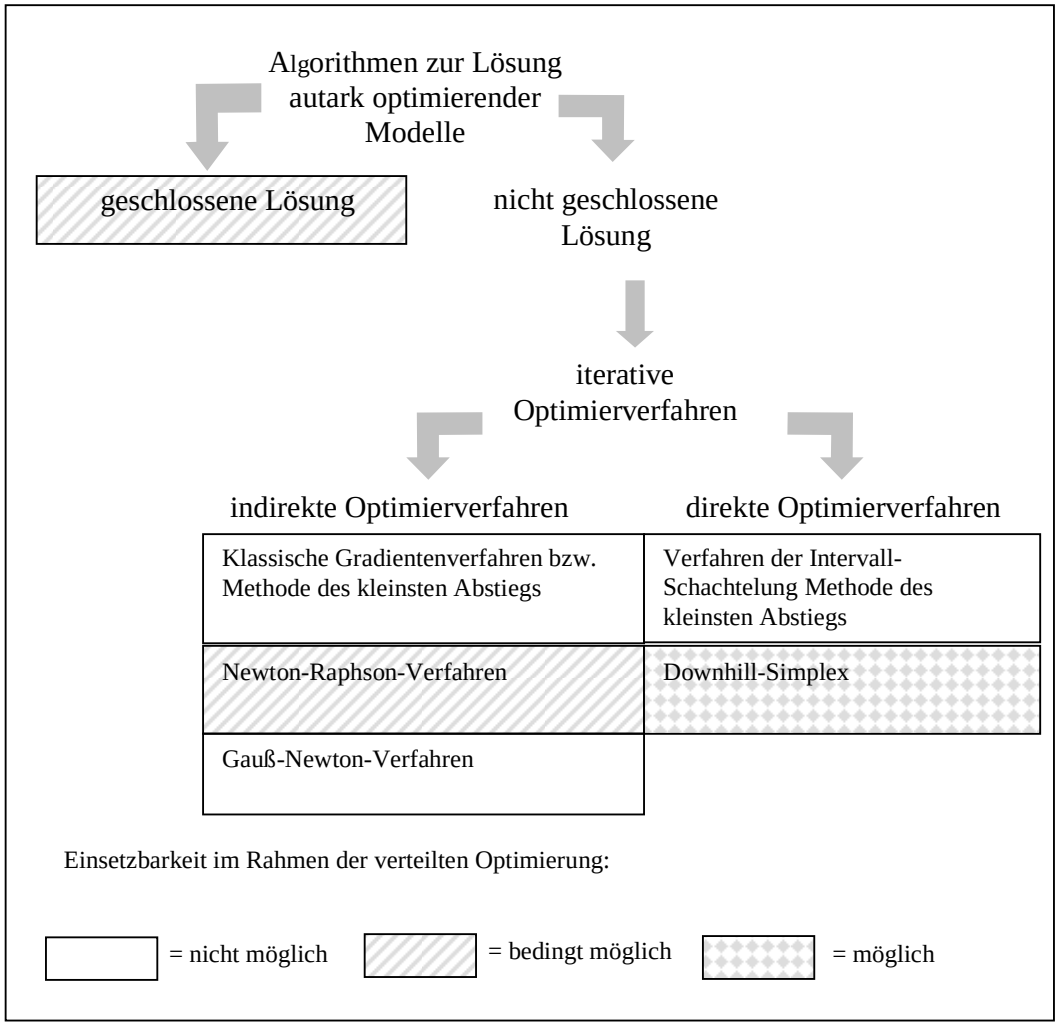


Abbildung 4 - Einsetzbarkeit vorgestellter Optimierverfahren im Rahmen der verteilten Optimierung

Das letzte Kriterium zur Implementierung des Algorithmus der verteilten Optimierung ist schließlich der Verbreitungsgrad des Planungssystems, in das der Algorithmus implementiert werden soll.

Zwar ist es technisch möglich, den erarbeiteten Algorithmus in jedes Optimiersystem, welches rekursiv in asymthotischer Weise die gewünschte Extremstelle der Zielfunktion ermittelt, einzubringen, allerdings geschieht dies in Abhängigkeit der Ausprägung des jeweiligen Optimiersystems. Wie in Kapitel 7.2.1 noch ausführlich beschrieben wird, muss ein Teil des Algorithmus zur verteilten Optimierung in die Optimierungssysteme implementiert werden, um den Optimierungsprozess zwischen zwei Rekursionen anhalten und Daten exportieren und schließlich wieder importieren zu können. Dieser Programmteil muss für jede Software-Lösung separat programmiert werden. Um die Anzahl der Programmvarianten klein zu halten und dennoch eine hohe Beteiligung zu erreichen, soll der Algorithmus der verteilten Optimierung in ein System mit hohem Verbreitungsgrad implementiert werden.

Nimmt man die in Kapitel 3 vorgestellten Planungssysteme als Basis, so lässt sich feststellen, dass in den Bereichen Operative Planung, Budgetplanung und vor allem Produktionsplanung LP-Systeme zum Einsatz kommen. In der rohölverarbeitenden Industrie decken hier nur zwei Softwarelösungen einen überwiegenden Teil der LP-Systeme ab. Zum einen ist dies das System PIMS von der Firma ASPENTECH[32] und zum anderen GRTMPS von HAVERLY SYSTEMS[33]. Aspentech verzeichnet dabei den größten Marktanteil von nahezu 75% aller raffineriebezogenen LP-Systeme. Anzumerken ist, dass beide LP-Systeme hinsichtlich der Dateneingaben, der Form der Modelllösung sowie der Datenausgaben sehr ähnlich sind. Eine Implementierung des Algorithmus der verteilten Optimierung in eines der beiden LP-Systeme kann mit beschränktem Aufwand auf das andere System übertragen werden.

[32] ASPENTECH (http://www.aspentech.com)
[33] HAVERLY SYSTEMS INC. (http://www.haverly.com)

Planungssysteme, die aus dem Bereich des Schedulings oder gar des Advanced Controll kommen, sind zahlreich am Markt verfügbar. Die Adaption auf jedes einzelne System wäre aufwendig und würde aufgrund des geringen Verbreitungsgrades nur eine geringe Beteiligung erreichen.

So lässt sich hinsichtlich des Verbreitungsgrades von Planungssystemen formulieren, dass LP-basierte Systeme, die im Bereich der Operativen Planung, der Budgetplanung und vor allem in der Produktionsplanung zum Einsatz kommen, einen hohen Verbreitungsgrad und damit eine hohe Systembeteiligung bei begrenztem systemseitigem Implementierungssaufwand aufweisen.

Fasst man die Ergebnisse aus der durchgeführten Bewertung im Hinblick hoher Kompatibilität von Schnittstellen zwischen Planungssystemen, iterativer Optimieralgorithmen sowie eines hohen Verbreitungsgrades zusammen, ergibt sich der in Abbildung 5 dargestellte Zusammenhang.

Danach zeigen von den vorgestellten Planungssystemen lediglich die Operative Planung, die Budgetplanung und die Produktionsplanung Optimierverfahren mit einem iterativem Charakter im Sinne des Downhill-Simplex. Eine hohe Kompatibilität konnte nur bei den Systemen der Produktionsplanung und des Schedulings aufgezeigt werden. Den gewünschten Verbreitungsgrad schließlich wiesen die LP-basierten Systeme auf.

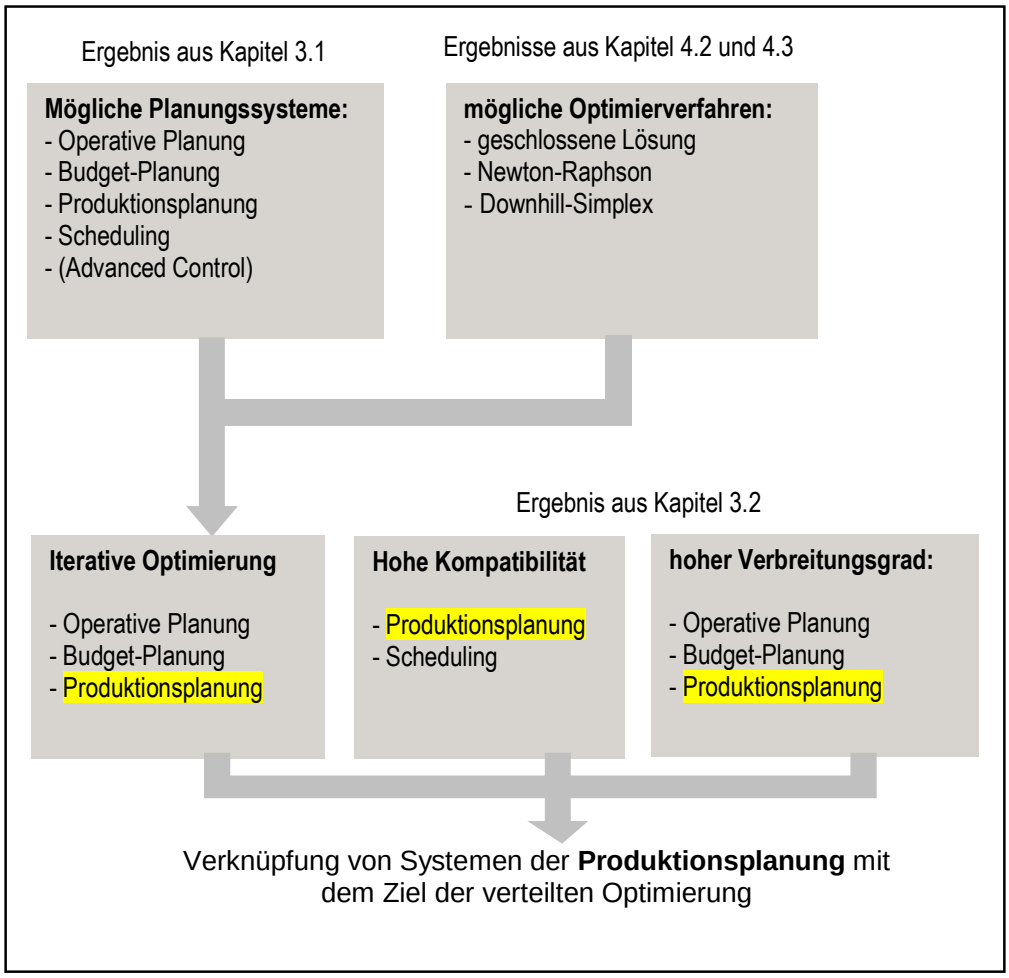


Abbildung 5 – Kriterien zur Implementierung des Systems der verteilten Optimierung

Fasst man die Ergebnisse aller drei Kriterien zusammen, entspricht lediglich das System der Produktionsplanung in Verbindung mit einem LP-System den Anforderungen zur Implementierung des Algorithmus der verteilten Optimierung.

Aufgrund der formulierten Mindestanforderungen lässt sich die weitere Vorgehensweise wie folgt festlegen: Es soll nachfolgend ein Algorithmus erarbeitet werden, der eine Verknüpfung beider Systeme mit dem Ziel der

verteilten Optimierung erlaubt. Der in Kapitel 7 erarbeitete Algorithmus wird dann in eben dieses System der Produktionsplanung und dem dazugehörigen LP-basierten Optimieralgorithmus implementiert, was in Kapitel 8 dargestellt wird.

6 Einordnung des Begriffs verteilter Systeme in bestehende Forschungsansätze

In den vorangegangenen Kapiteln wurde der Begriff der „verteilten Optimierung“ im Sinne der vorliegenden Arbeit definiert und erörtert. Nachfolgend soll dieser Begriff ergänzend in den Kontext aktueller Literatur eingeordnet werden. Im Besonderen soll dies im Hinblick auf die von Rosenberg publizierten Arbeiten erfolgen. Unter der Leitung von Dangelmaier, Rosenberg und Bock beschäftigt sich das Sonderforschungsprojekt 376 – Teilprojekt C2 an der Universität Paderborn[34] mit der Entwicklung praxisrelevanter Modelle und paralleler/verteilter Algorithmen zur echtzeitnahen Planung von vernetzten Produktionssystemen. Grundlage des Forschungsansatzes ist die Annahme, dass global agierende Produktionsunternehmen in ein Supply-Netz eingebunden sind, welches eine unternehmensübergreifende Betrachtung und Optimierung des gesamten Netzwerkes erforderlich macht. Bestehende Planungs- und Steuerungsverfahren berücksichtigen dabei lediglich einzelne Unternehmenssegmente und führen so zur Berechnung lokaler Optima – Aussagen über das globale Optimum können nicht gemacht werden. Ziel des Teilprojektes C2 sowie dieser Arbeit ist die Entwicklung von unternehmensübergreifenden Planungssystemen, die mittels verteilter Algorithmen eine Bestimmung des globalen Optimums für das Produktionsnetzwerk ermöglichen.

Bezogen auf das Gebiet der verteilten Systeme sollen in diesem Zusammenhang Veröffentlichungen von Stefan Bock[35] näher betrachtet werden. Darauf aufbauend wird der Themenbereich der vorliegenden Arbeit in den Kontext der o.g. Forschungsaktivitäten eingeordnet und abgegrenzt. Abschließend wird

[34] Beispielhaft sei hier auf Veröffentlichungen wie „Using distributed systems to control mixed-model assemly lines in realtime [2003]“ von S. Bock, O. Rosenberg, T. van Brackel verwiesen.
[35] Bock, S., 2000

untersucht, in welcher Weise die Ergebnisse dieser Arbeit eine Ergänzung zu den Veröffentlichungen von Stefan Bock darstellen können.

Der in Kapitel 7.2 erarbeitete verteilte Algorithmus kommt in so genannten verteilten Systemen zur Anwendung. Um eine Einordnung des Algorithmus in diesem Zusammenhang zu erörtern, wird nachfolgend die logische Unterteilung verteilter Systeme dargestellt.

Eine der ersten Definitionen und Einteilungen im Bereich der verteilten Systeme lässt sich bei Michael Flynn[36] erkennen. Bereits 1966 formulierte Flynn auf Basis der IBM 704, 709 und 7090 wesentliche Definitionen für Instruktions- und Datensequenzen. Dabei legte Fynn fest, dass unter einem „Instruction Stream“ eine Sequenz von Instruktionen zu verstehen ist, die für die maschinelle Verarbeitung von Anweisungen bestimmt ist. Der „Data Stream“ hingegen stellt danach eine Sequenz von Daten dar, die durch einen oder mehrere „Instruction Streams“ abgerufen wird und eine definierte Weiterverarbeitung initiiert.

Nach dieser Form der Definition der Begriffe „Instruction Stream“ und „Data Stream“ lässt Flynn theoretisch zu, dass ein „Instruction Stream“ sowohl einen als auch mehrere „Data Streams“ zur Verarbeitung auslöst. Ferner können damit mehrere „Instruction Streams“ auch einen bzw. mehrere „Data Streams“ auslösen. Diese Matrix ähnliche Verknüpfungsmöglichkeit war Anfang der 70-ger Grundlage für eine rasante Entwicklung unterschiedlicher Rechner-architekturen. Die Forschungsaktivitäten der expandierenden Computerent-wickler und –hersteller konzentrierte sich dabei stark auf die Entwicklung anwendungsspezifischer Rechnerarchitekturen. Der Entwicklungsschwerpunkt lag dabei eindeutig auf der Steigerung der anwendungsrelevanten Rechner-leistung und noch nicht auf der Nutzung von Synergien in Entwicklung und

[36] Flynn, M. J., 1966, S. 1901 - 1909

Produktion von Rechnersystemen, wie sie im weiteren zeitlichen Verlauf im Vordergrund standen.

Auf derBasis der Definition von „Instruction Stream“ und „Data Stream“ strukturierte Flynn die Fülle der neu entwickelten Kombinationen. Nach Flynn lassen sich Rechnerarchitekturen damit grundsätzlich in vier Klassen von Rechnertypen unterteilen:

1. Single Instruction Stream - Single Data Stream (SISD)
2. Single Instruction Stream - Multiple Data Stream (SIMD)
3. Multiple Instruction Stream - Single Data Stream (MISD)
4. Multiple Instruction Stream - Multiple Data Stream (MIMD)

In die SISD-Klasse werden im wesentlichen Neumannrechner eingeordnet, die mit so genannten sequenziellen Algorithmen arbeiten. Zur ausführlichen Darstellung sei hier auch auf A.Y. Zomaya verwiesen[37].

Die Klasse der SIMD-Rechner zeichnet sich hingegen durch den parallelen Einsatz von mehreren Prozessoren aus. Dabei verarbeiten alle beteiligten Prozessoren den selben „Instruction Stream“, jedoch mit jeweils eigenem Datensatz. In diesem Fall arbeiten die Prozessoren synchron, wodurch SIMD-Rechner in die Kategorie der parallelen Systeme eingeordnet werden können. Hinsichtlich der Speicherzuordnung lassen sich nach Zomaya bestehende SIMD-Systeme in „Shared-memory“ (SM) und „Local-Memory“ (LM) Rechner unterscheiden. Im Gegensatz zu den LM-SIMD-Rechnern, greifen bei SM-SIMD-Rechnern alle Prozessoren auf einen gemeinsamen Speicher zu. Diese simultane Verarbeitung von Datensätzen führt zu einer hohen Leistungsfähigkeit der SIMD-Rechner, die einen sehr zweckgebundenen Einsatz vor allem im Bereich der Bildverarbeitung oder Datenbankverwaltung möglich machen.

In die Gruppe der MISD-Rechner fallen Systeme, in denen mehrere Prozessoren gleichzeitig unterschiedliche Instruktionen, allerdings auf der Basis gleicher Daten verarbeiten können und somit als typische Parallelrechner bezeichnet werden dürfen.

Die Parallelrechner der Kategorie MIMD arbeiten im Gegensatz zu den MISD-Rechnern auf mehreren Prozessoren bei unterschiedlichen Instruktionen auf unterschiedlichen Datensätzen.

Verfolgt man in den Jahren nach der Entwicklung der unterschiedlichen verteilten Systeme deren Ausbreitung, lassen sich im Nachgang folgende Verbreitungsgrade feststellen[38]: Im Vergleich haben sich im Bereich der Parallelrechner die MIMD-Systeme am stärksten etabliert, gefolgt von SIMD-Rechnern. Die in den vorangegangenen Kapiteln dieser Arbeit geforderte hohe Rechnerkompatibilität ist die Grundlage für die Umsetzung des erarbeiteten Ansatzes der Verknüpfung mathematischer Modelle. Aufgrund des hohen Verbreitungsgrades der Systeme MIMD und SIMD sollen diese daher nachfolgend näher betrachtet werden.

Als Kategorien, in denen ein oder mehrere „Instruction Streams“ gleichzeitig, d.h. parallel, verarbeitet werden können, fasst Lüling[39] in seiner Dissertation genau die Kategorien MIMD und SIMD unter den Begriff der „horizontalen Parallelität“ zusammen. So definiert Lüling, dass bei horizontaler Parallelität mehrere „Instruction Streams“ gleichzeitig verarbeitet werden bzw. ein einzelner „Instruction Stream“ parallel auf mehrere „Data Streams“ angewandt werden kann.

---

[37] Zomaya, A.Y., 1996, S. 7
[38] BBN Advanced Computers Inc., 1989
[39] Lüling, R., 1996, S. 2

Eine vertikale Parallelität tritt hingegen dann auf, wenn einzelne Anweisungen eines „Instruction Streams“ nicht sequentiell, sondern parallel in einem oder mehreren Verarbeitungseinheiten (hier Prozessoren) abgearbeitet werden können. Aus bereits genannten Gründen der Kompatibilität und des Verbreitungsgrades sollen die Systeme der vertikalen Parallelität nicht weiter betrachtet werden.

Bevor weiter auf die Unterteilung horizontaler Systeme eingegangen wird, soll zur Veranschaulichung auf die grafische Darstellung der beschriebenen Struktur verteilter Systeme verwiesen werden (siehe dazu Abbildung 6).

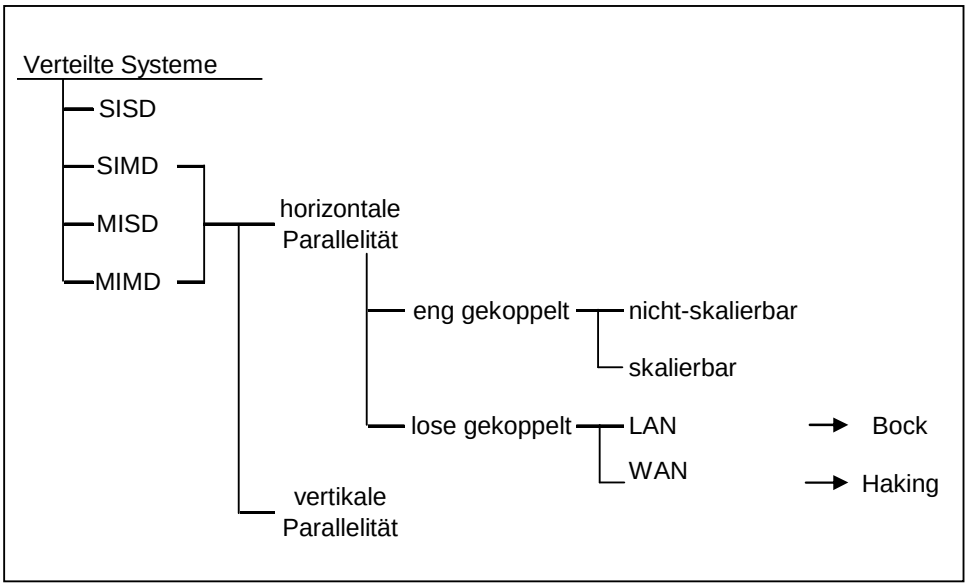


Abbildung 6 – Einteilung verteilter Systeme

Wie in Abbildung 6 dargestellt, wird in der Literatur im Bereich der Systeme horizontaler Parallelität in „enge“ und „lose“ gekoppelte Systeme unterschieden. Beide Systemformen sollen hinsichtlich der inhaltlichen Einordnung dieser Arbeit sowie hinsichtlich der Veröffentlichungen von Lüling und Bock näher erörtert werden.

Wie bei Zomaya[40], Lüling[41] und auch Bock[42] dargestellt wurde, haben sich in den vergangenen Jahren bei Parallelrechnern zwei Systemgruppen entwickelt: Seit Mitte der 80er Jahre haben sich Systeme, die eine größere Anzahl von Prozessoren über ein leistungsfähiges Kommunikationsnetzwerk koppeln, etabliert. Beispielhaft seien hier Paragon von Intel und SP2 von IBM genannt. Diese Systeme lassen sich als „eng gekoppelt“ bezeichnen.

Im Bereich der eng gekoppelten Parallelrechner lassen sich bestehende Systeme wiederum hinsichtlich der prinzipiellen Architektur der eingebundenen Prozessoren in die Kategorien der nicht „skalierbaren“ bzw. „skalierbaren Parallelrechner“ unterteilen.

Nicht skalierbare Parallelrechner nutzen als Kommunikationsinstrument Bus- oder gemeinsame physikalische Speichermedien mit dem Ziel einer sehr hohen Rechenleistung. In der Regel geht dies mit einer festen baulichen Anordnung von Prozessoren einher, deren Verbindung zum gemeinsamen Bus bzw. Speicher einen stark permanenten Charakter hat. Ein temporäres Lösen und Einbinden einzelner Prozessoren über eine Netzwerkstruktur ist bei nicht skalierbaren Parallelrechnern technisch in nur wenigen Fällen sinnvoll. So fallen bei dem vom IBM gebauten Supercomputer Blue Gene/L ca. die Hälfte der Rechenleistung der 65.536 Prozessoren auf Netzwerkfunktionen zwischen den verbundenen Einzelracks, wobei sich immer 1024 der Prozessoren in einem Rack zusammenfassen lassen. Hinsichtlich der exponentiell wachsenden Rechenleistung ist in den vergangen Jahren ein regelrechter Wettbewerb entstanden. So stellte noch 2002 das Regatta-System mit 752 Power-4-Prozessoren der Max-Planck-Gesellschaft (MPG) mit 3,9 Teraflop[43] in der Sekunde das leistungsfähigste System in Deutschland dar. Führend war zu diesem Zeitpunkt der japanische Earth-Simulator von NEC mit fast 36 Teraflop in der

[40] Zomaya, A.Y., 1996, S. 492 und S. 949
[41] Lüling, R., 1996, S. 5 ff.
[42] Bock, S., 2000, S. 33 ff.
[43] Flop = FloatingPoint Operations per Second. Ein Teraflop erfordert die Ausführung von

Sekunde. Im Oktober 2005 baute IMB schließlich den Supercomputer Gene/L im Lawrence Livermore National Laboratory mit einer Rechenleistung von 280,6 Teraflop. Die Investitionen derartiger Rechner liegen in der Regel bei über 100 Millionen US Dollar bei einem Stromverbrauch von bis zu 10 Megawatt.

Die Technologie nicht skalierbarer Parallelrechner findet aufgrund der hohen Rechenleistung in Verbindung mit hohen Kosten nur in sehr speziellen Anwendungen ihren Einsatz. Dazu zählen oftmals rechenintensive Simulationen in den Bereichen Strömungslehre, Elementarphysik, Klima- und Umweltforschung.

Betrachtet man die Nachteile nicht skalierbarer Parallelrechner, so lässt sich Folgendes zusammenfassen: Nicht skalierbare Parallelrechner zeigen eine stark eingeschränkte Flexibilität im Hinblick auf die Anpassungsfähigkeit beteiligter Prozessoren. Eine hohe Anzahl an permanent beteiligten Prozessoren kann zwar integriert werden, allerdings ist die absolute Anzahl technisch begrenzt. Ferner ist es beim Einsatz nicht skalierbarer Parallelrechner nicht sinnvoll, große Distanzen zwischen den verknüpften Prozessoren zu überbrücken, da die begrenzte Übertragungsrate der eingebundenen Bus- und Transfersysteme den großen Vorteil der Leistungsfähigkeit überproportional aufhebt.

Im Hinblick auf die in Kapitel 5 dargestellten Systemanforderungen zur Verknüpfung mathematischer Modelle mit dem Ziel der verteilten Optimierung ist der Ansatz nicht skalierbarer Parallelrechner aufgrund der eingeschränkten Flexibilität, der räumlich stark gebundenen Anordnung beteiligter Prozessoren und vor allem aufgrund der stark überdimensionierten Rechenleistung keine geeignete Plattform zur Umsetzung des erarbeiteten Algorithmus.

---

einer Trillionen Operationen pro Sekunde.

Neben den nicht skalierbaren stellen die skalierbaren Parallelrechner eine zweite Variante eng gekoppelter Systeme im Bereich der horizontalen Parallelrechner dar.

Skalierbare Parallelrechner nutzen im Gegensatz zu den nicht skalierbaren prinzipiell ein Netzwerk zum Datenaustausch zwischen den Prozessoren. Das bedeutet, dass zur Erreichung der angestrebten Rechenleistung Prozessoren nicht mehr unmittelbar in ein Rechnermodul implementiert werden müssen, sondern über einen permanenten Datentransfer verbunden sind. Beispielhaft sei hier der 2004 in Betrieb genommene Linux-Cluster mit 16 Servern des Typs rx2600 an der Universität Karlsruhe genannt. Jeder Server ist dabei mit zwei Prozessoren ausgestattet. Im Jahr 2006 wird dieses System auf insgesamt 334 Knoten ausgebaut. Die Rechenleistung in Relation zu den nicht skalierbaren Parallelrechnern liegt dann bei elf Teraflop. Im Gegensatz zu den nicht skalierbaren Systemen kann zwar die räumliche Distanz vergrößert werden, allerdings stellt dieser Transfer der Daten zwischen den Prozessoren aufgrund der begrenzen Übertragungsrate wiederum die die Rechnerleistung begrenzende Größe dar. Der Linux-Cluster der Universität Karlsruhe arbeitet beispielsweise mit einer Übertragungsgeschwindigkeit von 2 Gigabyte pro Sekunde.

Skalierbare sowie nicht skalierbare Parallelrechner sind konzeptionell nicht dafür ausgelegt, unterschiedliche Rechner flexibel im Sinne einer stark fluktuierenden und temporär begrenzten Einbindung von Rechnern über große Distanzen zu verbinden. Demgegenüber kann die extrem hohe Rechenleistung nicht sinnvoll genutzt werden. Eng gekoppelte Systeme scheiden somit sowohl für die Implementierung des in dieser Arbeit dargestellten Algorithmus als auch für den von Bock entwickelten Ansatz aus.

Neben den „eng gekoppelten Systemen“ stellen die „lose gekoppelten Systeme“ die zweite Gruppe im Bereich der horizontalen Parallelität dar.

Unter „lose gekoppelten Systemen“ versteht man das Verbinden von Personal Computern (PC) durch Netzwerke. Diese Netzwerke zeichnen sich im Gegensatz zu den bei eng gekoppelten Systemen eingesetzten Netzwerken dadurch aus, dass die Anzahl der an dem Netzwerk beteiligten Knoten grundsätzlich unbegrenzt ist. Da bei lose gekoppelten Systemen zunächst nicht die Rechnerleistung, sondern vielmehr die Möglichkeit einer flexiblen Kommunikation von Rechnern im Vordergrund steht, setzt man auf Kosten der Übertragungsraten auf eine hohe Kompatibilität unterschiedlicher PC-Konfigurationen sowie auf einen hohen Verbreitungsgrad der Netzwerke. Damit einhergehend setzt man im ersten Schritt bei lose gekoppelten Rechnernetzwerken weniger auf Spezialanwendungen mit hoher Rechnerleistung, sondern auf Flexibilität hinsichtlich Einbindung neuer Netzwerkknoten und Kompatibilität.

Die etabliertesten Beispiele lose gekoppelter Systeme sind sicherlich LAN- und WAN-Netzwerke, die im Weiteren im Hinblick auf die Implementierung des Ansatzes der verteilten Optimierung untersucht werden sollen.

Die im Gegensatz zu den WAN-Netzen (wide area network) räumlich begrenzten LAN-Netze (local area network) haben sich in den letzten drei Jahrzehnten schwerpunktmäßig in industriellen und wissenschaftlichen Einrichtungen etabliert.

Bock konzentriert sich in seiner Arbeit vor allem auf die Nutzung von PC-Clustern in vorhandenen LAN-Systemen. Im Speziellen untersucht Bock dabei die Clusterung von vollständigen PCs als Alternative zum Erwerb eines separaten Parallelrechners bzw. zum Erwerb von Rechnerzeiten auf bestehenden Parallelrechnern.

Aufbauend auf bestehende LAN-Netzwerke können diese wiederum miteinander verbunden werden und somit abermals ein Netzwerk bilden. Dieser Prozess

der kaskadenförmigen Verknüpfung von LAN-Netzwerken führt schließlich zur Entstehung von WAN-Netzen. Als das etablierteste WAN-Netz ist das Internet zu nennen. Eine hohe Verfügbarkeit und Flächendeckung der WAN-Netze macht diese Netzform für die Ermittlung netzübergreifender, globaler Optima interessant. Der Markt für multiregionale WAN-Netze wird in den nächsten Jahren im Durchschnitt um 4,5 bis 6,5 Prozent wachsen[44]; Schwerpunkt der Ausweitung ist die Vernetzung der Standorte global agierender Unternehmen. Dabei decken lediglich neun global operierende Netzwerkanbieter nahezu die Hälfte des Gesamtmarktes für multiregionale WAN-Services ab[45].

In Ergänzung zu der sich auf LAN-Systeme konzentrierenden Arbeit von Bock zielt diese Arbeit auf das Verbinden von WAN-Netz basierenden Systemen ab, in denen iterativ optimierende Modelle Daten zwischen den einzelnen Iterationsschritten über eine im Internet platzierte Plattform austauschen. Es sei an dieser Stelle auf das Kapitel 8.2 verwiesen, in dem lediglich zur Vereinfachung der im Rahmen dieser Arbeit durchgeführten Versuchsanordnung zu Testzwecken auf ein LAN-Netz zurückgegriffen wird. Eine Übertragung der Ergebnisse auf eine WAN-basierte Anwendung kann – wie im Folgenden noch dargestellt wird – ohne Einschränkung erfolgen.

Fasst man die Einordnung dieser Arbeit in den Zusammenhang verteilter Systeme zusammen, so lässt sich Folgendes formulieren: Zwei der vier vorgestellten Gruppierungen von verteilten Systemen lassen sich unter dem Begriff der horizontalen Parallelität subsumieren. Im Rahmen der Systeme horizontaler Parallelität wurden die WAN-Netze im Bereich der lose gekoppelten Systeme als eine für die Implementierung des erarbeiteten Algorithmus geeignete Plattform herausgearbeitet. In Bezug auf die zunehmende Globalisierung von Unternehmen wurden im Forschungsbereich von Rosenberg Lösungen

[44] Müller, D., 2005
[45] Die neun größten Netzwerkanbieter: AT&T, British Telecom, Cable and Wireless, Equant,

erarbeitet, die unter Generierung von Rechnerleistung durch das Clustern von LAN-verbundenen PC-Einheiten die Ermittlung möglichst guter Lösungen zum Ziel haben. Der in dieser Arbeit entwickelte Algorithmus stellt dazu eine Ergänzung im Bereich der WAN-Netze ohne Erweiterung der Rechenleistung dar.

BT Infonet, MCI, NTT Communications; Sprint, T-Systems (Stand: 01.01.2006)

7 Grundlage einer Software-Lösung zur Umsetzung des Ansatzes der verteilten Optimierung

7.1 Darstellung und Abgrenzung der Grundlagen einer Software-Lösung

Im Rahmen der Produktionsplanung wird in rohölverarbeitenden Raffinerien unter Einsatz eines LP-Modells in Verbindung mit bereits beschriebenen Optimierwerkzeugen die wirtschaftlichste Fahrweise der Raffinerie ermittelt. Das optimierte Ergebnis der Zielfunktion stellt die optimale Lösung für den im Modell abgebildeten Problemkreis dar. Wird zwischen zwei vollständig getrennt optimierenden Systemen ein Zwischen- bzw. Endprodukt ausgetauscht, entstehen in jedem der beteiligten Systeme Variablen, wie z.B. Qualität, Menge oder Preis. Eine Optimierung der Einzelsysteme führt zu Werten, die für das jeweilige lokale System das lokale Optimum darstellen. Da die Variablen einer Komponente durch die unterschiedlichen lokalen Optima beim Zu- bzw. Verkauf unterschiedliche Werte annehmen können, bleibt es dem Verhandlungsgeschick der Ein- bzw. Verkäufer überlassen, einen Wert für die Variablen zu bestimmen. Ein globales Optimum, welches auch volkswirtschaftlich sinnvoll wäre, wird nur zufällig erreicht. Jede andere Lösung kann zwar für den Einzelnen eine Verbesserung darstellen, in der Summe der Lösungen aller Einzelsysteme weicht diese jedoch vom globalen Optimum ab. Vernachlässigt man Faktoren wie Datenschutz und Zeitaufwand, so wäre es theoretisch möglich, die Modelle der Einzelsysteme in ein Gesamtmodell zusammenzufügen, dieses zu optimieren und so das globale Optimum für die gewünschte Variable der ausgetauschten Komponente zu ermitteln. Als Beispiel für eine Variable können Menge, Qualitätseigenschaft oder Preis der ausgetauschten Komponente angenommen werden. Betrachtet man beispielhaft die Eigenschaft des Schwefelgehaltes einer auszutauschenden Komponente, so können aufgrund der unterschiedlichen Wasserstoffsituationen der beteiligten Raffinerien auch die Kosten der Entschwefelung und damit der Schwefelgehalt der betreffenden Komponente wirtschaftlich relevant sein. In

einem Gesamtmodell, welches die Einzelmodelle der beteiligten Raffinerien umfasst, kann der vor dem Hintergrund des wirtschaftlichen Gesamtoptimums optimale Schwefelgehalt der ausgetauschten Komponente ermittelt werden. Bei der in der Praxis verbreiteten Optimierung von getrennten Einzelsystemen kann der Ansatz der verteilten Optimierung dazu dienen, hinsichtlich der Variablen einer ausgetauschten Komponente zu einem globalen Optimum über die beteiligten Einzelsysteme zu kommen. Die Modelle müssen dabei nicht in einem Gesamtmodell zusammengefügt werden.

Bevor mit der Darstellung des Algorithmus der verteilten Optimierung begonnen wird, sollen zunächst die Grundlagen für die Umsetzung der beschriebenen Grundidee dargestellt werden. Im ersten Schritt zählt dazu die Beschreibung des mathematisch funktionalen Zusammenhangs. Dieser soll grundsätzlich die Funktionsweise des Ansatzes der verteilten Optimierung definieren und die theoretische Korrektheit dieses Ansatzes beweisen.

Im zweiten Schritt werden die Strukturen verbundener Planungssysteme, genauer die Systemarchitekturen, im Hinblick auf die Implementierung eines verteilt optimierenden Gesamtsystems dargestellt und erörtert.

7.1.1 Mathematisch funktionaler Zusammenhang der verteilten Optimierung zum Nachweis der theoretischen Richtigkeit der Grundidee

Ein wichtiger Aspekt der Grundidee der verteilten Optimierung ist, dass beim Austausch einer Komponente zwischen den beteiligten Raffinerien derzeit kein Lösungsansatz im Sinne einer Software existiert, der ein globales Optimum bestimmt, ohne beide Raffineriemodelle in *ein* Modell zusammenzufügen. Nachfolgend soll mathematisch gezeigt werden, dass mit dem in dieser Arbeit dargestellten Ansatz ein globales Optimum über die unabhängigen Systeme

der beteiligten Raffinerien ermittelt werden kann, ohne eine vollständige Transparenz der beteiligten Modelle zu erzwingen .

Die Basisannahme dabei ist, dass das ermittelte Optimum zweier separater Modelle nicht zwingend bei dem gleichen Eigenschaftswert (x) vorliegt. Das globale Optimum über beide Modell kann, muss aber nicht dem Optimum der Einzelmodelle entsprechen.

Mit Hilfe der Modulation des bestehenden Deckungsbeitragsverlaufes durch den Ansatz der verteilten Optimierung tritt sowohl das globale als auch das Optimum der autarken Modelle bei dem selben Eigenschaftswert (x) auf. Das bedeutet, dass zur Bestimmung des globalen Optimums nicht mehr beide Modelle in ein zentrales Modell zusammengeführt werden müssen, sondern beide Einzelmodelle separat das globale Optimum erreichen.

Dies ermöglicht den an einer verteilten Optimierung teilnehmenden Modellen eine Bestimmung des globalen Optimums hinsichtlich der betrachteten Variable ohne das Modell des anderen an der Optimierung teilnehmenden Partners zu kennen. Jedes Modell bleibt autark.

Es ist nachfolgend mathematisch nachzuweisen, dass im Falle des lokalen Optimums in den beteiligten Einzelmodellen die betrachtete Eigenschaft unter Anwendung des Ansatzes der verteilten Optimierung den gleichen Wert annimmt wie im globalen Optimum.

Abbildung 7 und Abbildung 8 zeigen beispielhaft den nachfolgenden mathematisch funktionalen Zusammenhang, der im Anschluss hergeleitet und erläutert wird.

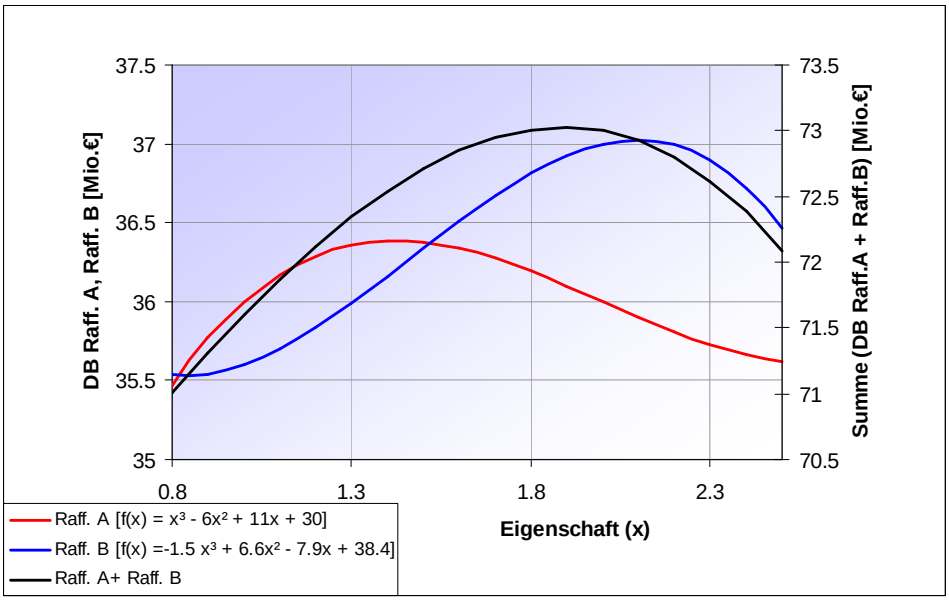


Abbildung 7 – Beispielhafte Darstellung des mathematisch funktionalen Zusammenhangs *ohne* Anwendung des Ansatzes der verteilten Optimierung

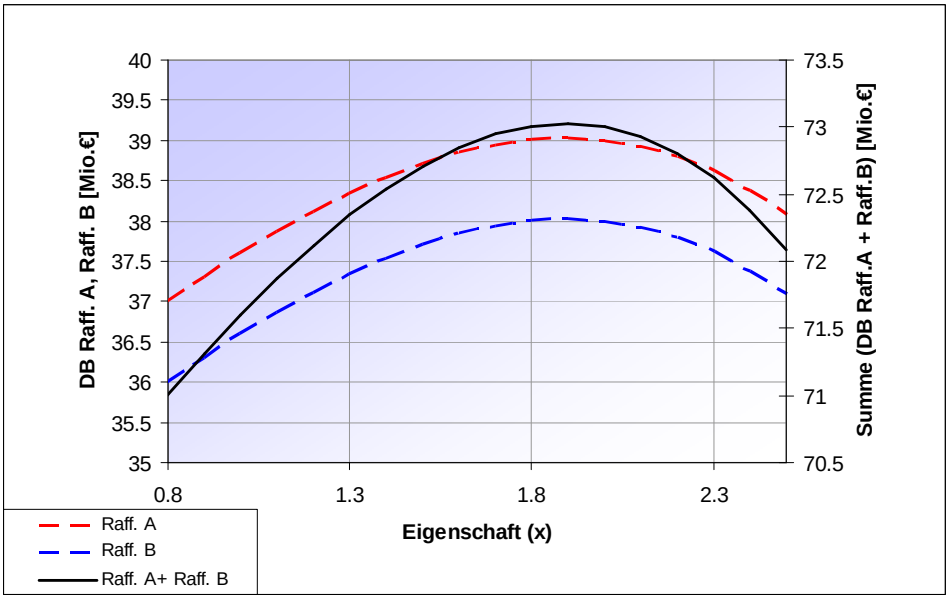


Abbildung 8 – Beispielhafte Darstellung des mathematisch funktionalen Zusammenhangs *mit* Anwendung des Ansatzes der verteilten Optimierung

Erläuterung zu Abbildung 7 und Abbildung 8:

Die Deckungsbeiträge der beteiligten Raffinerien mit Austausch der betrachteten Komponente in Abhängigkeit der entsprechenden Komponenteneigenschaft sind wie folgt dargestellt:

_____ Raffinerie A + B (dieser Gesamtdeckungsbeitrag würde berechnet werden, wenn beide Einzelmodelle in *einem* Gesamtmodell zusammengefasst werden würden)

_____ Raffinerie A **ohne** Anwendung des Ansatzes des verteilten Optimierung

_ _ _ _ _ _ Raffinerie A **mit** Anwendung des Ansatzes des verteilten Optimierung

_____ Raffinerie B **ohne** Anwendung des Ansatzes des verteilten Optimierung

_ _ _ _ _ _ Raffinerie B **mit** Anwendung des Ansatzes des verteilten Optimierung

Mathematische Beweisführung:

Der Deckungsbeitrag zweier isolierter Raffinerien *A* und *B* wird ohne Austausch der Komponenten als konstant angenommen:

$$f_A, f_{B = const.} \tag{4}$$

Es wird jeweils nur eine Eigenschaft, über die optimiert wird, betrachtet. Als Definitionsbereich gilt :

$$\mathbb{D}(f_A) = \mathbb{D}(f_B) = \mathbb{R}^n \rightarrow \mathbb{R} \tag{5}$$

Kommt es zu einer Wechselwirkung zwischen den beiden beteiligten Raffinerien und damit zwischen den beteiligten Funktionen, soll gelten:

$$g_A(\vec{x}), g_B(\vec{x}) : \mathbb{R}^n \rightarrow \mathbb{R} \tag{6}$$

Dabei hängen beide Funktionen von folgenden Größen ab:

$$g_A(\vec{x}) = F\left(f_A, v_A(\vec{x}), \mathbb{D}(v_A)\right) \tag{7}$$

analog $g_B(\vec{x}) = F\left(f_B, v_B(\vec{x}), \mathbb{D}(v_B)\right)$ (8)

Es sind $v_A(\vec{x})$, $v_B(\vec{x})$ als modellspezifische Parametervariationen der beteiligten Raffinerien zu verstehen. Die Teildefinitionsbereiche $\mathbb{D}(v_A)$ und $\mathbb{D}(v_B)$ sind durch sich selbst definiert und ein Teil von $\mathbb{D}(g_A) = \mathbb{R}^n$ bzw. $\mathbb{D}(g_B) = \mathbb{R}^n$, d.h.:

$$\mathbb{D}(v_A) = [v_A(\vec{u}); v_A(\vec{o})] \tag{9}$$

$$\mathbb{D}(v_B) = [v_B(\vec{u}); v_B(\vec{o})] \qquad \text{mit } \vec{u} \text{ und } \vec{o} \text{ als Grenzen} \tag{10}$$

Berücksichtigt man diese Intervallgrenzen bei den oben gezeigten Funktionen, so ist eine Fallunterscheidung einzuführen.

$$g_A(\vec{x}) = \begin{cases} f_A = const. & \forall \vec{x} \notin \mathbb{D}(v_A) \\ f_A + v_A(\vec{x}) & \forall \vec{x} \in \mathbb{D}(v_A) \end{cases} \tag{11}$$

$$g_B(\vec{x}) = \begin{cases} f_B = const. \quad \forall \vec{x} \notin \mathbb{D}(v_B) \\ f_B + v_B(\vec{x}) \quad \forall \vec{x} \in \mathbb{D}(v_B) \end{cases} \tag{12}$$

Für die Wechselwirkung zwischen den Funktionen der Raffinerien bedeutet das, dass nur innerhalb des sich aus dem Raffineriemodell definierenden Definitionsbereiches $\mathbb{D}(v_A)$ bzw. $\mathbb{D}(v_B)$ eine entsprechende gegenseitige Einflussnahme der Funktionen $g_A(\vec{x})$ und $g_B(\vec{x})$ erfolgt (siehe dazu auch die Gleichung 4 und 5). In der Praxis können technische oder logistische Restriktionen als Unter- und Obergrenze der Definitionsbereiche gelten.

Um ein globales Optimum über die Summe der Deckungsbeiträge beider an dem Produktaustausch beteiligten Raffinerien zu finden, ist es theoretisch denkbar, dass die Funktionen f_A, f_B, $g_A(\vec{x})$ und $g_B(\vec{x})$ in ein zentrales Planungsmodell zusammengefasst werden, um dann dieses hinsichtlich des Gesamtdeckungsbeitrages mit Hilfe eines Algorithmus zu optimieren. Allerdings ist, wie eingangs schon beschrieben, in der Praxis kaum ein Unternehmen bereit, zwecks Bestimmung des globalen Optimums seinem Verhandlungspartner sein Modell als Abbild der gesamten Produktion zu überlassen.

Gibt allerdings jedes der an dieser Wechselwirkung beteiligten Unternehmen zumindest das Delta $[g_A(\vec{x}) - f_A]$ bzw. $[g_B(\vec{x}) - f_B]$ an den Verhandlungspartner weiter und implementiert dieser additiv das Delta in seine Zielfunktion, so finden beide Algorithmen getrennt voneinander ein x, bei dem ebenfalls ein globales Optimum zu finden ist.

$$h_A(\vec{x}) = g_A(\vec{x}) + [g_B(\vec{x}) - f_B] \tag{13}$$

$$h_B(\vec{x}) = g_B(\vec{x}) + [g_A(\vec{x}) - f_A] \tag{14}$$

Wie in Abbildung 7 und Abbildung 8 zu erkennen ist, stellen die Funktionen $h_A(\vec{x})$ und $h_B(\vec{x})$ eine Translation der Summenfunktion $g_A(\vec{x}) + g_B(\vec{x})$ in y-Richtung dar. Das bedeutet, dass zu zeigen bleibt, dass $h_A(\vec{x})$ und $h_B(\vec{x})$ ebenfalls durch Translation in y-Richtung ineinander übergehen. Das wiederum muss bedeuten, dass das Delta dieser beiden Funktionen konstant ist.

Es muss demnach gelten:

$$h_A(\vec{x}) - h_B(\vec{x}) = const. \qquad (15)$$

Dies bedeutet:

$$h_A(\vec{x}) - h_B(\vec{x}) = g_A(\vec{x}) + g_B(\vec{x}) - f_B - g_B(\vec{x}) - g_A(\vec{x}) + f_A = f_A - f_B = const.$$

q.e.d **(16)**

Es konnte somit gezeigt werden, dass sowohl die Funktion der beteiligten Raffinerien als auch die globale Funktion nach Anwendung des Ansatzes der verteilten Optimierung an der gleichen Stelle das Optimum erreichen. Das bedeutet, dass der im globalen Sinne optimale Wert der betrachteten Eigenschaft gleich dem Wert der optimalen Eigenschaft der involvierten Planungssysteme ist.

7.1.2 Struktur verbundener Planungssysteme im Hinblick auf die Implementierung eines verteilt optimierenden Gesamtsystems

In den vorangegangenen Kapiteln wurde auf die Voraussetzungen für ein System der verteilten Optimierung eingegangen. Aufgrund dargestellter Prämissen wurde die Verknüpfung zweier LP-gestützter Planungssysteme als geeignet definiert. Auch der mathematische Ansatz der verteilten Optimierung wurde dargestellt. In diesem Kapitel sollen nun der Aufbau und die Struktur, d.h. die Architektur eines verteilt optimierenden Gesamtsystems dargestellt und erörtert werden.

In den bisher angeführten Beispielen und Erläuterungen wird von der Verknüpfung zweier Planungssegmente ausgegangen. Generell ist der Ansatz der verteilten Optimierung, was noch nachzuweisen ist, auf die Teilnahme einer beliebigen Anzahl an Planungssegmenten erweiterbar. Um diesem Erweiterungsgedanken Rechnung zu tragen, soll auf eine netzförmig aufgebaute Verbindung verzichtet werden (siehe dazu Abbildung 9).

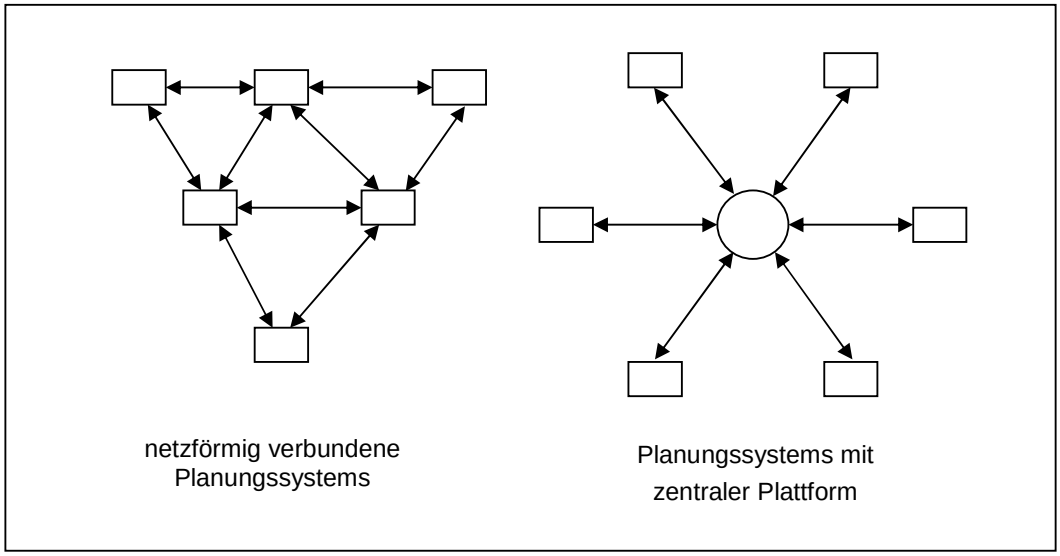


Abbildung 9 – Strukturen zur Verbindung von Planungssegmenten

Vielmehr soll durch die Einrichtung einer zentralen Plattform die Möglichkeit geschaffen werden, eine Information möglichst vielen Teilnehmern parallel zur Verfügung zu stellen. Bei netzförmig verbundenen Planungssegmenten würde eine nicht mehr handhabbare Anzahl an Knoten entstehen. Die zentrale Plattform ist im Rahmen dieser Arbeit als Datenbank zu verstehen, die netzwerktechnisch eingebunden ist.

Die Phase der Verbindung zwischen Planungssystemen und Plattform in der verteilten Optimierung ist in drei Phasen zu gliedern:

1. die Kontaktphase,
2. die Iterationsphase und schließlich
3. die Abschlussphase.

Im Falle einer zentralen Plattform kann das Planungssystem eines Segmentes eine Anfrage oder ein Angebot (im Weiteren mit „Call“ bezeichnet) zum Austausch eines Produktes zentral auf der Plattform hinterlegen, ohne mit einem anderen definierten Planungssegment direkt in Verbindung zu treten. Liegt nun auf der zentralen Plattform ein Call vor, so müssen dazu eine Reihe von Daten bezüglich des betreffenden Produktes zusätzlich auf der Plattform hinterlegt werden.

Diese zu spezifizierenden Daten lassen sich in zwei Gruppen einteilen. Auf der einen Seite sind die fixen Größen, wie Produktanforderungen und Spezifikationen usw. zu hinterlegen, auf der anderen Seite ist die zu optimierende Variable zu definieren. Wie in der Beschreibung der mathematischen Herleitung erläutert, haben verschiedene Variablen eines Produktes einen unterschiedlichen Wert für das jeweilige Planungssegment. So kann die Einhaltung eines Eigenschaftswertes in einem Planungssegment höhere Kosten verursachen als in einem anderen. Dies hängt von den technischen und marktwirtschaftlichen Möglichkeiten der Planungssegmente ab.

Um so wichtiger ist es, entsprechende Produktspezifikationen eindeutig und ausreichend zu definieren. Bei der Verwendung von technischen Parametern ist es wichtig auf die metrische Einheit und die Analysemethode zu achten. Ebenfalls ist es sinnvoll, Informationen zu logistischen Besonderheiten anzuführen. Die Gesamtheit dieser produktbezogenen Informationen sind vor Beginn der eigentlichen Optimierung auf der zentralen Plattform zu hinterlegen und auszutauschen bzw. vor jeder neuen Optimierung wieder zu aktualisieren.

Ist nun mit den beiden genannten Informationsgruppen der Call auf der zentralen Plattform hinreichend definiert, so können alle mit der Plattform verbundenen Teilnehmer in ihrem Planungssegment prüfen, ob prinzipiell ein Austausch des vakanten Produktes produktionstechnisch und logistisch möglich ist. Erst dann kann der Call mit einer Antwort - hier als Recall definiert - beantwortet werden. Ist ein positiver Recall erfolgt und somit ein technisch und logistisch realistischer Partner gefunden, ist die Phase der Kontaktaufnahme abgeschlossen. Der im Rahmen dieser Arbeit entwickelte und in Kapitel 7.2.1 vorgestellte Algorithmus kann mit der Iterationsphase zwischen den beiden Planungssegmenten beginnen.

Ist in der Iterationsphase eine optimale Lösung gefunden worden, bleibt es den beteiligten Produktionssystemen vorbehalten, ob die so ermittelte Lösung akzeptiert wird oder ob eine erneute Kontaktphase zu einem weiteren positiven Recall aufgebaut werden soll. Theoretisch ist es möglich, alle eingegangenen Recalls abzuarbeiten, um diese abschließend zu vergleichen. Bei der Abarbeitung mehrer Recalls ist allerdings die notwendige Rechenzeit zu beachten und individuell zu beurteilen. Mit Beendigung der Iterationsphase wird im Rahmen der Abschlussphase ein Protokoll zwischen den beteiligten Planungssegmenten ausgetauscht, um sicherzustellen, dass alle Beteiligten abschließend die gleichen Informationen besitzen. Die Optimierung bezüglich des Austausches der betrachteten Komponente ist damit abgeschlossen.

Definiert man eine Struktur zur verteilten Optimierung in unterschiedlichen Planungssegmenten, so bleibt neben der prinzipiellen Anordnung von Planungssegment und zentraler Plattform auch die Frage, an welcher Stelle der in dieser Arbeit erarbeitete Algorithmus implementiert werden muss.

Wie in Abbildung 10 dargestellt, sollte der Algorithmus direkt in das LP-Modell des jeweiligen Planungssystems implementiert werden. Die zentrale Plattform dient lediglich zur Speicherung und Verwaltung der anfallenden Daten. Der Grund für die direkte Implementierung des Algorithmus in das LP-System ergibt sich aus der Notwendigkeit unmittelbar in den Iterationsprozess des LP-Systems eingreifen zu können. Ein wesentlicher daraus resultierender Vorteil ist, dass die zentrale Plattform systemunabhängig ist.

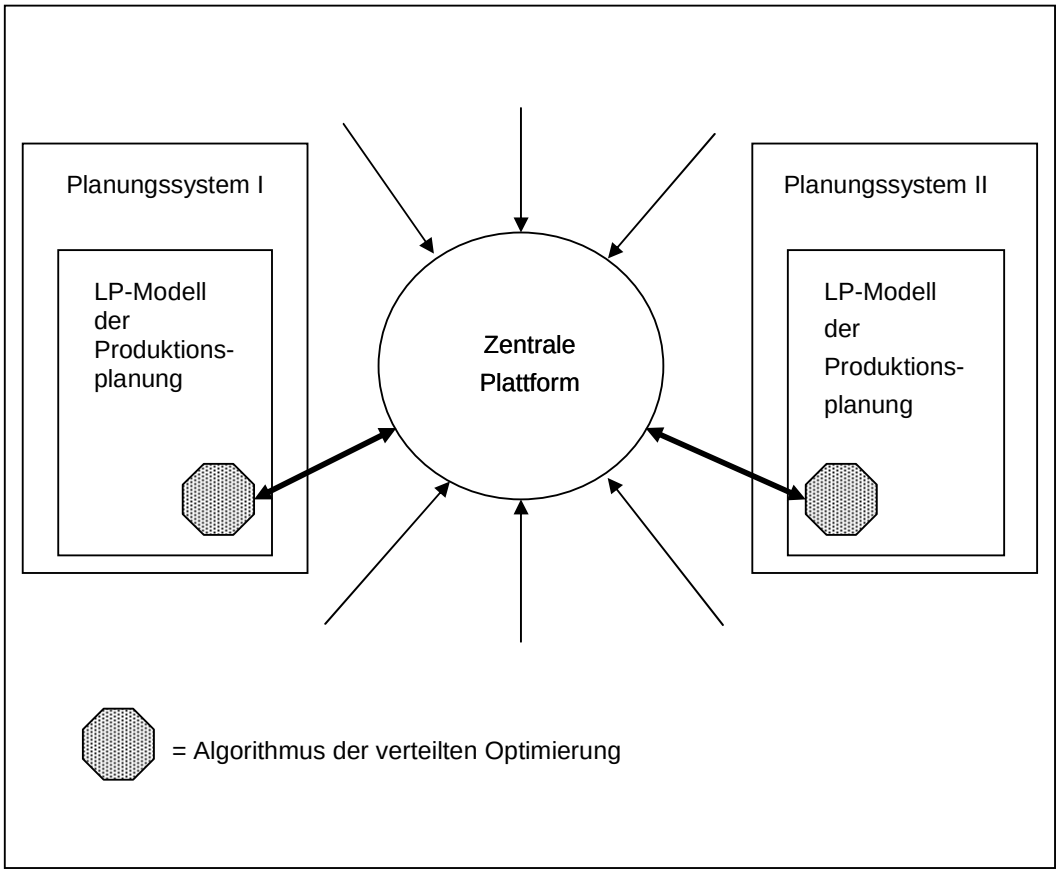


Abbildung 10 – Implementierung des Algorithmus zur verteilten Optimierung in die Planungssegmente

Das bedeutet, dass jedes System Daten auf der zentralen Plattform hinterlegen und abrufen kann, wenn definierte Regeln an die Form der ausgetauschten Daten eingehalten werden. So kann z.B. die LP-gestützte Produktionsplanung des einen Planungssegmentes mit der LP-gestützten Simulation eines zweiten Planungssegmentes zusammengeführt werden. Aus Gründen der Vereinfachung soll jedoch weiterhin von der Verknüpfung zweier Systeme der Produktionsplanung ausgegangen werden.

Bei der beschriebenen Anordnung von unterschiedlichen Planungssegmenten mit einer unabhängigen zentralen Plattform ist eine Erweiterung auf eine größere Anzahl an verknüpften Planungssegmenten denkbar. Auch wenn in dieser Arbeit nur beispielhaft das Testsystem mit zwei Planungssegmenten und einer zentralen Plattform betrieben wurde, so können prinzipiell mehrere Planungssegmente ein Angebot auf der zentralen Plattform hinterlegen. Ein nachfragendes Planungssystem könnte dann seriell alle verfügbaren Angebote prüfen, ggf. den Prozess der verteilten Optimierung starten und die errechneten Ergebnisse vergleichen.

Vergrößert man die Anzahl der teilnehmenden Systeme, kann ein immer weiter expandierendes Optimierungsnetzwerk entstehen. Die Optimierung über ein solches Netzwerk an Planungssystemen verhält sich dabei ähnlich dem Nachfrage- und Angebotsschema, d.h. den Marktmechanismen, einer freien Marktwirtschaft. Von volkswirtschaftlicher Bedeutung ist, dass sich hinsichtlich einer auszutauschenden Komponente nicht nur ein globales Optimum zwischen den zwei beteiligten Systemen einstellt, sondern dies über alle Teilnehmer des Optimierungsnetzwerks geschieht.

Diese Form der verteilten Optimierung soll wegen ihrer Komplexität nicht Gegenstand der weiteren Betrachtung sein. Eine derartige Entwicklung wird lediglich im Ausblick dieser Arbeit nochmals kurz aufgegriffen.

7.2 Umsetzung der Grundidee der verteilten Optimierung in Form eines Algorithmus

7.2.1 Darstellung der Funktionsweise des Algorithmus zur verteilten Optimierung und der Implementierung in ein bestehendes LP-Modell

Wie im Rahmen der Beschreibung der Architektur in Kapitel 7.1.2 erarbeitet, ist der Algorithmus der verteilten Optimierung in ein bestehendes LP-System zu implementieren, um in den interaktiven Optimierungsprozess zwischen den Iterationen eingreifen zu können. Da die Darstellung der Funktionsweise des Algorithmus der verteilten Optimierung sich inhaltlich nur schwer von der Beschreibung der Implementierung in ein LP-System trennen lässt, soll in diesem Kapitel der Gesamtvorgang einer Optimierung, d.h. LP-System und verteilte Optimierung, anhand eines idealisierten Ablaufes, beschrieben werden (siehe dazu Abbildung 11).

Zur Optimierung eines Produktionsplanes werden die abgebildeten Restriktionen und Möglichkeiten, wie in Kapitel 4.3 beschrieben, als Nebenbedingungen formuliert. Der im Rahmen des LP-Systems arbeitende Optimieralgorithmus (z.B. Downhill-Simplex) benötigt als Grundlage die Daten in Form einer Matrix, so dass ein systemeigenes Programm die Vorgabedaten in die entsprechende Matrix umbrechen muss.

Neben den Daten des LP-Systems wird entsprechend des Algorithmus der verteilten Optimierung noch vor der ersten Rekursion ein Datensatz[46], bestehend aus Konstanten und Variablen, von der zentralen Plattform in die Dateien des betrachteten LP-Systems transferiert.

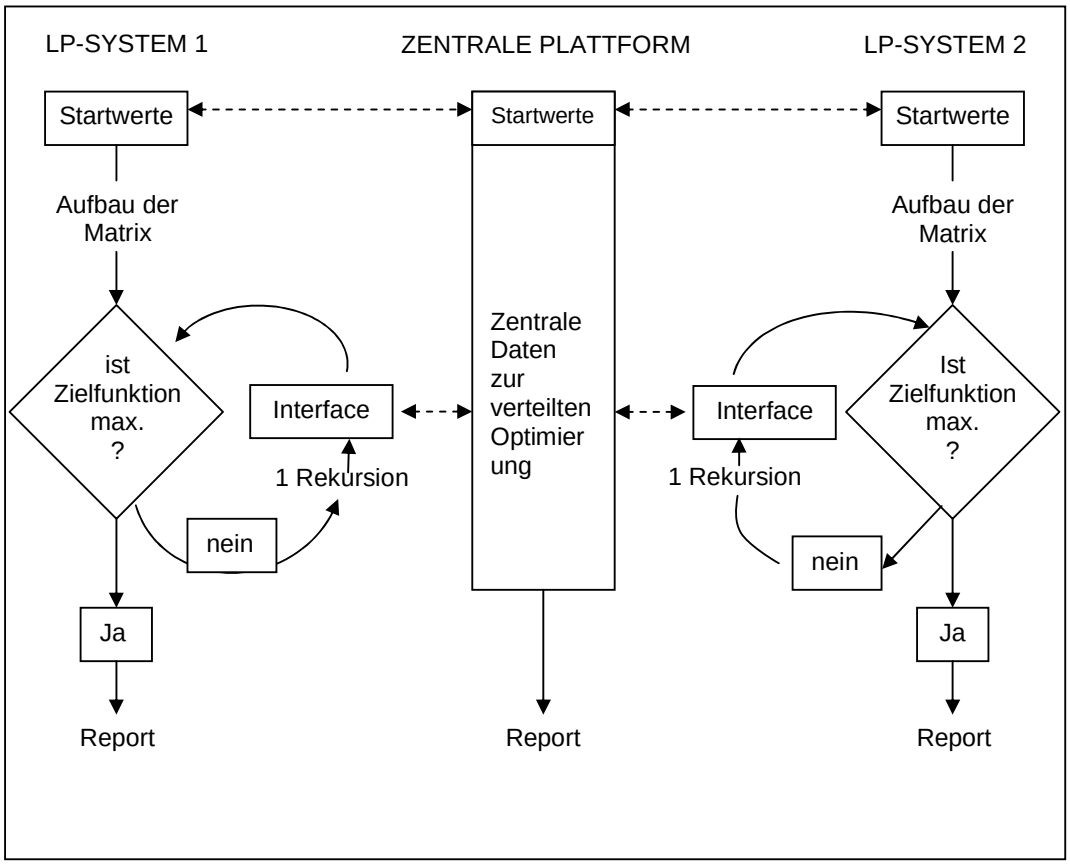


Abbildung 11 – Idealisierter Ablauf nach Implementierung der verteilten Optimierung

Das heißt wiederum, sollen im Rahmen der Optimierung eines Planungssegmentes Daten über eine zentrale Plattform im Sinne der verteilten Optimierung ausgetauscht und so über den global optimalen Austausch einer Ware optimiert werden, dann kann dies aus zwei Gründen erfolgen: Zum einen kann ein Planungssegment einen Datensatz neu auf der zentralen Plattform hinterlegen, zum anderen kann es auf einen bereits vorhandenen Datensatz antworten. Es ist dabei unabhängig, ob die Ware gekauft oder verkauft werden soll. Der Datensatz kann sowohl zu einem Angebot als auch zu einem Gesuch

[46] Der Inhalt des kopierten Datensatzes wird im nachfolgenden Kapitel ausführlich beschrieben.

einer Ware hinterlegt werden. Wichtig ist bezüglich des hinterlegten Datensatzes, dass die Konstanten der betreffenden Ware eindeutig definiert werden (z.B. als Spezifikation) und entsprechend die zu optimierende Eigenschaft als Variable mit den geltenden minimalen oder maximalen Grenzen versehen wird.

Im Weiteren soll folgender Fall betrachtet werden, dass ein LP-System II einen Datensatz über ein Warenangebot auf der zentralen Plattform hinterlegt hat und dessen gezeigte Konstanten einschließlich der Grenzen und der zu optimierenden Variablen aus Sicht des LP-Systems I als technisch möglich eingestuft wird (siehe auch Anhang 1).

Der Begriff des Transferierens soll nachfolgend im Sinne der allgemeinen technischen Übermittlung von Datensätzen verwandt werden. Dabei hängt die Form des Datentransfers stark von derjenigen der zentralen Plattform ab. Im nachfolgend betrachteten Fall reicht ein Kopierbefehl aus: Liegt die zentrale Plattform in Form einer Datenbank vor, so sind lediglich die entsprechenden Routinen zum Abfragen und Hinterlegen von Datensätzen zu installieren.

Sind sowohl die ursprünglichen Basisdaten des LP-Modells, als auch die Daten von der zentralen Plattform transferiert worden, werden diese in die zur Optimierung notwendigen Matrix umgewandelt.

Der Optimieralgorithmus des LP-Systems I versucht nun den gewünschten Extremwert der Zielfunktion zu ermitteln. Dies geschieht auf die in Kapitel 4.3.2 beschriebene Weise, bis die erste Rekursion abgeschlossen ist. Bei der darauf folgenden Rekursion wird der Optimieralgorithmus versuchen, das Ergebnis der Optimierung hinsichtlich der Zielfunktion zu verbessern. Zwischen diesen beiden Rekursionen findet allerdings eine Besonderheit statt, die in gängigen LP-Systemen sehr verbreitet ist: Eigenschaften, die sich nicht linear abbilden lassen, werden nach jeder Rekursion über ein so genanntes ‚Simulating-Interface' in einen externen Programmteil übertragen, berechnet

und in die Matrix zurückgegeben, um wieder in die nächste Rekursion der Optimierung einzufließen.

An genau dieser Stelle setzt der Algorithmus der verteilten Optimierung wieder an: Es wird der für die Kommunikation mit der zentralen Plattform notwendige Datensatz, der auch vor der ersten Rekursion transferiert wurde, wieder zur zentralen Plattform zurückgegeben. Der Inhalt dieses Datensatzes entspricht nun nicht mehr dem ursprünglichen, sondern dem eines im LP-System I im Rahmen einer Rekursion optimierten Datensatzes. Nach erfolgtem Datentransfer zur Plattform wird der Optimieralgorithmus des LP-Systems I solange gestoppt, bis ein neuer, vom LP-System II modifizierter Datensatz zurücktransferiert wird.

Während so das LP-System I nach Durchlauf einer Rekursion und nach dem Transfer eines optimierten Datensatzes in einer Warteposition verbleibt, prüft das LP-System II, welches den ursprünglichen Datensatz in die zentrale Plattform gestellt hat, ob der veröffentlichte Datensatz sich verändert hat. Ist dies im Rahmen der Optimierung des LP-Systems I erfolgt, so erkennt das LP-System II diese Veränderung durch periodische Abfrage eines Triggers. Das LP-System II kann sich nun in zwei möglichen Zuständen befinden: Entweder ist der Optimieralgorithmus des LP-Systems II noch nicht gestartet oder dieser befindet sich wie die Optimierung des LP-Systems I zwischen zwei Rekursionen. Im ersten Fall wird der aus der zentralen Plattform transferierte Datensatz noch in die Basisdaten des LP-Modells implementiert, um dann in eine zu optimierende Matrix umgewandelt zu werden. Im zweiten Fall wird der Datensatz mit Hilfe des Simulating-Interface von der zentralen Plattform direkt in die Matrix des LP-Systems II transferiert. Dabei ist der direkte Transfer der Daten in die Matrix wichtig, d.h. diese wird während einer Optimierung bewusst verändert, da zwischen den einzelnen Rekursionen kein erneute Umsetzung der Daten in die notwendige Matrixform erfolgt.

Als Anmerkung lässt sich formulieren, dass entsprechende Tests im Rahmen dieser Arbeit gezeigt haben, dass die betrachteten Variablen in der Matrix beim Start nie einen Wert haben dürfen, allen Konstanten hingegen immer ein Wert zugeordnet sein muss.

Ist nun dieser Datentransfer in die Matrix des LP-Systems II abgeschlossen, erfolgt eine Rekursion im LP-System II, d.h. eine Optimierung der Matrix mit veränderten Daten. Danach findet der gerade beschriebene Datentransfer in umgekehrter Richtung statt. Mit dem Eintreffen des Datensatzes in das in der Optimierung gestoppte LP-System I kann somit eine erneute Rekursion erfolgen. Daraus ergibt sich ein wechselseitiger Optimierprozess sowohl innerhalb als auch zwischen den beiden beteiligten LP-Systemen. Somit wird die globale Optimierung über die betrachtete betrachteten Eigenschaft unter Beteiligung beider LP-Systeme durchgeführt; es kommt zur gewünschten verteilten Optimierung.

Eine Wiederholung der beschriebenen Rekursion findet solange statt, bis die Zielfunktion in ausreichendem Maße erfüllt worden ist. Das bedeutet, dass im Algorithmus der beteiligten LP-Systeme ein Kriterium zum Abbruch der beschriebenen Rekursionen definiert werden muss. Dieses Kriterium kann zweierlei Form haben: Zum einen kann die Optimierung nach einer maximalen Anzahl an Rekursionen beendet werden (in der Praxis sind es zwischen 20 und 50 Rekursionen), zum anderen kann aber auch eine Toleranz hinsichtlich der Gleichheit zwischen den Ergebnissen zweier aufeinanderfolgender Rekursionen definiert werden. Sind die Änderungen des in der Zielfunktion zu optimierenden Parameters zwischen zwei aufeinanderfolgenden Rekursionen klein genug, wird davon ausgegangen, dass das LP-System gegen die gesuchte Lösung konvergiert. Die Optimierung ist damit beendet. Abweichend von dem Abbruchkriterium bei autarken Systemen ist es wichtig, dass die Optimierung der beteiligten LP-Systeme so beginnt, dass mit Vollendung der ersten Rekursion des einen LP-Systems unmittelbar die erste Optimierung des zweiten LP-Systems beginnt. Ferner ist bei verteilt optimierenden Systemen

darauf zu achten, dass die Anzahl der Rekursionen zuvor definiert, vor der Optimierung beiden beteiligten Systemen bekannt wird und die Anzahl der Rekursionen bei beiden Systemen gleich ist. Der Grund für diese Übereinstimmung ist, dass ein Datenaustausch zwischen den einzelnen Rekursionen zwischen den beteiligten LP-Systemen stattfindet. Ist die Optimierung des einen LP-Systems früher abgeschlossen als die des anderen, fehlt der ‚Gegenspieler' in diesem Optimierprozess.

Aus Sicht der beiden beteiligten LP-Systeme und des Algorithmus der verteilten Optimierung stellt die zwischen den Systemen zu optimierende Variable eine Gleichung dar, deren Veränderung einer Deckungsbeitrags-änderung gegenübersteht. Das heißt, die Zielfunktion stellt eine mathematische Funktion in Abhängigkeit der zu optimierenden Variablen dar.

Die absolute Veränderung des Deckungsbeitrages wird in der verteilten Optimierung zwischen den Systemen ausgeglichen. Damit unterscheidet sich diese Variable nicht von den originalen Variablen, die aus den Nebenbedingungen des LP-Systems resultieren. Aus Sicht des Optimieralgorithmus des LP-Systems ist auf Grund der Tatsache, dass der Algorithmus während der externen Optimierung unterbrochen wird, kein Unterschied zwischen den internen und den aus der verteilten Optimierung resultierenden Gleichungen zu erkennen. Allein diese Tatsache ermöglicht es dem LP-System eigenen Algorithmus, dass es einen systemexternen funktionalen Zusammenhang als einen systeminternen erscheinen lässt und diesen über alle Funktionen gleichermaßen optimiert.

Ein wesentlicher Aspekt, der die Funktionsweise der verteilten Optimierung charakterisiert, ist die Bedeutung von Variablen und Konstanten in den beteiligten Matrizen. Nachfolgend soll kurz darauf eingegangen werden.

Wie bereits beschrieben, setzen sich die Gleichungen und Ungleichungen eines LP-Systems u.a. aus Variablen und Konstanten zusammen. Die Variablen

können dabei im Rahmen der Optimierung zum Erreichen der Zielfunktion verändert werden. Die Konstanten dagegen bleiben während des Optimieralgorithmus des LP-Systems unverändert und sind vor dem Start des Optimiervorganges zu definieren. Das wiederum bedeutet, dass in der Matrix, die vor jeder Optimierung erzeugt wird, sowohl variable als auch konstante Elemente enthalten sind. In der beschrieben Funktionsweise des Algorithmus der verteilten Optimierung ist es allerdings notwendig, dass ein Wert in der Matrix, der in der Einzeloptimierung als Konstante definiert worden ist, mit demjenigen der Variable des zweiten LP-Systems überschrieben wird. Daraus ergibt sich, dass ein zuvor während des Optimierungsprozesses unbeeinflussbarer Wert im Rahmen der verteilten Optimierung zwischen den einzelnen Rekursionen gelesen und überschrieben wird.

Als Beispiel dafür soll die Optimierung des Schwefelgehaltes eines zwischen zwei Systemen auszutauschenden Zwischenproduktes dienen. Eine Erkenntnis, die sich durch Testläufe im Rahmen dieser Arbeit ergeben hat, ist, dass die in der Optimierung zu bestimmende Eigenschaft einen Startwert besitzen muß. Betrag und Vorzeichen des Startwertes haben auf das Endergebnis keinen Einfluss, lediglich die Anzahl der zur Optimierung notwendigen Rekursionen kann mit dem Startwert variieren. Mit diesem Startwert, in diesem Fall der des Schwefelgehalts, wird die dafür vorgesehene Konstante des einen LP-Systems noch vor der Umsetzung in die Matrix mit dem Startwert belegt. Der Grund für den Einsatz einer Konstante ist, dass das erste LP-System lediglich die Auswirkungen auf die Zielfunktion – hier den Deckungsbeitrag – berechnet, nicht aber den Schwefelgehalt des zugekauften Zwischenproduktes verändert. Diese Aufgabe obliegt lediglich dem zweiten verkaufenden LP-System. Nach Erzeugen der Matrix und der ersten Rekursion wird der Schwefelwert mit dem ermittelten Delta des Deckungsbeitrages (Delta DB) an das zweite LP-System übermittelt. Im zweiten, dem verkaufenden LP-System ist der betrachtete Schwefelwert allerdings als Variable definiert und kann unter Einbeziehung des Delta-DB’s des ersten Systems und der eigenen Zielfunktion variiert werden. Testreihen haben ergeben, dass diese Variable wiederum keinen

Startwert haben darf, sondern vom Optimieralgorithmus des zweiten LP-Systems selbständig bestimmt werden muss. Der korrigierte Schwefelwert inklusive des entsprechenden Delta-DB's des zweiten Systems wird in bereits beschriebener Weise über das Simulating-Interface über die zentrale Plattform an das erste LP-System transferiert. In die Matrix der ersten LP-Systems geht dieser neue Schwefelwert allerdings wieder als Konstante ein, mit der das erste LP-System abermals die damit verbundene Deckungsbeitragsänderung berechnet.

Die Werte von Konstanten in einer Matrix zwischen zwei Rekursionen zu lesen und zu überschreiben ist somit Grundvoraussetzung bei der Umsetzung der verteilten Optimierung. Eine schematische Darstellung des eben beschriebenen Datenaustausches ist dem Anhang 1 zu entnehmen.

7.2.2 Entwicklung des Algorithmus zur verteilten Optimierung

Aufbauend auf die im vorangegangenen Kapitel beschriebene Funktionsweise und basierend auf dem dargestellten Ablauf der verteilten Optimierung soll nun der eigentliche Algorithmus schrittweise beschrieben werden.

Um die in Kapitel 8 beschriebenen empirischen Untersuchungen durchführen zu können, wurde der nachfolgend beschriebene Algorithmus im Rahmen dieser Arbeit in VISUAL BASIC umgesetzt. Diese Form der Umsetzung resultierte daraus, dass die Daten zum LP-Modell in Excel-Tabellen vorlagen und sowohl diese Tabellen als auch das LP-Systems selbst unter MICROSOFT betrieben wurden. Allerdings soll die Beschreibung des Algorithmus unabhängig von der Programmiersprache in einem Pseudocode erfolgen, da zum einen der Schwerpunkt dieser Arbeit nicht auf der Programmierung eines Algorithmus liegt und zum anderen sollte der Algorithmus unabhängig von einer spezifischen Programmiersprache formuliert werden.

Der Algorithmus soll nachfolgend auf Basis des in Anhang 1 gezeigten Ablaufs exemplarisch dargestellt werden.

Es seien zwei LP-Systeme aus dem Bereich der Produktionsplanung von Raffinerien gegeben. Sowohl das LP-System 1 als auch das LP-System 2 verwenden das LP-System PIMS[47]. Zur Vereinfachung werden die beiden Planungssysteme mit PIMS 1 und PIMS 2 bezeichnet. PIMS 1 ist im nachfolgenden Beispiel als kaufendes, PIMS 2 als verkaufendes System zu verstehen.

So weist PIMS 2 ein Zwischen- oder Hauptprodukt aus, welches für den Verkauf geeignet und ausreichend innerhalb der beteiligten Planungssysteme spezifiziert ist. Das bedeutet, dass im LP-Modell geeignetes Datenmaterial vorliegt, um die Produkteigenschaften des zu verkaufenden Produktes geeignet zu beschreiben. In der Praxis liegen beim Verkauf von Halb- und Fertigfabrikaten detaillierte Produktspezifikationen vor, die einen Handel möglich machen, ohne das betreffende Produkt zuvor verarbeitet zu haben. Die Produktspezifikation enthält Grenzwerte von definierten Produkteigenschaften inklusive der Angabe zu den entsprechenden Prüfmethoden. Beispielhaft können Eigenschaften wie Dichte, Siedebereich, Aromaten-, Benzol- und Schwefelgehalt genannt werden.

Es ist darauf hinzuweisen, dass die Auswahl des Produktes weder vom Planungssystem noch mittels des Algorithmus der verteilten Optimierung automatisch erfolgen kann. Mit Hilfe eines LP-Systems können lediglich Zwischen- oder Hauptprodukte ermittelt werden, die sich für den Verkauf eignen, in dem Sinne, dass das betreffende Produkt nicht oder nur unter dem vermuteten Marktwert produziert und ggf. weiterverarbeitet wird.

Vor dem Austausch von Datensätzen mit der zentralen Plattform in Verbindung mit dem Algorithmus der verteilten Optimierung ist

[47] PIMS von der Firma ASPENTECH - Houston, USA

sicherzustellen, dass beide LP-Systeme eine vollständige Optimierung durchlaufen haben. Dies ist aus zwei Gründen wichtig: Einerseits ist anhand des Ergebnisreports ein Zwischen- oder Fertigprodukt erkennbar, welches ausgetauscht werden kann und andererseits ist ein Basis-Deckungsbeitrag, d.h. ohne Austausch von Produkten notwendig. Mit jeder Rekursion in dem Algorithmus der verteilten Optimierung kann so die entsprechende relative Deckungsbeitragsänderung, die zur Ermittlung des globalen Optimums notwendig ist, bestimmt werden.

Im nachfolgenden Beispiel eines Algorithmus zur verteilten Optimierung stellt der Preis des Produktes *i* die Variable dar. Die zu maximierende Zielfunktion ist der Gesamtdeckungsbeitrag über die beteiligten LP-Systeme. Das PIMS 2 bildet das verkaufende und PIMS 1 das kaufende LP-System ab:

Der erste Schritt nach Auswahl des **Produktes i** ist das Kopieren der **Produktspezifikation i** vom PIMS 2 in die zentrale Plattform.

```
Copy (Produktspezifikation i) von (PIMS 2) nach
     (zentrale Plattform)
```

Gleichzeitig wird ein Trigger auf der zentralen Plattform auf den Betrag [1] gesetzt, um kenntlich zu machen, dass der Datensatz noch von keinem weiteren System verändert wurde. Solange dieser Trigger den Betrag [1] hat, verbleibt PIMS 2 in einer Warteschleife (hier: **breaktime**). Im Rahmen der durchgeführten Tests wurde die zentrale Plattform durch eine Excel-Tabelle realisiert. Im Gegensatz zu einer Datenbank kann eine Excel-Tabelle nicht von zwei Usern gleichzeitig gelesen und gespeichert werden. Befindet sich die zentrale Plattform im Eingriff vom PIMS 1, kann PIMS 2 nicht im vollen Umfang - inklusive der Fähigkeit zu speichern - auf die zentrale Plattform zugreifen. Es ist zu prüfen, ob die zentrale Plattform verfügbar ist.

Für PIMS 2 gilt:

```
Setze trigger=1

While trigger=1 or (zentrale Plattform) nicht
verfügbar
     Do wait breaktime
```

In dem kaufenden Planungssystem, welches von dem LP-System PIMS 1 optimiert wird, ist analog zum Planungssystems 2 das in der zentralen Plattform zu suchende Produkt zu definieren. Ähnlich dem PIMS 2 kann beim PIMS 1 die autarke Optimierungsrechnung dem Planer Aufschluss über die Art des gesuchten Produktes geben. Aus der Definition des zu suchenden Produktes ist eine damit verbundene Mindestspezifikation abzuleiten.

Ist die auf der zentralen Plattform hinterlegte Produktspezifikation innerhalb der im PIMS 1 definierten Mindestanforderungen, so wird der vom Planungssystem 2 hinterlegte Datensatz von der zentralen Plattform in die Tabelle des LP-Systems 1 transferiert, vom Simulating-Interface gelesen und in die Matrix des PIMS 1 gebracht. Ist dieser Datentransfer abgeschlossen, wird der Optimieralgorithmus des LP-Systems 1 gestartet. Die Suche nach dem geeigneten Produkt und den damit verbundenen Datensatz in der zentralen Plattform ist nun beendet, d.h. die Phase der verteilten Optimierung beginnt.

Für PIMS 1 gilt:

```
i := 1
while i <> NIL or i <> 0 do
begin
     Copy (Produktspezifikation i) von (zentrale
```

```
          Plattform) nach (PIMS1)
     if   (Produktspezifikation i) innerhalb
          (Mindestanforderung)
     then  begin
               copy (datensatz i) von (zentrale
                    Plattform) nach (PIMS 1)
               start LP-Optimierung
               exit
          end
     else i:=i+1
end
```

Nach der ersten Rekursion wird das Simulating-Interface (SI) aufgerufen. Damit werden Excel-Tabellen gestartet, in denen wiederum bei deren Öffnen selbstaktivierende Makros aktiviert werden, die den in der vorangegangenen Rekursion berechneten Wert der Eigenschaft ‚Preis' aus der Matrix des PIMS1 herauslesen und in eine Tabelle mit dem Namen OUTPUT_DAT_1 schreiben. Das Kopieren von Daten aus der Matrix des LP-Systems in eine Tabelle ist mit den üblichen Makrobefehlen eines Tabellenkalkulations-Programms möglich, wenn die Matrixadressen der auszulesenden Daten bekannt und definiert sind. Die für den Austausch relevanten Daten bestehen aus dem Wert, aus der zu optimierenden Eigenschaft (ITEM_PIMS) und aus der Deckungsbeitrags-änderung (DELTA_DB1), die sich aus der abgeschlossenen Rekursion ergibt.

Die Tabelle OUTPUT_DAT_1 muss schließlich aus Gründen der Eindeutigkeit in die Tabelle TRANSFER_DAT und diese wiederum zur zentralen Plattform kopiert werden. Ist dieser Vorgang abgeschlossen, wird der Trigger in der zentralen Plattform auf den Wert O („Null") gesetzt, um dem zweiten LP-System PIM 2 zu signalisieren, dass ein veränderter Datensatz existiert.

Die entsprechende Umsetzung in Form eines Algorithmus lässt sich für das PIMS 1 wie folgt formulieren:

```
when open SI-File
     do begin
          schreibe (ITEM_PIMS, DELTA_DB1)
               von (MATRIX_1) nach (OUTPUT_DAT_1)
          copy (OUT_DAT_1) nach (TRANSFER_DAT)
          copy (TRANSFER_DAT) nach (zentrale
               Plattform)
          setze Trigger = 0
        end
```

Die erste Rekursion im System der verteilten Optimierung ist somit abgeschlossen. Das LP-System 1 verbleibt in einer Warteschleife, bis der Trigger wieder den Wert [1] annimmt und damit signalisiert, dass die Daten der Datei INPUT_DAT auf der zentralen Plattform verändert wurden.

Für PIMS 1 gilt:

```
While trigger = 0 or (zentrale Plattform) nicht
verfügbar
     Do wait breaktime
```

Das LP-System 2 ist während der zuvor beschriebenen Rekursion des LP-Systems 1 in einer Warteschleife verblieben. Erst dann, wenn sich die Datei INPUT-DAT auf der zentralen Plattform nicht mehr im Eingriff des PIMS 1 befindet und der Trigger den Wert 0 („null“) hat, wird die Datei INPUT_DAT in das LP_System 2 kopiert.

Für PIMS 2 gilt:

```
While trigger = 1 or (zentrale Plattform) nicht
verfügbar
     Do wait breaktime

Copy  (TRANSFER_DAT) nach (INPUT_DAT_2)
```

Analog zum PIMS 1 wird nach Beendigung des Kopiervorganges der Optimiervorgang des PIMS 2 gestartet. Ebenso wie beim PIMS 1 wird auch beim PIMS 2 nach einer abgeschlossenen Rekursion das Simulating-Interface gestartet. Wiederum werden in bekannter Weise die ermittelten Werte für die zu optimierende Eigenschaft ITEM_PIMS und die nach der abgeschlossenen Rekursion zu ermittelnde Deckungsbeitragsänderung DELTA_DB2 zwischen der letzten Rekursion (n_B) und der vorletzten Rekursion (n_B -1) aus der MATRIX_2 in die Tabelle OUTPUT-DAT_2 geschrieben. Diese Datei wird wieder in das Format der auszutauschenden Datei TRANSFERDAT_DAT gebracht und in die zentrale Plattform kopiert. Auch hier wird nach abgeschlossenem Kopiervorgang der Trigger abermals auf den Wert 0 („null“) gesetzt.

```
Start LP-Optimierung
when open SI-File
     do begin
          schreibe (ITEM_PIMS, DELTA_DB2)
               von (MATRIX_2) nach (OUTPUT_DAT_2)
          copy (OUTPUT_DAT_2) nach (TRANFER_DAT)
          copy   (TRANFER   _DAT)   nach   (zentrale
     Plattform)
          setze Trigger = 0
```

```
        end
```

Das LP-System 2 verbleibt wieder in bekannter Warteschleife.

```
While trigger =1 or (zentrale Plattform) nicht
verfügbar
     Do wait breaktime
```

Bei der nächsten Abfrage des Triggers von PIMS 1 beginnt eine weitere Rekursion in beschriebener Weise.

Die Abbruchbedingung für den beschriebenen rekursiven Algorithmus der verteilten Optimierung ist vor der Optimierung und für beide beteiligten LP-System gleich vorzugeben ($n_A = n_B$). Zwar besteht generell in LP-Systemen die Möglichkeit den Abbruch der Optimierung so zu definieren, dass bei der Unterschreitung einer vorgegebenen Deckungsbeitragsänderung zwischen zwei aufeinanderfolgenden Rekursionen und bei Erfüllung aller formulierten Gleichungen die Optimierung beendet wird. Optimieren jedoch im Rahmen der verteilten Optimierung zwei LP-Systeme wechselseitig, ist es notwendig, dass beide LP-Systeme die gleiche Anzahl an Rekursionen durchzuführen haben, um auszuschließen, dass eines der beiden LP-Systeme die Optimierung bereits beendet hat, bevor das zweite zu einer Lösung gekommen ist. So ist die Anzahl der Rekursionen fest vorzugeben. Wichtig ist eine Größenordnung zu wählen, die beiden LP-Systemen eine realistische Möglichkeit gibt, das Optimum zu erreichen. Allerdings sollte die Anzahl der Rekursionen auch nicht zu hoch sein, da die Rechenzeit sich je nach Umfang der beteiligten Modelle und der Leistungsfähigkeit der eingesetzten Rechner um ein Vielfaches pro Rekursion erhöhen kann. Im Rahmen der empirischen Untersuchungen mit der verteilten Optimierung haben sich 50 Rekursionen als sinnvoll erwiesen. Es hat sich gezeigt, dass sich aufgrund des asymptotischen Charakters das Ergebnis nur

marginal ändert, wenn das endgültige Ergebnis bereits deutlich früher als bei der 50. Rekursion gefunden wurde. Eine Optimierung über 50 Rekursionen ist in keinem der untersuchten Fälle notwendig gewesen, da LP-Systeme, die bis zur 50. Rekursion ermitteln konnten, auch mit einer höheren Anzahl an Rekursionen zu keiner Lösung gekommen wären.

Die gezeigte Form des Algorithmus der verteilten Optimierung diente als Anhalt zur praktischen Umsetzung. Auf die Beschreibung der Umsetzung wird im Kapitel 8 näher eingegangen. Ferner werden in den nachfolgenden Kapiteln die Ergebnisse empirischer Tests darstellt und die daraus ermittelten Möglichkeiten und Grenzen des aufgezeigten Algorithmus in der Praxis beschrieben.

8 Ergebnisse empirischer Untersuchungen bei Umsetzung der verteilten Optimierung und daraus resultierende Möglichkeiten und Grenzen

8.1 Darstellung der Untersuchungsergebnisse in Bezug auf den praktischen Einsatz der entwickelten Software-Lösung

In den vorangegangenen Kapiteln wurde der Grundgedanke der verteilten Optimierung vorgestellt; es wurde die Architektur, der Aufbau eines Systems der verteilten Optimierung erörtert und schließlich ein Algorithmus erarbeitet, der als Vorlage zur Umsetzung des erarbeiteten Ansatzes dient.

Nachfolgend wird die praktischen Umsetzung der in Kapitel 7.2.2 erarbeiteten Algorithmusvorlage dokumentiert und ausgewertet. Die Ergebnisse der empirischen Untersuchungen mit dem System der verteilten Optimierung hinsichtlich der Möglichkeiten und Grenzen werden abschließend erörtert und zusammengefasst.

Zur Umsetzung der in Kapitel 7.1.2 beschriebenen systemseitig notwendigen Architektur wurden zwei LP-Systeme der Firma ASPENTECH mit der Bezeichnung PIMS inklusive eines entsprechenden LP-Modells verwendet. Als zentrale Plattform diente eine auf einem Server hinterlegte Excel-Tabelle. Beide PIMS-Systeme wurden auf zwei getrennten Rechnern installiert und über ein LAN-Netzwerk mit dem Server und der dort gespeicherten zentralen Plattform verbunden. Die auf Excel-Tabellen basierenden PIMS-Systeme wurden um den im vorherigen Kapitel beschriebenen Algorithmus erweitert. Die Umsetzung des Algorithmus erfolgte in Visual-Basic, um eine problemlose Handhabung der PIMS-internen Excel-Tabellen gewährleisten zu können. Aus Gründen der Homogenität und zur Vereinfachung wurde die zentrale Plattform ebenfalls als Excel-Tabelle angelegt. Der Einsatz einer Datenbank als zentrale Plattform ist für einen Einsatz in der Praxis aufgrund der Zugriffszeiten und

des bei vielen Systembenutzern hohen Datenaufkommens generell zu empfehlen. Für den Testfall stellte sich allerdings eine Excel-Tabelle als ausreichend dar. Der Aufwand zur Programmierung einer Datenbank hätte den Nutzen nicht gerechtfertigt, da die Ergebnisse durch den Einsatz einer simplen Excel-Tabelle unverändert bleiben.

Wie bereits beschrieben besteht ein LP-System im Wesentlichen aus zwei Teilen. Einerseits ist ein Solver inklusive verschiedener Algorithmen zur Optimierung der Matrix eingebettet in eine Bedieneroberfläche notwendig, andererseits ist ein Modell als Abbild des zu optimierenden Problems erforderlich.

Für die empirischen Untersuchungen wurde ein Modell der Erdöl-Raffinerie Emsland GmbH (Deutsche BP AG Lingen) als Ausgangsmodell gewählt. Dieses Ausgangsmodell wurde in zwei spezifische Modelle umgewandelt. Beide sind in der abgebildeten Raffineriestruktur gleich, mit Ausnahme des Anlagenteils „Cyclohexan-Anlage“. Ohne näher auf den technischen Aufbau einer solchen Anlage einzugehen, ist in diesem Zusammenhang lediglich wichtig, dass in dem untersuchten Modell die Cyclohexan-Anlage Benzol zu Cyclohexan umwandelt unter Einbeziehung von Energie und Wasserstoff. Das Modell des LP-Systems I stellt die Versorgung der Cyclohexan-Anlage zu einem Teil aus eigenproduziertem Benzol und zu einem zweiten Teil aus zugekauftem Benzol dar. Die maximale Einsatzkapazität der Cyclohexan-Anlage wurde mit 20 Tonnen pro Stunde (= t/h) definiert. Es wurde keine Mindestauslastung der Anlage angenommen. Die Eigenbenzolmenge wurde mit der unteren Grenze von 7 t/h und einer oberen Grenze von 10 t/h angegeben. Die Zukaufsmenge konnte zwischen 0 t/h und 13 t/h variieren. Der Cyclohexan-Preis wurde als konstant und nicht mengenabhängig gestaffelt festgelegt.

Das Modell des LP-Systems II entspricht mit Ausnahme der Cyclohexan-Anlage und dem Preis-Erlös-Szenario dem Modell des LP-Systems I. Die im

LP-System II modellierte Cyclohexan-Anlage kann **ausschließlich** aus dem Eigenbenzol versorgt werden, ein Zukauf ist nicht möglich. Die Anlagenkapazitäten wurden mit 0 t/h bis 10 t/h gewählt. Das eigenproduzierte Benzol kann innerhalb der Anlagengrenzen zum Verkauf zur Verfügung gestellt werden.

Fasst man die LP-Systeme I und II zusammen, so kann Benzol vom LP-System II an I verkauft werden, um unter Rücknahme der Produktion im LP-System II die Cyclohexan-Anlage des LP-Systems I über das Maß der eigenproduzierten Benzolmengen hinaus auszulasten. Als Aufgabe soll somit die für beide LP-Systeme in der Summe deckungsbeitragsoptimale Verarbeitungsmenge von Benzol ermittelt werden. Das System der verteilten Optimierung kann den Ort (wo das Cylohexan produziert werden soll) und die Menge (wie viel an den jeweiligen Standorten produziert werden soll) frei optimieren.

Unter dem Ausdruck „Preis-Erlös-Szenario“ sollen im Folgenden die Preise aller im Modell zugekauften Produkte sowie die Erlöse aller im Modell verkauften Produkte zusammengefasst werden. Das Preis-Erlös-Szenario der beiden an den empirischen Untersuchungen beteiligten LP-Systeme wurde von einem gemeinsamen Szenario abgeleitet und dann um 0-5% nach dem Zufallsprinzip einmalig verändert, um zwei ähnliche, aber nicht gleiche Preis-Erlös-Szenarien zu erzeugen. Die so konstruierten Szenarien blieben bei allen Untersuchungen gleich. Grundsätzlich kann der Algorithmus der verteilten Optimierung auch bei Gleichheit der verwendeten Preis-Erlös-Szenarien angewandt werden. Jedoch sind die Untersuchungsergebnisse nur wenig aussagekräftig, da eine Interaktion zwischen den beteiligten Modellen nicht auf direkte wirtschaftliche Vorteile, sondern allenfalls auf eine technisch günstigere Verarbeitungsmöglichkeit abzielt. Grade der differierende Wert verschiedener Komponenteneigenschaft in einem Planungssystem (im weiteren: innerer Wert) soll in den nachfolgend beschriebenen Untersuchungsergebnissen herausgearbeitet werden.

Um die Funktionsfähigkeit des Ansatzes der verteilten Optimierung unter Berücksichtigung des in Kapitel 7.2.2 dargestellten Algorithmus zu untersuchen, wurden im Rahmen dieser Arbeit praktische Untersuchungen durchgeführt. Diese Untersuchungen wurden in 4 Phasen unterteilt:

Phase I: Testen der Funktionsfähigkeit der beteiligten LP-Systeme als autark optimierende Systeme ohne Einbindung in die Architektur der verteilten Optimierung.

Phase II: Verifizieren des korrekten Datenaustausches zwischen den beteiligten LP-Systemen und der zentralen Plattform ohne Durchführung von Optimierungungsrechnungen.

Phase III: Testen des grundsätzlichen korrekten Ablaufs einer vollständigen Optimierung (wie in Kapitel 7.2.1 beschrieben) unter der speziellen Beachtung der im Algorithmus genannten Start- und Abbruchbedingungen.

Phase IV: Verhalten der beiden beteiligten Solver im Rahmen der verteilten Optimierung im Hinblick auf verschiedene Funktionsverläufe und Ermittlung der sich daraus ergebenden Möglichkeiten und vor allem der Grenzen des in dieser Arbeit erarbeiteten Ansatzes.

Sieht man das System der verteilten Optimierung als Gesamtsystem, so kann im Rahmen von empirischen Untersuchungen die Startsituation, d.h. die eingesetzten LP-Modelle, Mengen, Produktqualitäten, Preis-Erlös-Szenarien usw., untersucht und systematisch variiert werden. Die durch die Optimierung ermittelten Ergebnisse werden anschließend analysiert und interpretiert. Da die beschriebene Architektur des Optimierungssystems aus mehreren Komponenten (Planungssysteme, zentrale Plattform, implementierte Algorithmen usw.) besteht, wurden zunächst die Einzelkomponenten und dann

das Gesamtsystem analysiert. Zur Umsetzung dieser komponentenweisen Betrachtung wurden zwei Objekte in die bereits in Abbildung 9 gezeigte Systemarchitektur implementiert (siehe dazu Abbildung 12). Das erste ist ein auf der Programmiersprache VISUAL BASIC basierender Viewer, der die Beträge ausgewählter Variablen zwischen den abgeschlossenen Rekursionsschritten darstellt. Die Programmierung des Viewers geht dazu bis in den Algorithmus der verteilten Optimierung hinein. Das zweite zu Testzwecken installierte Objekt ist ein auf einer Excel-Tabelle basierender Testsimulator. Mit Hilfe dieses Simulators kann das Verhalten des Planungssystems II kontrolliert simuliert werden. Das bedeutet, dass das Planungssystem II von der zentralen Plattform getrennt und der Testsimulator eingefügt wird. Inhaltlich wird der Testsimulator bei der nachfolgenden Beschreibung der Testphase IV näher erläutert.

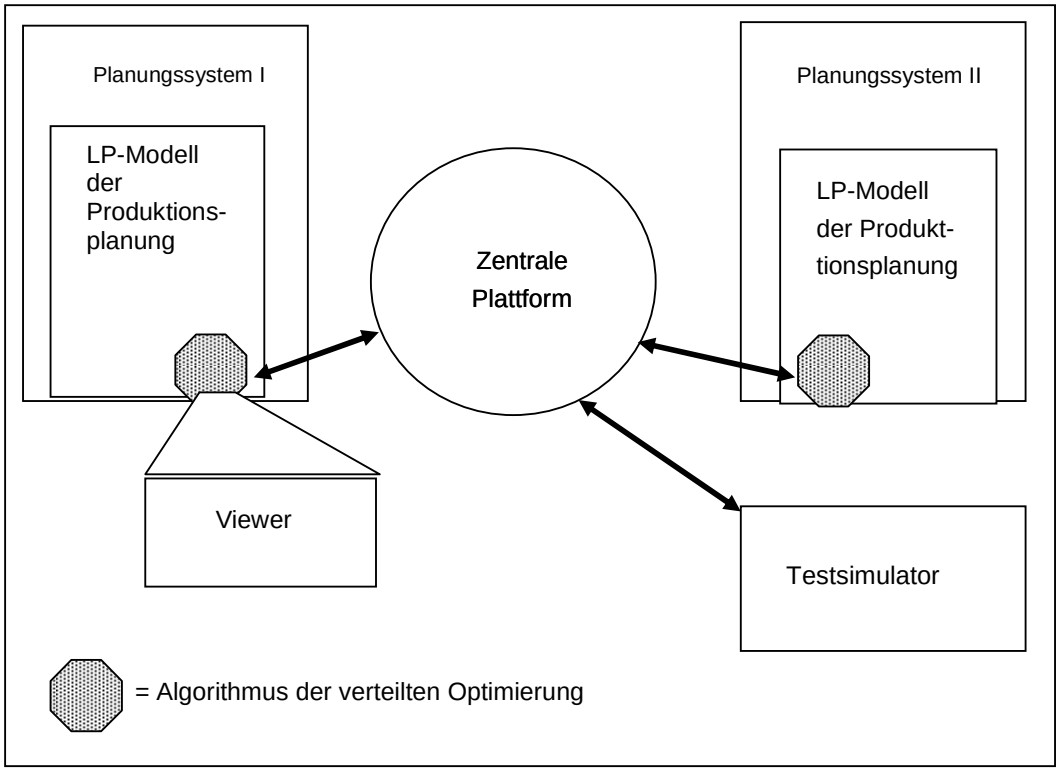


Abbildung 12 – Implementierung eines Viewers und eines Testsimulators zur komponentenweisen Austestung

Zu Phase I - Testen der Funktionsfähigkeit der beteiligten LP-Systeme:

Bevor die LP-Systeme in die Architektur der verteilten Optimierung sinnvoll eingebunden werden können, ist sicherzustellen, dass die beteiligten LP-Modelle im autarken Fall zu einer lokal optimalen, feasiblen Lösung kommen. Dazu sollten folgende Faktoren erfüllt sein:

- Die autarke Optimierung kommt innerhalb von maximal 20 Iterationen zur Lösung. Eine höhere Anzahl an Iterationen würde bei den verbreiteten Modellen auf einen zu kleinen Lösungsraum schließen lassen. Dieser kleine Lösungsraum würde im Rahmen der verteilten Optimierung nochmals eingeschränkt, so dass die Anzahl der zur Lösung des Systems notwendigen Iterationen nicht mehr realistisch handhabbar sind.

- Die Zielfunktion der in dieser Arbeit betrachteten Modellen stellte immer den zu maximierenden Gesamtdeckungsbeitrag dar. In der autarken Lösung der beteiligten Modelle sollten die Ergebnisse (hier: Deckungsbeiträge oder DB) in einem realistischen Rahmen sein. So kann es durch die ungünstige Wahl der in das Modell einfließenden Parameter der Fall sein, dass Gleichungen entstehen, die einen stark nichtlinearen Charakter aufweisen. Auch ausgereifte Solver können in diesem Fall ein lokales Optimum als Lösung ausweisen. Der Einsatz eines derartigen LP-Modells in die Architektur der verteilten Optimierung kann zu stark verfälschten Ergebnissen führen. In den durchgeführten Tests zeigte sich in einem derartigen Fall, dass aufgrund der veränderten Startlösung im Falle der verteilten Optimierung kein lokales Optimum mehr auftrat. So wurde der verteilt optimierte Fall grundsätzlich deutlich besser, als der autarke Fall, was zu groben Fehlern bei der Berechnung des „Delta-DBs“ führte. Eine korrekte Berechnung des zum Basisfall wirtschaftlich sinnvollen Austausches der optimierten Komponente kann nicht erfolgen.

- Aus den Tests mit den zunächst autark optimierenden LP-Modellen zeigte sich ebenfalls, dass die Variablen, die im Rahmen der verteilten Optimierung variiert werden[48], nicht nur modelltheoretisch, sondern auch praktisch alle Werte innerhalb des gesamten Definitionsbereiches annehmen können. Das bedeutet, dass in den beteiligten Modellen in autarken Optimierungen die als minimal bzw. maximal definierte Cyclohexan-Produktion zu Testzwecken erzwungen werden muss. Es ist sicherzustellen, dass diese Grenzen technisch, wenn auch unter Einbußen hinsichtlich des Gesamtdeckungsbeitrages, von der betreffenden Variablen des Modells angenommen werden kann. Gleiches gilt bei den Mengenvariablen Benzolzukauf, Benzol-Eigenproduktion und Benzolverkauf. Beispielhaft zeigte sich in den durchgeführten Tests, dass sowohl im autarken wie im verteilt optimierten Fall trotz augenscheinlich wirtschaftlicher Vorteile in dem betrachteten Modell die rechnerisch zu produzierende Cyclohexanmenge immer den Wert der unteren Grenze annahm. Die Ursache lag jedoch an dem fehlenden Wasserstoff, der für die Cyclohexanherstellung notwendig ist. Eine größere Menge als die Mindestmenge konnte technisch nicht realisiert und das wirtschaftliche erwünschte Optimum nicht erzielt werden. Das Ergebnis der unveränderten Cyclohexan-Produktion hätte ohne den o.g. Test nicht als Problem eines einzelnen LP-Modells erkannt werden können, sondern wäre im ungünstigsten Fall als Ergebnis der verteilten Optimierung ausgewiesen worden.

  So kann als Ergebnis formuliert werden, dass die für die verteilte Optimierung relevanten Variablen hinsichtlich der modellseitig abgebildeten Grenzen in den autarken LP-Modellen systematisch

[48] im dargestellten Beispiel folgende Mengen: Cyclohexan-Produktion, Benzolzukauf, Benzol-Eigenproduktion und Benzolverkauf

auszutesten sind. Es empfiehlt sich eine entsprechende Matrix zu erstellen, wie sie Abbildung 13 zeigt.

Menge in [t/h]		PIMS I	PIMS II
Cyclohexan-produktion	min	0	0
	max	20	10
Benzol-zukauf	min	0	
	max	13	
Benzol-eigenproduktion	min	7	5
	max	10	10
Benzol-verkauf	min	7	5
	max	10	10

Abbildung 13 – Grenzen für die verteilte Optimierung relevanten Variablen

Die Werte in Abbildung 13 zeigen die modellseitig formulierten und zu Beginn dieses Kapitels erwähnten Grenzen. Für die durchzuführenden Tests bedeutet das, dass bei dem LP-System PIMS I in autarker Weise die zu produzierende Cyclohexanmenge in einem ersten Schritt mit 0 t/h und in einem zweiten mit 20 t/h fix vorgegeben wird. Wichtig ist darauf zu achten, dass die Variablen Benzolzukauf, Benzoleigenproduktion und Benzolverkauf die in Abbildung 13 vorgegebenen Grenzen auch entsprechend annehmen können. Sollten sowohl die minimalen als auch die maximalen Grenzen der Cyclohexan-Produktion vom LP-Modell sinnvoll realisierbar sein, werden in gleicher Weise alle relevanten Variablen getestet. Es bleibt darauf hinzuweisen, dass jeweils nur eine der o.g. Variablen fix vorgegeben werden darf, da sonst schnell ein überbestimmtes, nicht mehr lösbares System erzeugt wird.

- Zu den Tests der Phase I zählt auch die Betrachtung der minimalen und maximalen Preise bzw. Erlöse für das zuzukaufende (PIMS I) bzw. zu verkaufende Benzol (PIMS II). Für ein Zustandekommen eines Austausches von Komponenten im Rahmen der verteilten Optimierung ist es u.a. notwendig, dass die Preisspanne des Zukaufsbenzols sich mit der Preisspanne des Verkaufsbenzols in Teilen überschneidet:

  Es soll gelten:

  $$P_{Verkauf\text{-}max} > P_{Zukauf\text{-}min} \qquad (17)$$

  und

  $$P_{Zukauf\text{-}max} > P_{Verkauf\text{-}min} \qquad (18)$$

  mit:

  $P_{Verkauf\text{-}min}$ = minimaler Verkaufserlös

  $P_{Verkauf\text{-}max}$ = maximaler Verkaufserlös

  $P_{Zukauf\text{-}min}$ = minimaler Zukaufspreis

  $P_{Zukauf\text{-}max}$ = maximaler Zukaufspreis

  Die Grenzen für die genannten Erlöse können zwar modellseitig vorgegeben werden, **müssen aber nicht zwingend mit den vom LP-System realisierbaren Grenzen übereinstimmen**. Das bedeutet, dass z.B. modellseitig ein maximaler Zukaufpreis vorgegeben wurde, dieser allerdings vom LP-System niemals ausgenutzt wird, da die Unterlassensalternative, d.h. das Benzol nicht in dem betreffenden Umfang zuzukaufen, wirtschaftlich sinnvoller und/oder technisch nicht anders realisierbar ist. Es ist somit bei den beteiligten autarken Modellen unbedingt folgendes auszutesten:

PIMS I:
Wo liegt der maximale Zukaufspreis von Benzol, bei dem das LP-System kein Benzol mehr zukauft ?

PIMS II:
Wo liegt der minimale Verkaufspreis von Benzol, bei dem das LP-System kein Benzol mehr verkauft ?

Die aus den beiden genannten Testzuständen resultierenden Deckungsbeiträge der autarken LP-Systeme können von denen der frei optimierten Deckungsbeiträge der einzelnen LP-System stark abweichen.

Bei der Durchführung der aufgezeigten genannten Tests lassen sich die Ergebnisse eines einzelnen Testlaufes anhand des nach Beendigung der Optimierung standardmäßig erstellten Reports beurteilen. Es können allerdings nur die im Endergebnis erzielten und im Report dargestellten Werte der relevanten Variablen geprüft werden. Im Rahmen dieser Arbeit erwies es sich als sinnvoll, das vorhandene autark arbeitende LP-System um eine hier als **Viewer** definierte Eigenentwicklung zu erweitern.

Der im Rahmen dieser Arbeit entwickelte Viewer hat auf das LP-System und auf die damit verbundene Optimierung keinen Einfluss. Der Viewer dient lediglich dazu, die Werte definierter Variablen nach jeder Iteration während der Optimierung zu erfassen und zu archivieren. Realisiert wurde dies mit Hilfe eines auf der Programmiersprache VISUAL BASIC basierenden Programms, welches in das LP-Modell implementiert wurde und das während der Optimierung nach jeder Iteration aufgerufen wird. Das bedeutet, dass bei den durchgeführten Tests nicht nur das Endergebnis selbst, sondern auch die Qualität dieses Ergebnisses beurteilt werden kann. Als Qualität eines Ergebnisses sind in diesem Zusammenhang die Stabilität und die Konvergenz der ausgewiesenen Wertereihen der betrachteten Variablen zu verstehen. So ist

Voraussetzung für die verteilte Optimierung, dass die Ergebnisse der durchgeführten Tests der autarken LP-Systeme einen konvergierenden Verlauf zeigen. Ein schwingendes System, d.h. eine periodische Veränderung der Ergebnisse, ist als nicht stabil einzustufen und Indiz dafür, dass der Solver nicht die gewünschte Zielfunktion optimal berechnen konnte. Moderne LP-Systeme versuchen zwar eine derartig unerwünschte Lösung des Modells zu vermeiden, sind andererseits aber auch getrieben eine feasible Lösung zu ermitteln. Aufgrund des im Rahmen dieser Arbeit eingesetzten Viewers lässt sich die Empfehlung formulieren, die Wertereihen der Variablen im Gesamtzusammenhang der Optimierung zu prüfen, bevor ein LP-System in ein System der verteilten Optimierung implementiert wird.

Zusammenfassend lässt sich formulieren, dass bei den an der verteilten Optimierung beteiligten LP-Systeme auf autarker Basis

- die Anzahl der Iterationen,
- die Höhe des Deckungsbeitrages,
- die definierten Grenzen der betrachteten Variablen
- und das Lösungsverhalten hinsichtlich Konvergenz

unter Einsatz eines entwickelten Viewers zu prüfen sind.

Zu Phase II – Verifizieren des korrekten Datenaustausches:

Nachdem die an der verteilten Optimierung beteiligten LP-Systeme im autarken Betrieb in der Phase I getestet wurden, sollen in der Phase II die im Rahmen dieser Arbeit durchgeführten Untersuchungen und die daraus gewonnenen Ergebnisse hinsichtlich des korrekten Datenaustausches dargestellt werden.

Im Speziellen wurde der Datenaustausch zwischen den beteiligten Komponenten einer verteilten Optimierung betrachtet. Mit Verweis auf

Abbildung 10 und Abbildung 11 lassen sich folgende Schnittstellen definieren, die einen zu prüfenden Datenaustausch aufweisen:

- Tabellen mit den Eingangsdaten ↔ PIMS I bzw. PIMS II
- PIMS I bzw. PIMS II ↔ zentrale Plattform
- PIMS I bzw. PIMS II ↔ Report

Als hilfreich hat sich bei den durchgeführten Tests der Einsatz des bereits in Phase I beschriebenen Viewers erwiesen. Mit Hilfe des Viewers, der die errechneten Werte relevanter Variablen zwischen den einzelnen Iterationen ausweist, lässt sich nach einer Optimierung verifizieren, ob ein korrekter Datenaustausch mit dem Modell stattgefunden hat. Alle anderen an der Optimierung beteiligten Komponenten (Eingangstabellen, zentrale Plattform und der Report) sind im Gegensatz zum LP-Modell unmittelbar einsehbar. Die Form der durchgeführten Tests wird nachfolgend beschrieben.

Der Transfer der Modelleingangsdaten in die beiden LP-Modelle PIMS I und PIMS II erfolgte in den untersuchten Modellen vom Standardalgorithmus der LP-Modelle. Es wurden lediglich die in einer Tabellenform vorliegenden Daten in die notwendige Matrixstruktur umgebrochen. Dieser Datentransfer gilt als Routine des LP-Systems und wird in diesem Zusammenhang als unkritisch angesehen. Es war lediglich zu prüfen, ob die Eingangsdaten sich in der korrekten Weise im Modell wiederfinden und damit sowohl der Wert als auch die Variablenbezeichnung korrekt übernommen wurden.

Hauptaugenmerk in der Phase II der Tests sollte auf die Verifizierung des korrekten Datenaustausches zwischen den LP-Modellen und der zentralen Plattform gelegt werden. Als LP-Modell ist in diesem Zusammenhang das um den in Kapitel 7.2.2 beschriebenen Algorithmus erweiterte System zu verstehen. Es wurden folgende Test durchgeführt:

- Wie bereits beschrieben, ist für das Zustandekommen einer verteilten Optimierung von einem der beteiligten LP-Systeme ein Datensatz auf der zentralen Plattform zu hinterlegen. Zu diesem Zwecke ist die verteilte Optimierung in einem LP-System zu starten - das zweite LP-System wird deaktiviert. Das hat zur Folge, dass das aktivierte LP-System noch vor der ersten Optimierung den Startdatensatz in die zentrale Plattform kopiert, den Trigger auf den Wert 1 setzt und in eine Warteschleife geht, solange der Trigger nicht den Wert 0 annimmt. Aufgrund des deaktivierten zweiten LP-Systems ist nun ausreichend Zeit, die zentrale Plattform zu öffnen und die ausgetauschten Daten zu überprüfen.

- Ist der Eingangsdatensatz verifiziert, so kann bei weiter deaktiviertem PIMS II der Trigger in der zentralen Plattform manuell auf den Wert 0 gesetzt werden. Es wird so der zum zweiten LP-System hergestellte Kontakt simuliert. Das hat zur Folge, dass das erste LP-System am Ende einer Warteschleife erkennt, dass der Datensatz geändert worden ist, d.h. die nächste Iteration der Optimierung vom LP-System durchgeführt wurde. Der nach erfolgter Iteration in der zentralen Plattform hinterlegte Datensatz kann wiederum manuell aufgerufen und geprüft werden. Parallel sind die Werte der zentralen Plattform mit den Werten aus dem Viewer zu vergleichen. Nur so kann geprüft werden, ob die während der Iteration im Modell berechneten Werte auch korrekt in die zentrale Plattform exportiert wurden. Es ist zu prüfen, ob die richtigen Datensätze ausgetauscht wurden sind und ob die ausgetauschten Datensätze vollständig und korrekt sind. Diese zuletzt beschriebenen Schritte des Tests haben sich bei den empirischen Untersuchungen als sehr sinnvoll für einen korrekten Einsatz der verteilten Optimierung herausgestellt, da ein Fehler im Datentransfer ggf. nur unauffällige Veränderungen des Ergebnisses zur Folge haben kann und nur schwer zu identifizieren ist.

  Schließlich wurde in der Testphase II die Zugriffszeit vom LP-Modell auf die zentrale Plattform untersucht. Die maximale Zugriffszeit sollte nicht

länger als 10 Sekunden betragen. Unter Zugriffszeit ist dabei die Zeit vom Öffnen der zentralen Plattform bis zur Vollendung des Schließens inklusive des notwendigen Speicherns zu verstehen. Eine längere Zugriffszeit sollte überdacht werden, da eine Erhöhung zur Verlängerung der Gesamtlaufzeit führt. Es ist zu berücksichtigen, dass die Verlängerung der Zugriffszeit bei jedem Datentransfer zur zentralen Plattform mit der Anzahl der zur Optimierung notwendigen Iterationen zu multiplizieren ist.

- Die letzte betrachtete Schnittstelle zwischen LP-Modell und Report ist ähnlich der zu Beginn beschriebenen Schnittstelle zwischen Eingangsdaten und LP-System eine Routine des LP-Systems und wurde hinsichtlich eines fehlerhaften Datentransfers als unkritisch eingestuft. Es wurde lediglich überprüft, ob die im Viewer aus der letzten Iteration gespeicherten Werte betrachteter Variablen auch im erzeugten Report ausgewiesen wurden.

Zusammenfassend lässt sich festhalten, dass der Datentransfer an drei wesentlichen Schnittstellen hinsichtlich

- der korrekten Übernahme der Eingangsdaten in die Matrix,
- des Hinterlegens des ersten Datensatzes auf der zentralen Plattform
- sowie des korrekten Datenaustausches mit der zentralen Plattform zwischen den Iterationen unter Einhaltung der definierten maximalen Zugriffszeiten und
- des Erzeugens eines korrekten Reports

unter Einsatz eines Viewers verifiziert wurde.

Zu Phase III – Testen des korrekten Ablaufes einer vollständigen Optimierung:

In der dritten Phase des Testablaufes wurde der korrekte Ablauf einer vollständigen Optimierung (wie in Kapitel 7.2.1 beschrieben) unter der

speziellen Beachtung der im Algorithmus genannten Start- und Abbruchbedingungen untersucht.

Basis dieser Testphase waren die zu Beginn dieses Kapitels beschriebenen LP-Systeme. Auf das definierte Preis-Erlös-Szenario wurde ebenfalls eingegangen.

Zur Durchführung des Tests wurde das LP-System I mit dem Modell PIMS I auf einem der mit dem LAN-Netzwerk verbundenen Rechner gestartet. Analog des Algorithmus zur verteilten Optimierung sollte im Test mit Hilfe des Viewers verfolgt werden, wie der erste Datensatz auf der zentralen Plattform hinterlegt wird. Die inhaltliche Prüfung des hinterlegten Datensatzes wurde in den vorhergehenden Testphasen absolviert und war an dieser Stelle nicht mehr nötig, so dass sich diese Phase des Tests ausschließlich auf den reinen Ablauf konzentrierte.
Wurde der erste Datensatz auf der zentralen Plattform hinterlegt, so war in einem nächsten Schritt zu prüfen, ob und wann das LP-System II mit dem Modell PIMS II den Datensatz von der zentralen Plattform in das LP-System kopieren würde. Bei den durchgeführten Tests zeigte sich dafür eine in der zentralen Plattform programmierte Routine als hilfreich. Diese auf VISUAL BASIC programmierte Routine schrieb bei jedem Öffnen der zentralen Plattform die Zeit (Datum und Uhrzeit) sekundengenau in eine Tabelle. So konnte der zeitliche Verlauf der Zugriffe auf die zentrale Plattform nach Beendigung der verteilten Optimierung nachvollzogen werden.

Ferner galt zu sicherzustellen, dass das LP-System II, auch wenn noch kein zu kopierender Datensatz auf der zentralen Plattform hinterlegt wurde, den beschriebenen Algorithmus weiterverfolgt, indem es die Warteschleife fortsetzt. Ein weiterer auf Auswirkungen zu prüfender Fall war der zeitgleiche Zugriff beider LP-Systeme auf die zentrale Plattform. Zwar wurde dieser Aspekt in dem Algorithmus zur verteilten Optimierung berücksichtigt, führte allerdings in der Praxis in wenigen Fällen zum unkontrollierten Abbruch des

Systems. Aus dieser Erfahrung heraus lässt sich die Empfehlung einer datenbankgestützten zentralen Plattform wiederholen.

Hat das LP-System II den auf der zentralen Plattform hinterlegten Datensatz korrekt ins LP-Modell II kopiert, bleibt zu untersuchen, ob das LP-System II mit der ersten Rekursion der Optimierung beginnt und nach Abschluss dieser das notwendige Simulating Interface startet, welches das Auslesen eines neuen Datensatzes aus der Matrix und das anschließende Kopieren in die zentrale Plattform zur Folge hat. Der Verlauf der Rekursionen wird auf dem Monitor während der Optimierung dargestellt. Wie bereits beschrieben, wurden die an der verteilten Optimierung beteiligten LP-Systeme auf getrennten Computern mit LAN-Verbindung betrieben. Für diese Testphase zeigt es sich als zweckmäßig, die Monitore der beiden Computer nebeneinander zu installieren. Bei gleichzeitiger Betrachtung beider Monitore konnte so das Wechselspiel der LP-Systeme an den Iterations-Report in Echtzeit verfolgt werden.

Sind die beschriebenen Kriterien für die Beendigung der verteilten Optimierung erfüllt, ist sicherzustellen, dass beide LP-Systeme die Optimierung nach der Durchführung direkt aufeinanderfolgender Rekursionen beenden und jeweils einen korrekten Report erstellen. Es war im Speziellen zu prüfen, ob die verknüpften LP-Systeme zu einem kontrollierten Abschluss der Optimierung kommen, auch wenn eines der beiden Systeme nicht regelgerecht arbeitete. Darunter waren z.B. Systemausfall eines der beiden Computer, Unterbrechung der LAN-Verbindung, Ausfall der zentralen Plattform und unvorhergesehener Abbruch der Optimierung durch eine Störung im LP-Modell zu verstehen.

Zusammenfassend lassen sich folgende in der Testphase III zu prüfende Elemente aufzeigen:

- Hinterlegen eines ersten vollständigen Datensatzes auf die zentrale Plattform unter Einsatz eines Viewers.
- Verfolgen der wechselseitigen Rekursion der LP-Systeme

- Sicherstellen der korrekten Umsetzung der definierten Abbruchbedingungen
- Erstellen zweier inhaltlich und formal korrekter Reports
- Reaktion der LP-Systeme bei technischen Problemen während der Optimierung.

Zur Phase IV – Verhalten der beteiligten Solver:

In der vierten Phase der Testreihen standen die Untersuchung des Verhaltens der beteiligten Solver im Rahmen der verteilten Optimierung sowie die Ermittlung und Darstellung der sich daraus ergebenden Möglichkeiten und vor allem Grenzen des in dieser Arbeit erarbeiteten Ansatzes im Mittelpunkt.

Die Notwendigkeit zur Untersuchung des Zusammenspiels zwischen den beteiligten Solvern lässt sich wie folgt begründen:

Wie in Kapitel 2 erörtert, ist in dem beschriebenen Anwendungsfall ein LP-Modell ein abstrahiertes Abbild der zu optimierenden Realität. Die für das Optimierproblem relevanten Möglichkeiten und Grenzen werden in Form von Gleichungen mit entsprechenden Variablen und Konstanten formuliert. Jede zu optimierende Eigenschaft wird im Allgemeinen in einer eigenständigen Gleichung dargestellt. Diese Gleichung findet im Modell in der vor jeder Gesamtoptimierung erzeugten Matrix eine definierten Koordinate. Das bedeutet, jeder modellierten Eigenschaft steht ein Eintrag in der Matrix gegenüber. Die Einträge in der Matrix können sowohl Variablen als auch Konstanten sein. Die Variablen werden während der Optimierung eines autarken LP-Modells zum Zweck der Maximierung der Zielfunktion verändert, die Konstanten nicht. Die Funktionsverläufe der abgebildeten Eigenschaften entsprechen den bereits beschriebenen Anforderung eines LP-Modells.

Diese Anforderungen an die Gleichungen eines LP-Modells lassen sich bei der verteilten Optimierung nur stark eingeschränkt einhalten. Entsprechende

Probleme ergeben sich bei dem Einsatz des systeminternen Solvers. Aus diesem Grund steht eine Testreihe zum Verhalten des Solvers bei unterschiedlichen Funktionsverläufen im Mittelpunkt der letzten Testphase.

Bevor der Inhalt der Testphase IV näher beschrieben wird, soll vorbereitend theoretisch auf das Problem näher eingegangen werden.

Aus der Sicht des LP-Systems I wird bei Einsatz der verteilten Optimierung die Zielfunktion mit Hilfe des Solvers auf Extremstellen (in diesem Fall auf ein Maximum) untersucht. Soll über eine Eigenschaft, wie z.B. der Schwefelgehalt der ausgetauschten Komponente, optimiert werden, so zeigte sich in durchgeführten Tests, dass der funktionale Zusammenhang zwischen Eigenschaft und Deckungsbeitragsänderung in den seltensten Fällen linear ist. Vielmehr ergab eine nähere Betrachtung der Deckungsbeitragsfunktion, die im Rahmen dieser Arbeit bei einem stark simplifizierten LP-Modell untersucht wurde, dass die Funktion des Gesamtdeckungsbeitrages in Abhängigkeit der variierten Eigenschaften als Überlagerung mehrerer Einzelfunktionen definiert werden kann. Die Einzelfunktionen werden durch Intervallgrenzen voneinander getrennt. Bezieht sich dieser Zusammenhang nur auf ein autarkes LP-Modell, sind die standardmäßig in den LP-Systemen implementierten Solver ausreichend, um einen derartigen funktionalen Zusammenhang verarbeiten und die gewünschte Extremstelle ermitteln zu können.

Bei einer verteilten Optimierung hingegen wird die Deckungsbeitragsfunktion des LP-Systems II in Abhängigkeit einer Eigenschaft in ihrer gesamten Komplexität zu einer einzigen Eigenschaft und damit nur auf einem einzigen Eintrag in der Matrix des LP-Systems I reduziert (Abbildung 14).

Die übrigen funktionalen Beziehungen der Eigenschaften im LP-System II bleiben bestehen, so dass die zuvor in autarkem Zustand vom Solver lösbare Funktion in der verteilten Optimierung nur noch bedingt lösbar ist.

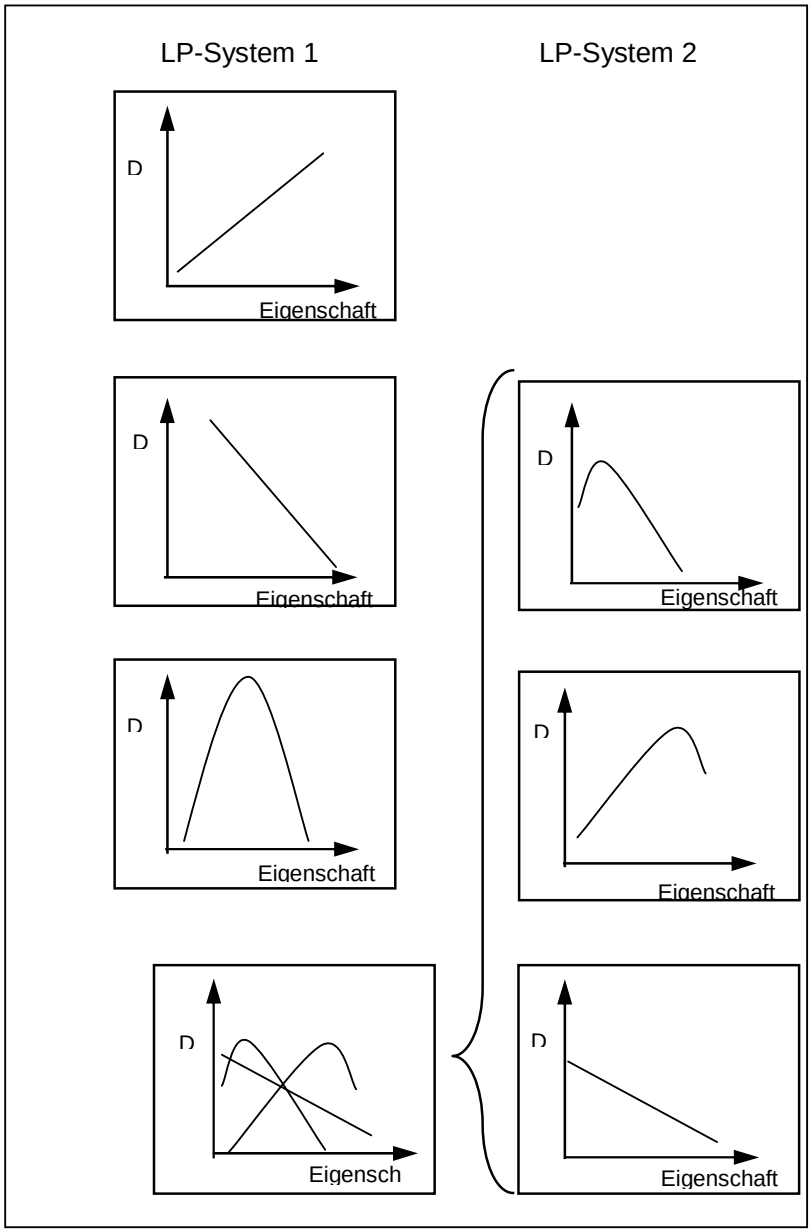


Abbildung 14 – Zusammenfassung mehrerer Eigenschaften im LP-System II auf einen Eintrag in der Matrix des LP-Systems I

Da die Möglichkeiten eines gängigen Solvers wesentlichen Einfluss auf die Umsetzbarkeit des Ansatzes haben, ist es wichtig, Kenntnisse über das Verhalten des Solvers beim Einsatz unterschiedlicher Funktionsverläufe zu erlangen. Die Möglichkeiten und Grenzen des Solvers haben wesentlichen Einfluss auf die Möglichkeiten und Grenzen der verteilten Optimierung. Aus diesem Grund wurden im Rahmen dieser Arbeit Tests zu den Möglichkeiten

und Grenzen des Solvers durchgeführt. Der eingesetzte Solver ist Teil des LP-Systems PIMS der Firma ASPENTECH.

Als Hilfsmittel wurde dazu im Rahmen dieser Arbeit ein vereinfachter Testsimulator entwickelt, der mit dem Hinweis auf Abbildung 12 nachfolgend kurz erörtert werden soll.

Ziel des entwickelten Testsimulators ist es, komplexe Funktionsverläufe resultierend aus dem LP-Systems II zu simulieren und das LP-System II während der Tests zu ersetzen. Das bedeutet, dass über der zentralen Plattform zwischen den einzelnen Rekursionen nicht die Daten mit dem LP-System II ausgetauscht werden, sondern mit dem Testsimulator.

Für den technischen Aufbau des Testsimulators bedeutet das, dass der Teil des entwickelten Algorithmus zur verteilten Optimierung, der das LP-System II betrifft, auf den Testsimulator angepasst wird. So wird der Teil des Algorithmus modifiziert, der den Datenaustausch zwischen der zentralen Plattform und dem LP-System II regelt[49]. Anstelle der LP-typischen Rekursionen tritt schließlich die zu testende Funktion, jedoch ohne Rekursion, lediglich als Zusammenhang zwischen der untersuchten Eigenschaft und der Deckungsbeitragsveränderung. Wie bei dem entwickelten Viewer zeigte sich in diesem Teil der Testphase IV ebenfalls eine Visualisierung der Zwischenergebnisse zwischen den durchgeführten Rekursionen des LP-Systems I als hilfreich. Reaktionen des Solvers auf verschiedene Funktionsverläufe ließen sich so grafisch darstellen. Für die spätere Auswertung der Versuche, wurden die Zwischenergebnisse im Testsimulator über den gesamten Versuchsverlauf gespeichert.

Die Funktionsverläufe, die den Zusammenhang zwischen der im Rahmen der verteilten Optimierung betrachteten Eigenschaft und der Deckungsbeitragsveränderung darstellten, waren somit nicht mehr Resultat

[49] siehe dazu nochmals Anhang 1

eines komplexen LP-Systems. Vielmehr konnten mittels des Testsimulators mathematische Funktionen vorgegeben und zuvor manuell auf Extremstellen untersucht werden. Die Vorgehensweise gliederte sich in folgende Schritte: In einem ersten Schritt wurden Funktionsverläufe von unterschiedlicher Komplexität erstellt und auf Extremstellen untersucht. Im nächsten Schritt wurden diese Funktionen in den Testsimulator implementiert, das LP-System II durch den Testsimulator ersetzt und der Algorithmus der verteilten Optimierung angewandt.

Die durchgeführten Tests zeigten hinsichtlich der Auswertbarkeit der eingesetzten Funktionsverläufe kein zufriedenstellendes Ergebnis. Ursache dafür war die Überlagerung der eingebrachten Funktionsverläufe mit den bestehenden Funktionen des LP-Systems I. Zwar konnte in der Regel eine maximale Extremstelle in der Gesamtzielfunktion des LP-Systems I vom Solver gefunden werden, allerdings war es nicht möglich Rückschlüsse darauf zu ziehen, ob das zuvor errechnete Maximum der Funktion im Testsimulator auch wirklich vom Solver des LP-Systems I korrekt gelöst wurde.

Die Frage, ob der eingesetzte Solver geeignet war komplexe Funktionen korrekt auf Extremstellen zu untersuchen, blieb somit unbeantwortet.

Der in Abbildung 12 dargestellte Testaufbau musste nochmals vereinfacht werden. Zu diesem Zweck wurden die Funktionen des LP-Systems I (siehe dazu auch Abbildung 14) durch die einzige Funktion

$$f(e_i) = K \qquad mit\ \mathbb{D}(e_i) \in \mathbb{R} \quad und\ e_i = Eigenschaften\ im\ LP\text{-}System$$

$$wobei\ K = const. \qquad mit\ \mathbb{D}(K) \in \mathbb{R}^+$$

ersetzt (Abbildung 15). Somit hat eine Veränderung der Eigenschaften im LP-System I keinen Einfluss auf die Zielfunktion – der zu maximierende Deckungsbeitrag im LP-System I bleibt im autarken Betrieb immer gleich.

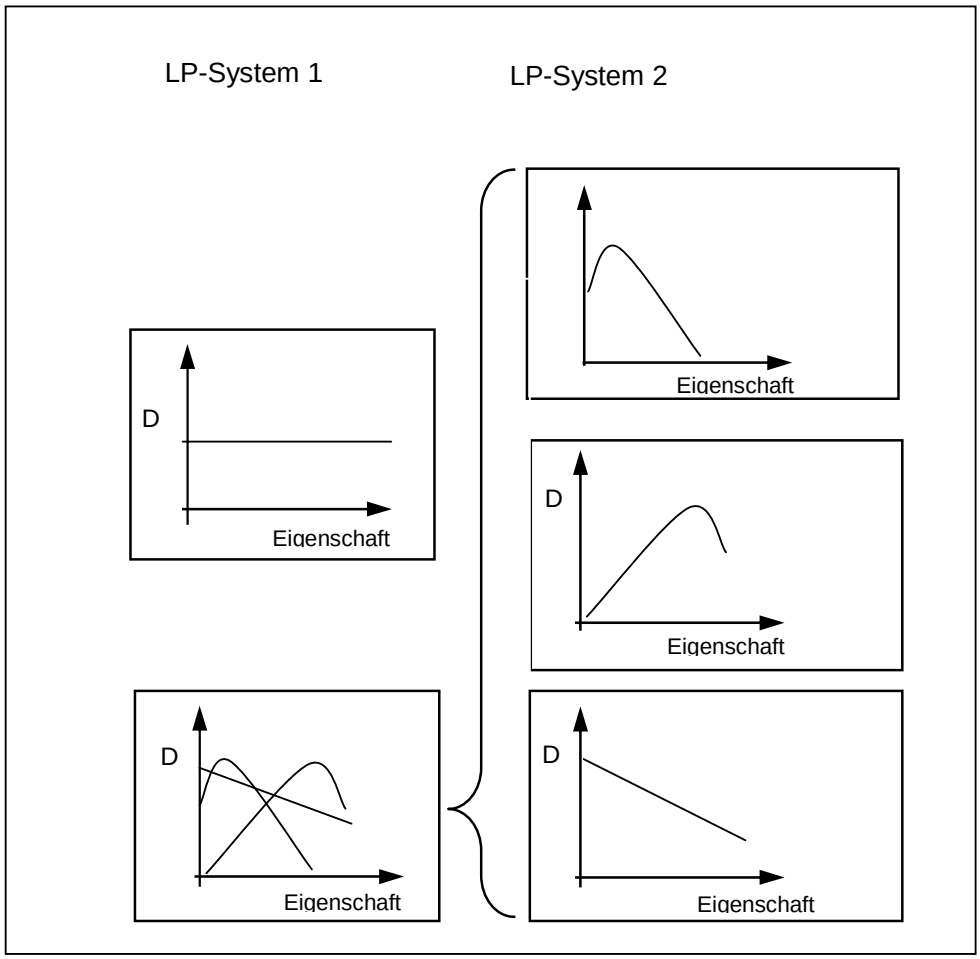


Abbildung 15 – zu Testzwecken reduzierte Form der Abbildung 14

Wird nun im Rahmen der Testphase IV wie zuvor eine verteilte Optimierung unter Berücksichtigung des beschriebenen Testsimulators durchgeführt, können nur die Funktionsverläufe des Testsimulators Einfluss auf das Ergebnis der verteilten Optimierung nehmen. Mit Hilfe des errechneten Ergebnisses der Gesamtoptimierung lassen sich die notwendigen Rückschlüsse auf die

Funktionsweise des eingesetzten Solvers ziehen, d.h. darauf, ob und bei welchen Funktionen die korrekte maximale Extremstelle ermittelt werden konnte.

Die Funktionen für den Testsimulator lassen sich in zwei Gruppen teilen. Zum einen in die Gruppe der Funktionen ohne Intervallgrenzen und zum zweiten in die Gruppe der Funktionen aus zusammengesetzten Funktionen mit Intervallgrenzen zwischen den Einzelfunktionen.

Bei den Funktionen ohne Intervallgrenzen wurden folgende Funktionstypen untersucht:

I. Ganzrationale Funktionen 1. Grades
   Allgemeine Funktion: $f(x) = a_0x + b$ $\quad$ *mit* $a_i \neq 0$ *und* $i = 0$

II. Ganzrationale Funktionen 2. Grades
   Allgemeine Funktion: $f(x) = a_1x^2 + a_0x + b$ $\quad$ *mit* $a_i \neq 0$ *und* $i = 0,1$

III. Ganzrationale Funktionen 3. Grades
   Allgemeine Funktion: $f(x) = a_2x^3 + a_1x^2 + a_0x + b$ *mit* $a_i \neq 0$ *und* $i = 0, 1, 2$

IV. Exponentialfunktionen

Um den Ursache-Wirkungs-Zusammenhang möglichst transparent zu halten, wurden im ersten Schritt einfache mathematische Funktionen ausgewählt.

Als Beispiel einer ganzrationalen Funktion 1. Grades diente folgende Funktion:

$$f(x) = 5x + 10 \qquad \text{für } x \in [1;5]$$

Bei der Implementierung dieser Funktion in beschriebener Weise als Ersatz für das LP-System II konnte die korrekte Extremstelle sicher vom Solver ermittelt werden (siehe Abbildung 16).

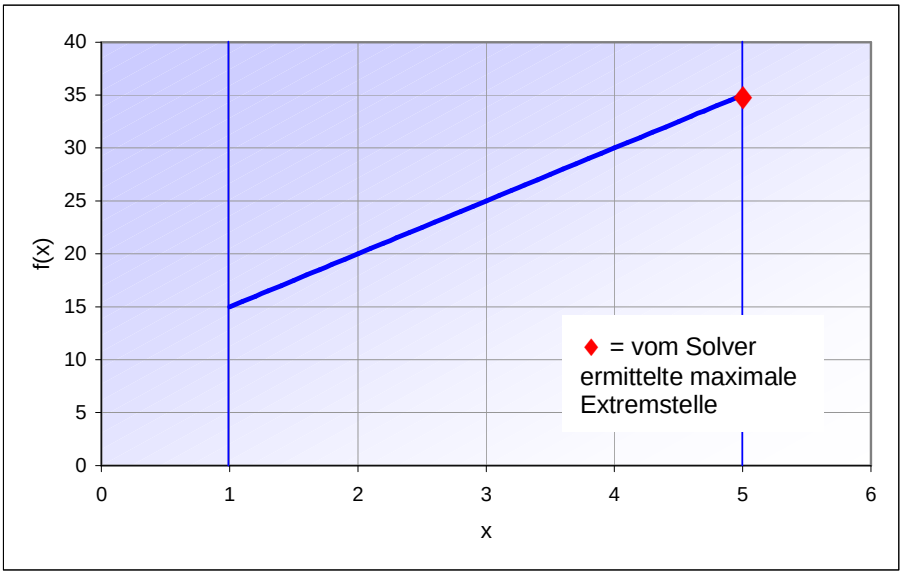


Abbildung 16 – Beispiel einer ganzrationalen Funktionen 1. Grades

Als Beispiel einer ganzrationalen Funktion 2. Grades wurde folgende Funktion verwendet:

$$f(x) = (x-3)^2 + 1 \qquad \text{für } x \in [1{,}5;5]$$

Vor dem Starten des Solvers muss die Eigenschaft einen Startwert besitzen. Der in diesem Beispiel definierten Eigenschaft (x) muss ebenfalls ein Startwert manuell vorgegeben werden. Als ein wesentliches Ergebnis aus den Untersuchungen ergab sich, dass die Wahl des Startwertes in diesem Beispiel einen wesentlichen Einfluss auf das Ergebnis hat. Unter Einsatz des Viewers ließ sich Folgendes erkennen (siehe Abbildung 17):

- Wenn der Startwert x_{start} der Eigenschaft x kleiner als x_{min} ist, entspricht das vom Solver ermittelte Maximum $f_{opt}(x)$ der unteren Intervallgrenze $f(x_u)$.

  d.h.: *wenn* $x_{start} < x_{min}$ *dann gilt* $f_{opt}(x) = f(x_u)$

- Wenn der Startwert x_{start} der Eigenschaft x größer als x_{min} ist, entspricht das vom Solver ermittelte Maximum $f_{opt}(x)$ der oberen Intervallgrenze $f(x_o)$.

  d.h.: *wenn* $x_{start} > x_{min}$ *dann gilt* $f_{opt}(x) = f(x_o)$

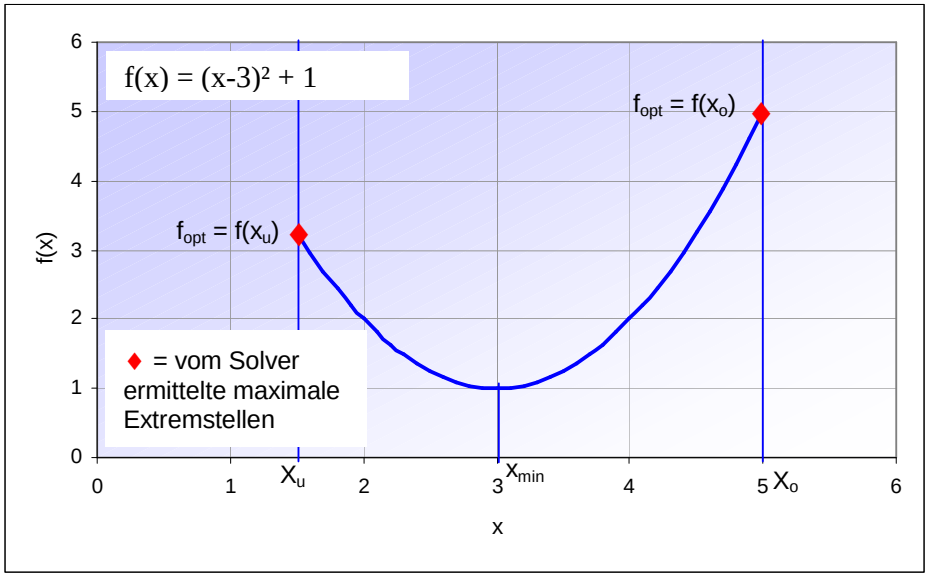


Abbildung 17 - Beispiel A einer ganzrationalen Funktion 2. Grades

Legt man bei den Tests hingegen folgendes Beispiel einer ganzrationalen en Funktion 2. Grades zugrunde, kann die maximale Extremstelle vom Solver korrekt bestätigt werden (siehe Abbildung 18):

$f(x) = -(x-3)^2 + 5$ *für* $x \in [1,5;5]$

Das bedeutet: $f_{opt}(x) = f(x_{max})$.

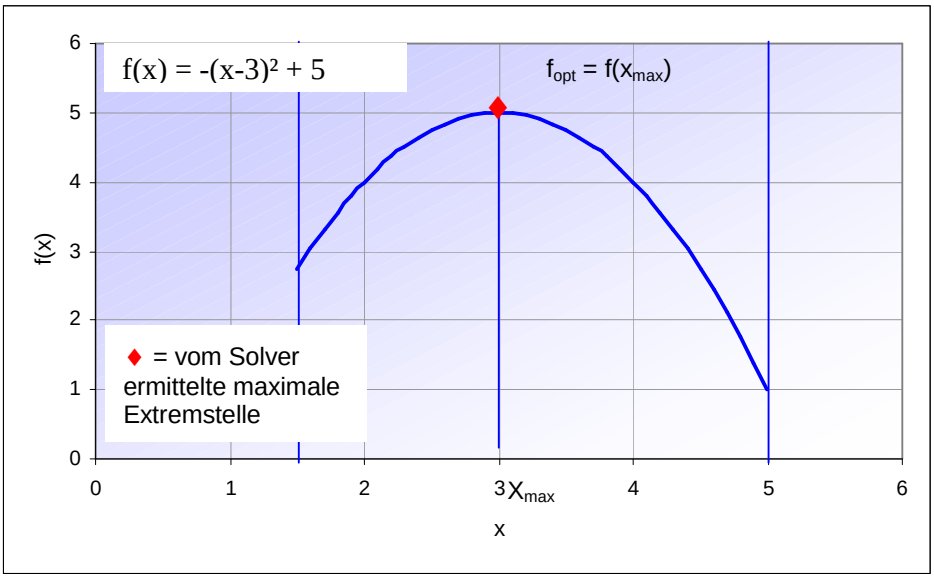


Abbildung 18 – Beispiel B einer ganzrationalen Funktionen 2. Grades

Hinsichtlich der ganzrationalen en Funktionen 2. lässt sich somit das Ergebnis formulieren, dass die Bestimmung der maximalen Extremstelle durch den Solver bei einer ganzrationalen en Funktion 2. Grades nur korrekt erfolgt, wenn der Funktionswert der Extremstelle größer ist als beide Funktionswerte der Intervallgrenzen. Dies stellt eine wesentliche Restriktion im Einsatz der verteilten Optimierung dar.

Bei der Überprüfung des Solververhaltens im Rahmen der Bestimmung der maximalen Extremstelle bei ganzrationalen Funktionen 3. Grades zeigte sich ein Ergebnis, welches im Wesentlichen dem der Tests bei ganzrationalen Funktionen 2. Grades entsprach. Die Beispielfunktion lautete:

$$f(x) = x^3 - 6x^2 + 11x - 5 \qquad \text{für } x \in [0{,}8;3{,}4]$$

Das Testergebnis für diese Funktion lässt sich wie folgt zusammenfassen (siehe dazu
Abbildung 19):

- Wenn der Startwert x_{start} der Eigenschaft x kleiner als x_{min} ist, entspricht das vom Solver ermittelte Maximum $f_{opt}(x)$ einem lokalen, aber nicht dem globalem Optimum.

  d.h.: *wenn* $x_{start} < x_{min}$ *dann gilt* $f_{opt}(x) = f(x_{lokal\ opt})$

- Wenn der Startwert x_{start} der Eigenschaft x größer als x_{min} ist, entspricht das vom Solver ermittelte Maximum $f_{opt}(x)$ der oberen Intervallgrenze $f(x_o)$.

  d.h.: *wenn* $x_{start} > x_{min}$ *dann gilt* $f_{opt}(x) = f(x_o)$

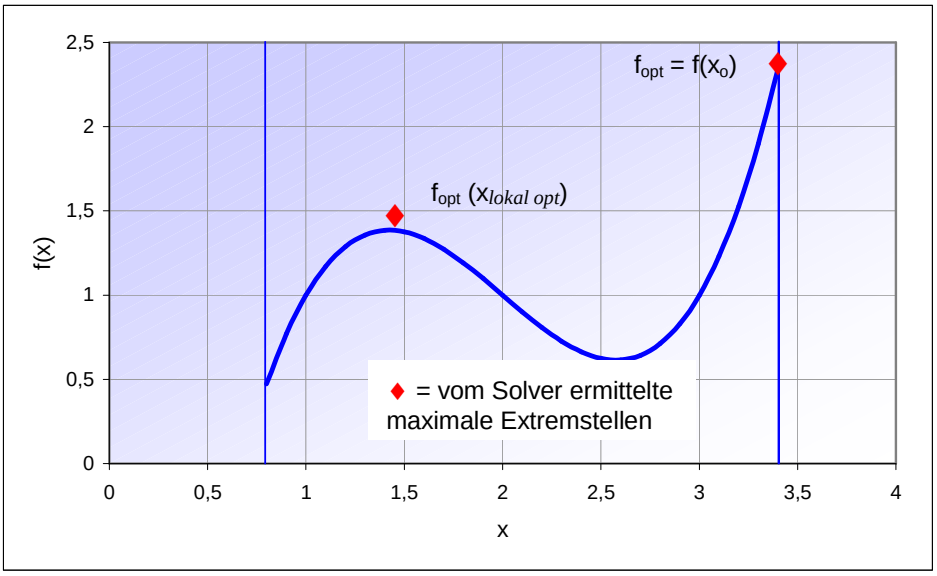


Abbildung 19 - Beispiel B einer ganzrationalen Funktionen 3. Grades

Somit lässt sich wie schon bei den ganzrationalen Funktionen 2. Grades formuliert auch bei den ganzrationalen Funktion 3. Grades erkennen, dass die Bestimmung der maximalen Extremstelle durch den Solver nur korrekt erfolgt, wenn der Funktionswert der Extremstelle größer ist als beide Funktionswerte der Intervallgrenzen. Darüber hinaus ist anzumerken, dass je nach Festlegung des Startwertes bei den ganzrationalen Funktionen 3. Grades in erheblichem Maße das Problem der lokalen Optima deutlich wird. Die Problematik des Solververhaltens bei lokalen Optima und möglichen Lösungsansätzen aus der aktuellen Literatur wird in Kapitel 9 nochmals aufgriffen.

Zur Vervollständigung wurde schließlich das Solververhalten bei Exponentialfunktionen untersucht. Als Beispielfunktion diente dazu (siehe Abbildung 20):

$f(x) = 3^x + 5$ für $x \in [0{,}5;3{,}5]$

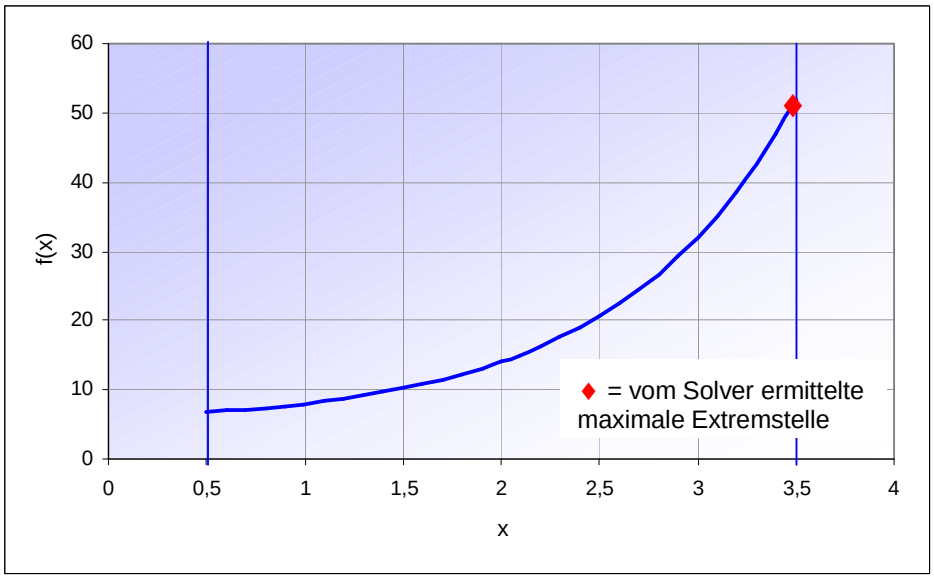


Abbildung 20 - Beispiel einer Exponentialfunktion

Die korrekte Extremstelle im definierten Intervall der Exponentialfunktion konnte vom eingesetzten Solver bestimmt werden. Das Sollverhalten ließ sich aufgrund des fehlenden lokalen Optimums mit dem Lösungsverhalten bei der Anwendung bei ganzrationalen Funktionen 1. Grades vergleichen.

Wie bei der Beschreibung der Abbildung 14 erörtert, zeigte sich bei der näheren Betrachtung des funktionalen Gesamtzusammenhangs - als Überlagerung der Einzelfunktionen des LP-Systems II – diese nicht als eine Funktion entsprechend der vorgestellten Beispielfunktionen I bis IV. Vielmehr konnte ein komplexer Verlauf beobachtet werden, der eher dem Charakter einer zusammengesetzten Funktion entsprach.

So wurden in einem zweiten Schritt zur Steigerung der Komplexität Teile der vier genannten Funktionstypen intervallweise zu Gesamtfunktionen zusammengesetzt und getestet.

Nachfolgend werden repräsentativ aus den durchgeführten Untersuchungen drei charakteristische Funktionen dargestellt, die sich aus den Funktionen I bis IV zusammensetzen. Entsprechend wird das gezeigte Lösungsverhalten des im Rahmen der verteilten Optimierung eingesetzten Solvers dokumentiert.

Testfunktion Typ 1:

Die erste von drei zusammengesetzten Funktionen setzt sich aus drei ganzrationalen Funktionen 2. Grades intervallweise zusammen. Diese Funktionsform zeichnet sich im Wesentlichen durch das Auftreten sowohl einer lokalen als auch einer globalen Extremstelle aus.
Eine den Tests zugrunde gelegte Beispielfunktion des Typs 1 war:

$$f(x)= \begin{cases} -3(x-3)^2 + 7 & \text{für } x \in [1,5;4] \\ 3(x-5)^2 + 1 & \text{für } x \in [4;6] \\ -1,2(x-8)^2 + 9 & \text{für } x \in [6;8,5] \end{cases}$$

Die grafische Darstellung ist wie folgt:

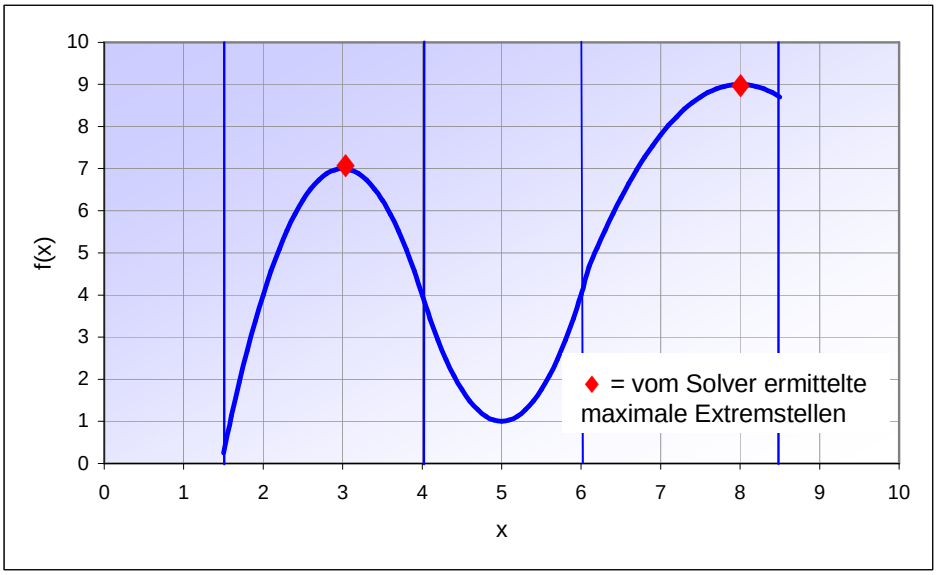


Abbildung 21 – Zusammengesetzte Funktion Typ 1

Bei den mittels Testsimulator durchgeführten Tests der Funktion des Typs 1 zeigte sich, dass je nach Startwert sowohl das absolute als auch das lokale Maximum als maximale Extremstelle vom Solver ausgewiesen wurde.

Eine sichere Bestimmung des absoluten Maximums war im Zusammenhang der verteilten Optimierung nicht möglich.

Testfunktion Typ 2:

Die zweite der drei zusammengesetzten Testfunktionen setzte sich aus einer ganzrationalen Funktionen 0. Grades sowie aus zwei ganzrationalen Funktion 1. Grades zusammen. Diese Funktionsform zeichnet sich durch zwei Funktionsabschnitte mit der Steigung 0 aus, wie sie bei der funktionalen Darstellung von annähernd sprungfixen Eigenschaften auftritt.

Eine beispielhafte Funktion des Typs 2 lautet:

$$f(x)= \begin{cases} 2x + 1 & \text{für } x \in [1;3] \\ 7 & \text{für } x \in [3;6] \\ 5x - 23 & \text{für } x \in [6;9] \end{cases}$$

Die grafische Darstellung ist wie folgt:

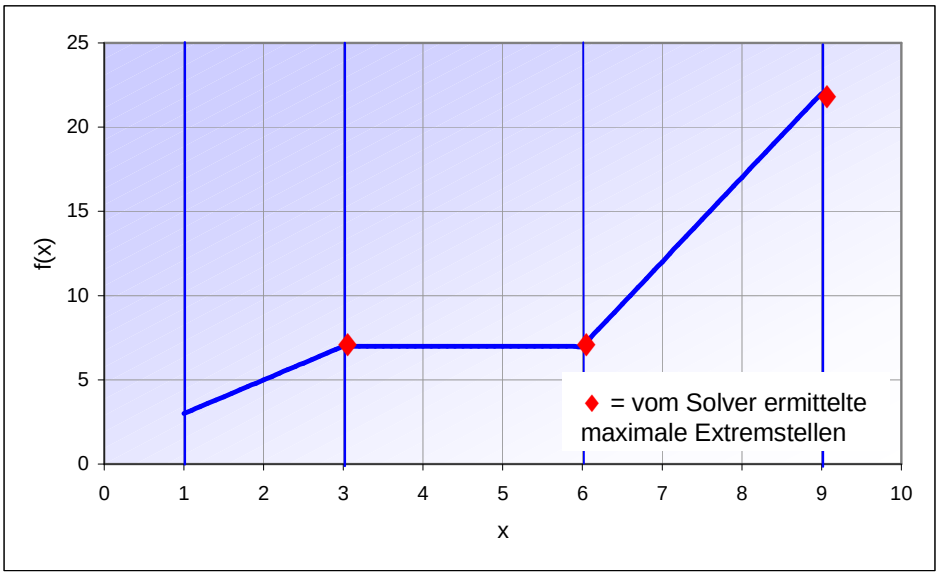


Abbildung 22 - Zusammengesetzte Funktion Typ 2

Bei den Tests auf Basis der zusammengesetzten Funktion des Typs 2 zeigten sich zwei unterschiedliche Verhaltensweisen des Solvers. Je nach Startwert zeigte sich entweder ein sogenanntes schwingendes Lösungsverhalten, welches zwischen den beiden Endpunkten der Funktion mit der Steigung 0 hin- und herpendelte oder aber die korrekte Ermittlung der maximalen Extremstelle.

Ähnlich der Funktion des Typs 1 konnte im Rahmen der Tests bei Funktionen des Typs 2 die globale, maximale Extremstelle nicht verlässlich bestimmt werden.

Testfunktion Typ 3:

Die dritte von drei zusammengesetzten Funktionen setzt sich aus zwei ganzrationalen Funktionen 2. Grades sowie aus einer ganzrationalen Funktion 0. Grades zusammen. Diese Funktionsform zeichnet sich wie der Funktionstyp 2 ebenfalls durch einen Funktionsabschnitt mit der Steigung 0 aus, allerdings entspricht die maximale Extremstelle nicht der Intervallgrenze.

Eine beispielhafte Funktion des Typs 3 lautet:

$$f(x)= \begin{cases} -0{,}5(x-4)^2 + 5 & \text{für } x \in [1{,}5;4] \\ 5 & \text{für } x \in [4;6] \\ -(x-8)^2 + 9 & \text{für } x \in [6;9] \end{cases}$$

Die grafische Darstellung ist wie folgt:

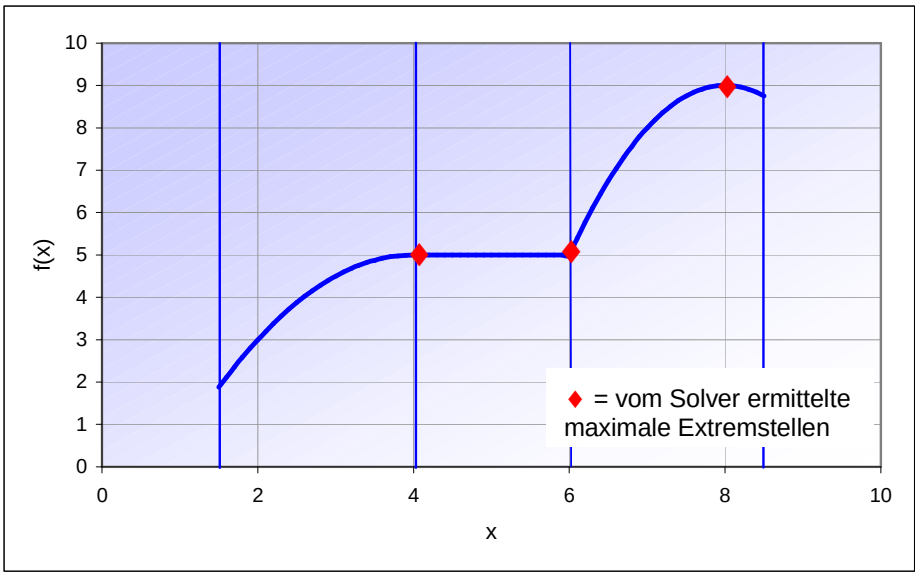


Abbildung 23 - Zusammengesetzte Funktion Typ 3

Ähnlich dem Solververhalten bei der Testfunktion 2 lässt sich bei der Gruppe der Testfunktionen des Typs 3 formulieren, dass je nach Startwert sich

entweder ein schwingendes Lösungsverhalten zeigt, welches zwischen den beiden Endpunkten der Funktion mit der Steigung 0 hin- und herpendelte oder aber der Solver die maximale Extremstelle ermittelte. Ein sicheres Ermitteln der globalen, maximalen Extremstelle war auch bei diesem Funktionstyp nicht erkennbar.

Ergebnis:

Basierend auf den durchgeführten Tests lässt sich hinsichtlich der über das Simulating Interface eingetragenen mathematischen Funktionen und der möglichen Untersuchung dieser Funktionen auf Extremstellen mittels Solver folgendes Ergebnis formulieren.

Für die aus dem LP-System II in das LP-System I eingetragenen Funktionen müssen folgende Bedingungen gelten, um eine korrekte Bestimmung der maximalen Extremstelle im Rahmen der verteilten Optimierung zu ermöglichen:

1. Die Steigung der Funktion sollte nicht in mehr als einem Punkt gleich dem Wert 0 sein.
2. Lokale Optima sind zu vermeiden (siehe Abbildung 21)
3. Entspricht die maximale Extremstelle der Gesamtfunktion nicht einem der beiden Intervallgrenzen, muss der Funktionswert der maximalen Extremstelle größer sein als der Funktionswert mindestens einer Intervallgrenze (siehe Abbildung 17)

Konsequenz:

Fasst man die in den vorigen beschriebenen Testphasen gewonnene Erkenntnis zusammen, so lässt sich Folgendes feststellen:

Die Zielfunktion eines praxisnahen Modells, welches sich durch komplexe, überlagerte und mathematisch nur schwer beschreibbare Funktionen auszeichnet, enthält Intervalle mit der Steigung 0 sowie lokale Optima.

Dieses durch Tests untermauerte Ergebnis lässt den Schluss zu, dass der erarbeitete Algorithmus zur Verknüpfung mathematischer Modelle mit dem Ziel der verteilten Optimierung für den praktischen Einsatz mit dem getesteten Solver nur eingeschränkt geeignet ist.

Eine Verbesserung in der praktischen Umsetzung kann unter Umständen durch die Weiterentwicklung der eingesetzten Solver erfolgen, womit auf Kapitel 9 dieser Arbeit, dem Ausblick, verwiesen werden soll.

8.2 Aufwandsabschätzung hinsichtlich der praktischen Umsetzung des Ansatzes der verteilten Optimierung

Im vorangegangenen Kapitel wurde der in dieser Arbeit entwickelte Ansatz der verteilten Optimierung auf Möglichkeiten und Grenzen hinsichtlich des praktischen Einsatzes untersucht. In diesem Kapitel soll eine grobe Aufwandsabschätzung der Installierung dieses Ansatzes erfolgen.

Zur Abschätzung des Aufwandes einer Systemimplementierung gibt es in der Literatur eine Reihe von Ansätze. Beispielhaft seien hier nach Litke[50] folgende Methode angeführt:

[50] Litke, H.-D., 2004, S. 111

- Analogiemethode
- Relationenmethode
- Multiplikatormethode
- Gewichtungsmethode
- Parametrische Schätzungsgleichung
- Prozentsatzmethode

Aber auch komplexere Ansätze, wie das Function-Point-Verfahren[51] oder auch die COCOMO-Methode[52] dienen der Aufwandabschätzung.

Im Hinblick auf die Implementierung des Systems der verteilten Optimierung ist der Aufwand als überschaubar anzusehen, so dass keines der genannten Methoden zur Anwendung kommen soll. Es wird lediglich eine grobe Abschätzung über den Umfang gemacht, sowie betroffene Bereiche aufgezeigt. Heinrich[53] bietet dazu einen hilfreichen Ansatz. Nach Heinrich umfasst die Installierung eines Informatik-Projektes nicht nur die reine technische Implementierung einer Softwarelösung. Vielmehr ist die Installation als Ganzes im Sinne von Mensch, Aufgabe und Technik/System zu sehen.

Bei der Betrachtung der Umsetzung des Algorithmus der verteilten Optimierung wurde der Ansatz nach Heinrich angepasst und die Betrachtung der Praxiseinführung in folgende Bereiche unterteilt:

1. den hardwaretechnischen Bereich
2. den softwaretechnischen Bereich
3. und den organisatorischen Bereich.

[51] Poensen, B., Bock, B., 2005
[52] Boehm, B., 1995
[53] Heinrich, L.J., 1994, S. 352

Die hardwaretechnischen Voraussetzungen für den Einsatz einer verteilten Optimierung bestehen im Wesentlichen aus drei Komponenten: Zwei Rechner, ein Netzwerk sowie eine zentraler Server.

Die erste Komponente stellen zwei Rechner im Sinne von PCs dar, auf denen je ein autarkes LP-System betrieben werden kann. Die technischen Mindestvoraussetzungen dieser PCs hängen von der Art der LP-Software und vor allem vom Umfang des Modells ab und sind sinnvollerweise auf das autark zu optimierende Problem abzustimmen. Generell gilt: In den zu dieser Arbeit durchgeführten praktischen Tests zeigte sich, dass der Vorgang der verteilten Optimierung im Vergleich zum bestehenden LP-Modell nur vernachlässigbar geringe Systemressourcen beansprucht. So kann an dieser Stelle unterstellt werden, dass ein System, welches ausreichend für das Betreiben eines LP-Systems ist, auch ohne das Schaffen zusätzlicher Ressourcen im Rahmen einer verteilter Optimierung betrieben werden kann.

Lediglich ein Netzanschluss ist für den Austausch von Daten mit der zentralen Plattform notwendig. Die Form dieses Netzanschlusses hängt wiederum davon ab, wie die zentrale Datenbank zum Austausch der Daten zwischen den einzelnen LP-Systemen definiert und eingerichtet wird.

Prinzipiell sind zwei verschiedene Formen zur Anbindung an eine zentrale Plattform denkbar und sinnvoll:

1. Einbindung über das Internet
2. Datenaustausch über ein lokales Netzwerk

Im ersten Fall erfolgt der Datenaustausch zwischen den optimierenden LP-Systemen über das Internet. Das bedeutet, dass ein entsprechender zentraler Internetserver als Plattform dient. Dieser kann von den Modellrechnern kontaktiert werden. Nach der Einwahl eines LP-Systems wurde gerade im Hinblick auf einen derartigen Internetbetrieb an den Anfang des Algorithmus

zur verteilten Optimierung die Phase der Kontaktaufnahme gesetzt (siehe Kaptitel 7.2.2). In dieser Phase wird vom einen „Optimierungspartner“ suchenden System wie beschrieben ein Trigger auf den Server gesetzt – erst bei erfolgreicher Kontaktaufnahme eines zweiten LP-Systems wird der Trigger in der Weise verändert, dass die eigentliche Optimierung beginnen kann. Der Abstand zwischen dem ersten Einwählen und dem Finden eines geeigneten Optimierpartners ist zunächst zeitlich nicht begrenzt. Ähnlich einem Marktplatz kann ein LP-System seinen Trigger auf dem Server hinterlassen und ohne temporäre Limitierung auf ein Kontaktangebot eines anderen LP-Systems warten.

Wichtig ist, dass eine möglichst schnelle und vor allem stabile Internetverbindung installiert werden kann, um einen reibungslosen Durchlauf des in Kapitel 7.2.2 darstellten Algorithmus gewährleisten zu können.

Auf etwaige Risiken durch Übertragung von Viren im Zuge des Datenaustausches über das Internet soll hier nicht eingegangen werden, da in ausreichendem Maße Standardsoftware zur Lösung dieses Problems am Markt verfügbar ist.

Bei einem Datenaustausch über ein lokales Netzwerk, welches in der Regel einem LAN-Netzwerk entspricht, gelten im Wesentlichen die gleichen Voraussetzungen wie bei einer reinen Internetanwendung. Das bedeutet, dass auch bei einem lokalen Netzwerk ein Rechner als zentrale Plattform dient und jeweils zwei oder mehrere LP-Rechner über das Netzwerk mit dem zentralen Rechner kommunizieren. Anders als bei einer Internetanbindung haben heutige LAN-Netzwerke i.d.R. eine höhere Übertragungsrate als herkömmliche Internetverbindungen. Grade bei einer großen Anzahl an Iterationsschritten kann dies von Vorteil sein. Auch das Risiko der Übertragung von Viren kann als gering angesehen, jedoch auch nicht ganz ausgeschlossen werden. Insgesamt sind beide Netzformen in reiner, aber auch gemischter Form möglich; z.B. ist es denkbar, dass einer der beiden Optimierungspartner und

der Rechner mit der zentralen Plattform in einem LAN-Netzwerk betrieben werden, diese aber wiederum mit einem dritten Rechner über das Internet verbunden sind. Es sei an dieser Stelle auf die in Kapitel 6 beschriebene Abgrenzung verschiedener Systeme der verteilten Optimierung verwiesen.

Zu Versuchen bezüglich verteilt optimierender Systeme bietet sich, wie im Rahmen dieser Arbeit durchgeführt, zur Vereinfachung des Versuchsaufbaus (beschrieben in Kapitel 8.1) das LAN-Netzwerk an. In der Praxis kann die Idee der verteilten Optimierung und der damit einhergehenden Ermittlung von systemselbständig ermittelten Handelspreisen bzw. –erlösen allerdings nur im Internet in vollem Umfang ausgenutzt werden, da im Internet sich naturgemäß ein freier Handel von unternehmensunabhängigen Systemen ohne große Probleme beteiligen und damit das System der Verteilten Optimierung erst mit Leben füllen kann.

Nachfolgend wird der Umfang des softwaretechnischen Aufwandes zur Umsetzung eines Systems der verteilten Optimierung kurz dargestellt. Im Wesentlichen sind dabei zwei Softwarekomponenten notwendig:

1. Das LP-Programm inkl. der Implementierung des Algorithmus der verteilten Optimierung.
2. Das Datenbanksystem als zentrale Plattform

Geht man davon aus, dass ein vollständiges, lokal optimierendes LP-System installiert ist, dann muss wie beschrieben und in der Abbildung 11 dargestellt, das Simulating Interface des jeweiligen LP-Systems aktiviert werden. Wie in Kapitel 7.2.1 beschrieben, werden Eigenschaften, die sich nicht ohne weiteres linear berechnen lassen, nach jeder Rekursion über dieses Simulating-Interface in einem externen Programmteil berechnet und in die Matrix zurückgegeben, um wieder in die nächste Rekursion der Optimierung einzufließen. Diese Anpassung des Simulating Interface auch im Hinblick des in dieser Arbeit entwickelten Algorithmus ist bei der praktischen Umsetzung zu

berücksichtigen und kann in Abhängigkeit des jeweiligen LP-Systems stark hinsichtlich des Aufwands schwanken.

Mit der Einrichtung einer zentralen Plattform soll, wie in Kapitel 7.1.2 dargestellt, die Möglichkeit geschaffen werden, eine Information möglichst vielen Teilnehmern parallel zur Verfügung zu stellen. Die zentrale Plattform ist im Rahmen dieser Arbeit als Datenbank zu verstehen, die zwar netzwerktechnisch eingebunden, aber auf von Planungssegmenten unabhängigen Rechnern lokalisiert ist.

Zwar wurde bei den im Rahmen dieser Arbeit durchgeführten Test zur Vereinfachung eine Exceltabelle als zentrale Plattform auf einem unabhängigen Computer eingesetzt, allerdings ist für einen realistischen Praxiseinsatz eines Systems zur verteilten Optimierung in jedem Fall eine Datenbank notwendig.

Der Begriff Datenbank soll hierbei im Sinne eines elektronischen Systems zur Speicherung und Verwaltung umfangreicher Datenmengen verstanden werden, wobei nachfolgend nur wesentliche Elemente aus dem Bereich der Datenbanken erörtert werden. So sei u.a. an dieser Stelle auf die weiterführende Literatur von Kemper/Eickler[54], Silberschatz/Korth/Sudarshan[55] und Ramakrishnan/Gehrke[56] verwiesen.

Allgemein besteht eine Datenbank aus der Datenbasis, in der die Datenbestände zentral i.d.R. auf Magnetplattenspeichern gehalten werden und einem Datenbankmanagementsystem (kurz DBMS), das den Zugriff auf die Daten verwaltet. Zentrale Idee von DBMS ist die Trennung in eine logische und eine physikalische Beschreibung von Daten[57]. Über das DBMS kann die

[54] Kemper A./Eickler A.(2004)
[55] Silberschatz A.,Korth H.F., Sudarshan S. (2001)
[56] Ramakrishnan R., Gehrke J. (2002)
[57] Pernul G., Unlaud R. (2003), S. 21 ff.

Datenbasis mit Hilfe von Abfragesprachen (englisch query language) erschlossen werden.

Hinsichtlich des Verbreitungsgrades spielen relationale Datenbanken eine zentrale Rolle. Relationale Datenbanken bestehen aus einer Speicherungs- und einer Verwaltungskomponente. Die Speicherungskomponente dient dazu, sowohl Daten als auch Beziehungen zwischen diesen Elementen lückenlos in Tabellen abzulegen. Die Verwaltungskomponente enthält als wichtigsten Bestandteil eine relationale Datendefinitions- und Datenmanipulations-sprache.[58] Objektorientierte Datenbanken, welche definierten Regeln und Grundsätzen der Objektorientierung und Datenhaltung- bzw. Nutzung unterliegen, erweitern diesen Ansatz[59].

Als wesentliche Kriterien für die Auswahl der Datenbank zum Einsatz als zentrale Plattform sind folgende Punkte zu nennen, die nachfolgend weiter ausgeführt werden:

- schnelle Zugriffszeiten
- unkompliziertes Lesen und Schreiben von Datensätzen
- Kompatibilität zu den verknüpften LP-Systemen

Bei der Auswahl einer für eine zentrale Plattform im Rahmen der verteilten Optimierung geeigneten Datenbank hat die Zeitspanne zur Kontaktaufnahme des anschließenden Datenaustausches eine große Bedeutung. Dabei spielen neben den Übertragungszeiten der Daten vor allem die Zugriffszeiten bei Datenbanken eine wesentliche Rolle im Hinblick auf die Laufzeit eines Optimierungsverlaufes. Als Zugriffszeit ist dabei der Zeitraum zu verstehen, den ein LP-System nach erfolgreicher Verbindung mit der zentralen Plattform braucht, um einen Datensatz auf diesem zentralen Rechner zu hinterlegen oder

[58] Meier, A. (2004), S. 7 ff.
[59] Meier, A.; Wüst T., (2000), S. 5

zu lesen. Form und Art der Datenbank haben je nach Aufbau und Systematik unterschiedliche Zugriffszeiten.

Ein anderes Kriterium bei der Auswahl einer geeigneten Datenbankform ist die benötige Form des Datensatzes, der in der Datenbank hinterlegt werden kann. Diese von unterschiedlichsten LP-Systemen gesendeten und gelesenen Datensätze sollten mit möglichst vielen LP-Systemen kompatibel sein. Nur dann kann ein unkompliziertes und damit schnelles Schreiben und Lesen von Datensätzen auf die zentrale Plattform gewährleistet werden. Muss zu ihrer Anpassung zwischen dem LP-Systemen und der zentralen Plattform noch ein Zwischenschritt zur Normierung der transferierten Datensätze implementiert werden, erhöht sich gerade bei Optimierungen mit hoher Iterationszahl die Laufzeit der Gesamtoptimierung erheblich.

Erstrebenswert ist die Schaffung eines Standards zur Definition der Form auszutauschender Datensätze, um die Auswahl der Datenbankform für die zentrale Plattform zu vereinfachen. Damit empfiehlt sich die Auswahl einer Datenbankform mit einer hohen Kompatibilität zu den beteiligten LP-Systemen, um unnötige Datenanpassungsroutinen zu vermeiden.

Bei der Aufwandsabschätzung zur praktischen Umsetzung des Systems zur verteilten Optimierung lässt sich im Hinblick auf den softwaretechnischen Aufwand zusammenfassen, dass beide betrachteten Elemente (Modellanpassung und Datenbankauswahl/-installation) wesentlichen Einfluss auf die Systemzuverlässigkeit haben. Eine entsprechend sorgfältige Abarbeitung dieser beiden Punkte ist damit Grundvoraussetzung für die Funktionalität dieses Systems.

In einem letzten Punkt zur Beschreibung des Aufwandes zur praktischen Umsetzung des Systems der verteilten Optimierung sollen die organisatorischen Elemente näher betrachtet werden.

Dazu zählen:

- Schulung der Mitarbeiter
- Identifizieren eines Partners mit passender LP-Struktur
- Abgleich von Produktspezifikationen

Durch die Verknüpfung verteilt optimierender Systeme wird in bestehende klassische Optimierungssysteme (siehe dazu Kapitel 3.1) erheblich und in umfangreichem Maße eingegriffen. Dieses stellt nicht nur eine erhöhte Anforderung an die technischen Systeme, sondern auch an die Benutzer dieser Neuentwicklung. Im Gegensatz zu den zuvor oftmals lokal optimierenden Systemen findet hier nun eine Verknüpfung von Optimierungsbereichen mit unterschiedlichen Zielfunktionen statt.

So ist bei der Schulung der mit dem System der verteilten Optimierung betrauten Mitarbeiter auf zwei verschiedenen Ebenen zu agieren. Erstens ist dies die Ebene des unmittelbaren , technischen Wissens, zum anderen die Ebene des vernetzten Denkens. Im ersteren, technischen Teil der Schulung sollte nach Möglichkeit vor Einführung des Systems ein Schulungsplan aufgestellt werden, der Personenkreis festgelegt und wesentliche Teile des Systems geschult sein, um nicht die Akzeptanz und damit auch die Funktionalität zu gefährden. Der zweite Teil der Schulung sollte schließlich die Anforderungen an ein vernetztes Denken im Rahmen eines Optimiersystems abdecken. Aus persönlicher Erfahrung heraus ist festzuhalten, dass dieser Teil der Schulung den bis dahin lokal optimierenden Mitarbeitern schwer fällt. So stellen sich in der Praxis schnell Fragen betreffend des Wertes von Produkten im eigenen und im fremden System.

Auch eine Schulung über das eigene System hinaus kann für das vernetzte Denken sinnvoll sein. Beispielsweise kann es nützlich sein, das Optimierungssystem des Optimierungspartners im Wesentlichen zu kennen,

um potentielle Produkte, die sich für den Austausch eignen, beim Optimierungspartner zu identifizieren. Denn beim System der verteilten Optimierung kann zwar – wie mehrfach beschrieben – das globale Optimum einer zuvor definierten Eigenschaft über die beteiligten System optimiert werden, allerdings muss zuvor vom Bediener des Systems das Produkt ausgewählt werden. Dazu ist das Wissen um den Nutzen der diskutierten Komponenten sowohl im eigenen als auch in den an der Optimierung noch beteiligten Systeme unerlässlich.

Ein weiterer wesentlicher organisatorischer Teil, der bei der Aufwandsabschätzung einer praktischen Umsetzung erwähnt werden muss, ist das Identifizieren eines Partners mit passender LP-Struktur.

Dazu ist zunächst zu erörtern, was genau eine passende LP-Struktur kennzeichnet. In Kapitel 5 wurden so genannte Mindestanforderungen an Optimierverfahren zur Implementierung des Systems der verteilten Optimierung definiert, welche an dieser Stelle nicht nochmals aufgegriffen werden sollen. Lediglich soll darauf hingewiesen werden, dass diese Mindestanforderungen an alle beteiligten Systeme gestellt werden.

Sollen es nun über die beschriebene zentrale Plattform zu einem iterativen Datenaustausch kommen, ist zuvor ein entsprechender Optimierungspartner zu suchen und auf die beschriebenen Mindestanforderungen zu prüfen. Denn nur ein reibungsloser und damit effizienter Datenaustausch über die zentrale Plattform bildet die Voraussetzung für das Funktionieren des Systems der verteilten Optimierung.

Auf der Grundlage der praktischen Untersuchungen im Rahmen dieser Arbeit ist nachdrücklich darauf hinzuweisen, die technischen systemseitigen Voraussetzungen zu prüfen. Ein Fehler während des rekursiven Optimierlaufes zwischen den beteiligten Systemen findet in der anschließenden Praxis ohne unmittelbare Beteiligung des Anwenders, nur im Dialog zwischen den

verknüpften Optimierungssystemen statt. Ob und inwieweit das errechnete globale Optimum realistisch ist, kann der einzelnen Anwender nur schwer einschätzen, da ihm der Inhalt des beteiligten Optimierungssystems in den meisten Fällen unzugänglich bleibt. Um so mehr muss er sich darauf verlassen, dass die an der Optimierung beteiligten Systeme abgestimmt und im Sinne der Anwender arbeiten.

Der letzte Parameter der organisatorischen Maßnahmen bei der Abschätzung des Aufwandes zur praktischen Umsetzung ist der Abgleich der Produktparameter. Als Produktparameter sollen hier alle Elemente verstanden werden, die zur eindeutigen Definition des Produktes und den damit verbundenen Eigenschaften gehören.

Wenn ein globales Optimum über eine Produkteigenschaft im Rahmen der verteilten Optimierung ermittelt werden soll, ist diese Eigenschaft in Form von Wert, Dimension, Messmethode oder Ähnliches in der Weise zu definieren, dass eine Eindeutigkeit der betrachteten Eigenschaft zwischen den an der Optimierung teilnehmenden Partnern vorhanden ist. Soll z.B. ein einzustellender Schwefelgehalt eines bekannten Produktes über mehrere Systeme optimiert werden, so ist die Dimension (z.B. ppm) und/oder die Eigenschaft (z.B. welcher Schwefel) sowie die Meßmethode (z.B. DIN) zu definieren.

Bei den im Rahmen dieser Arbeit durchgeführten Praxistests zeigte sich, dass es sinnvoll ist, auf der zentralen Plattform eine Stoffdatenbank mit standardisiertem Aufbau anzulegen. Eine derartige Stoffdatenbank enthält wesentliche Produktparameter zu der an der Optimierung beteiligten Produktkomponenten und sollte als Datensatz zur Verfügung stehen. Dieser sollte von den an der verteilten Optimierung beteiligten LP-Systemen vor Beginn der Optimierung von der zentralen Plattform herunter geladen bzw. auf die zentrale Plattform zum Herunterladen abgelegt werden. Nur so kann gewährleistet werden, dass vor Beginn der Optimierung allen LP-Systemen der

gleiche Datensatz zur Verfügung steht. Es hat sich dabei als sinnvoll herausgestellt, in den Namen des Produktes das letzte Änderungsdatum des dazugehörigen Datensatzes zu implementieren. Beispielhaft sei hier „Alkylat-2004-07-01“ genannt, d.h. dieser Datensatz wurde am 01. Juli 2004 aktualisiert.

Die Definition des Datensatzes ist deshalb Grundvoraussetzung für eine korrekte Optimierung, da auch bei eindeutigem Namen eines betrachteten Produktes zwar über eine korrekt definierte Eigenschaft (z.B. Produktdichte in kg/m^3) optimiert wird, allerdings hinsichtlich einer als konstant anzunehmenden Produkteigenschaft die beteiligten LP-Systeme von unterschiedlichen Werten ausgehen können, was vermieden werden muss.

In Bezug auf die in Kaptitel 7.1.2 beschriebene Kontaktphase bedeutet dies, dass bei Hinterlegen eines so definierten Angebotes mit dem Ziel der verteilten Optimierung über eine Eigenschaft zu dem betreffenden Produkt immer ein aktueller Datensatz mit allen relevanten Daten auf der zentralen Plattform hinterlegt ist. Diese Veröffentlichung der Produkteigenschaften kann aus Sicht des Datenschutzes ein Problem darstellen. Für den Fall, dass kein vollständiger Datensatz über die Produkteigenschaften ausgetauscht werden darf, fehlt die Basis für den Einsatz der verteilten Optimierung.

Bei der Betrachtung der organisatorischen Maßnahmen soll an dieser Stelle nicht auf vertragsrechtliche Aspekte, wie sie beim Austausch von Gütern in der allgemeinen Warenwirtschaft üblich sind, eingegangen werden.

Insgesamt zeigte sich bei dem testweisen Einsatz des Systems zur verteilten Optimierung, dass sowohl die software- als auch die hardwaretechnischen Implementierung einen umfassenden, aber begrenzten Umfang darstellten. Die Elemente des organisatorischen Bereiches hingegen bedürfen einer permanenten Pflege und Anpassung auf die aktuellen Optimierungsbedingungen und stellen somit hinsichtlich des Aufwands zur

praktischen Umsetzung des Ansatzes zur verteilten Optimierung einen nicht zu unterschätzenden Aspekt dar.

8.3 Zusammenfassung definierter Testmerkmale und der damit verbundenen Erfahrungen bei der Umsetzung

Bei der Umsetzung des Algorithmus der verteilten Optimierung wurden, wie in den vorangegangenen Kapiteln beschrieben, definierte Merkmale getestet, um einen fehlerfreien Ablauf der Gesamtoptimierung gewährleisten zu können. Diese Merkmale, die eingesetzten Hilfsprogramme und die gewonnenen Erkenntnisse werden nachfolgend nochmals kurz zusammengefasst.

Sollen zwei LP-Systeme über eine zentrale Plattform zum Zweck einer verteilten Optimierung verbunden werden, so ist sicherzustellen, dass die beteiligten LP-Systeme auch autark – ohne Anbindung an die zentrale Plattform – in einer begrenzten Anzahl an Iterationen, stabil und ohne Schwingen[60] zu einer Lösung kommen. Es stellte sich im Rahmen von autark optimierenden Systemen unter Einsatz eines Viewers die Überprüfung folgender Parameter als sinnvoll heraus:

- die Anzahl der benötigten Iterationen zur Optimierung
- die Höhe des Deckungsbeitrages sowie
- die definierten Grenzen der betrachteten Variablen.

Denn nur, wenn alle an einer verteilten Optimierung teilnehmenden Systeme diese Anforderungen im autarken Betrieb erfüllen, ist die Basis für eine reibungslos verteilte Optimierung geschaffen.

[60] Schwingen: Während der iterativen Optimierung pendelt nach einer asymptotischen Annäherung die maximale Ausprägung der Zielfunktion zwischen zwei Lösungen hin und her ohne zu einer finalen Lösung zu gelangen.

Hinsichtlich des Datenaustausches zwischen LP-System und zentraler Plattform zeigte sich die Überprüfung folgender Punkte als sinnvoll:

- die korrekte Übernahme der Eingangsdaten in die Matrix,
- das Hinterlegen des ersten Datensatzes auf der zentralen Plattform,
- die Korrektheit des Datenaustausches mit der zentralen Plattform zwischen den Iterationen unter Einhaltung der definierten maximalen Zugriffszeiten und
- das Erzeugen eines korrekten Reports.

Ein dritter Block, der sich bei der praktischen Umsetzung des Algorithmus zur verteilten Optimierung als sensibel herausstellte, war der technisch korrekte Ablauf eines Optimierzyklus unter spezieller Beachtung der im Algorithmus genannten Start- und Abbruchbedingungen. Entsprechend sollten folgende Elemente des Algorithmus einer Überprüfung unterzogen werden:

- das Hinterlegen eines ersten vollständigen Datensatzes auf der zentralen Plattform,
- das Verfolgen der wechselseitigen Rekursionen der LP-Systeme,
- das Sicherstellen der korrekten Umsetzung der definierten Abbruch-bedingungen,
- das Erstellen zweier inhaltlich und formal korrekter Reports und
- die Reaktion der LP-Systeme bei technischen Problemen während der Optimierung.

Bei der vierten und letzten Phase der durchgeführten Testreihen zeigte sich, dass ein praxisnahes Modell der verteilten Optimierung, welches sich durch komplexe, überlagerte und mathematisch nur schwer beschreibbare Funktionen auszeichnet, nur bedingt für ein System verknüpfter Modelle mit paralleler Optimierung geeignet ist.

Im Hinblick auf die Aufwandsabschätzung bei der praktischen Umsetzung lässt sich zusammenfassen, dass im Wesentlichen Arbeitsgebiete hardwaretechnisch, softwaretechnisch und organisatorisch betrachtet werden müssen.

Im Bereich der Hardware wurden die Elemente zentraler Server, Modellrechner und Netzwerke betrachtet. Dieses Gebiet zeigte einen hohen Aufwandsanteil im Verhältnis zum Gesamtprojekt zur Einführung des Systems der verteilten Optimierung, weil die technischen Anforderungen eine unerwartet hohe Komplexität aufwiesen.

Im Bereich der Software wurde einerseits der Aufwand der notwendigen Anpassung des LP-Modells erläutert, andererseits wurde auf den Aufbau einer zentralen Datenbank als zentrale Plattform eingegangen. Dabei konnte die Erkenntnis gewonnen werden, dass im Gegensatz zum Aufbau einer Datenbank als zentrale Plattform die Anpassung des LP-Modells in Bezug auf das Simulating Interface in diesem Projekt ein hohes Maß an Aufmerksamkeit erfordert.

Schließlich wurde der organisatorische Bereich mit den Komponenten ‚Auswahl eines Optimierungspartners', ‚Schulung' sowie der sich im Rahmen dieser Arbeit als relativ komplex herausstellende Punkt des ‚Abgleichs der Produktparameter' betrachtet. Im Gegensatz zu den technischen Elementen (Hardware und Software) zeichnet sich der organisatorische Teil dadurch aus, dass der Aufwand über eine einmalige Leistung hinaus permanenten Einsatz in Form von Schulung, Anpassung der Produktparameter usw. erfordert.

Um einen fehlerfreien Ablauf einer verteilten Optimierung gewährleisten zu können, wurden der autarke Betrieb, der Datenaustausch, der Einsatz des Algorithmus sowie die überlagerten mathematischen Funktionen betrachtet. Ebenso wurde der software- und hardwaretechnische Aufwand einer Systemeinführung beschrieben.

Die bei der Implementierung der Testversionen dargestellten Ergebnisse zeigten, dass eine Fülle von Einzelbetrachtungen notwendig ist, um einen stabilen Betrieb des Systems zu gewährleisten. Diese Systemstabilität ist vor allem vor dem Hintergrund einer WAN-Netz basierten Anwendung wichtig. Im Gegensatz zu eng gekoppelten Parallelrechnern ist bei lose gekoppelten Systemen eine hohe Stabilität trotz steigender Anzahl an Systemknoten die Grundvoraussetzung.

Ein weitere Grundvoraussetzung in diesem Zusammenhang stellt die Kompatibilität der beteiligten Systeme dar. Nimmt man den Fall einer Implementierung im Rahmen einer WAN-Verbindung an, was in der Praxis in den meisten Fällen einer Internetverbindung entspricht, so muss man davon ausgehen, dass unterschiedliche LP-Systeme sich an dem Prozess der verteilten Optimierung beteiligen. Die Umschreibung dieser unterschiedlichen Systeme ist dabei in zweierlei Hinsicht zu verstehen. Einerseits können, wie bereits erwähnt, LP-Systeme unterschiedlicher Anbieter zur Anwendung kommen, anderseits ist auch jedes LP-Modell in Struktur und Aufbau anders. Dennoch muss der Ansatzes der verteilten Optimierung derart umgesetzt werden, dass eine hohe Systemverfügbarkeit in Verbindung mit einer hohen Kompatibilität erreicht werden kann. Denn nur so kann eine möglichst hohe Anzahl an beteiligten Systemen ermöglicht und im martktechnische Sinne ein globales Optimum erreicht werden. Weiterführend sei an dieser Stelle allerdings auf Kapitel 9 verwiesen.

9 Ansätze zur Bestimmung des globalen Optimums in der Literatur zur verbesserten Umsetzung des Ansatzes der verteilten Optimierung

Bei den durchgeführten Praxistests wurde in Kapitel 8.1 die Feststellung formuliert, dass ein praxisnahes Modell aufgrund der Komplexität der mathematischen Funktionen nur eingeschränkt für das System der verteilten Optimierung geeignet ist. Als Grund für die nicht ausreichende Extremstellenbestimmung im Rahmen der verteilten Optimierung erwies sich das eingeschränkte Lösungsverhalten des Simplex Algorithmus bei der Bestimmung der maximalen Extremstelle der Zielfunktion. Ziel dieser Arbeit soll nicht sein, den von ASPENTECH erstellten Solver zu verbessern. Vielmehr soll nachfolgend auf Entwicklungen in der Literatur hingewiesen werden, die einen Lösungsansatz zur Bestimmung des globalen Optimums bieten, um so eine Realisierung des Algorithmus der verteilten Optimierung ermöglichen zu können.

Wie in der Einleitung dieser Arbeit erwähnt, gilt es hier, das globale Optimum als maximalen Gesamtdeckungsbeitrag unter Variation einer oder mehrerer Eigenschaften über zwei Planungsmodelle hinweg zu bestimmen. Der Begriff des globalen Maximums soll nachfolgend wie bei Mautz[61] als Funktionswert eines Punktes verstanden werden, der im Vergleich größer ist, als alle anderen Punkte aus dessen lokaler Umgebung.

Liegen alle beteiligten Funktionen in *einem* Planungsmodell vor, über das optimiert werden kann, gibt es in der Literatur ausreichend Lösungsansätze zur Bestimmung des globalen Optimums. Beispielhaft seien hier die Untersuchungen von Ratz[62], Ratschek[63] und Törn/Zilinska[64] genannt. Alle drei

[61] Mautz R., 2000, S. 11
[62] Ratz D., 1992
[63] Ratschek, H., 1988, S. 18
[64] Törn/Zilinska, 1989, S. 23

Untersuchungen legen zu Grunde, dass über die Betrachtung mathematischer Sachverhalte hinaus Optimierungsprobleme aus der Praxis aufgegriffen werden. Dabei wird ein wesentlicher Schritt weg von theoretischen Formulierungen hin zu Funktionsverläufen als Abbild alltäglicher Zusammenhänge gemacht. Die mit der Praxisnähe einhergehende zunehmende Komplexität stellt sich dabei schon beim Modellieren einfachster Zusammenhänge ein.

So weisen die drei genannten Autoren eindringlich darauf hin, dass mit zunehmender Komplexität der mathematischen Funktionen sich die Dichte der untersuchten Punkte und damit auch die Komplexität der Modellfunktionen stark erhöht und damit die Laufzeiten zur Lösung so genannter „Real-World-Probleme“, wie sie Ratz nennt, exponentiell ansteigen. Aber gerade die Abbildung und Modellierung von praxisbezogenen Aufgabenstellungen stellt eine zentrale Rolle dar, da die Verfügbarkeit von Prozessdaten in der Vergangenheit stark angestiegen ist.

Ein wesentlicher Grund für das steigende Datenvolumen in vielen Bereichen ist die zunehmende elektronische Unterstützung der Sammlung, Archivierung und Auswertung von Daten. Für die Industrie sei hier beispielhaft die Entwicklung im Bereich der Mess- und Regeltechnik erwähnt. Durch die signifikanten Verbesserungen bei der Datenaufnahme (Onlineanalytik, Prozessparameter, Positionsmessungen usw.) können Daten exakter und mit höherer Frequenz aufgenommen werden. Auch die Datenübermittlung, die Entwicklungen wie Breitbandtechnik oder Lichtfaserkabel hervorgebracht hat, wurde auf den Transfer von entsprechend hohen Datenmengen ausgebaut. Hier ist die Entwicklung im Bereich der Datenbanken als zentral im Hinblick auf die steigende Komplexität von Funktionen basierend auf „Real World Problemen“ zu nennen. Erst das systematische Speichern und Verwalten von sehr großen Datenmengen macht eine mathematische Analyse und Auswertung der Daten möglich, um entsprechend wirtschaftlich nutzbare Informationen daraus ableiten zu können.

Aus diesem Grunde bieten sowohl ASPENTECH als auch HAVERLY inzwischen Datenbank basiert LP-Systeme an. Dazu können sowohl Eingangsdaten aus Datenbanken entnommen werden als auch die Ergebnisse vor dem nächsten Weiterverarbeitungsschritt, wie z.B. Scheduling bedarfsgerecht gespeichert werden. Wie von Ratz[65] und Ratschek[66] erwähnt, nimmt die Komplexität mathematischer Modelle und den damit verbundenen LP-Systemen stark überproportional zu, wenn die technischen Entwicklung hinsichtlich des steigenden Datenvolumens genutzt wird. Um so aufwendiger wird die Anforderung an den Solver zur Ermittlung der gewünschten Extremstelle der Zielfunktion. Im Rahmen dieser Arbeit stand ein komplexes Raffineriemodell mit umfangreicher Datenstruktur in Verbindung mit linearen sowie nichtlinearen Funktionen zur Verfügung. Steht ein derartiges Modell nicht zur Verfügung, sollte in der Testphase auf mathematische Grundfunktionen zurückgegriffen werden.

In der Literatur wird im Rahmen von vergleichenden Systemtests oftmals auf Levy-Funktionen zurückgegriffen. Diese zeichnen sich im Wesentlichen durch ihre Komplexität bei gleichzeitig einfacher mathematischer Darstellbarkeit des mathematischen Zusammenhangs, aus.

Auf Basis derartiger Levy-Funktionen kann die Effizienz bestehender Algorithmen zur Lösung von mehrdimensionalen Funktionen mit lokalen Optima bestimmt und verbessert werden. Auch der Algorithmus zur verteilten Optimierung könnte ausgehend von Levy-Funktionen hinsichtlich der Verbesserung des Lösungsverhaltens von „Real-World-Problemen“ weiterentwickelt werden.

Eine Alternative zu den Levy-Funktionen stellen Funktionen dar, die im Bereich der Simulationstechnik in Raffinerien Anwendung finden und deren

[65] Ratz D., 1992
[66] Ratschek, H., 1988, S. 18

Ursprung unter anderem in Ableitungen statistischer Methoden zu suchen ist. An dieser Stelle sei die stark voranschreitende Entwicklung der neuronalen Netzwerktechnik genannt. Bei diesem Ansatz werden aus der Praxis z.B. in einem industriellen Prozess eine Fülle von zunächst unabhängigen Datenreihen über die gleiche Zeitachse aufgenommen. In einem zweiten Schritt wird nach Korrelationen zwischen den Datenreihen gesucht. Mehrdimensionale Zusammenhänge können so identifiziert und in mathematischem Zusammenhang formuliert werden. Der Vorteil neuronaler Netze liegt in der Möglichkeit, innerhalb kurzer Zeit ein mathematisches Modell zu erstellen. Bei Vorhandensein komplexer Strukturen, bei denen lineare oder einfache funktionelle Ansätze für eine Erklärung der Zusammenhänge nicht ausreichen, ist mit der neuronalen Netzwerktechnik häufig eine Verbesserung der Prognosequalität zu erreichen. Neuronale Netze werden in vielen Bereichen der Wissenschaft untersucht. Ohne detailliert auf neuronale Netze in der Wissenschaft einzugehen, soll hier lediglich auf die entsprechende Literatur verwiesen werden; siehe dazu [67,68,69,70,71].

Speziell in einem LP-Modell, welches der wirtschaftlichen Optimierung einer Raffineriefahrweise dient, spielt das beschriebene Blending von Komponenten zu Endprodukten eine wesentliche Rolle. Eine Reihe von Eigenschaften der Blendingkomponenten lassen sich linear berechnen. Dazu zählen z.B. Dichte und Viskosität. Eigenschaften wie Dampfdruck oder Cetanzahl einer Komponente werden in der Regel indirekt über einen Index (Dampfdruckindex, Cetanzahlindex) bestimmt. Dieser ist allerdings nur ein Maß dieser Einheit – stellt aber nicht die Einheit selbst dar. Zur Abbildung eines direkten funktionalen Zusammenhangs stößt man ohne neuronale Netzwerktechnik schnell an die mathematischen Grenzen.

[67] M. Baret, 2000, S.55 ff
[68] B. Chaudhuri, J.M. Modak (1998), S. 19
[69] E.M. Cortez, S.C. Park S. Kim (1995), S. 31 u. 789
[70] S. Jin, K.M. Ye, K. Shimizu, J. Nikawa (1996), S. 81
[71] Z.C. Lin (1996), S. 10 u. S. 21

Die Veröffentlichung von Yin, Luo und Zhou[72] zeigt Lösungsansätze, in denen mathematische Formulierungen des Blendingproblems durch die neuronale Netzwerktechnik im Bereich des Kohleblendings realisiert wurden. Bei systematischem Vergleich zwischen dem Blenden von Kohle und dem von Mineralölprodukten zeigt sich, dass in Anlehnung an die veröffentlichte Arbeit eine Anwendung der neuronalen Netzwerktechnik im Bereich des petrochemischen Blendings technisch möglich und sinnvoll ist. Eine entsprechende Veröffentlichung konnte im Rahmen der Literaturrecherche nicht gefunden werden.

Die Überführung der erwähnten „Real-World-Probleme“ in formelmäßige Zusammenhänge mittels der neuronalen Netzwerktechnik bietet eine höhere Übereinstimmung zwischen Realität und mathematischer Funktion. Allerdings können die auf diese Weise bestimmten Funktionen von hoher Komplexität sein, was neben den Problemen bestehender Solver mit dem Ansatz der verteilten Optimierung auch Verbesserungspotential bei Einsatz der Funktionen aus der neuronalen Netzwerktechnik aufzeigen würde.

Wie eingangs dieses Kapitels erklärt, sollen Ansätze aus der Literatur aufgezeigt werden, mit denen ausgehend von den zuvor beschriebenen Funktionen eine Bestimmung des globalen Optimums möglich ist. Sie geben einen Ausblick, in welcher Weise eine Weiterentwicklung des in dieser Arbeit an die Grenzen gelangten Solvers denkbar wäre, um eine vollständige Umsetzung des in dieser Arbeit entwickelten Algorithmus zu ermöglichen.

Zur Verbesserung der Lösungsalgorithmen in den Solvern betrachteter LP-Systeme, zeichnet sich in der Literatur ein Lösungsansatz ab, der die Vereinfachung der Funktionsverläufe vor Einsetzen der Lösungsalgorithmen und die damit einhergehende Reduzierung der Laufzeiten zum Ziel hat. In der Literatur gibt es diesbezüglich eine Reihe von Ausarbeitungen. Beispielhaft sei hier ein in der Praxis erfolgreicher Ansatz einer heuristischen Glättungsprozedur zur

[72] C.G. Yin, Z.Y. Luo, Z.H. Zhou, K.F. Cen (2000), S. 78 ff.

Vermeidung von lokalen Optima von Hilding[73] genannt. In der von ihm dargestellten ‚Heuristic smoothing procedure' (HSP) kommt es zur sequenziellen Lösung von Optimierungsproblemen. Im Gegensatz zum Downhill-Simplex-Algorithmus als direktem Optimierungsverfahren setzt HSP auf Gradientenverfahren, wie sie bereits in Kapitel 4.2 dargestellt wurden. Als klassisches Gradientenverfahren wurden die indirekt optimierenden Ansätze ‚Newton-Raphson-Verfahren' und ‚Gauß-Newton-Verfahren' erörtert. Hinsichtlich des Einsatzes beider Verfahren im Bereich der Produktionsplanung und in der verteilten Optimierung ließ sich feststellen, dass das Newton-Raphson-Verfahren inklusive seiner Erweiterung durch Levenberg[74] in der Lage ist, auch höherdimensionale Probleme zu lösen und die gewünschte Extremstelle der Zielfunktion zu ermitteln. Allerdings nimmt die Rechenzeit bei komplizierten, realitätsnahen Funktionen in einem Maße zu, dass sich diese Form der Gradientenverfahren nicht für den Einsatz in der industriellen Produktionsplanung eignet (siehe dazu Kapitel 4.2.2). Neuere Gardientenverfahren weisen deutlich verbesserte Rechenzeiten aus. Beispielhaft seien hier das SQP (sequential quadratic programming) von Bazaraa[75] und der geschilderte Ansatz MMA (method of moving asymptotes) von Svanberg[76] genannt. Verbindet man diese verbesserten Ansätze der Gradientenverfahren mit der Methodik der heuristischen Glättungsprozedur nach Hilding, so kann die Zielfunktion durch Glättung vereinfacht und damit die Optimierzeiten minimiert werden, ohne Einschränkungen beim Lösungsraum einräumen zu müssen. Unter Berücksichtigung der aktuellen Entwicklung bei Gradientenverfahren kann dies nach Hilding zu einer Alternative des bei Solvern sehr verbreiteten Simplex-Algorithmus führen.

Ein weiterer Ansatz zur Ermittlung des globalen Optimums bei nicht-linearen Funktionsverläufen mit lokalen Optima wird von Williams[77] aufgezeigt. Williams geht wie Hilding davon aus, dass optimale Lösungen immer auf den

[73] D. Hilding, 2000, S. 29-36
[74] K. Levenberg, 1944
[75] M.S. Bazaraa, 1993
[76] K. Svanberg, 1987, S. 359 ff.

Grenzen des Feasibilitätsbereiches liegen. Sind jedoch nicht nicht-lineare Funktionen im Modell enthalten, können auch die Grenzen des Feasibilitätsbereiches nicht-linear sein (siehe dazu Abbildung 24). In diesem Fall besteht die Gefahr der Ermittlung eines lokalen Optimums an Stelle der gewünschten absoluten Extremstelle.

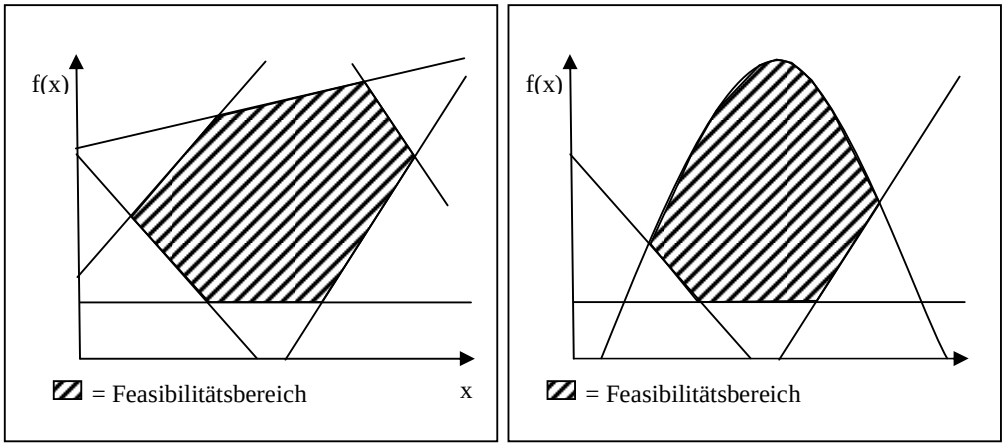


Abbildung 24 – Feasibilitätsbereiche mit linearen und nicht-linearen Grenzen

Das Implementieren von nicht-linearen Funktionen in das Modell ist die Basis der so genannten Nicht-linearen-Programmierung (Non-linear-programming [NLP]). Die durch das Verknüpfen linearer Modelle im Rahmen dieser Arbeit entstandenen Modelle verlassen den Bereich der Linearität. LP-Modelle werden zu NLP-Modellen[78], dieser Zusammenhang wurde in den in Kapitel 8.1 dokumentierten Tests ausführlich beschrieben.

Das in Kapitel 8.3 formulierte Ergebnis entspricht exakt der Darstellung von Williams: Sind unmodifizierte Simplex Algorithmen bei konvexen nicht-linearen Modellen noch in der Lage, das absolute, globale Optimum zu ermitteln, so ist dies bei nicht konvexen Modellen nicht mehr sicher und reproduzierbar möglich.

---

[77] H. Williams, 1993

[78] In bestehenden LP-Modellen sind z.B. durch die Modellierung von nicht-linearen Blendingeigenschaften bei der Berechnung des Dampfdruckes durch den Doppellogarithmus der Waltherformel nicht lineare Elemente enthalten.

Zur Ermittlung des globalen Optimums bei nicht konvexen Funktionen im Zusammenhang eines NLP-Modells untersucht Williams in einem ersten Schritt den Ansatz der separablen Funktionen. Eine Funktion kann dabei als separabel definiert werden, wenn diese durch Approximation als Summe von EInzelfunktionen dargestellt werden kann. Zur Verdeutlichung wird in Abbildung 25 die stückweise lineare Approximation anhand der folgenden nicht-linearen Funktion beispielhaft dargestellt.

Beispielfunktion: $y = 5x^3 - 14x^2 + 11x$

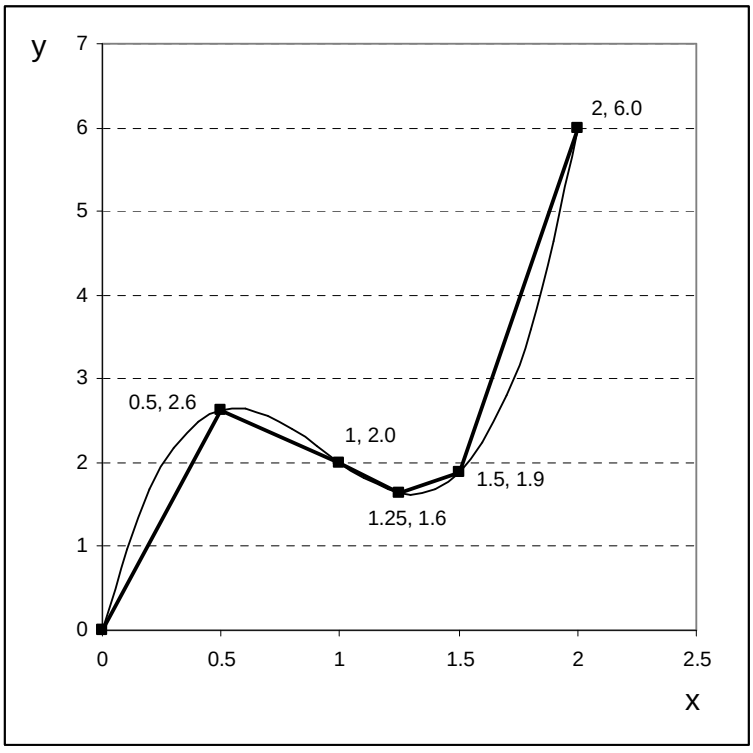


Abbildung 25 – Stückweise Approximation der Funktion $y = 5x^3 - 14x^2 + 11x$ nach Williams[79]

[79] H. Williams, 1993, S. 229

Wird der Ansatz der separablen Funktionen bei nicht-konvexen NLP-Modellen eingesetzt, so kommt Williams zu der Erkenntnis, dass nur dann das korrekt globale Optimum ermittelt wird, wenn zwischen Startwert und dem globalen Optimum kein lokales Optimum liegt. Dies entspricht den Erfahrungen im Rahmen dieser Arbeit mit den bestehenden Solvern. Es wird von Williams empfohlen mehrere Rechnungen mit unterschiedlichen, nach dem Zufallsprinzip ermittelten Startwerten durchzuführen, um dann das maximale der ermittelten Ergebnisse als globales Optimum zu definieren. Da die so ermittelte Extremstelle nur zufällig dem globalen Optimum entspricht, wird dieser Ansatz von ihm erweitert. Williams konvertiert den Ansatz der separablen Funktion zu einem Modell der Integer Programmierung, indem er den Branch-and-Bound-Algorithmus einsetzt[80], wodurch die Ermittlung des globalen Optimums bei NLP-Modellen sichergestellt werden kann[81].

Unterstellt man eine Adaption bestehender LP-Solver in der beschriebenen Weise, ist eine praxisnahe Realisierung des Ansatzes der verteilten Optimierung im Sinne meiner Ausführungen möglich.

Analog zu den ermittelten Testergebnissen hinsichtlich des eingeschränkten Lösungsverhaltens betrachteter Solver (siehe Kaptitel 8.1, speziell Abbildung 21) kommt Ratz[82] unter anderem am Beispiel der 1-dimensionalen Shubert-Funktion[83] ebenfalls zu dem Schluss, dass trotz zahlreicher Theorien auf dem Gebiet der linearen und nicht linearen Optimierung eine Bestimmung des globalen Optimums bei so genannten real-world-Problemen nicht zuverlässig möglich ist.

Aufbauend auf die Definition von Törn[84], nach der globale Optimierungsverfahren bei einer stetigen Funktion f nur dann gegen das

[80] H. Wiliams, 1993, S. 235
[81] Weiterführende Literatur zum Branch-and-Bound Algorithmus siehe:
R.S. Garfinkel, 1979, M. Jeffreys, 1974 , L. Ten-Hwang, 1984, P. Brucker 1994
[82] D. Ratz, 1992, S. 9
[83] Die 1-dimensionale Shubert-Funktion hat 20 lokale maximal Extremstellen, von denen drei zu den globalen Lösungen zählen.

globale Optimum konvergieren, wenn die Folge der durch den Algorithmus generierten Testpunkte überall dicht im kompakten Minimierungsbereich liegen, zeigt Ratz, dass die Intervallrechnung ein Hilfsmittel zur Ermittlung globaler Extremstellen über Teilbereiche des Optimierungsgebietes darstellt. Dabei können in einer einzigen intervallarithmetischen Funktionsauswertung für alle Punkte innerhalb eines Intervalls Aussagen über die Lage der Funktionswerte gemacht werden. Die Intervallauswertung ersetzt somit (unendlich) viele reelle Funktionsauswertungen.

Die von Ratz unter dem Thema der globalen Optimierung beschriebenen Intervallrechnungen gründen wie bei Williams auf dem Algorithmus des Branch-and-Bound. Nachdem bereits 1974 Skelboe[85] die Branch-and-Bound Methodik mit der Intervallarithmetik für die globale Optimierung von Moore[86] kombinierte, gingen Anfang der Neunziger die Forschungsaktivitäten vielmehr in die Verbesserung der Effizienz.

Zusammenfassend lässt sich formulieren, dass Systeme zur Ermittlung des globalen Optimums in der Literatur umfassend vorgestellt werden. Es lassen sich neben den Ansätzen zur Optimierung eines einzelnen Modells ohne lokales Optimum ebenfalls Ausführungen zur Untersuchung von Zielfunktionen mit einer oder mehrer Extremstellen in der in der Literatur finden. Exemplarisch wurden drei Ansätze vorgestellt.

Im ersten Verfahren zur Bestimmung des globalen Optimums wurde eine heuristische Glättungsprozedur zur Vermeidung von lokalen Optima dargestellt. In Verbindung mit Gradientenverfahren wie HSP (Heuristic smoothing procedure), SQP (sequential quadratic programming) und MMA (method of moving asymptotes) kann durch die Methodik der heuristischen Glättung das globale Optimum in realistischer Rechenzeit ermittelt werden.

---

[84] A. Törn, 1989; S. 30-38
[85] S. Skelboe, 1974

In einem zweiten vorgestellten Ansatz werden die separierten Funktionen konvertiert und durch die Implementierung des Branch-and-Bound Algorithmus zu einem Modell der Integer-Programmierung erweitert, wodurch eine Ermittlung des globalen Optimums bei NLP-Modellen verbessert werden kann.

Auch der dritte vorgestellte Ansatz geht vom Algorithmus des Branch-and-Bound aus, findet seinen Schwerpunkt allerdings in der Intervallarithmetik, verbunden mit Verfahren wie Cut-Off-, Monotonie- und Konkavitätstest zur Effizienzsteigerung.

Aus Gründen der Vollständigkeit sei abschließend noch die von Mitra, Levkovitz und Tamiz[87] dargestellte Methode IPM (Interior Point Method) als Alternative zum Simplex-Algorithmus genannt. IPM ist danach stabiler und aufgrund der geringeren Anzahl an notwendigen Iterationsschritten effizienter als der Simplex-Algorithmus. Anderson, Mitra und Parksinson[88] sehen eine derart hohe Effizienz von IPM, dass ein Einsatz von IPM im Rahmen der massiven Parallelität empfohlen wird. Aber auch die Kombination von IPM- und Simplex- Algorithmus wird von Bixby[89] als Möglichkeit aufgezeigt, die Kapazität des Simplex-Algorithmus zu erweitern und die Bestimmung globaler Optima zu verbessern.

Alle vorgestellten Verfahren zeichnen sich durch die korrekte Ermittlung maximaler Extremstellen bei Funktionen mit lokalen Optima aus. Eine Erweiterung des betrachteten Solvers um diese Verfahren trägt damit zur Verbesserung einer praxisnahe Realisierung des Ansatzes der verteilten Optimierung im Sinne dieser Arbeit bei.

---

[86] R.E. Moore, 1966
[87] Mitra,G.; Levkovitz, R, Tamiz, M (1991)
[88] AndersonJ.H.; Mitra, G; Parksinson, D. (1991)
[89] Bixby, R.E.; Gregory, J.W.; Lustig, I.J; Marstend, R.E.; Shanno (1991)

10 Zusammenfassung der Ergebnisse

Wie eingangs herausgestellt ist Gegenstand dieser Arbeit die Entwicklung eines Systems der verteilten Optimierung durch die Verknüpfung von unabhängigen mathematischen Modellen. Die verteilte Optimierung stellt dabei eine Möglichkeit zur Bestimmung des globalen Optimums im Sinne der Ermittlung der gemeinsamen, maximalen Extremstelle aller beteiligten Modelle dar.

Einleitend wurden dazu die in der Rohöl verarbeitenden Mineralölindustrie etablierten Planungssysteme vorgestellt. In Bezug auf mathematische Modelle als Hilfsmittel zur Produktionsplanung in Rohölverarbeitenden Raffinerien ließen sich die Optimierung der Bereiche Logistik und Blending sowie die LP-basierte Optimierung der Gesamtfahrweise unter Maximierung des Deckungsbeitrages als zentrale Anwendungen festlegen.

Zur Ermittlung eines technisch geeigneten Systems zur Implementierung des im Rahmen dieser Arbeit entwickelten Systems der verteilten Optimierung wurden vorgestellte Planungsformen hinsichtlich der Kompatibilität der Schnittstellen, des Verbreitungsgrades des Systems und des eingesetzten Optimieralgorithmus untersucht. Bei der weiterführenden Betrachtung der indirekten und direkten Verfahren zeigten sich alle betrachteten indirekten Verfahren und von den betrachteten direkten Verfahren nur das Downhill-Simplex-Verfahren als geeignet für die Implementierung des entwickelten Algorithmus. Aus dem Bereich der Produktionsplanung erwies sich schließlich die Verknüpfung zweier LP-Systeme für die Umsetzung des Ansatzes der verteilten Optimierung als geeignet.

Als Nachweis der theoretischen Richtigkeit des Ansatzes der verteilten Optimierung konnte ein mathematischer Beweis angeführt werden. Darauf

aufbauend wurde der Algorithmus zur verteilten Optimierung entwickelt und die Implementierung in bestehende LP-Systeme beschrieben.

Hinsichtlich der Praxiseinführung des Systems der verteilten Optimierung wurden der hardwaretechnische, der softwaretechnische und der organisatorische Bereich betrachtet. Umfang und Aufwand der Einführung wurden erörtert.

Basierend auf dem entwickelten System wurden zur Ermittlung der Möglichkeiten und Grenzen systematische Testreihen mit vereinfachten Beispielfunktionen durchgeführt. Als Ergebnis der Testreihen konnten folgende Mindestanforderungen an die Zielfunktion formuliert werden, um eine bestimmungsgemäße Ermittlung des globalen Optimums gewährleisten zu können:

- Die Steigung der Funktion darf nicht in mehr als einem Punkt den Wert 0 annehmen.
- Lokale Optima zwischen den Intervallgrenzen der Zielfunktion sind zu vermeiden.
- Entspricht die maximale Extremstelle der Gesamtfunktion nicht einem der beiden Intervallgrenzen, muss der Funktionswert größer sein, als der Funktionswert mindestens einer Intervallgrenze.

Bei Testreihen mit praxisnahen LP-Modellen zeigte sich, dass sich bei der Verknüpfung zweier Modelle durch Überlagerung Zielfunktionen ergeben, die die oben definierten Mindestanforderungen nicht erfüllen. Eine unmittelbare Umsetzung des Systems der verteilten Optimierung zeigte sich als nur stark eingeschränkt möglich. Dabei ließ sich die begrenzte Lösungskapazität des LP-internen Solvers als begrenzender Faktor identifizieren.

Ansätze zur Verbesserung des LP-internen Solvers wurden anschließend aus der Literatur zitiert und erörtert. Die vorgestellten Verfahren zur Bestimmung

des globalen Optimums zeichnen sich durch die korrekte Ermittlung maximaler Extremstellen bei Zielfunktionen aus, die nicht den oben genannten Mindestanforderungen entsprechen.

Eine Erweiterung des betrachteten Solvers um eines der aufgeführten Verfahren lässt damit eine praxisnahe Realisierung des Ansatzes der verteilten Optimierung im Sinne dieser Arbeit zu.

So lässt sich abschließend formulieren, dass im Rahmen dieser Arbeit ein Algorithmus entwickelt wurde, der entsprechend der eingangs formulierten Zielsetzung die Verknüpfung von mathematischen Modellen mit dem Ziel der verteilten Optimierung ermöglicht.

11 Ausblick

In nachfolgenden Ausblick soll kurz auf denkbare Entwicklungsmöglichkeiten des Ansatzes der verteilten Optimierung im Sinne der Ergebnisse dieser Arbeit eingegangen werden.

In den vorangegangenen Kapiteln wurde bewusst ein modellhafter Charakter im Wesentlichen zu Testzwecken und aus Gründen der Systemtransparenz zu Grunde gelegt. Bei Implementierung einer systemtechnisch unabhängigen zentralen Plattform ist eine Erweiterung auf eine größere Anzahl an verknüpften Planungssegmenten technisch möglich und denkbar. Auch wenn in dieser Arbeit nur beispielhaft das Testsystem mit zwei Planungssegmenten und einer zentralen Plattform betrieben wurde, so können prinzipiell mehrere Planungssegmente ein Angebot auf der zentralen Plattform hinterlegen. Ein nachfragendes Planungssystem kann dann seriell alle verfügbaren Angebote prüfen, ggf. den Prozess der verteilten Optimierung starten und die errechneten Ergebnisse vergleichen.

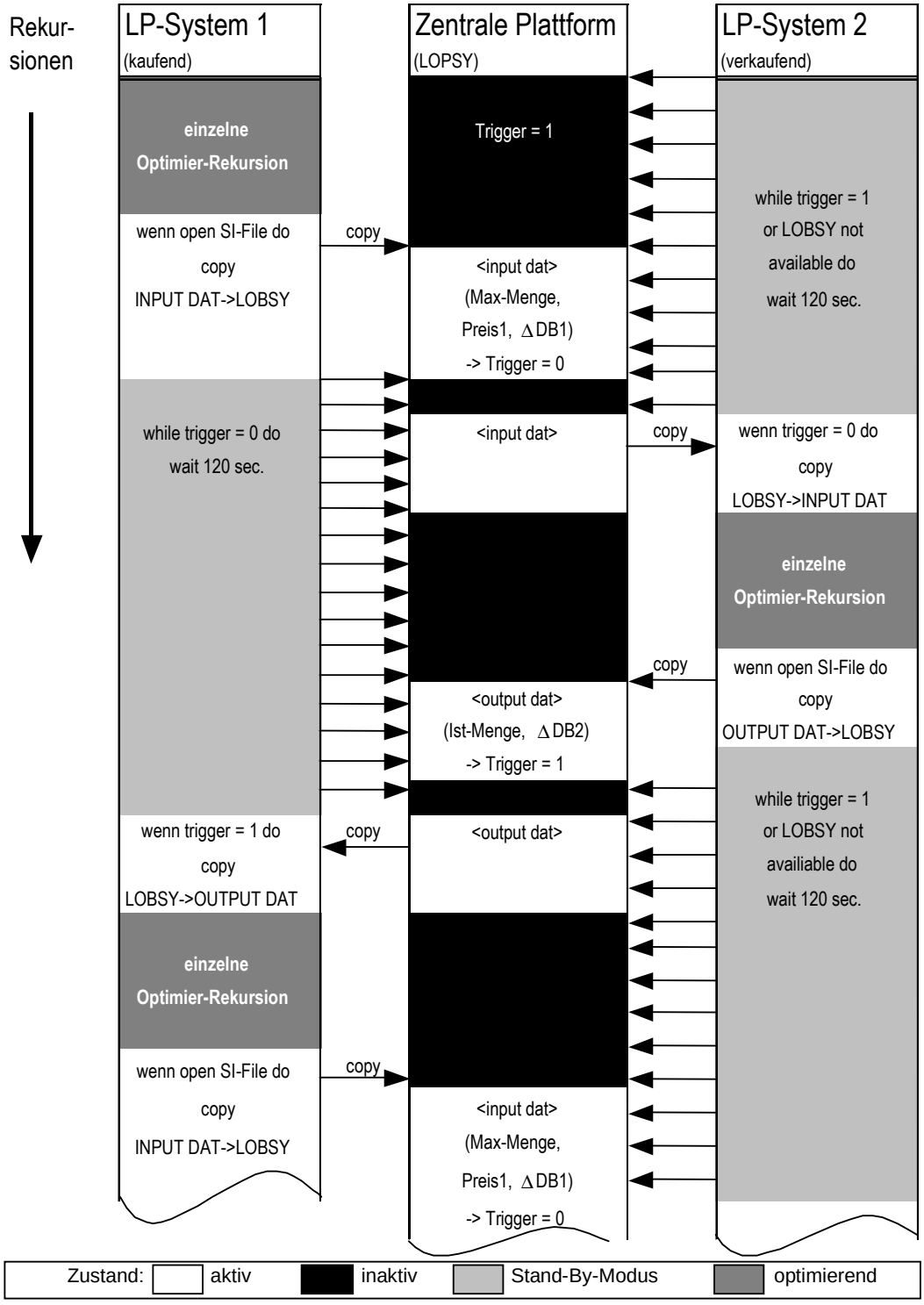
Vergrößert man so die Anzahl der teilnehmenden Systeme, kann ein selbständig expandierendes Optimierungsnetzwerk gleich dem World-Wide-Web entstehen.

Unterstellt man bei den beteiligten Systemen ein entsprechendes Nachfrage- und Angebotsvolumen, so wirken schließlich die Mechanismen der freien Marktwirtschaft. Von volkswirtschaftlicher Bedeutung ist, dass sich hinsichtlich der auszutauschenden Komponenten nicht nur ein globales Optimum zwischen den zwei Systemen ergibt, sondern dass dies über alle Teilnehmer des Optimierungsnetzwerks, die an der Optimierung der betreffenden Komponente beteiligt sind, möglich ist. Das so ermittelte globale Optimum stellt aus wertschöpfender Sicht das wirtschaftliche Optimum dar;

persönliche, politische Präferenzen können dabei ausgeschlossen werden. Ferner sind den beteiligten Systemen im Netzwerk durch das Verhalten des Optimieralgorithmus alle modelltechnischen Restriktionen der Netzwerkteilnehmer, wenn auch implizit, bekannt.

Diese Transparenz unter Ausschluss persönlicher Einflussfaktoren bei theoretisch unendlich erweiterbarer Anzahl an teilnehmenden Systemen kann dann markwirtschaftlich das übergeordnete Optimum ermitteln und so dazu beitragen, Güter und Ressourcen effizient für die Volkswirtschaft nutzbar zu machen.

Anhang 1 – Implementierung des Algorithmus der verteilten Optimierung auf die beteiligten LP-Systeme



Abbildungsverzeichnis:

Literaturverzeichnis:

AndersonJ.H.; Mitra, G; Parksinson, D. (1991)	The Scheduling of Sparse Matrix-Vector Multiplication on a massively parallel DAP computer, Brunel University London
BBN Advanced Computers Inc. (1989)	TC2000 Product Summary, Cambridge
Baker, T.E., Lasdon, L.S. (1985)	Successive linear programming at Exxon. Management Science, 31(3)
Baret, M. (2000)	"Application of neural networking calibrations to an halide ISE array", Talanta 51
Bazaraa, M.S., Sherali, H.D., Shetty, C.M. (1993)	Nonlinear programming: Theory and algorithms
Bixby, R.E.; Gregory, J.W.; Lustig, I.J; Marstend, R.E.; Shanno (1991)	Very Large scale linear Programming: a case study in combination interior point and simplex methods, Rice University, Texas 1991
Blömer, F.; Günther, H.-O. (1998)	LP-based heuristics for scheduling chemical batch processes
Blörk, A.; Germund, D. (1972)	Numerische Methoden
Bock, S. (2000)	Modell und verteilte Algorithmen zur Planung getakteter Fließlinien, Dissertation, Uni Paderborn
Boehm, B. (1995)	COCOMO II – Model Definition Manual
Bomze, I.;Grossmann, W. (1993)	Optimierung – Theorie und Algorithmen
P Brucker, B Jurisch, B Sievers (1994)	A branch and bound algorithm for the job-shop scheduling problem, in: Discrete Applied Mathematics archive Volume 49 , Issue 1-3 (March 1994)
Buchberger, B. (1965)	Ein Algorithmus zum Auffinden der Basiselemente des Restklassenringes nach einem nulldimensionalen Polinomideal, Dissertation der Universität Insbruck

Catchpole, A.R. (1962)	The Application of linear programming to integrated supply problems in the oil industry, Operational Research Quarterly, Vol. 13, no. 2, pp.161-169, 1962
Chaudhuri, B.; Modak J.M. (1998)	"Optimization of fed-batch bioreactor using neural network model", Bioprocess Engineering 19
Chiang, A.C. (1992)	Elements of dynamic optimization
Cortez, E.M.; Park, S.C.; Kim, S. (1995)	The hybrid application of an inductive learning method and a neural network for intelligent information retrieval, Information Processing and Management
Dantzig, G.B. (1949)	Programming in a Linear Structure – Econometrica 17
Dantzig, G.B. (1951)	Maximization of a linear function of variables subject to linear inequalities, In; Activity analysis of production and allocation
Dantzig, G.B. (1955)	Linear programming under uncertainty, Management Science 1
Dantzig, G.B. (1963)	Linear Programming and Extensions, Princeton University Press, Princeton, N.J.
Dantzig, G.B., Thapa, M.N. (1997)	Linear Programming1: Introduction
Diwekar, U.M., Madhavan, K.P., Swaney, R.E. (1989)	Optimization of Multicomponent Batch Destillation Columns – Ind. Eng. Chem. Res., 1001-1017
Eiselt, H. Pederzoli, G., Sandblom C.-L. (1987)	Continuous optimization models (Operations research)
Flynn, Michaeal J. (1966)	Very high speed computing. In: Proceedings of the IEEE, Vol. 54, No. 12, S. 1901-1909
Garfinkel, R.S. (1979)	Branch-and-Bound Methods for Integer Programming.
Garvin, W., Crandall, H., John, J., Spellmann, R. (1957)	Applications of Linear Programming in Oil Industry, Management Science, Vol. 3, 1957, S. 407-430
Gdalevitch, Semen (1994)	Practical applications of linear programming duality
Göpfert, A. (1986)	Optimierung und optimale Steuerung

Haarbrücker, G. (2000)	Sequentielle Optimierung verfeinerter Approximation in der mehrstufigen stochastischen linearen Programmierung (Dissertation)
Heinrich, L.J. (1994)	Systemplanung – Planung und Realisierung von Informatik-Projekten. Band 2. Prozess der Grobprojektierung, der Feinprojektierung und Installation
Hilding, D. (2000)	"A heuristic smoothing procedure for avoiding local optima in optimization of structures subject to unilateral constraints", Structural and multidisciplinary optimization, 2000, S. 29-36
Hungenberg, H. (2001)	Strategisches Management in Unternehmen, 2. Aufl.,Wiesbaden
Jeffrey, M. (1974)	Some ideas on formulation strategies for integer programming problems so as to reduce the nodes generated by a branche-and-bound algorithm, Working paper 74/2, Wootton, Jeffreys and Partners, London
Jin, S.; Ye, K.M.; Shimizu, K.; Nikawa, J. (1996)	Application of artificial neural network and fuzzy control for feedbatch cultivation of recombinant Saccharomyces cerivisiae, Journal Of Fermentation And Bioengineering
Kallrath, J., Wilson, J.M. (1997)	Business optimization using mathematical programming
Kall, P. (1992)	System modeling and optimization
Kemper A./Eickler A. (2004)	Datenbanksysteme
Körkel M. (1999)	Effiziente Verfahren zur Lösung unkapazitierter Standort-Probleme - Dissertaion
Kosmol, P. (1989)	Methoden zur numerischen Behandlung nichtlinearer Gleichungen und Optimierungsaufgaben
Pernul, G., Umland, R. (2003)	Datenbanken in Unternehmen: Analyse, Modellierung und Einsatz
Levenberg, K. (1944)	„A method for the solution for certain nonlinear problems in least-squares", The Quarterly Journal of Mechanics and Applied Mathematics, Vol. 2, S. 164-168
Lin, Z.C.; Chang, D.Y. (1996)	Application of an neural network machine learning model in the selection system of sheet metal bending tooling, Artificial Intelligence In Engineering
Litke, H.-D. (2004)	Projektmanagement: Methoden, Techniken, Verhaltensweisen

Lüling, R. (1996)	Lastverteilungsverfahren zur effizienten Nutzung paralleler Systeme. Dissertation, Universität Paderborn
Magoulas, K., Marinos-Kouris, D., Lygeros, A. (1988), S. 44-48	Actual Blending Liner Programming Model is Developed, Oil & gas Journal, Vol. 86, No. 29, 1998, 44-48
Mautz, R. (2000)	Zur Lösung nichtlinearer Ausgleichsprobleme bei der Bestimmung von Frequenzen in Zeitreihen
Meier, A. (2004)	Relationale Datenbanken
Meier, Andreas; Wüst Thomas (2000)	Objektorientierte und objektrelationale Datenbanken
Mitra,G.; Levkovitz, R, Tamiz, M (1991)	Integration of IPM within Simplex, Brunel University London
Moore, R.E. (1966)	Interval Analysis. Prentice Hall, Engelwood Cliffs, New Jersey
Müller, D. (2005)	CNET Network Inc., Juni 2005, (www.zdnet.de)
Ratschek, H. (1988)	New computer Methods for Global Optimization
Ratz, D. (1992)	Automatische Ergebnisverifikation bei globalen Optimierungsproblemen (Dissertation)
Plantenga, T.D. (1999)	A trust region method for nonlinear programming based on primal interior point. Techniques. In: Sci. Comput. 20
Poensgen, B., Bock, B. (2005)	Function-Point-Analyse
Press, W. H. (1992)	Numerical recipes in C, Cambridge University, Cambridge
Ramakrishnan, R., Gehrke, J. (2002)	Database Management Systems
Ramsey, J.R., Truesdale, P.B. (1990)	Blend Optimization Integration into Refinery-wide Strategy, Oil & Gas Journal, Vol. 88, No. 12, 1990, S. 40-44
Schäffer, U.,Hoffmann, D. (1999)	Handbuch Technikfolgenabschätzung, Berlin
Schöllholm, M. (1999)	Die Optimierung des Blending-Prozesses bei der Benzinproduktion durch verbesserte stationäre Modelle und eine Qualitätsregelung
Shanno, D.F. (1993)	“Computational Methods for Linear Programming” in RUTCOR Reaearch Report RRR19-93, August 1993

Silberschatz, A., Korth, H.F., Sudarshan, S. (2001)	Database System Concepts
Skelboe, S. (1974)	Computation of Rational Interval Funkctions. BIT 4, 87-97
Stepan, A., Fischer E.O.	Betriebswirtschaftliche Optimierung – Einführung in die quantitative Betriebswirtschaftslehre
Stubenvoll, R. (1995)	Helmertransformation zwischen terretrischen und GPS-Netzen und die Bestimmung relativer Geoidundulation, Technische Universität Berlin
Svanberg, K. (1987)	The method of moving asymptotes – a new method for structural optimization . In: Int. J. Num. Meth. Eng. 24, 359-373
Symonds, G.H. (1955)	Linear Programming – The solution of Refinery Problems, ESSO Standard Oil Company
Symonds, G.H. (1956)	Linear Programming Solves Gasoline Refinery and Blending Problems, Ind. And Eng. Chemistry, Vol. 48, No. 3, 1956
Ten-Hwang Lai (1984)	Anomalies in parallel branch-and-bound algorithms, in: Communications of the ACM archive Volume 27 , Issue 6 (June 1984)
Teunissen, P. (1990)	„Nonlinear least squares“, in manusscripta geodaetica Vol. 15, Springer-Verlag
Törn, A., Zilinskas, A. (1989)	Global Optimization. Lecture Notes in Computer Science, No. 350, Springer Verlag Berlin
Tomlin, J.A. (1971)	An Improved Branch-and-Bound Method for Integer Programming, Oper. Res., Vol. 19, 1971
Yin, C.G.; Luo, Z.Y.; Zhou, J.H.; Cen, K.F. (2000)	A novel non–linear programming-based coal blending technology for power plants, Chemical Engineering Research And Design
Törn, A., Zilinskas, A. (1989)	Global Optimization
Vanderbei, R.J. (1997)	Linear Programming: Foundations and Extensions. Internaltional series in Operations Research & Management Science
Williams, H. (1993)	Model solving mathematical programming
Zomaya, A.Y. (1996), Editor	Parallel and Distributed Computing Handbook. McGraw Hill (USA)