Ist besser wirklich besser?

Über das Einfügeglück und die Pfadabhängigkeit im Kontext des DVRPTW

Jürgen Müller[1], Richard Lohwasser[2], Andreas Lackner[3], Alexander Zeier[1], Hasso Plattner[1]

[1]Hasso Plattner Institut für IT Systems Engineering, Prof.-Dr.-Helmert-Str. 2-3, 14482 Potsdam

[2]Externer Doktorand, RWTH Aachen, Deisenhofener Str. 83, 81539 München

[3]DB Mobility Logistics AG, Gallusanlage 8, 60329 Frankfurt am Main

1 Einleitung

Einen Problembereich im Verkehrssektor stellen Tourenplanungsprobleme dar. Diese entstehen, wenn Personen oder Güter von einem Transportmittel zu verschiedenen Orten gebracht und/oder abgeholt werden müssen. Im Allgemeinen geht es darum Aufträge zu erfüllen und die Anzahl der nötigen Touren sowie die Gesamtstrecke zu minimieren (Christofides et al. (1979)). Ziel dieser Arbeit ist es

- ausgewählte Konstruktions-, Einfüge- und Verbesserungsverfahren für die dynamische Tourenplanungsoptimierung zu evaluieren sowie
- auftretende Effekte bei einer Parallelisierung bei der Erstellung von Zwischenlösungen eines Tourenplanuns zu analysieren.

Die Arbeit gliedert sich wie folgt: In Abschnitt 2 gehen wir auf das Problem der dynamischen Tourenplanung mit Zeitfensterrestriktionen ein. Abschnitt 3 beschreibt von uns verwendete Verfahren. Anschließend werden die Implementierung und der Versuchsaufbau beschrieben, bevor in Abschnitt 5 die Ergebnispräsentation und -analyse durchgeführt wird. Der Beitrag schließt mit einem Fazit und Ausblick.

2 Dynamische Tourenplanung mit Zeitfensterrestriktionen

Die Grundlage des *Dynamic Vehicle Routing Problem with Time Windows* (DVRPTW, Psaraftis (1995) sowie Cordeau und Laporte (2003)) sind das *Handelsreisendenproblem* (Travelling Salesman Problem, TSP, Hoffmann (1985)), das *Capacitated Vehicle Routing Problem* (CVRP, Bodin et al. (1983)) sowie das *Vehicle Routing Problem with Time Windows* (VRPTW, Solomon (1987)).

Bei dem DVRPTW sind bei der Erstellung der Ausgangslösung von insgesamt *N* Kunden nur *n* Kunden bekannt. Die restlichen *N-n* Kunden offenbaren sich später und können somit erst zum Zeitpunkt ihrer Nachfrage in bestehende laufende Touren integriert werden. Die Anzahl der im Verlauf des Planungshorizontes hinzukommenden Kunden entspricht dem Dynamikgrad der Problemstellung. Die Schwierigkeit ist es, den bestehenden Tourenplan so abzuändern, dass diese Kunden integriert werden. In der vorliegenden Arbeit werden Einsammeltouren betrachtet. Bei dem DVRPTW gibt es eine festgelegte Maximalanzahl an Touren. Weiterhin darf ein Kunde auch nach dem Ende seines Bedienzeitfensters bedient werden kann. Entsteht jedoch solch eine Verspätung, wird diese in der sekundären Zielfunktion berücksichtigt.

Es ergibt sich für das DVRPTW folgendes Optimierungsproblem mit lexikographischer Zielfunktionsordnung:

- ZF I: Anzahl abgewiesener Kunden → min,
- ZF II: Summe der Länge aller Touren + Summe aller Verspätungen → min

unter den Nebenbedingungen

- NB I: Start- und Endpunkt einer jeden Tour ist das Depot,
- NB II: Jede Tour darf frühestens beim Zeitpunkt 0 anfangen und muss bis zum Depotzeitfensterende beendet sein,
- NB III: Die Anzahl der verwendeten Touren muss kleiner oder gleich der verfügbaren Touren der Problemstellung sein,
- NB IV: Kein Kunde darf vor seinem Zeitfensterbeginn bedient werden und
- NB V: Die Ladekapazität je Tour darf nicht überschritten werden.

Zur Evaluierung der Algorithmen werden die 56 Solomon-Benchmark-Probleme (Solomon (1987)) mit den Einfügedateien für die dynamiasche Problemstellung von Lackner (2004) verwendet.

3 Algorithmische Grundlagen

Im DVRPTW wird mit den bereits bekannten Kunden mithilfe eines Konstruktionsverfahrens eine Ausgangslösung geschaffen. Sobald ein neuer Kunde nachfragt

wird dieser gemäß einer Einfügestrategie in den Tourenplan integriert. Anschließend wird der Tourenplan optimiert. Dies wiederholt sich solange bis keine weiteren Kunden mehr vorhanden sind. Die in dieser Arbeit verwendeten Konstruktions-, Einfüge- und Optimierungsverfahren sollen kurz beschrieben werden.

Als Konstruktionsverfahren wird zum Einen das von Clarke and Wright (1964) entwickelte *Savings*-Verfahren verwendet, welches von Solomon (1987) für das DVRPTW-Problem erweitert wurde. Zum Anderen wird das Konstruktionsverfahren *Solomon I1* (Solomon (1987)) angewendet.

Es werden drei Einfügestrategien verwendet. Zunächst gibt es das klassische *Optimieren ohne Strategie*, bei der nach dem Einfügen jedes einzelnen Kunden der Tourenplan optimiert wird. Die Strategie *Optimieren nach Sammeln* integriert zunächst eine bestimmte Anzahl von Kunden (#Kunden) in den Tourenplan ohne eine Optimierung anzustoßen. Diese erfolgt erst, sobald #Kunden einen Grenzwert K* erreicht Es gibt also K/K* Optimierungen. Die Strategie, *Optimierung nach Zeit* verfolgt das Prinzip, dass nach jeweils x% des Planungszeitraums optimiert wird. Ist solch ein Zeitpunkt t* erreicht, so wird der Tourenplan optimiert. Es kommt insgesamt zu 100/x Optimierungen. Im Vergleich zur Optimierung ohne Strategie reduzieren die anderen Einfügestrategien die Anzahl der Optimierungen. Dadurch kann die Anzahl der aufwendigen Tourenplanoptimierungen reduziert werden. Bspw. sind bei der Optimierung ohne Strategie und einem Dynamikgrad von 50% insgesamt 50 Tourenplanoptimierungen durchzuführen. Bei der Strategie nach Sammeln von zwei Kunden sin des nur noch 25.

Von uns verwendete Verbesserungsverfahren sind die Knotentauschverfahren Or-Opt (Or (1976)) und One-Interchange (Osman (1993)) sowie das Kantentauschverfahren Two-Opt (Potvin and Rousseau (1995)).

4 Programmablauf und Versuchsaufbau

Im Rahmen dieser Arbeit wurde zur Durchführung der Tourenplanungsoptimierung ein Computerprogramm einer ereignisgesteuerten Simulation in Java implementiert mit dem die Versuche durchgeführt werden.

4.1 Allgemeiner Programmablauf

Nach dem Programmstart wird der Einfügedatei des entsprechenden Solomon-Problemdatensatzes geladen und die Ausgangslösung erzeugt. Für die dynamische Problemstellung wird anschließend eine ereignisgesteuerte Simulation gestartet.

Wenn ein Kunde nachfragt, so werden alle Einfügestellen auf Gültigkeit in Bezug auf die Nebenbedingungen überprüft. Unter allen gültigen Einfügestellen wird die potentielle Änderung des sekundären Zielfunktionswertes verglichen. Die Position mit der geringsten Verschlechterung stellt die zu realisierende Einfügeposition dar. Sollte keine gültige Einfügeposition existieren, wird eine neue Pendeltour zu

diesem Kunden eröffnet. Sollte das Maximum an erlaubten Touren bereits erreicht worden sein, wird dieser Kunde abgewiesen (Wang et al. 2007).

Anschließend werden die Verbesserungsverfahren auf den Tourenplan angewendet. Wird nur isoliert ein Verbesserungsverfahren angewendet, geschieht dies solange bis der Zielfunktionswert sich nicht mehr verbessert. Bei der Optimierung mit allen Verbesserungsverfahren wird zunächst One-Interchange auf den Tourenplan angewendet. Ist dadurch eine Verbesserung eines Zielfunktionswertes entstanden, so wird die Anwendung wiederholt. Ansonsten werden in zufälliger Reihenfolge die Verbesserungsverfahren Or-Opt, Two-Opt oder One-Interchange angewendet bis es zu keiner Zielfunktionswertverbesserung mehr kommt.

Wird ein Kunde erreicht, so wird dieser markiert und es wird zum nächsten Kunden gefahren, sofern dieser dadurch nicht vor Öffnung seines Bedienfensters erreicht würde. Ansonsten wird bei dem gerade bedienten Kunden gewartet.

4.2 Parallelisierung

Eine parallele Abarbeitung desselben Problems zielt i.d.R. auf eine Beschleunigung eines Algorithmus ab. Wir verfolgen mit der parallelen Bearbeitung dahingegen eine Verbesserung des Ergebnisses (Bergman et al. (2003)). Da im DVRPTW die Verbesserungsverfahren eine herausragende Rolle spielen sollen diese Verfahren in einer Parallelisierung im Rahmen dieser Arbeit bei jeder Optimierung des Tourenplans variiert werden. Wenn also eine Optimierung durchgeführt werden soll, so werden insgesamt 12 verschiedene Optimierungen durchgeführt und es wird die beste Lösung übernommen. Somit sollen bessere Gesamtergebnisse erreicht werden als bei der sequenziellen Durchführung.

Die Umsetzung des Parallelisierungsansatzes geschieht mit 12 Variationen der Verbesserungsverfahren, bei denen jeweils drei unterschiedliche Kombinationen von Verbesserungsverfahren vorbestimmt sind. Wenn eines der Verbesserungsverfahren keine weiteren Zielfunktionswertverbesserungen erreicht, wird das nächste angewendet. Nach dem dritten Verbesserungsverfahren werden die weiteren Verfahren wie im sequentiell laufenden Programm zufällig angewendet.

4.3 Versuchsaufbau

Im Rahmen dieser Arbeit wurden die 56 Benchmark-Probleme von Solomon mit unterschiedlichen Parametern simuliert. Eine Übersicht über die Simulationen, die in der statischen Problemstellung durchgeführt wurden, gibt Tabelle 1. Sofern nicht gegensätzlich angegeben, wurden die Standardparameter für Solomon I1 verwendet ($\mu = 1$, $\lambda = 1$, $\alpha = 0{,}5$). Die letzte Spalte gibt die Anzahl der unterschiedlichen Parameterkonstellationen pro Solomon-Probleminstanz an.

Tabelle 1: Übersicht über die Simulationen der statischen Problemstellung

Beschreibung	Konstante Parameter	Variable Parameter	#/P
Ohne Verbesserungsverfahren	Dynamikgrad = 0	Konstruktionsverfahren	2
Mit Verbesserungsverfahren	Dynamikgrad = 0	Konstruktionsverfahren, Verbesserungsverfahren = {alle Verfahren sequenziell initiiert mit One-Interchange, Or-Opt exklusiv. One-Interchange exklusiv, Two-Opt exklusiv}	8
Nur Solomon I1 ohne Verbesserungsverfahren	Dynamikgrad = 0; Solomon I1 als Konstruktions-verfahren	Parameter von Solomon I1 (partielle Faktorvariation) mit Kombinationen von μ: 0, 0.1, 0.3, 0.5, 0.7, 0.9; λ: 0, 0.1, 0.3, 0.5, 0.7, 0.9, 1.3, 1.5, 1.7, 2.0, 2.3, 2.5, 2.7, 3.0; α : 0, 0.1, 0.3, 0.7, 0.9, 1.	26

Anschließend wurde die dynamische Problemstellung untersucht. Eine Übersicht über durchgeführten Simulationen gibt Tabelle 2. Es wurden die Standardparameter für Solomon I1 und die Dynamikgrade 10, 30, 50, 70 und 90% verwendet.

Tabelle 2: Übersicht über die Simulationen der dynamischen Problemstellung

Beschreibung	Konstante Parameter	Variable Parameter	#/P
Ohne Strategie	Strategie = Optimierung ohne Strategie	Dynamikgrade, Konstruk-tionsverfahren, Verbesse-rungsverfahren	40
Optimierung nach Sammeln	Konstruktionsverfahren = Solomon I1; Strategie = Optimierung nach Sammeln, Verbesserungsverfah-ren = sequenzielle Anwendung aller Verfahren	Dynamikgrade, Parameter der Einfügestrate-gie = {2, 3, 4, 5, 6}	25
Optimierung nach Sammeln	Konstruktionsverfahren = Savings; Strategie = Optimierung nach Sammeln; Parameter der Einfügestrategie = 3; Verbesserungsverfahren = sequenzielle Anwendung aller Verfahren	Dynamikgrade	5
Optimierung nach Zeit	Konstruktionsverfahren = Solomon I1; Strategie = Optimierung nach Zeit,Verbesserungsverfahren = sequenzielle Anwendung aller Verfahren	Dynamikgrade, Einfügestrategieparameter = {1, 2, 3, 4, 5, 7.5, 10, 15}	45

Letztendlich wurde die Parallelisierung in die Untersuchung aufgenommen (Tabelle 3). Die Parameterkonstellationen wurden hierbei so ausgewählt, dass zu allen Problembereichen (Konstruktionsverfahren, Einfügestrategien und Verbesserungsverfahren) Ergebnisse gewonnen werden konnen.

Tabelle 3: Übersicht über die Simulationen mit dem parallelisierten Programm

Beschreibung	Konstante Parameter	Variable Parameter	#/P
Ohne Strategie	Strategie = keine Strategie	Konstruktionsverfahren, Dynamikgrad	10
Optimierung nach Sammeln	Konstruktionsverfahren = Solomon I1 Strategie = Sammeln nach Kunden	Dynamikgrad, Strategieparameter = {3, 5}	10
Optimierung nach Zeit	Konstruktionsverfahren = Solomon I1, Strategie = Sammeln nach Zeit, Strategieparameter = 3%	Dynamikgrad	5

5 Präsentation und Analyse der Ergebnisse

Insgesamt wurde das implementierte Programm mit 36 Parametervariationen der statischen Problemstellung, 115 Parametervariationen der dynamischen Problemstellung und 25 Parametervariationen unter Zuhilfenahme der Parallelisierung für jedes der 56 Solomon-Benchmark-Probleme ausgeführt. Somit konnten zu 9.856 unterschiedlichen Simulationsläufen des DVRPTW Ergebnisse gesammelt werden. Die Ergebnisse wurden jeweils auf aggregierter als auch auf heruntergebrochen auf einzelne Solomon-Problemklassen und bei entsprechender Fragestellung auch bis auf Solomon-Probleminstanz-Ebene analysiert. Aus Platzgründen konnen hier nur die wesentlichen Erkenntnisse dargelegt werden. Der interessierte Leser kann bei den Autoren gerne Einsicht in weitere Ergebnisse erlangen.

5.1 Statische Problemstellung

In der statischen Problemstellung wurden Konstruktions- und Verbesserungsverfahren analysiert. Dabei stellt sich heraus, dass das Konstruktionsverfahren Solomon I1 im Durchschnitt 9,05 Touren benötigt wohingegen das Savings-Verfahren 10,95 Touren benötigt. Die Anwendung von Verbesserungsverfahren kann diese Diskrepanz nicht egalisieren und verbessert die durchschnittliche Anzahl an Touren bei Anwendung aller Verbesserungsverfahren auf 8,3 (Solomon I1) bzw. 10,1 (Savings).
Für die statische Problemstellung kann gesagt werden:

- Das Konstruktionsverfahren Solomon I1 ist bei der lexikographischen ZF-Ordnung Touren vor Entfernung wesentlich besser als Savings.
- Dies wird durch Verbesserungsverfahren nicht egalisiert, da die Differenz lediglich von 1,9 auf 1,8 Touren gesenkt werden kann.
- Der deutliche Nachteil des Solomon I1 Verfahrens bei der Entfernung wird durch Verbesserungsverfahren bis auf 5% eliminiert.

Die Situation der Durchschnittsbetrachtung über alle Problemklassen bestätigt sich auch innerhalb der Problemklassen, allerdings in unterschiedlicher Stärke.

5.2 Dynamische Problemstellung

Im Folgenden sollen die Ergebnisse der dynamischen Problemstellung präsentiert und analysiert werden. Es wird zunächst auf die Wirkungen der unterschiedlichen Konstruktionsverfahren unter sequenzieller Anwendung aller Verbesserungsverfahren im dynamischen Fall eingegangen. Darauf aufbauend wird anschließend stets Solomon I1 als Konstruktionsverfahren angewendet, um die Verbesserungsverfahren in Abhängigkeit des Dynamikgrades zu analysieren.

Analyse der Konstruktionsverfahren

In der Durchschnittsbetrachtung über alle Problemklassen ist Solomon I1 dem Savings-Verfahren vorzuziehen. Innerhalb der jeweiligen Problemklasse ergeben sich hingegen große Unterschiede. Beide Verfahren sind absolut gesehen etwa gleich häufig vorzuziehen (Abbildung 1). Damit unterscheiden sich die Ergebnisse stark zur statischen Betrachtung, in der Solomon I1 unabhängig von der Problemklasse stets vorzuziehen war.

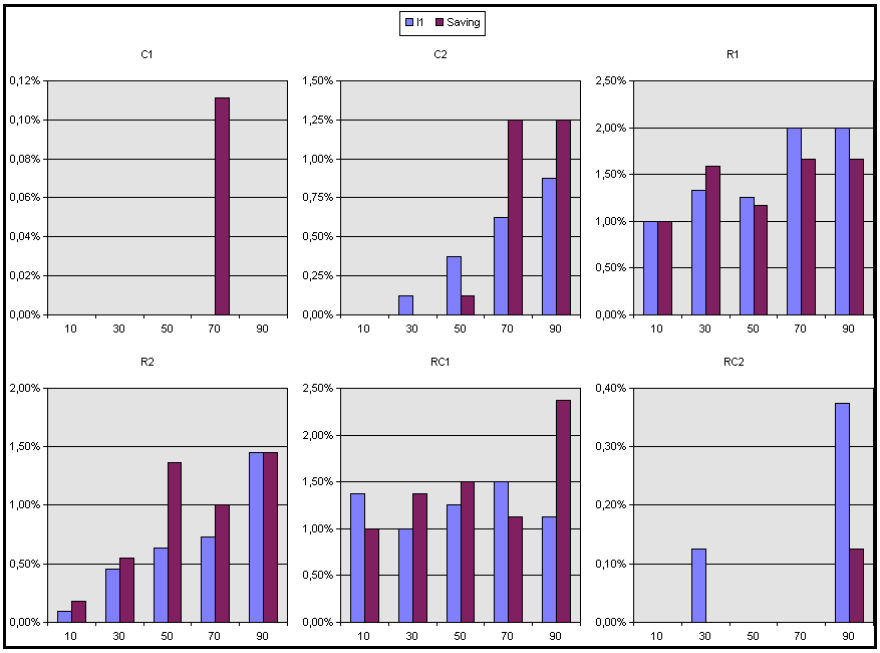

Abbildung 1: Vergleich der Konstruktionsverfahren nach Dynamikgrad, Solomon-Problemklasse und durchschnittlicher Anzahl abgewiesener Kunden

Analyse der Verbesserungsverfahren

Im Folgenden wird die Leistung der verschiedenen Verbesserungsverfahren unter konstant gehaltenem Konstruktionsverfahren analysiert. Aufgrund der Ergebnisse obiger Analyse wird dafür das im Mittel über alle Problemklassen und Dynamikgrade bessere Verfahren Solomon I1 gewählt.

Wie in nachstehender Tabelle 4 aufgeführt stellt sich ein Vorteil der sequenziellen Anwendung aller Verbesserungsverfahren ein, welcher im Allgemeinen mit dem Dynamikgrad steigt. So klettert der Unterschied zu einzelnen Verfahren auf 0,4 Prozentpunkte bei Dynamikgrad 90 (1,43% auf 1,05%, relativ also 36%). Entsprechend steigt der Vorsprung der sequenziellen Anwendung aller Verfahren.

Tabelle 4: Vergleich der Verbesserungsverfahren nach Dynamikgrad und durchschnittlicher Anzahl abgewiesener Kunden

Dynamikgrad	Verbesserungs-verfahren	Abg. Kunden	Unterschied zu allen Verbesserungsverfahren in Prozentpunkten
10	Alle	0,43%	0%
	One-Interchange	0,39%	-0,04%
	Or-Opt	0,50%	0,07%
	Two-Opt	1,09%	0,66%
30	Alle	0,55%	0%
	One-Interchange	0,66%	0,11%
	Or-Opt	0,91%	0,36%
	Two-Opt	3,73%	3,18%
50	Alle	0,63%	0%
	One-Interchange	0,89%	0,26%
	Or-Opt	1,38%	0,75%
	Two-Opt	4,77%	4,14%
70	Alle	0,88%	0%
	One-Interchange	0,93%	0,05%
	Or-Opt	1,46%	0,58%
	Two-Opt	4,46%	3,58%
90	Alle	1,05%	0%
	One-Interchange	1,43%	0,38%
	Or-Opt	1,73%	0,68%
	Two-Opt	5,23%	4,18%
Durchschnitt	Alle	0,71%	0%
	One-Interchange	0,86%	0,15%
	Or-Opt	1,20%	0,49%
	Two-Opt	3,86%	3,15%

Bei der Untersuchung des Verbesserungsverfahren One-Interchange fällt auf, dass dieses bei einem Dynamikgrad von 10 besser ist als die sequenzielle Anwendung aller Verfahren. Hierin liegt der Beweis des *Einfügeglücks*, also der Aussage, dass eine bessere Zwischenlösung nicht zwangsläufig zu einem besseren Endergebnis führt. Dies tritt nicht nur im gerade gezeigten Fall sondern auch bei anderen Verbesserungsverfahren und Dynamikgraden auf, dann allerdings nicht in einer gesamten Durchschnittsbetrachtung, sondern nur innerhalb einzelner Problemklassen.

Zusammenfassend kann für die Analyse der Verbesserungsverfahren gesagt werden, dass

- innerhalb der einzelnen Verbesserungsverfahren One-Interchange im Allgemeinen Or-Opt vorzuziehen ist. Nur in Einzelfällen wird mit Or-Opt ein besseres Ergebnis erzielt. Das Verbesserungsverfahren Two-Opt ist hingegen stets um ein vielfaches schlechter.
- Die sequenzielle Anwendung aller Verfahren bringt durch Ausnutzung der individuellen Stärken in dem jeweiligen Problem gegenüber One-Interchange bzw. Or-Opt bis zu 0,4 bzw. 0,8 Prozentpunkte weniger abgewiesene Kunden (relativ: 40% bzw. 120%). Es ist jedoch nicht in jedem Problem immer besser als die einzelne Anwendung von Verfahren.

Analyse der Einfügestrategien

Die Unterschiede bzgl. der Verwendung von Strategen sind jedoch bei sinnvoller Parameterwahl (Optimieren nach ≤ 3 Kunden bzw. ≤ 4% der Zeit) nicht gravierend im Vergleich zur Verwendung ohne Strategie. Bei diesen Parametern erzielen das Sammeln nach Kunden und Zeit ein nur um jeweils ca. 0,05 Prozentpunkte schlechteres Ergebnis. Da eine kurze Zurückhaltung von Kunden zu einem lediglich minimal schlechteren ZF-Wert führt, ergibt sich Potenzial zum Einsparen von Rechenleistung mit geringem Qualitätsverlust.

5.3 Parallelisierungseffekte

In diesem Abschnitt sollen die bisherigen Ergebnisse der Verbesserungsverfahren, also die zufällige Anwendung aller Verbesserungsverfahren, mit den Ergebnissen des parallelisierten Programms verglichen werden. Es wird erwartet, dass die Ergebnisse des parallelisierten Programms besser sind, als die des nicht parallelisierten, da in jedem Optimierungsschritt das Ergebnis des parallelisierten Programms gleich gut oder besser ist. Es könnte dazu kommen, dass eine bessere Zwischenlösung zu einer schlechteren Gesamtlösung führt. Jedoch wird angenommen, dass die Überlegenheit des parallelisierten Programms sich durchsetzen wird.

Die Ergebnisse des vorherigen Abschnittes bzgl. Konstruktionsverfahren und Einfügestrategien lassen sich auf die Simulationsläufe mit Parallelisierung übertragen. Wesentliche Erkenntnisse ergeben sich jedoch bei der Betrachtung der Verbesserungsverfahren.

Ergebnisse

Interessanterweise liefert das parallelisierte Programm über alle Dynamikgrade und Problemklassen hinweg ein um 0,06 Prozentpunkte schlechteres Ergebnis bei den abgewiesenen Kunden (0,70% zu 0,76%). Die Aufschlüsslung nach Dynamikgraden zeigt keine signifikanten Schwankungen, wohingegen die Betrachtung der Problemklassen zeigt, dass das parallelisierte Programm insbesondere in der Problemklasse RC1 schlecht abschneidet. Eine genauere Betrachtung der Problemklasse RC1 zeigt, dass dies insbesondere bei hohen Dynamikgraden der Fall ist.

Die Betrachtung des zweiten Zielfunktionswertes zeigt ein analoges Bild. Diese Unterlegenheit des parallelisierten Programms bei der Bewertung resultiert aus höheren Verspätungen, da die Fahrzeuge Kunden oftmals erst nach dem Zeitfensterende erreichen. Abbildung 2 zeigt dies für die Probleminstanz C101, bei der diese Zusammenhänge besonders deutlich werden.

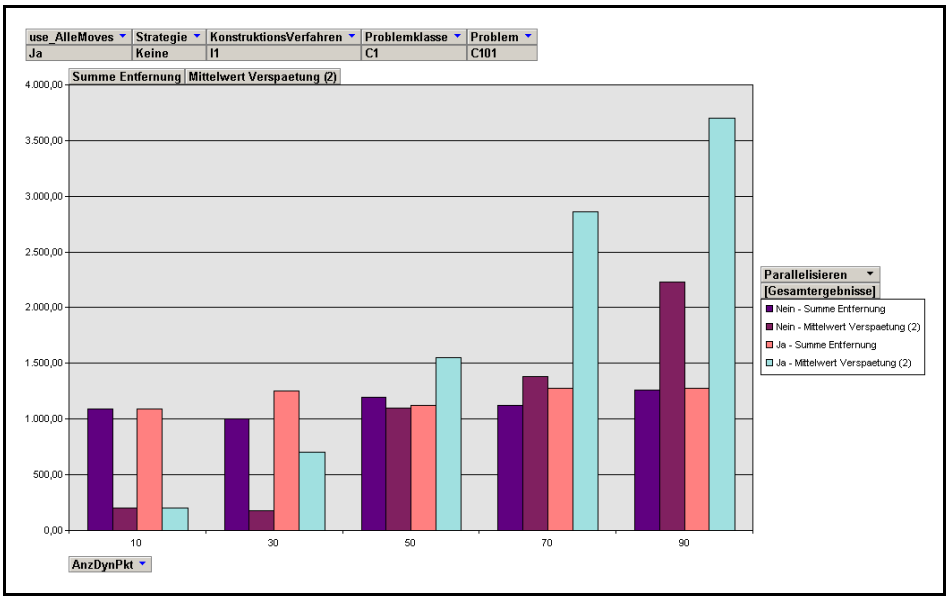

Abbildung 2: Vergleich von Entfernung und Verspätung des sequentiell ab laufenden und parallelisierten Programms für Problem C101 bei Variation der Dynamikgrade

Analyse

Die vorliegenden Ergebnisse entsprechen nicht den Erwartungen, dass das parallelisierte Programm gleich gute oder bessere Ergebnisse erreicht. Das nicht parallelisierte Programm schneidet fast durchgängig besser ab als das parallelisierte Programm. Bei dem parallelisierten Programm ist durch die unterschiedliche Vorgabe der ersten drei Verbesserungsverfahren bis auf ein statistisches Minimum sichergestellt, dass eine bessere oder gleich gute Zwischenlösung im Vergleich zum sequenziell ablaufenden Programm erreicht wird.

Diese Ergebnisse können wie folgt erklärt werden: Eine gute Bewertung entsteht immer dann, wenn die Strecke zwischen allen zu bedienenden Kunden möglichst klein wird. Das bedeutet gleichzeitig, dass der besten Bewertung eine Tour zugrunde liegt, die so konzipiert ist, dass möglichst wenige Umwege vorhanden sind. Gerade diese Umwege in den Zwischenlösungen sind es aber letztendlich, die bei dem DVRPTW die zukünftige Integration neuer Kunden erleichtern und somit das erste Zielfunktionswertkriterium verbessern.

Das Vorgehen, möglichst gute Bewertungen der Zwischenergebnisse zu erreichen, zahlt sich in Hinblick auf die Gesamtlösung nicht aus. Es ist vielmehr so, dass durch die Erzielung eines möglichst guten Zwischenergebnisses der Tourenplan auf eine Art und Weise optimiert wird, die das Einfügen späterer Kunden erschwert.

Es zeigt sich also, dass für einen Tourenplan über seinen Lebenszyklus von der Depoteröffnung bis zum Depotzeitfensterende eine *Pfadabhängigkeit* besteht. Es werden Entscheidungen getroffen, die nicht revidierbar sind und die zukünftige Güte des Tourenplans maßgeblich beeinflussen. Bei den etablierten Verbesserungsverfahren wird im Wesentlichen diese Frage gestellt: *Wie kann der Tourenplan optimiert werden, so dass ein möglichst gutes Zwischenergebnis realisiert wird?* Richtigerweise müsste diese Frage aber lauten: *Wie kann der Tourenplan optimiert werden, so dass im Hinblick auf eine möglichst gute Gesamtlösung die besten Voraussetzungen geschaffen werden?*

Bei der Kreierung von Zwischenlösungen sollten demnach weitere Kriterien als nur die Anzahl der Touren und die zurück gelegte Strecke berücksichtigt werden. Dies ist zum Einen die Flexibilität, mit der neue Kunden in den Tourenplan integriert werden können. Weiterhin schlagen wir eine Zeitkomponente vor, die in die Tourenplanoptimierung eingebunden werden, die von der Bewertung her suboptimale, aber flexible Tourenpläne am Anfang des Planungszeitraums bevorzugt und erst zum Ende des Planungszeitraumes die Flexibilität des Tourenplanes zugunsten der Bewertung verringert.

Die beobachtete Beziehung, dass eine bessere Zwischenlösung ein schlechteres Gesamtergebnis ergibt, kann nicht für alle Fälle verallgemeinert werden. So erzielt das Verbesserungsverfahren Two-Opt die schlechtesten Zwischenlösungen und die schlechtesten Gesamtlösungen.

6 Zusammenfassung und Ausblick

Im Rahmen der vorliegenden Arbeit wurden mit Hilfe eines dafür erstellten Computerprogramms die Ergebnisse der 9.856 Simulationsläufe der Solomon-Benchmarks präsentiert und analysiert. Dabei hat sich herausgestellt, dass

(1) das Konstruktionsverfahren Solomon I1 dem Konstruktionsverfahren Savings i.d.R. überlegen ist,

(2) Einfügestrategien keine besseren Ergebnisse erzielen, jedoch bei geeignet gewählten Strategieparametern die Anzahl der Optimierungen verringern ohne nennenswert mehr Kunden abzuweisen,

(3) eine kombinierte Anwendung aller Verbesserungsverfahren der Anwendung einzelner Verbesserungsverfahren überlegen ist,

(4) das Erreichen besserer Zwischenlösungen in der vorliegenden Problemstellung im Durchschnitt kontraproduktiv ist und

(5) die bisherigen Verfahren zur Bestimmung der Güte einer Zwischenlösung nicht ausreichend sind.

Zukünftige Arbeiten sehen wir in der Erstellung neuer Verbesserungsverfahren für das DVRPTW, die einen ganzheitlichen Ansatz verfolgen. Es können auch rechenintensivere Verfahren angewendet werden, wobei eine Kompensation dadurch

erreicht werden kann, dass ein Einfügen nach Sammeln oder Zeit durchgeführt wird, was bei marginaler Zielfunktionswertverschlechterung die Anzahl der rechenintensiven Tourenlpanoptimierungen einspart.

Literatur

Bodin, L.; Golden, B.; Assad, A.; Ball, M. (1983) Routing and Scheduling of Vehicles and Crews, The State of the Art. In: Computers & Operations Research, Vol. 10 No. 2, S. 63-201.

Christofides, N.; Mingozzi, A.; Toth P. (1979) The Vehicle Routing Problem. In: Christofides, N.; Mingozzi, A.; Toth, P.; Sandi, C. (Hrsg.): Combinatorial Optimizations, S. 315-338, New York: John Wiley & Sons.

Clarke, G.; Wright, W. (1964) Scheduling of Vehicles from a Central Depot to a Number of Delivery Points, In: Operations Research, 12(1964), 4, S. 568-581.

Cordeau, J.-F.; Laporte, G. (2003) A Tabu Search Heuristic for the Static Multi-vehicle Dial-a-ride Problem. In: Transportation Research Part B, 37(2003), 6, S. 579–594.

Lackner, A. (2004) Dynamische Tourenplanung mit ausgewählten Metaheuristiken. Culliver: Göttingen.

Or, I. (1976) Traveling Salesman-type Combinatorial Problems and their Relation to the Logistic of Blood Banking. Ph.D. thesis, Northwestern University Evanston.

Osman, I. H. (1993) Metastrategy Simulated Annealing and Tabu Search Algorithms for the Vehicle Routing Problem. In: Annals of Operations Research, 41(1993), S. 421-452.

Potvin, J. Y.; Rousseau, J. M. (1995) An Exchange Heuristic for Routing Problems with Time Windows. In: Journal of the Operational Research Society, 46.

Psaraftis, H.N. (1995) Dynamic Vehicle Routing: Status and Prospects. In: Annals of Operations Research 61, S. 143–164.

Solomon, M. M. (1987) Algorithms for the Vehicle Routing and Scheduling Problems with Time Window Constraints. In: Operations Research, 35(2), S.254 - 265.

Wang, J.; Tong, X.; Li, Z. (2007) An Improved Evolutionary Algorithm for Dynamic Vehicle Routing Problem with Time Windows. In: Lecture Notes in Computer Science, vol. 4490, pp. 1147-1154, Springer.