Konrad-Zuse-Zentrum für Informationstechnik Berlin
Heilbronner Str. 10, D-10711 Berlin - Wilmersdorf

Atef Abdel–Aziz Abdel–Hamid   Norbert Ascheuer
Martin Grötschel   Herbert Schorer

Simulation und Optimierung einer PC–Fertigung unter Echtzeitbedingungen

Erscheint in:
A. Bachem, M. Jünger und R. Schrader (Hrsg.): *Mathematik in der Praxis – Fallstudien aus Industrie, Wirtschaft, Naturwissenschaften und Medizin*, Springer Verlag

Preprint SC 95–4 (February 1995)

Simulation und Optimierung einer PC–Fertigung unter Echtzeitbedingungen

Atef Abdel–Aziz Abdel–Hamid * Norbert Ascheuer † Martin Grötschel ‡
Herbert Schorer §

1 Problembeschreibung

Die Business Unit PC in Augsburg ist die zentrale Produktionsstätte der Siemens–Nixdorf Informationssysteme (SNI) AG für Personal Computer sowie für einige Periphärgeräte. Das Werk, entworfen nach modernen CIM/CAI–Konzepten (Computer Integrated Manufacturing/ Computer Aided Industry), wurde 1987 errichtet. Bald zeigte sich jedoch, daß es für ein zu geringes Produktionsvolumen ausgelegt war und einige Komponenten des Systems Engpässe im Produktionsbetrieb darstellen. Das Management suchte nach Möglichkeiten, den Produktionsfluß zu verbessern, ohne teure technische Änderungen am System vornehmen zu müssen.

Eine Forschungsgruppe des Konrad–Zuse–Zentrums für Informationstechnik (die ehemals an der Universität Augsburg ansässig war) analysierte, unterstützt von einigen Studenten und Ingenieuren der SNI, den Produktionsfluß und lokalisierte Schwachstellen. Basierend auf diesen Erkenntnissen wurden mathematische Fragestellungen erarbeitet und auf mathematischen Optimierungsverfahren basierende Softwarepakete entwickelt, die jetzt teilweise bei SNI im Einsatz sind.

Im folgenden werden einige dieser Fragestellungen, deren Modellierung und mathematische Behandlung beschrieben. Einige der Ansätze, die hier dargestellt werden sollen, sind teilweise schon in Grötschel ([Grö92]) angesprochen worden.

1.1 Layout der Fertigungshalle

In diesem Abschnitt soll der Produktionsfluß, soweit er für das weitere Verständnis erforderlich ist, skizziert werden. Abbildung 1 zeigt einen Teil des Fertigungssystems, wie wir es zu Projektbeginn 1989 vorfanden. Im ganzen umfaßt diese Halle drei Hochregallager (HRL). Auf jeder Seite der HRL befindet sich eine Montagelinie auf denen PCs, Monitore, Datensichtgeräte und Tastaturen gefertigt werden.

*Konrad–Zuse–Zentrum für Informationstechnik Berlin (ZIB), Heilbronner Str. 10, D–10711 Berlin–Wilmersdorf, on leave from Department of Engineering, Mathematics & Physics, Cairo University, Egypt

†Konrad–Zuse–Zentrum für Informationstechnik Berlin (ZIB), Heilbronner Str. 10, D–10711 Berlin–Wilmersdorf

‡Konrad–Zuse–Zentrum für Informationstechnik Berlin (ZIB), Heilbronner Str. 10, D–10711 Berlin–Wilmersdorf, and Technische Universität Berlin

§Siemens–Nixdorf Informationssysteme AG, Business Unit PC, Postfach 11 18 40, D–86043 Augsburg

Abbildung 1: Systemlayout

Die Anlieferung der zur Montage der Geräte benötigten Einzelteile (Leiterplatten, Bildröhren, Kabel, etc.) an der **Warenannahme** erfolgt in genormten Behältern (TE = Transporteinheit), die hier datentechnisch erfaßt werden. Die Identifikation der TE im System erfolgt über einen Strichcode, der an den Kisten angebracht ist und von Scannern gelesen werden kann. Abhängig von der Montagelinie, an der die Teile benötigt werden, wird bei der Einbuchung in einem der drei Hochregallager ein Lagerplatz gesucht und für diese TE reserviert.

Über Rollenbänder wird die TE zu einem Übergabepunkt gebracht, wo sie von einem der zwölf Fahrzeuge des **Fahrerlosen Transportsystems** (FTS) aufgenommen und an die Einlagerstation des Hochregallagers gebracht wird. Die batteriebetriebenen Fahrzeuge des FTS stehen über Funk mit einem zentralen Rechner in Verbindung, der u.a. die Zuweisung von Fahrzeugen zu Fahraufträgen durchführt und über die Fahrtroute der Fahrzeuge entscheidet.

Die **Hochregallager** (HRL) dienen hauptsächtlich als Materialpuffer zwischen der Warenannahme und den Montagelinien, die sich zu beiden Seiten der Hochregallager befinden. Ein HRL besteht aus zwei gegenüberliegenden Regalreihen, in denen für die drei verschieden großen TE-Arten Lagerplätze vormontiert worden sind. In dem zentralen Gang fährt ein **Regalbediengerät** (RBG), welches alle Transportaufgaben innerhalb des HRL verrichtet. Nachdem eine TE die Einlagerstation des HRL erreicht hat, wird automatisch ein Transportauftrag (TA) generiert und an den RBG–Kontrollrechner übermittelt. Der Rechner entscheidet, ob dieser Auftrag oder ob ein anderer vorliegender Auftrag, wie Auslagern oder Umlagern, als nächster ausgeführt wird.

Nachdem die TE eingelagert worden ist, bleibt sie so lange auf ihrem Lagerplatz, bis die in ihr enthaltenen Teile an der Montagelinie benötigt werden. Dies sind i.d.R. 1–3 Tage. Im unteren Bereich der HRL befindet sich ein Entnahmebereich, in der die TE an der Stelle, an der sie an der Montagelinie benötigt wird, bereitgestellt werden kann. Die Nachlieferung neuer TE in den Entnahmebereich, d.h., die Generierung eines entsprechenden TA für das RBG erfolgt entweder automatisch, falls eine Lichtschranke feststellt, daß der Entnahmeplatz leer ist, oder durch manuelle Eingabe in das Softwaresystem.

Auf jeder Seite der HRL befindet sich eine **Montagelinie**. Eine dieser Montagelinien ist vollautomatisiert, d.h., dort wird die Montage von Robotern durchgeführt. An allen anderen Linien erfolgt die Montage manuell. Bevor ein Produktlos auf der Montagelinie gestartet wird, wird überprüft, ob die benötigten Teile im entsprechenden HRL verfügbar sind. Bei einem Wechsel des Produktloses kann eine Neubestückung des Entnahmebereiches notwendig sein.

Fertig montierte Geräte (außer Tastaturen) werden nach einer kurzen Funktionsprüfung über Rollenbänder in einen Dauertestbereich gebracht, wo unter erschwerten Bedingungen (z.B. unter erhöhter Temperatur) für einen geräteabhängigen Zeitraum (6–24 Stunden) Testprogramme ablaufen, die die volle Funktionsfähigkeit der Geräte überprüfen.

Nach der Auswertung der Dauertestergebnisse und einem kurzen Endtest, werden die Geräte verpackt und zum Versand transportiert. Falls im Test ein Gerätefehler festgestellt worden ist, wird dieser behoben, und das Gerät wieder in den Testbereich eingeschleust.

1.2 Konkrete Fragestellungen

Nach einer detaillierten Analyse des Systems kristallisierten sich einige Fragestellungen heraus, von denen wir der Meinung waren, daß sie durch den Einsatz von Verfahren der mathematischen Optimierung, und hier insbesondere der Kombinatorischen Optimierung, verbessert werden können. Nach Diskussionen mit Ingenieuren und Informatikern von SNI wurden einige dieser auch mathematisch sehr interessanten Probleme wieder fallengelassen, da ihre Umsetzung zu große Änderungen im Softwaresystem nach sich gezogen hätte. Im folgenden wollen wir einige der verbleibenden Fragestellungen beschreiben.

Fahrbewegungen des RBG.

In der Regel hat das Regalbediengerät (RBG) eine Liste von noch nicht abgearbeiteten Transportaufträgen vorliegen, die im Normalbetrieb bis zu 5 Aufträge umfaßt. Nach Störungen des Geräts kann diese Liste durchaus auf 30–40 (und mehr) Aufträge anwachsen. Um den Zeitverzug an den Montagelinien und den Rückstau an der Warenannahme so gering wie möglich zu halten, sollen diese nach Beseitigung der Störung natürlich so schnell wie möglich abgearbeitet werden. Es zeigte sich jedoch, daß das RBG im Hochlastbetrieb einen Engpaß darstellt. Dies war im wesentlichen dadurch bedingt, daß das RBG aufgrund einer starren Prioritätensteuerung bei der Abarbeitung dieser Transportaufträge einen hohen Anteil an Leerfahrten aufzuweisen hatte. Das Ziel war es nun, eine Strategie zu entwerfen, bei der der Anteil der Leerfahrten so gering wie möglich gehalten wird.

Einlagerung von TE in die Hochregallager.

Sobald eine TE an der Warenannahme eingebucht wird, wird ihr ein Lagerplatz in einem der Hochregallager zugewiesen. Dieser Platz sollte so gewählt werden, daß die Fahrzeiten des RBG zur Einlagerung der TE und zum Nachfüllen des Entnahmebereichs so gering wie möglich sind. Zwischen Platzzuweisung und endgültiger Einlagerung liegen aber, bedingt durch Stau– und Transportzeiten, oft über zwei Stunden. Dieser Lagerplatz ist in der Zwischenzeit für eine weitere Verwendung blockiert und günstigere Lagerplätze, die in der Zwischenzeit frei geworden sind, finden keine Berücksichtigung mehr. Zudem ist die vorhandene Zuweisungsmethode ein lokales Suchverfahren; globale Zuweisungsaspekte, in die alle noch nicht eingelagerten TE einbezogen werden, werden nicht berücksichtigt. Dies kann zu einer höheren Fahrzeitsumme führen. Unser Ziel war es, eine "globale" Einlagerstrategie zu entwickeln, die die Summe der RBG–Fahrzeiten so gering wie möglich hält.

Dauertestoptimierung.

Ein Produktionsziel ist es, die Verweildauer der Geräte im Dauertestbereich so gering wie möglich zu halten. Da die Testzeit der Geräte fest vorgegeben ist, besteht im wesentlichen nur eine Einflußmöglichkeit auf die folgenden drei Zeiten: Die Fahrzeit des RBG zum und vom Testplatz, die Wartezeit auf den Beginn des Testzyklus und die Wartezeit nach Ende der Testzeit bis zur Auslagerung. Da die RBG–Fahrzeit im Vergleich zu den anderen beiden Zeiten sehr kurz ist, kann sie für die Minimierung der Verweildauer vernachlässigt werden. Dies war aber das Entscheidungskriterium, das bisher bei der Testplatzzuweisung Verwendung fand.

Das Dauertestregal ist in Stromzyklen eingeteilt, die jeweils im Abstand von zwei Stunden neu bestromt werden. Die Testzeit eines Gerätes beginnt nicht ab dem Zeitpunkt der Einlagerung, sondern ab dem Zeitpunkt der Neubestromung. Dies bedeutet, daß sich für Geräte, die auf ungünstigen Lagerplätzen eingelagert werden, die Testzeit um bis zu zwei Stunden verlängern kann.

Gesucht werden Verfahren, die für ankommende Geräte einen Testplatz bestimmen und festlegen, welches Gerät als nächstes ausgelagert werden soll, so daß die Summe der Verweilzeiten minimiert wird und evtl. eine maximale Verweildauer nicht überschritten wird.

2 Modellierung

Aus Platzgründen können hier nicht alle Modellierungsaspekte für die in Kapitel 1 genannten Probleme beschrieben werden. Für das erstgenannte Problem, der Optimierung der Fahrbewegungen des RBG, werden wir die Modellierung, die datentechnische Umsetzung und die erzielten Resultate detailliert beschreiben. Für alle anderen Fragestellungen werden wir das nur kurz skizzieren.

Für das weitere Verständnis setzen wir Grundbegriffe der Graphentheorie, der Linearen und Ganzzahligen Optimierung voraus. Für eine Einführung in diese Teilgebiete der Mathematik verweisen wir auf Standardwerke, wie z.B. Bondy und Murty ([BM76]), Jungnickel ([Jun87]), Chvátal ([Chv83]), Schrijver ([Sch86]) und Nemhauser und Wolsey ([NW88]).

Die klassische Theorie der Algorithmen geht davon aus, daß der Algorithmus die volle Information über die Inputdaten besitzt. Für die beschriebenen Anwendungen benötigen wir aber sogenannte **Online–Algorithmen**, die Entscheidungen treffen müssen, die nur auf unvollständiger Information basieren. Dies ist z.B. der Fall wenn der RBG–Controller entscheiden muß, welcher Transportauftrag als nächster auszuführen ist. Diese Entscheidung muß getroffen werden ohne Kenntnis darüber, welche Transportaufträge als nächste generiert werden. Die getroffene Entscheidung kann sich als ungünstig herausstellen, kann aber im nachhinein nicht mehr revidiert werden.

Da die zu entwerfenden Verfahren im täglichen Produktivbetrieb eingesetzt werden sollen, mußte bei deren Entwicklung besonderes Augenmerk auf ihre **Echtzeit-Tauglichkeit** gelegt werden. Dies bedeutet u.a., daß die Belastung der Rechner, auf denen gleichzeitig andere für die Produktion wichtige Prozesse laufen, möglichst gering gehalten wird und daß die Datenanforderung der Verfahren, die Berechnung und auch die Rückübertragung der berechneten Daten in einem vorgegebenen Zeitraum zu gewährleisten ist.

Die Theorie der Online–Algorithmen steckt noch in den "Kinderschuhen" und ist im Hinblick auf die vorliegenden Anwendungen wenig hilfreich. (Einen guten Überblick über den aktuellen Stand der Forschung liefern McGeoch und Sleator ([MS]) und Albers ([Alb93])). Zur Behandlung der genannten Fragestellungen mag es deshalb bessere Wege als die im folgenden vorgestellten geben, aber zur Erarbeitung von theoretisch besser fundierten Verfahren ist sicherlich weitere Grundlagenforschung erforderlich.

2.1 Modellierung des Gesamtsystems

Es sei noch einmal explizit betont, daß unser Ziel die Verbesserung des Gesamtproduktionsflusses und nicht nur die Optimierung von Teilkomponenten ist. Ein prinzipielles Problem bei solch komplexen Fertigungssystemen ist zu beurteilen, wie die Verbesserung einer Komponente sich im Hinblick auf das Gesamtsystem auswirken wird. Dies gilt insbesondere, wenn mehrere Optimierungsprogramme unabhängig voneinander ablaufen. Da das vorliegende Fertigungssystem zu groß und zu komplex ist, um es als Ganzes in vernünftiger Weise mathematisch modellieren zu können, entschlossen wir uns, auf ein **Simulationsmodell** zurückzugreifen, welches alle wesentlichen Komponenten des Systems und ihre Interdependenzen abbildet.

Um unsere Vorstellungen an Genauigkeit, Geschwindigkeit und Flexibilität des Simulationsprogramms gewährleistet zu sehen, griffen wir nicht auf kommerzielle Simulationssoftware zurück. Statt dessen implementierten wir, basierend auf Prinzipien der ereignisorientierten Simulation (siehe hierzu z.B. Page et al. ([1988])), ein auf unsere Anforderungen zugeschnittenes Simulationsprogramm.

Die Auswirkungen von Veränderungen im Verhalten von Teilkomponenten, die durch den Einsatz mathematischer Optimierungsverfahren oder durch Änderung technischer Parameter bedingt sind, sollen mit Hilfe des Simulationsprogramms überprüft werden. Um verläßliche Aussagen zu gewinnen, war eine Modellvalidierung unerläßlich. An dieser Stelle kann keine vollständige Beschreibung aller Validierungsergebnisse erfolgen, und wir wollen den (sehr aufwendigen) Validierungsprozeß deshalb nur für die Fahrbewegungen des RBG andeuten. Die Ergebisse sind in Tabelle 1 zusammengefaßt.

Über den Zeitraum einer Woche wurde in einem der HRL jeder generierte Transportauftrag (TA) und jede Fahrbewegung des RBG protokolliert. Die generierten TA dienten als Input für das Simulationsprogramm, und die realen Fahrten wurden mit den Fahrbewegungen in der Simulation verglichen. Am ersten Tag lagen 416 Transporte vor, ein Fahrzyklus, bestehend aus Leerfahrt, beladener Fahrt und TE–Handling, dauerte im realen System durchschnittlich 76,27 Sekunden, in der Simulation 76,83 Sekunden. Dies entspricht einer prozentualen Abweichung von 0,73 %. Die durchschnittliche Abweichung zwischen den Zeiten betrug 2,42 Sekunden, die maximale Abweichung 20 Sekunden.

Die Validierungsergebnisse wurden als sehr zufriedenstellend eingestuft, und wir kamen überein, Aussagen zu akzeptieren, die auf Simulationsergebnissen mit unserem System basieren. Für weitere Details sei auf Ascheuer ([Asc95]) verwiesen.

	# TA	ØSNI	ØSIM	ØDEV	max DEV	% DEV	Time
1	416	76,27	76,83	2,42	20	0,73	11,47
2	423	75,45	76,34	1,96	11	1,17	11,60
3	403	78,81	78,95	2,01	9	0,18	11,17
4	399	76,19	76,58	2,09	9	0,51	11,08
5	450	77,11	76,73	2,06	20	0,49	12,30

# TA	:	Anzahl der TA, die im vorliegenden Zeitraum generiert wurden
ØSNI	:	Durchschnittliche Dauer eines Fahrzyklus bei SNI
ØSIM	:	Durchschnittliche Dauer eines Fahrzyklus in der Simulation
ØDEV	:	Durchschnittliche Abweichung dieser beiden Zeiten
max DEV	:	Maximale Abweichung dieser beiden Zeiten
% DEV	:	Prozentuale Abweichung dieser beiden Zeiten
Time	:	CPU–Zeit in Sekunden auf SIEMENS MX 300 für die Simulation von 18 Stunden (Angabe in Sekunden)

Tabelle 1: Validierungsergebnisse der Hochregallager-Simulation

2.2 Optimierung der RBG–Fahrbewegungen

Wie im letzten Kapitel beschrieben, ist es das Ziel, unter Online–Bedingungen eine vorhandene Menge von Transportaufträgen so anzuordnen, daß die Summe der Leerfahrzeiten zwischen den einzelnen Aufträgen so gering wie möglich ist.

Mit jedem Transportauftrag und der aktuellen Position des RBG identifizieren wir einen Knoten in dem gerichteten Graphen (Netzwerk) $D = (V, A)$ mit Knotenmenge $V = \{1, ..., n\}$ und Bogenmenge A. Der Knoten 1 entspreche fortan der RBG–Position. Ein Bogen $(i, j) \in A$ repräsentiert die Möglichkeit, Transportauftrag j direkt nach i auszuführen. Da Knoten 1 immer der erste in einer zulässigen Sequenz von TA sein muß, erhalten wir

$$A := \{(i, j) \mid i, j \in V \setminus \{1\}\} \cup \{(1, j) \mid j \in V \setminus \{1\}\}.$$

Jeder Bogen $(i, j) \in A$ erhält einen Kostenkoeffizienten c_{ij}, der der Leerfahrzeit zwischen i und j entspricht. Offensichtlich gilt meistens $c_{ij} \neq c_{ji}$. Das Problem ist nun, einen Weg in D zu finden, der jeden Knoten genau einmal besucht und minimale Länge hat. Im graphentheoretischen Sinne entspricht dies der Lösung eines **Asymmetrischen Hamiltonschen Wege–Problems (AHWP)**.

Das AHWP läßt sich als binäres lineares Programm formulieren. Hierzu führen wir für jeden Bogen $(i, j) \in A$ eine binäre Variable x_{ij} ein, die wie folgt zu interpretieren ist:

$$x_{ij} := \begin{cases} 1, & \text{falls } j \text{ nach } i \text{ ausgeführt wird,} \\ 0, & \text{sonst.} \end{cases}$$

Im folgenden sind die Bogenmengen $A(W), \delta^-(v), \delta^+(v)$ wie folgt definiert:

$$\begin{aligned} A(W) &:= \{(i, j) \in A \mid i, j \in W\}, \\ \delta^-(v) &:= \{(i, v) \in A \mid i \in V \setminus \{v\}\}, \\ \delta^+(v) &:= \{(v, j) \in A \mid j \in V \setminus \{v\}\}. \end{aligned}$$

Für eine gegebene Kantenmenge $B \subseteq A$ bezeichnet $x(B) := \sum_{(i,j)\in B} x_{ij}$.
Einer optimalen Sequenz der TA entspricht nun offensichtlich eine Lösung des folgenden linearen 0/1-Optimierungsproblems:

$$
\begin{array}{rlll}
 & \min c^T x & & \\
s.\,t.\ (1) & x(A) & = n-1 & \\
(2) & x(\delta^-(i)) & \leq 1 & \forall\, i \in V \\
(3) & x(\delta^+(i)) & \leq 1 & \forall\, i \in V \\
(4) & x(A(W)) & \leq |W|-1 & \forall\, W \subseteq V,\ 2 \leq |W| \leq n-1 \\
(5) & x_{ij} & \in \{0,1\} & \forall\, (i,j) \in A
\end{array}
$$

Wir haben eine lineare Zielfunktion unter den Nebenbedingungen (1) – (5) zu minimieren. Hinter Gleichung (1) verbirgt sich der Umstand, daß jeder TA in einer zulässigen Sequenz enthalten sein muß. Ungleichungen (2) bzw. (3) bewirken, daß jeder Transportauftrag höchstens einen Vorgänger bzw. Nachfolger hat. Das Ungleichungssystem (4) verhindert Zyklen innerhalb der Auftragssequenz. Die Bedingung (5) erzwingt die Ganzzahligkeit der Lösung.

Das AHWP gehört im komplexitätstheoretischen Sinne zur Klasse der $\mathcal{NP}$–schwierigen Probleme. Jedoch sind für die vorliegenden Größenordnungen (bis zu 50 Knoten) einige Verfahren bekannt, welche die Problembeispiele in vernünftiger Zeit optimal lösen.

Praktische Umsetzung.

Die optimale Sequenz der TA ist jeweils neu zu berechnen, falls ein neuer TA generiert oder ein noch nicht abgearbeiteter TA storniert wird. Da es nicht passieren darf, daß das RBG auf das Ende der Berechnung der optimalen Sequenz wartet, implementierten wir einen dreistufigen Optimierungsprozeß. Wir beschreiben hier die Funktionsweise für den Fall, daß das RBG eine Liste von TA vorliegen hat und ein neuer TA generiert wird.

In einer ersten Stufe wird eine schnelle Heuristik aufgerufen, die versucht, den neuen TA möglichst günstig in die bestehende Abarbeitungssequenz einzuordnen. Diese Heuristik liefert praktisch ohne Zeitverzug eine zulässige Lösung. Falls noch Rechenzeit zur Verfügung steht, wird in einer zweiten Stufe ein aufwendigeres heuristisches Verfahren aufgerufen. In einer dritten Stufe wird das AHWP mittels eines Branch&Bound–Codes von Fischetti und Toth ([FT92]) optimal gelöst. Falls in einer dieser Berechnungsstufen eine bessere Sequenz gefunden wird, wird diese dem RBG–Controller zur Verfügung gestellt. Falls während der Stufen zwei oder drei jedoch ein neuer TA generiert wird, wird der Optimierungsprozeß abgebrochen, und mit dem neuen Problem wieder gestartet.

Die Rechenergebnisse zeigten aber, daß die implementierten Verfahren so schnell sind, daß es so gut wie nie erforderlich war, eine Stufe des Optimierungsprozesses vorzeitig zu beeenden. Für einen Vergleich verschiedener Heuristiken für dieses Problem sei auf Abdel–Hamid, Ascheuer und Grötschel ([AAG95]) verwiesen.

Rechenergebnisse.

Anhand der realen Produktionsdaten, die auch für die Simulationsvalidierung aufgenommen wurden, sollte das Optimierungsverfahren mit der alten SNI–Strategie verglichen werden. Die Ergebnisse, die in der Simulation erzielt wurden, sind in Tabelle 2 zusammengefaßt.

Ein Blick in die Spalte mit der prozentualen Verbesserung (V %) zeigt, daß diese lediglich bei 6–8% lag. Dies waren natürlich enttäuschende Ergebnisse. Eine Analyse der Inputdaten

	TA	LF-S	LF-U	V %	MAX-TA	ØTA
1	416	20,72	19,00	8,30	8	2,08
2	423	20,42	19,08	6,56	8	1,82
3	403	20,37	18,87	7,36	7	2,19
4	399	20,09	18,49	7,96	9	1,91
5	450	20,91	19,26	7,89	9	2,00

TA : Anzahl der TA, die im vorliegenden Zeitraum generiert wurden
LF-S : durchschnittliche Leerfahrzeit zwischen den TA bei SNI–Prioritätensteuerung (Angabe in Sekunden)
LF-U : durchschnittliche Leerfahrzeit bei Steuerung der Abarbeitung über das beschriebene Optimierungsmodul (Angabe in Sekunden)
V % : Verbesserung in % ($\frac{\text{(LF-S) - (LF-U)}}{\text{(LF-S)}} \cdot 100$)
MAX-TA : Maximale Anzahl von TA, die sich gleichzeitig im Auftragspool befanden.
ØTA : durchschnittliche Anzahl von TA, die sich gleichzeitig im Auftragspool befanden.

Tabelle 2: Ergebnisse mit Originaldaten

zeigte aber, daß in dem Zeitraum, in dem die Daten aufgenommen worden sind, keine Hochlastsituation aufgetreten ist und es zudem auch keine wesentlichen Ausfälle des RBG gab. So lagen im Maximum nur 7–9 TA, im Durchschnitt nur zwei TA, gleichzeitig vor. Es gab also praktisch kaum "Optimierungsspielraum".

Im Unterschied zum obigen Systemzustand besteht ein echter Bedarf an Optimierungsverfahren aber im Hochlastbetrieb, wenn wirklich Engpässe auftreten. Da eine erneute Datenaufnahme zu aufwendig erschien, entschlossen wir uns, innerhalb der Simulation künstlich Hochlastbedingungen zu erzeugen. Wir fügten einfach einen zweistündigen Ausfall des RBG zu den Inputdaten hinzu und beschränkten unsere Analyse auf den so erzeugten Zeitraum mit Hochlastbetrieb. Die Ergebnisse sind in Tabelle 3 zusammengefaßt.

Wir stellen fest, daß die erzielten Verbesserungen hier signifikant sind. Sie liegen im Bereich von 30–40%. Falls in dem vorgegebenen Zeitraum mehr TA vorlagen, konnten in gleicher Zeit auch mehr TA ausgeführt werden (siehe Spalte "Plus").

Zusammenfassend kann festgestellt werden, daß die Umstellung von einer einfachen Prioritätensteuerung auf ein mathematisches Optimierungsverfahren zu einer Entschärfung eines potentiellen Engpasses geführt hat. Das beschriebene Optimierungsmodul ist seit Anfang 1992 bei SNI in sechs Hochregallagern im Produktiveinsatz. Die in der Simulation erzielten Ergebnisse konnten im täglichen Produktivbetrieb bestätigt werden.

Modellkritik.

Obwohl die erzielten Verbesserungen beeindruckend sind, wollen wir an dieser Stelle einige kritische Bemerkungen zu der verwendeten Modellierung anbringen. Es sollte aufgefallen sein, daß sowohl das Modell als auch die Optimierungsverfahren statischer Natur sind. Wir haben für eine gegebene Menge von TA eine optimale Sequenz berechnet, und gingen stillschweigend

	TA	LF-S	LF-U	V %	MAX-TA	ØTA	Plus
1	50	17,98	11,50	36,00	29	13,32	3
2	49	19,48	14,72	24,44	20	8,32	0
3	50	19,74	11,33	42,60	26	12,23	3
4	49	18,14	10,34	43,00	31	15,24	3
5	50	19,31	13,77	28,69	25	13,08	1

Einträge wie in Tabelle 2

Plus : Anzahl der TA, die in der vorgegebenen Zeit zusätzlich ausgeführt werden konnten

Tabelle 3: Originaldaten mit künstlicher Störung

davon aus, daß diese auch in dieser Reihenfolge ausgeführt wird. In der Regel wird dies aber nicht der Fall sein, da die Generierung eines neuen TA eine Reoptimierung, d.h. die Berechnung einer neuen Sequenz, erfordert. Das vorliegende Problem ist somit also kein AHWP, sondern ein **Online–AHWP**.

Alle verwendeten Stufen des Optimierungsmoduls, auch der Branch&Bound–Algorithmus, der das AHWP optimal löst, sind somit lediglich heuristische Verfahren zur Lösung des Online–AHWP. Zur Abschätzung der Güte dieser Verfahren ist es notwendig, untere Schranken für den Optimalwert zu bestimmen. Ascheuer ([Asc95]) führte die Berechnung dieser Schranken auf die Bestimmung von Optimalwerten für AHWP mit Nebenbedingungen (Präzedenzrelationen, Zeitfenster) zurück, die über komplexe Branch&Cut–Algorithmen bestimmt werden. Die Verwendung statischer Verfahren für ein dynamisches Online–Problem spiegelt sich in den so erzielten Schranken wider.

Bei den vorliegenden Datensätzen lagen die mit dem Optimierungsmodul erzielten Lösungen höchstens 20 – 40%, in einem Beispiel bis zu 90% von einer bestmöglichen Online–Strategie entfernt. Selbst in Anbetracht der Tatsache, daß diese Schranken nicht scharf sind, glauben wir, daß dynamische Verfahren, basierend auf einem Online–Modell, sicherlich noch bessere Ergebnisse produziert hätten. Uns ist bis heute aber kein Modell bekannt, welches dies in vernünftiger Weise abbildet. Da aber viele Anwendungen einen solchen Online–Charakter aufweisen, ist weitere Grundlagenforschung auf diesem Gebiet unbedingt erforderlich.

2.3 Einlagerung von Transporteinheiten

Bei dieser Problemstellung ging es darum, ankommenden Transporteinheiten (TE) Lagerplätze in den Hochregallagern zuzuweisen. Da das RBG in Hochlastzeiten einen Engpaß darstellt, ist es das Ziel, seine Fahrzeiten im Hochlastbetrieb so gering wie möglich zu halten.

Verbesserte Zuweisungsstragie.

Eine erste Lagerplatzvergabe soll nach wie vor bei der Einbuchung an der Warenannahme erfolgen. Jedoch ist dieser Platz nicht fix für die TE reserviert, sondern kann (im Idealfall) bis zur endgültigen Einlagerung noch mit anderen Lagerplätzen getauscht werden, falls dies für das Gesamtsystem günstiger erscheint. In das optimierte Zuweisungsverfahren sollen alle noch nicht eingelagerten TE einbezogen werden.

Modellierung.

Es werden alle TE, die eingebucht, aber noch nicht eingelagert sind, zu einem Containerpool C zusammmengefaßt. Die freien Lagerplätze kommen in einen Lagerplatzpool L. Analog zu dem Modellierungsschritt bei der Leerfahrzeitminimierung werden 0/1–Entscheidungsvariablen x_{ij} für jede mögliche Zuweisung von TE zu Lagerplatz eingeführt, wobei gelten soll

$$x_{ij} := \begin{cases} 1, & \text{Container } i \in C \text{ wird auf Lagerplatz } j \in L \text{ eingelagert,} \\ 0, & \text{sonst.} \end{cases}$$

Zu jeder "Einlagermöglichkeit" x_{ij} assoziieren wir einen **Kostenkoeffizienten** c_{ij}, der der absoluten Fahrzeit entspricht, die entsteht, falls Behälter i auf Platz j eingelagert wird. Hierbei sind sowohl die Fahrzeit c_{ej} für die Einlagerung in den Lagerplatz j als auch die Fahrzeit c_{jp} für die Nachfüllung des Entnahmebereichs p zu berücksichtigen, d. h.

$$c_{ij} := \lambda_1 \cdot c_{ej} + \lambda_2 \cdot c_{jp} \ .$$

Die Kostenfunktion wird über Strafkostenparameter $\lambda_i \geq 1$ gesteuert, die dem aktuellen Systemzustand anzupassen sind. Im "Normalbetrieb" sollte $\lambda_1 < \lambda_2$ gelten; dadurch werden die TE nahe an ihrem Entnahmebereich gelagert. Falls es der Systemzustand erfordert, kann durch Änderung der Parameter λ_i ein anderes Verhalten erzwungen werden. Dies könnte z.B. der Fall sein, falls vor der Einlagerstation der Hochregallager im Bereich des FTS ein Rückstau entsteht. Dann kann durch $\lambda_1 > \lambda_2$ erzwungen werden, daß Lagerplätze nahe an der Einlagerstation bevorzugt werden. Dadurch wird der Prozeß der Einlagerung beschleunigt, und somit der Rückstau so schnell wie möglich aufgelöst. Die TE, die auf Lagerplätze zugewiesen werden mußten, die weit vom Entnahmebereich entfernt sind, können im Rahmen einer späteren Umlagerung auf günstigere Plätze transportiert werden.

Auf Lagerplätzen von zwei TE–Arten (Gitterboxen/Paletten und Sonderpaletten) kann nur eine TE pro Lagerplatz gelagert werden. Für die Zuweisung dieser TE–Arten ergibt sich somit folgendes binäres lineares Programm:

$$\begin{array}{lllll} & & \min c^T x & & \\ s.t. & (1) & \sum_{j \in L} x_{ij} & = 1 & \forall\, i \in C \\ & (2) & \sum_{i \in C} x_{ij} & \leq 1 & \forall\, j \in L \\ & (3) & x_{ij} & \in \{0,1\} & \end{array}$$

Gleichung (1) erzwingt, daß jedem Auftrag genau ein Lagerplatz zugewiesen wird; Ungleichung (2) bewirkt, daß jedem freien Lagerplatz höchstens eine TE zugewiesen wird.

Die Modellierung der Zuweisung der dritten TE–Art (Eurofixbehälter) auf die entsprechenden Lagerplätze ist etwas komplexer. Diese Behälter werden in Schächten gelagert, deren Gesamtkapazität drei Lagereinheiten entspricht. Ein kleiner Eurofixbehälter beansprucht eine Einheit, ein großer zwei. Der Auftragspool C muß somit aufgeteilt werden in Aufträge für kleine Behälter C_s und große Behälter C_t. In den Lagerplatzpool werden alle Schächte k aufgenommen, die vielleicht schon teilbelegt sind, aber immer noch eine positive Lagerkapazität cap_k besitzen.
Wir erhalten somit folgendes ganzzahlige lineare Programm:

$$\begin{array}{lllll} & & \min c^T x & & \\ s.t. & (1) & \sum_{j \in L} x_{ij} & = 1 & \forall\, i \in C \\ & (2) & \sum_{i \in C_t} \sum_{j \in C_s} (2 \cdot x_{ik} + x_{jk}) & \leq cap_k & \forall\, k \in L \\ & (3) & x_{ij} & \in \{0,1\} & \end{array}$$

Wir minimieren eine lineare Zielfunktion unter einer Reihe von Nebenbedingungen. Gleichung (1) erzwingt, daß jedem Auftrag genau ein Lagerplatz zugewiesen wird; Ungleichung (2) bewirkt, daß die Restkapazität des Lagerschachtes nicht überschritten wird.

Praktische Lösung des Problems.

Das erste Problem der Lagerplatzzuweisung entspricht im graphentheoretischen Sinne einem Zuordnungsproblem (Assignment Problem, AP). Das Zuordnungsproblem gehört zur Klasse der polynomial lösbaren Probleme, für die effiziente Algorithmen zur Berechnung einer Optimallösung bekannt sind.

Obwohl das Minimierungsproblem für die Behälter ähnlich aussieht, ist es ein $\mathcal{NP}$–schwieriges Optimierungsproblem, ein Generalized Assignment Problem (GAP). Im Unterschied zum AP kann nicht erwartet werden, daß es Algorithmen gibt, die dieses Problem in polynomialer Zeit lösen. Man ist deshalb darauf angewiesen, für die praktische Lösung dieser Probleme Heuristiken einzusetzen, die in kurzer Zeit suboptimale Lösungen liefern. Wir verwenden eine zweistufige Heuristik: Zuerst weisen wir für die großen Behälter Lagerplätze zu, fixieren diese Zuweisungen, führen einen Update des Lagerplatzpools durch und weisen anschließend die kleinen Behälter ihren Lagerplätzen zu. Jede Zuweisung entspricht der Lösung eines Assignment Problems.

Diese Zuweisungen müssen jeweils durchgeführt werden, wenn eine neue TE eingebucht oder ein Lagerplatz frei wird. Zur Lösung der Assignment Probleme verwenden wir den Code von Achatz, Kleinschmidt und Paparrizos ([AKP91]), der für die entstehenden Größenordnungen ($|C| \sim 20, |L| \sim 800$) praktisch ohne Zeitverzug eine optimale Lösung liefert.

Um die Güte der heuristischen Lösung abzuschätzen, implementierten wir für das GAP ein Schnittebenenverfahren. Dieses Verfahren löste fast alle Problembeispiele optimal oder lieferte sehr gute untere Schranken für deren Optimalwerte. Aus Laufzeitgründen ist dieses Verfahren jedoch für einen Einsatz bei SNI ungeeignet. Es zeigte sich, daß die Heuristiken durchweg zufriedenstellende Lösungen produzieren. Sehr oft wurde eine Optimallösung gefunden, oder der Optimalwert bis auf 1% erreicht. Weitere theoretische und algorithmische Untersuchungen zum GAP können in Abdel–Hamid ([Abd94]) gefunden werden.

Rechenergebnisse.

Analog zur Vorgehensweise bei der Leerfahrzeitminimierung des RBG verglichen wir die alte Strategie mit dem Optimierungsverfahren. Die erzielten Verbesserungen sind jedoch nicht so signifikant, wie es beim letzten Problem der Fall war. Sie lagen lediglich im 5%–Bereich. Da der Aufwand zur Umstellung des Softwaresystems zu groß erschien, wurde vorerst von einer Umsetzung bei SNI abgesehen.

Wie bei der Leerfahrzeitminimierung gesehen, haben die aufgenommenen Systemdaten, die als Basis für einen Vergleich dienen, eine große Auswirkung auf die erzielten Ergebnisse. Leider erwies sich hier der Prozeß der Datenaufnahme als sehr schwierig, da synchron mehrere Softwaresysteme “angezapft” werden mußten. Aus diesem Grunde konnten nur für wenige Tage Daten aufgenommen werden. Wir sind nach wie vor der Meinung, daß in dem beschriebenen Ansatz ein größeres Potential steckt, als es die Rechenergebnisse bisher zeigen. So kann es z.B. möglich sein, daß das Optimierungsverfahren eine gewisse Einschwingphase benötigt, bis alle Einlagerungen, die unter der alten Strategie erfolgt sind, durch optimierte

Einlagerungen ersetzt worden sind. Diesbezügliche Experimente innerhalb des Simulationsprogramms dauern momentan noch an.

Bemerkung.

In Zeiten, in denen für das RBG keine Transportaufträge vorliegen, können **Umlagerungen** vorgenommen werden, d.h., weit von ihrem Verbrauchsort im Entnahmebereich entfernt liegende TE können auf näher gelegene Lagerplätze transportiert werden. Dies kann in analoger Weise als ein AP bzw. GAP formuliert werden. In den Containerpool C werden alle TE aufgenommen, die in den Hochregallagern eingelagert sind. Der Lagerplatzpool L setzt sich aus allen freien und belegten Lagerplätzen des Hochregallagers zusammen. Durch Umlagerung kann im Prinzip ein für den RBG–Betrieb in der laufenden Produktion günstiger Ausgangszustand erreicht werden. Ob sich dieser Aufwand in der Praxis lohnt, ist jedoch noch nicht geprüft worden.

2.4 Ein– und Auslagerung im Dauertestbereich

Die Fragestellungen, die bei der Zuweisung von Geräten zu Testplätzen im Dauertestregal entstehen, sind ähnlicher Natur wie die bei der Einlagerung in die Hochregallager. Bei SNI gibt es zwei verschiedene Typen von Testregallagern. Im ersten (DTA) sind für die verschieden hohen Gerätetypen (PC, Monitor, Tower–PC, etc.) verschieden hohe Testplätze vormontiert. Es ist zulässig, daß Geräte in höhere Testplätze eingelagert werden. Im anderen Testbereich (DTN) sind die Testplätze nicht fest vormontiert, sondern es ist eine variable Einlagerung möglich.

Modellierung.

In Krippner und Matejka ([KM93]) sind Modelle und heuristische Verfahren beschrieben, die zum Ziel haben, die Verweildauer der Geräte im Dauertestbereich zu minimieren. Es ist prinzipiell möglich, die anstehenden Entscheidungen über Ein– und Auslagerungen über einen Entscheidungsbaum zu realisieren. Der Wurzel dieses Baumes entspricht der momentane Systemzustand, die Söhne stellen mögliche Entscheidungen dar, z.B. ein ankommendes Gerät auf einen bestimmten Platz einzulagern. Diese Knoten des Baumes werden dann wiederum als mögliche Ausgangssystemzustände angesehen und erhalten Söhne, die z.B. den Entscheidungen über mögliche Auslagerungen entsprechen, usw. Die Kanten zwischen Vater und Sohn erhalten einen Kostenkoeffizienten, der dem "Aufwand" entspricht, der mit dieser Entscheidung verbunden ist. Dies ist eine Kombination aus RBG–Fahrzeit und Wartezeit auf den Testbeginn bzw. auf die Auslagerung. Setzen wir diese Konstruktion fort, erhalten wir einen exponentiell großen Entscheidungsbaum. Die günstigste Sequenz von Entscheidungen entspricht einem kürzesten Weg von der Wurzel des Baumes zu einem seiner Endknoten. Dieses Verfahren ist natürlich aus Speicherplatz– und Rechenzeitgründen nicht praktikabel.

Die Kostenkoeffizienten ergeben sich aus einer Kombination aus Wartezeit auf den Testbeginn bzw. Auslagerung, und RBG–Fahrzeit. Der wesentliche Unterschied zur TE–Einlagerung besteht in dieser Kostenfunktion. War es bei der TE–Einlagerung so, daß die Kosten, eine gewisse TE auf einem festen Lagerplatz einzulagern, über die Zeit konstant waren, ist es hier, bedingt durch die Betaktungszyklen, so, daß die Kosten sich über die Zeit verändern. Ein Platz, der vor kurzem noch günstig war, wird nach Beginn des Betaktungszyklus "teuer", um dann über einen Zeitraum von zwei Stunden langsam wieder "billiger" zu werden. Ein kurzer

Verzug bei der Einlagerung, bedingt z.B. durch kurzzeitige technische Störungen oder eine Überbeanspruchung des Datennetzes, kann sich somit sehr nachteilig auf das Gesamtergebnis auswirken.

Aus diesen Gründen kann eine Realisierung eines Entscheidungsbaumes, die auf wenige Schichten beschränkt ist, eine sinnvolle Heuristik zur Lösung des Problems darstellen.

Eine weitere Möglichkeit, die Einlagerung von Geräten zu modellieren, ist die Formulierung der Aufgabe als Zuordnungsproblem (AP). Dies geschieht analog zu der Modellierung, die in Kapitel 2.3 beschrieben wurde. Geräte, die auf die Einlagerung warten (Gerätepool), werden Testplätzen (Lagerpool) zugewiesen. Für den DTN ergibt sich statt eines AP ein Generalized Assignment Problem (GAP).

Ein weiterer Modellierungsansatz zieht neben den Einlagerungen auch noch die möglichen Auslagerungen mit in Betracht. Hierzu wird ein geschichtetes Netzwerk $D = (V, B)$ mit der Knotenmenge

$$V := \{s,t\} \cup V_E \cup V_T \cup V_{T'} \cup V_A$$

konstruiert. Hier entsprechen s und t einer künstliche Quelle und Senke, die Knoten in V_E den einzulagernden Geräten, V_T den freien Lagerplätzen und V_A möglichen Auslagerungen von Geräten. $V_{T'}$ ist ein Duplikat von V_T.
Die Kantenmenge B ist wie folgt definiert

$$\begin{aligned} B := \quad & \{(s,i) | i \in V_E\} \\ \cup \quad & \{(i,j) | i \in V_E, j \in V_T, i \text{ kann auf } j \text{ eingelagert werden } \} \\ \cup \quad & \{(i,i') | i \in V_T, i' \in V_{T'}, i' \text{ ist das Duplikat von } i\} \\ \cup \quad & \{(i,j) | i \in V_{T'}, j \in V_A\} \\ \cup \quad & \{(i,t) | i \in V_A\} \end{aligned}$$

Die Kanten zwischen V_E und V_T erhalten einen Kostenkoeffizienten, der der Wartezeit auf den Testbeginn entspricht, die entsteht, falls das Gerät auf diesem Platz getestet wird. Die Kanten zwischen $V_{T'}$ und V_A erhalten Kosten, die einer Kombination aus der RBG–Fahrzeit von diesem Lagerplatz zu dem auszulagernden Gerät und der bisherigen Wartezeit auf die Auslagerung entsprechen. Alle anderen Kanten erhalten Gewicht 0. Ist auf allen Kanten ein maximaler Flußwert von 1 gegeben, ist ein kostenminimaler Fluß mit Wert $|E|$ in diesem Netzwerk gesucht.

Rechenergebnisse.

Zur Lösung der Assignment Probleme verwendeten wir die Implementierung des Algorithmus von Achatz, Kleinschmidt und Paparrizos ([AKP91]), zur Lösung der Min–Cost–Flow–Probleme eine Implementierung des Netzwerk–Simplexalgorithmus von Löbel ([Löb92]).

Die beschriebenen Optimierungsverfahren werden jeweils aufgerufen, wenn ein neues Gerät den Dauertestbereich erreicht und eingelagert werden muß, oder wenn ein Gerät ausgelagert wurde, und somit ein neuer Platz für mögliche Einlagerungen frei wurde. Zum Test der beschriebenen Verfahren stand ein Datenabzug für einen Produktionstag zur Verfügung. Es zeigte sich, daß durch Einsatz der Optimierungsverfahren und durch einige organisatorische Änderungen, wie z.B. eine andere Reihenfolge der Bestromung der Testzyklen, die Durchlaufzeiten von ca. 12,5 Stunden auf ca. 8,5 Stunden reduziert werden konnten. Wir möchten an dieser Stelle betonen, daß eine wesentliche Reduzierung alleine durch eine geänderte Betaktungsreihenfolge erzielt werden konnte. Über eine Umsetzung der vorgeschlagenen Änderungen und den Einsatz der Optimierungsverfahren wird derzeit nachgedacht. Weitere algorithmische Details und Rechenergebnisse sind in Krippner und Matejka ([KM93]) zu finden.

3 Zusammenfassung

Die mathematische Optimierung kann eine große Unterstützung bei der Ausnutzung der teuren Produktionsressourcen in flexiblen Fertigungssystemen sein. Sowohl in der Layout– als auch in der Betriebsphase bieten diese Systeme eine Vielzahl von Einsatzmöglichkeiten für Verfahren der mathematischen Optimierung. Wir berichteten über ein Projekt, das zum Ziel hatte, den Produktionsfluß in einem bereits bestehenden System zu verbessern.

Es wurde ein Optimierungsmodul zur Minimierung der Leerfahrzeiten eines Regalbediengeräts beschrieben, welches jetzt seit mehr als zwei Jahren im Produktivbetrieb im Einsatz ist. Durch dieses Modul konnten im Hochlastbetrieb die Leerfahrzeiten um ca. 30% reduziert werden. Ferner skizzierten wir eine Einlagerstrategie für Hochregallager und Ein– bzw. Auslagerstrategien für ein Testregal.

Am Beispiel der Bestromungsreihenfolge im Dauertest wurde gezeigt, daß manchmal auch schon einfache organisatorische Veränderungen zu einer Systemverbesserung führen. Zu deren Lokalisierung sind oft nur die "Augen eines neutralen Beobachters" notwendig.

Literatur

[AAG95] Abdel–Aziz Abdel-Hamid, A., Ascheuer, N., Grötschel, M.: Order picking in an automatic warehouse: Solving on–line asymmetric TSPs. Technical Report, Konrad–Zuse–Zentrum für Informationstechnik, Berlin (1995 – in preparation).

[Abd94] Abdel–Aziz Abdel–Hamid, A.: Combinatorial optimization problems arising in the design and management of an automatic storage system. Doktorarbeit, Technische Universität Berlin (1994).

[AKP91] Achatz, H., Kleinschmidt, P., Paparrizos, K.: A dual forest algorithm for the assignment problem. DIMACS Series in Discrete Mathematics and Theoretical Computer Science **4** (1991) 1-10

[Alb93] Albers, S.: The influence of lookahead in competitive on–line algorithms. Doktorarbeit, Universität des Saarlandes, Saarbrücken (1993).

[Asc95] Ascheuer, N. Hamiltonian path problems in the on–line optimization of flexible manufacturing systems. Doktorarbeit, Technische Universität Berlin (1995).

[BM76] Bondy, J.A., Murty, U.S.R.: Graph Theory with Applications. American Elsevier, New York and Macmillan, London (1976).

[Chv83] Chvátal, V.: Linear Programming. W.H. Freeman and Company, New York (1980).

[FT92] Fischetti, M., Toth, P.: An additive bounding procedure for the asymmetric TSP. Mathematical Programming **53** (1992) 173–197.

[Grö92] Grötschel, M.: Discrete Mathematics in Manufacturing. ICIAM'91: Proceedings of the Second International Conference on Industrial and Applied Mathematics (1992).

[Jun87] Jungnickel, D.: Graphen, Netzwerke und Algorithmen. B.I.–Wissenschaftsverlag (1987).

[KM93] Krippner, T., Matejka, H.: Online–Optimierung eines Testregallagers. Modellierung und Vergleich verschiedener Heuristiken. Diplomarbeit, Universität Augsburg (1993).

[MS] McGeoch, L.A., Sleator, D.D. (Hrsg.) : On–line algorithms. Proceedings of a DIMACS workshop. AMS (1992).

[Löb92] Löbel, A.: Implementierung und Analyse des Netzwerk–Simplexalgorithmus für das Minimalkosten–Flußproblem. Diplomarbeit, Universität Augsburg (1992).

[NW88] Nemhauser, G.L., Wolsey, L.A.: Integer and Combinatorial Optimization. John Wiley & Sons, New York (1988).

[1988] Page, B., Bölckow, R., Heymann, A., Kadler, R., Liebert, H.: Simulation und moderne Programmiersprachen. Fachberichte Simulation 8, Springer Verlag, Berlin (1988).

[Sch86] Schrijver, A.: Theory of Linear and Integer Programming. John Wiley & Sons, Chichester (1986).