Zuverlässige und effiziente Überwachung von Warenflüssen durch Analyse von RFID-Massendaten

zur Erlangung des akademischen Grades eines

Doktors der Ingenieurwissenschaften

von der Fakultät für Informatik
des Karlsruher Instituts für Technologie (KIT)

genehmigte

Dissertation

von

Leonardo Weiss Ferreira Chaves

aus Rio de Janeiro, Brasilien

Tag der mündlichen Prüfung:	7. Mai 2010
Erster Gutachter:	Prof. Dr. Klemens Böhm
Zweiter Gutachter:	Prof. Dr. Wolfgang Lehner

Leonardo Weiss Ferreira Chaves
Zuverlässige und effiziente Überwachung von Warenflüssen durch Analyse von RFID-Massendaten
Dissertation, Universität Fridericiana zu Karlsruhe (TH)
Karlsruhe, 2010

Abstract

Radio Frequency Identification (RFID) promises optimization of commodity flows in all industry segments. Especially in the retail segment, since retailers gather data manually. Therefore, data is seldom gathered and it contains errors. However, integration of RFID data into business processes is difficult: Due to physical constraints, RFID technology cannot detect all RFID tags from an assembly of items, and RFID tags can be identified by more than one RFID antenna, thus the location of such tags cannot be determined. Furthermore, the deployment of RFID results in large amounts of data that have to be processed efficiently. This thesis tackles these three problems, which are the main technical problems that arise when deploying RFID.

TagMark estimates the number of tagged items from samples like the sales history or the tags read by smart shelves. It adapts an estimation method from biology (mark-recapture) in order to provide guarantees for the accuracy of the estimation and bounds for the sample sizes. A study with RFID-equipped goods acknowledges that TagMark is effective in realistic scenarios, and database experiments with up to 1,000,000 items confirm that it can be efficiently implemented.

RFID Planogram Compliance Verification (**RPCV**) is a method that checks if tagged items that are identified by more than one RFID antenna comply with predefined layout plans, so called planograms. It is based on the observation that the number of times an antenna identifies each item of a certain product type roughly follows a normal distribution. RPCV produces one order of magnitude less wrong predictions than current state of the art, and it requires less data to yield good predictions. A study with RFID-equipped goods and smart shelves shows that RPCV is effective in realistic scenarios.

We use **Materialized Views** (MV) to optimize complex database queries over large amounts of distributed data resulting from the deployment of RFID. MVs refer to precomputed final or intermediate results of database queries. They can improve the query performance by avoiding re-computation of expensive query operations. For a small setting consisting of 24 tables distributed over 9 nodes, an exhaustive search needs 10 hours processing time. Our approach derives a comparable set of MVs within 30 seconds.

These methods were evaluated in real-world scenarios and under extreme conditions, using an actual RFID installation and simulations. Therefore, they should be directly applicable in practice. Besides solving technical problems, these methods also increase the quality of the RFID data. Since the economic viability of an RFID deployment depends on the benefit that the RFID data brings, these methods help making RFID deployments economically viable.

Inhaltsverzeichnis

Kapitel 1

Einleitung

Radiofrequenzidentifikation (RFID) [Fin03] ist eine Technologie für die automatische Identifikation von Objekten. Daten werden auf so genannten Funketiketten gespeichert, die drahtlos ausgelesen werden können. Die Technologie geht weit über die des Barcodes, der bis heute am weitesten verbreiteten Identifikationstechnologie, hinaus: Für das Auslesen von Daten ist kein Sichtkontakt notwendig, es können mehrere Funketiketten auf einmal erfasst werden und die Funketiketten haben eine deutlich höhere Speicherkapazität als der Barcode.

Aufgrund dieser Eigenschaften können mittels RFID Geschäftsprozesse in verschiedenen Branchen optimiert werden. Ein vielversprechendes Einsatzszenario ist die Verbesserung der Warenverfügbarkeit im Einzelhandel [GSH07]. Die Warenverfügbarkeit bezeichnet den Anteil der angebotenen Artikel, der tatsächlich vorhanden ist. Obwohl dieser im Filiallager 98-99% beträgt, liegt er in den Verkaufsflächen lediglich bei 90-93% [ECR03]. Dies ist auf eine ungenaue Bestandsführung zurückzuführen: Im Einzelhandel werden Daten manuell erfasst, was die Erfassung teuer und fehlerbehaftet macht. Der Wareneingang wird oft nur stichprobenartig kontrolliert, Artikel im Filiallager und in den Verkaufsflächen werden selten gezählt und Vorschriften für Produktplatzierungen werden nicht eingehalten [ABG+02].

Die oben beschriebenen Probleme können prinzipiell gelöst werden, wenn jeder einzelne Artikel mit einem Funketikett ausgestattet ist [HWM06]. So kann z.B. der Wareneingang automatisch erfasst und sowohl die Bestände im Filiallager als auch in den Verkaufsflächen können kontinuierlich überwacht werden. Zusätzlich ermöglicht RFID das Auffinden von fehlplatzierten und abgelaufenen Artikeln. Das Optimierungspotential ist groß, denn die schlechte Warenverfügbarkeit verringert den Umsatz eines Einzelhändlers um ca. 4% [GCB02].

Im Rahmen eines Pilotprojektes mit einem großen deutschen Einzelhändler wurden Teile des Sortiments einer Filiale mit Funketiketten ausgestattet, um die Machbarkeit und Vorteile eines RFID-Einsatzes zu bewerten. Auf technischer Ebene zeigten sich allerdings folgende Probleme, die die Integration von RFID-Daten in Unternehmenssoftware erschweren:

1. **Schwankende Anzahl erfasster Artikel:** Aufgrund von physikalischen Interferenzen können nicht alle Artikel in der Umgebung eines RFID-Lesers erfasst werden. Sogenannte Funklöcher entstehen, wenn elektromagnetische Wellen absorbiert werden, oder

wenn sie reflektiert werden und sich somit überlagern [FL04]. Allerdings sind genaue Daten eine Voraussetzung für die Optimierung vieler Prozesse, wie zum Beispiel Bestandsführung, Nachfüllung von Regalen, der Nachbestellung von Artikeln und Bestimmung der Größe von Lager- und Regalflächen.

2. **Ungenauer Ort der Erfassung:** Der Ort eines Artikels, der von einer RFID-Antenne erfasst wird, entspricht dem Ort dieser Antenne. Allerdings können aufgrund von Reflektionen der elektromagnetischen Wellen Artikel von mehreren RFID-Antennen gleichzeitig erfasst werden [FL04]. In solchen Fällen ist es nicht möglich, den genauen Ort eines Artikels zu bestimmen. Dies stellt ein Problem dar, da nicht entschieden werden kann, ob Artikel fehlplatziert sind, und da Artikel mit abgelaufenem Mindesthaltbarkeitsdatum nicht sofort gefunden werden können.

3. **Komplexe Abfragen auf verteilte und große Datenmengen:** Daten können an neuen Orten und in viel kleineren Zeitabständen automatisiert erfasst werden. Ferner sind diese auf verschiedene Rechnersysteme verteilt und müssen verknüpft werden. Eine Optimierung der Abfragen auf solche Daten ist notwendig, um Prozesse des Einzelhandels zu implementieren.

1.1 Beiträge dieser Arbeit

Im Rahmen dieser Arbeit wurden Verfahren entwickelt, um die drei zuvor genannten Probleme zu lösen. Die Verfahren erhöhen die Qualität der RFID-Daten und somit den Mehrwert, den die Technologie bietet. Ferner erfüllen sie die Anforderungen aus dem Einzelhandel, die bei der Integration von RFID-Daten entstehen. Sie wurden auf Datenbankebene (s. Abbildung 1.1) implementiert, so dass die Integration in Unternehmenssoftware durch Anpassung entsprechender Datenbankabfragen einfach erreicht werden kann. Die Verfahren werden im Folgenden skizziert.

Das Verfahren **TagMark** [WBB08] schätzt die Anzahl von Artikeln in der Umgebung eines RFID-Lesers. Es erweitert ein statistisches Verfahren aus der Biologie, das anhand von verschiedenen Stichproben die Anzahl von Individuen in einer Population schätzt. Existierende Schätzverfahren sind nicht anwendbar, da sie Annahmen treffen, die in typischen RFID-Szenarien nicht gelten, wie zum Beispiel statische Populationen, reine Zufallsstichproben oder Stichproben mit benutzerdefinierten Größen. Mit TagMark kann die Genauigkeit einer Schätzung mittels relativer Konfidenzintervalle definiert werden und es existieren obere Schranken für die Größe der benötigten Stichproben. Dies wird analytisch hergeleitet und bewiesen. Des Weiteren zeigen Experimente mit einer RFID-Installation, dass TagMark in realistischen Szenarien anwendbar ist, und Experimente mit synthetischen Daten zeigen, dass TagMark ein schnelles Verfahren ist und auf eine große Anzahl von Artikeln skaliert. Außerdem werden mögliche Angriffe auf TagMark untersucht.

RFID Planogram Compliance Verification (**RPCV**) [WBB10] ist ein Verfahren, das entscheidet, ob Artikel, die von mehr als einer RFID-Antenne erfasst wurden, sich am richtigen Ort

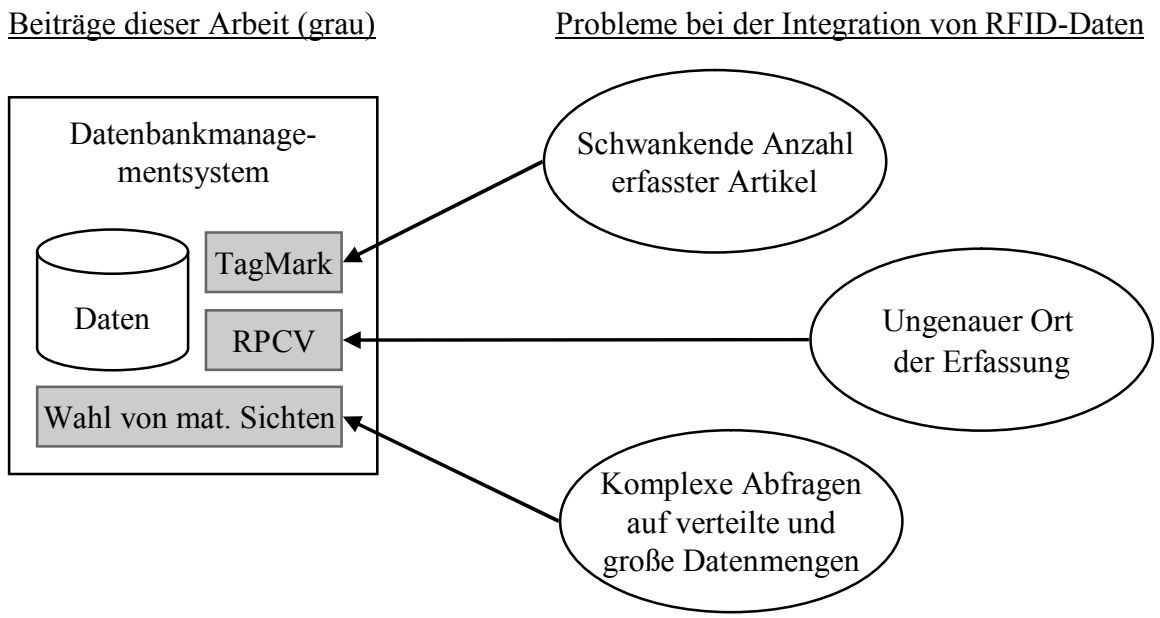

Abbildung 1.1: Architektur und entwickelte Komponenten

innerhalb eines Regals befinden. Dieser Ort ist in vordefinierten Layoutplänen, so genannten Planogrammen, spezifiziert. Das Verfahren basiert auf der Beobachtung, dass Artikel desselben Produkttyps relativ ähnliche Lesemuster aufweisen. RPCV clustert alle Artikel desselben Produkttyps und entscheidet dann, ob diese richtig oder fehlplatziert sind. RPCV erzeugt eine Größenordnung weniger falsche Schätzungen als verwandte Arbeiten, es ist schnell und benötigt weniger RFID-Daten, um gute Schätzungen zu liefern. Experimente mit Daten aus einer RFID-Installation und synthetische Daten zeigen, dass RPCV in realistischen und in extremen Szenarien anwendbar ist.

Die Optimierung von komplexen Abfragen auf verteilte und große Datenmengen erfolgt durch die **Wahl von geeigneten materialisierten Sichten** [WBHB09]. Eine materialisierte Sicht (MS) ist ein (Zwischen-)Ergebnis einer oder mehrerer Abfragen, das vorberechnet und gespeichert wird. Gespeicherte Ergebnisse können verwendet werden, um zukünftige Abfragen schneller zu beantworten. Da die Wahl von geeigneten MS NP-vollständig ist [Gup97], wird das Problem durch die Verwendung mehrerer Heuristiken gelöst, die den Lösungsraum so stark verkleinern, dass ein genetischer Algorithmus angewendet werden kann. Das Verfahren ist schnell: Während ein *Brute-Force*-Algorithmus für ein kleines Szenario 10 Stunden nach der optimalen Menge von MS sucht, liefert das vorgestellte Verfahren eine vergleichbare Lösung in 30 Sekunden. Und es ist skalierbar: eine gute Lösung für ein großes Szenario mit 400 Rechnerknoten wird in ca. 15 Minuten berechnet.

In dieser Arbeit wird insbesondere Wert darauf gelegt, die Anwendbarkeit der entwickelten Verfahren in der Praxis nachzuweisen. Die Verfahren werden aus diesem Grund nicht nur in realistischen Szenarien mit einer RFID-Installation evaluiert, sondern auch unter extremen Bedingungen mittels Simulationen, z.B. mit einer sehr großen und einer sehr kleinen Anzahl

von Artikeln.

Die Verfahren, die im Rahmen dieser Arbeit entwickelt wurden, lösen nicht nur technische Probleme bei dem Einsatz von RFID, sondern spielen auch ein wichtige Rolle bei betriebswirtschaftlichen Betrachtungen: Für die Entscheidung, ob sich der Einsatz von RFID rentiert, werden die Kosten der RFID-Installation ihrem Mehrwert gegenübergestellt. Verfahren wie TagMark und RPCV erhöhen die Qualität der RFID-Daten und somit den Mehrwert, der durch diese Daten erzielt werden kann. In manchen Szenarien wird der Einsatz von RFID erst dadurch rentabel, wenn die Verfahren aus der vorliegenden Arbeit angewendet werden.

1.2 Gliederung der Arbeit

Die Arbeit ist in vier Teile gegliedert. Kapitel 2 stellt die Grundlagen vor. Es werden die Funktionsweise und die Eigenschaften von RFID aufgezeigt. Danach wird dargestellt, wie Prozesse im Einzelhandel durch RFID optimiert werden können, und wie sich die Technologie in die bestehende Infrastruktur eines Einzelhändlers eingliedert. Dabei werden die Probleme erläutert, die bei der Integration von RFID-Daten entstehen.

In Kapitel 3 wird TagMark vorgestellt, ein Verfahren für die Schätzung der Anzahl von Artikeln in der Umgebung eines RFID-Lesers. Als erstes wird die so genannte Rückfangmethode vorgestellt, ein Schätzverfahren aus der Biologie auf dem TagMark aufbaut, das durch verschiedene Stichproben die Größe einer Population schätzt. Danach wird gezeigt, wie dieses Verfahren angepasst und erweitert werden muss, um Anforderungen von RFID-Szenarien im Einzelhandel zu erfüllen. Mit Daten aus einer RFID-Installation und mit synthetischen Daten werden verschiedene Parameter evaluiert, die die Qualität und die Geschwindigkeit der Schätzungen von TagMark beeinflussen.

RPCV, ein Verfahren zur Verifikation von Planogramm-Einhaltung, wird in Kapitel 4 präsentiert. Zunächst wird erläutert, was Planogramme sind und warum deren Einhaltung für Einzelhändler von großer Bedeutung ist. Danach wird der RPCV-Algorithmus vorgestellt. Es folgt eine ausführliche Auswertung von RPCV mit Daten aus einer RFID-Installation und synthetischen Daten. Dabei werden Fälle aufgezeigt, bei denen RPCV gute Schätzungen liefert, sowie die ungünstigsten anzunehmenden Fälle, bei denen das Verfahren herausgefordert wird.

Kapitel 5 stellt dar, wie komplexe Abfragen auf große und verteilte Datenbanken mittels materialisierter Sichten optimiert werden können. Das Verfahren besteht aus mehreren Schritten, die sukzessive den Lösungsraum derart verkleinern, dass zum Schluss ein genetischer Algorithmus angewendet werden kann. Das Verfahren wird in einer simulierten verteilten Datenbankumgebung evaluiert. Es ist schnell und liefert gute Lösungen. Des Weiteren ist es robust gegenüber falschen Schätzungen der Kosten von Datenbankoperatoren.

Die Arbeit schließt mit einer Zusammenfassung und einem Ausblick auf zukünftige Forschungsvorhaben.

Kapitel 2

Grundlagen

In diesem Kapitel werden die Funktionsweise von RFID und dessen Einsatzmöglichkeiten im Einzelhandel erläutert. Dabei werden Probleme bei der Datenerfassung aufgezeigt, die eine Integration von RFID-Daten in Unternehmenssoftware erschweren. Zusätzlich werden spezifische Anforderungen aus dem Einzelhandel vorgestellt, die bei der Integration von RFID-Daten erfüllt werden müssen.

2.1 Grundlagen der Radiofrequenzidentifikation

Radiofrequenzidentifikation (RFID) [Fin03] ist eine Technologie für die automatische Identifikation von Objekten. Daten werden auf so genannte Funketiketten gespeichert, die drahtlos ausgelesen werden können. Im Vergleich zur heute am weitesten verbreiteten Identifikationstechnologie, dem Barcode, bietet RFID viele Vorteile: Funketiketten können ohne Sichtkontakt ausgelesen werde, es können mehrere Funketiketten auf einmal erfasst werden und sie können mehr Daten als ein Barcode speichern. Des Weiteren besitzen Funketiketten eine Identifikationsnummer, die weltweit eindeutig ist.

Die Grundlagen der RFID-Technologie sind seit mehreren Jahrzehnten bekannt [Sto48], allerdings konnten die Herstellungskosten für die Technologie erst in den letzten 10 bis 15 Jahren derart gesenkt werden, dass ein massenhafter Einsatz der Technologie möglich wurde [Wan06].

2.1.1 Funktionsweise

Ein RFID-System besteht aus zwei Komponenten: RFID-Leser und Funketiketten. RFID-Leser sind Rechner, die die Protokolle für die Kommunikation mit den Funketiketten und anderen Rechnern implementieren. Ein RFID-Leser kann mehrere RFID-Antennen ansteuern (1:n Relation) und eine Antenne kann mit vielen Funketiketten gleichzeitig kommunizieren.

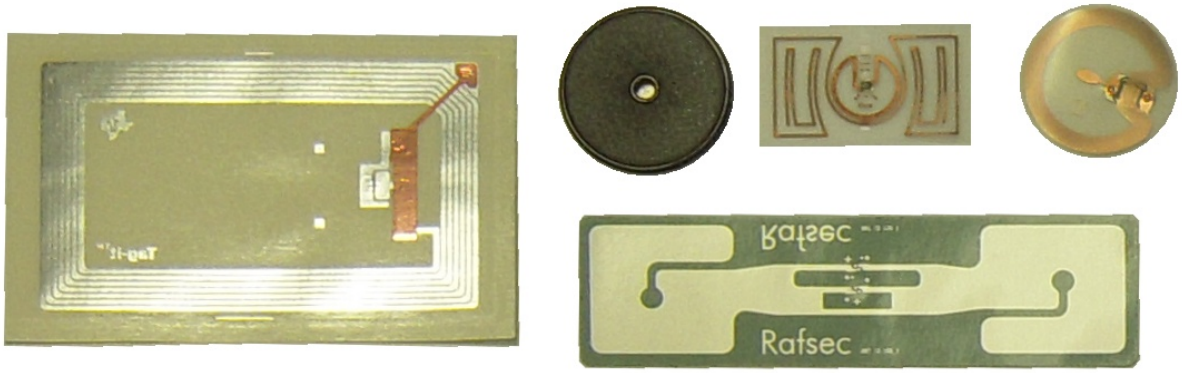

Abbildung 2.1: Beispiel verschiedener RFID-Funketiketten

Die Funketiketten, die in der Literatur auch als *Tag* oder *Transponder* bezeichnet werden, lassen sich in *aktive* und *passive* Funketiketten unterteilen. Aktive Funketiketten benötigen eine eigene Energiequelle, wie z.B. eine Batterie. Aus diesem Grund sind solche Funketiketten hochpreisig und die Batterie schränkt ihre Lebensdauer ein. Aktive Funketiketten werden beispielsweise bei Flugzeugen eingesetzt, um während des Anfluges ihr Herkunftsland zu ermitteln. Passive Funketiketten hingegen benötigen keine eigene Energiequelle und müssen daher nicht gewartet werden. Sie sind klein und günstig. Aus diesem Grund werden in der Regel nur passive Funketiketten im Einzelhandel eingesetzt. In dieser Arbeit wird deshalb der Fokus auf passive Funketiketten gelegt.

Im Wesentlichen bestehen passive Funketiketten aus drei Komponenten [Wan06]: einer Antenne, einem Mikrochip und einer Verpackung, die die Komponenten zusammenhält und schützt, vgl. Abbildung 2.1. Passive Funketiketten können in einer der folgenden Frequenzen operieren: *Low Frequency* (LF, 135kHz), *High Frequency* (HF, 13,56MHz), *Ultra High Frequency* (UHF, 868MHz oder 915MHz) und Mikrowelle (2,5GHz oder 5,8GHz). Die Kommunikation mit dem Leser funktioniert wie folgt: Der Leser erzeugt ein Energiefeld, das dem passiven Funketikett als Stromversorgung dient. Je nach Frequenz des Funketiketts erzeugt der Leser ein magnetisches (bei LF und HF) oder ein elektromagnetisches Feld (bei UHF und Mikrowelle). Ist das Funketikett mit Strom versorgt, kann es seine Daten drahtlos an den Leser übertragen. Die Reichweite, in der ein Funketikett mit dem Leser kommunizieren kann, hängt von dessen Frequenz ab. UHF erlaubt eine große Reichweite und ist daher besonders für den Einzelhandel geeignet. Aus diesem Grund wurden im Rahmen des erwähnten Pilotprojektes Funketiketten dieses Typs verwendet.

Da die Kommunikation mit den Funketiketten und deren Stromversorgung drahtlos erfolgt, kann es zu verschiedenen Arten von Störungen kommen, die im Folgenden dargestellt werden [Fin03, FL04]:

1. **Unzureichende Stromversorgung:** Die Funketiketten werden durch das elektromagnetische Feld des RFID-Lesers mit Strom versorgt. Das elektromagnetische Feld kann allerdings nur eine bestimmte Anzahl von Funketiketten mit Strom versorgen. Wenn viele

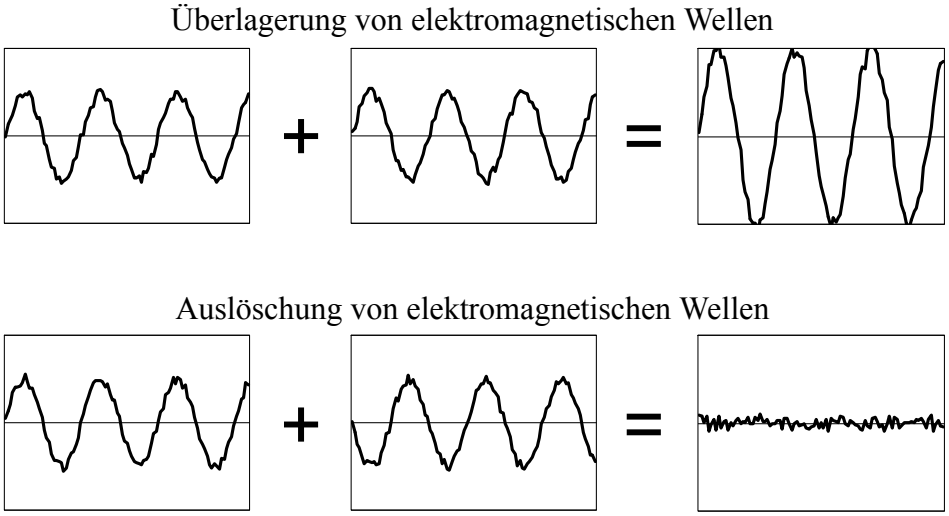

Abbildung 2.2: Überlagerung und Auslöschung von elektromagnetischen Wellen

Funketiketten zu nah beieinander platziert sind, können einzelne Funketiketten nicht mit Strom versorgt werden.

2. **Überlagerung und Auslöschung von elektromagnetischen Wellen:** Objekte in der Umgebung des RFID-Lesers können elektromagnetische Wellen reflektieren (z.B. Metalle) oder absorbieren (z.B. Flüssigkeiten). Dadurch können sich elektromagnetische Wellen überlagern oder auslöschen, vgl. Abbildung 2.2. Durch die Überlagerung können sich elektromagnetische Wellen weiter ausbreiten, so dass fälschlicherweise Funketiketten von benachbarten Orten erfasst werden. Und durch Auslöschungen entstehen so genannte Funklöcher, in denen eine Kommunikation nicht möglich ist.

Da diese Störungen aufgrund von physikalischen Effekten auftreten, ist nicht zu erwarten, dass sie durch eine zukünftige RFID-Technologie behoben werden.

2.1.2 RFID-Daten

Funketiketten können bis zu 64KB [Fuj08] speichern. Auf ihrem Speicher werden eine eindeutige Identifikationsnummer (ID) und Nutzdaten abgelegt. Die ID wird nach dem Standard des elektronischen Produktkodes (engl. *Electronic Product Code*, EPC) kodiert und belegt mindestens 96 Bits [EPC06]. Der restliche Speicher, dessen Verwendung nicht standardisiert ist, steht für Nutzdaten zur Verfügung.

Der EPC sieht verschiedene Arten der Kodierung einer ID vor. Für den Einzelhandel erweist sich die *Serialized Global Trade Item Number* (SGTIN) [EPC06] als zweckmäßig. Sie erweitert die im Barcode kodierte GTIN um eine Seriennummer, vgl. Abbildung 2.3. Dadurch ist die Interoperabilität mit bestehenden Systemen, die nur die GTIN und nicht die SGTIN unterstützen, einfacher: die SGTIN kann durch Weglassen der Seriennummer auf die GTIN abgebildet

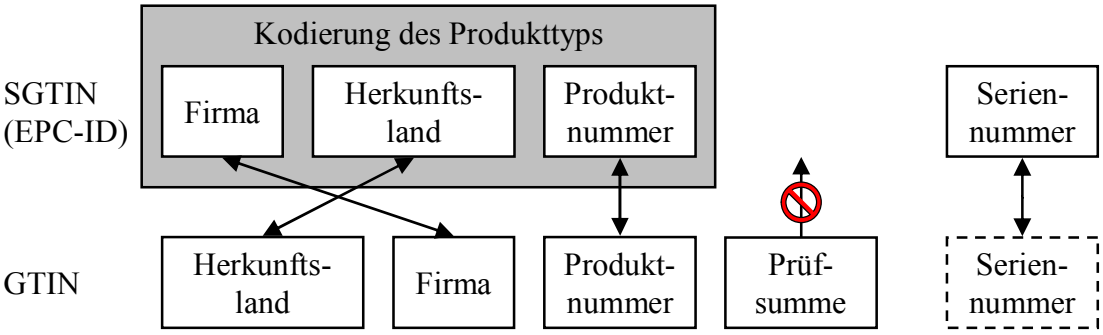

Abbildung 2.3: Struktur der RFID-Daten am Beispiel der SGTIN

werden. Ein Produkttyp wird eindeutig durch das Tupel bestehend aus Firma, Herkunftsland und Produktnummer kodiert. Ein einzelner Artikel kann anhand des Produkttyps und der Seriennummer eindeutig identifiziert werden.

2.2 Radiofrequenzidentifikation im Einzelhandel

In diesem Abschnitt wird beschrieben, wie Prozesse im Einzelhandel durch den Einsatz von RFID optimiert werden können. Der Fokus liegt auf Prozessen in der Filiale, da dort das größte Optimierungspotential existiert. Obwohl die Warenverfügbarkeit im Filiallager 98-99% beträgt, liegt die Warenverfügbarkeit in den Verkaufsflächen zwischen 90% und 93% [ECR03].

2.2.1 Szenariobeschreibung

Innerhalb eines Pilotprojektes mit einem großen deutschen Einzelhändler wurden verschiedene Szenarien für den Einsatz von RFID evaluiert. Viele Prozesse in der Filiale können optimiert werden, wenn alle Artikel mit einem Funketikett ausgestattet sind [ABG+02, GSH07, Loe05]. Die Erfassung dieser Artikel kann auf verschiedene Arten erfolgen:

- *RFID-Gates:* RFID-Antennen werden an Durchgängen oder Türen angebracht, um passierende Artikel zu erfassen. Ein RFID-Gate kann beispielsweise verwendet werden, um den Wareneingang zu kontrollieren. Artikel auf einer Palette werden durch einen RFID-Leser erfasst, wenn die Palette durch ein RFID-Gate bewegt wird.
- *Mobile RFID-Leser:* Ein tragbarer Rechner (z.B. ein *Personal Digital Assistant*, PDA) wird mit einem RFID-Leser ausgestattet. Angestellte können solche Geräte mit sich führen und beispielsweise in regelmäßigen Abständen Artikel in einem Regal erfassen. Jede Erfassung ist allerdings mit Personalaufwand verbunden.
- *Smart-Shelves:* In jedem Regal werden die Regalböden mit RFID-Antennen ausgestattet [DKB03]. So kann der Inhalt der Regale kontinuierlich erfasst und überwacht werden. Des Weiteren kann prinzipiell die Position eines Artikels im Regal bestimmt werden.

Abbildung 2.4: Ein Smart-Shelf bei Laborversuchen (links) und bei Feldversuchen (rechts)

Die Hardwarekosten von RFID-Gates und von mobilen RFID-Lesern sind im Vergleich zu Smart-Shelves gering, da nur eine kleine Anzahl von RFID-Lesern erforderlich ist. Allerdings liefern Smart-Shelves mehr Informationen: Zu jedem Zeitpunkt kann der Inhalt eines Regals bestimmt werden, und der Ort eines Artikels ist identisch mit dem Ort der Antenne, die den Artikel gerade erfasst. Die örtliche Auflösung bei RFID-Gates ist sehr grob: es kann beispielsweise bestimmt werden, wie viele Artikel in den Verkaufsbereich der Filiale transportiert wurden, es kann aber nicht bestimmt werden, ob diese Artikel tatsächlich im Regal liegen und dem Kunden zur Verfügung stehen. Artikel können entwendet werden, fehlplatziert sein oder im Einkaufswagen eines Kunden liegen. Bei mobilen RFID-Lesern ist die örtliche Auflösung vergleichbar mit derjenigen von Smart-Shelves, allerdings können die Artikel nicht kontinuierlich überwacht werden und jede Erfassung ist mit Personalaufwand verbunden. Aus den genannten Gründen werden Installationen von Smart-Shelves im Einzelhandel bevorzugt.

Im Folgenden wird von einer Datenerfassung durch Smart-Shelves ausgegangen. Abbildung 2.4 zeigt das Smart-Shelf, das in dem Pilotprojekt eingesetzt wurde. Es wurde bei Feld- und Laborversuchen eingesetzt und lieferte die Daten für die Auswertung der im Rahmen dieser Arbeit entwickelten Verfahren. Jeder Regalboden wurde mit zwei RFID-Antennen ausgestattet. Diese sind im linken Teil der Abbildung als graue Kästen, die an der Unterseite der Regalböden befestigt sind, zu sehen. In diesem Aufbau kann prinzipiell unterschieden werden, ob sich Artikel auf der linken oder der rechten Hälfte eines Regalbodens befinden.

Des Weiteren werden Kassen mit fest eingebauten RFID-Lesern ausgestattet. An jeder Kasse wird eine RFID-Antenne angebracht, so dass die Antenne die Artikel erfassen kann, die von dem Laufband entnommen und über der Antenne in den sich hinter der Kasse befindlichen Ablagebereich bewegt werden. So können Artikel einzeln erfasst werden, analog zur Erfassung mit heutigen Barcode-Systemen. Diese händische Erfassung ist notwendig, da bei Abverkäufen alle Artikel erfasst werden müssen. Bei einer automatischen Erfassung eines kompletten Warenkorbes könnten einzelne Artikel nicht erfasst werden. Gründe dafür werden in Abschnitt 2.3 vorgestellt.

2.2.2 Geschäftsprozesse im Einzelhandel

Durch den Einsatz von RFID können viele Prozesse optimiert werden, die sonst auf einer manuellen Datenerfassung beruhen, und daher sowohl teuer als auch fehlerbehaftet sind. Im Folgenden werden die Prozesse vorgestellt, die im Einzelhandel die meisten Vorteile bieten [BDF+07]. Diese bringen somit hohe Kosteneinsparungen und/oder hohe Umsatzsteigerungen mit sich.

Bei der **Bestandskontrolle** bzw. Inventur muss die Anzahl von Artikeln pro Produkttyp erfasst werden. Die Genauigkeit der Inventur bezeichnet die Anzahl von erfassten Artikeln geteilt durch die Anzahl von tatsächlich vorhandenen Artikeln. Es sind verschiedene Genauigkeiten gefordert, so dass der Bestand der Jahresinventur beispielsweise genauer sein muss, als der Bestand der regelmäßigen Inventur für die Nachfüllung der Regale. Die Bestandskontrolle ist ein zentraler Prozess, da die Bestandsdaten die Grundlage für viele weitere Prozesse bilden, welche die Warenverfügbarkeit beeinflussen, wie z.B. **Nachfüllung von Regalen**, **Bestellungen von Artikeln** und **Bestimmung der Größe von Lager- und Verkaufsflächen**.

Heute ist die Bestandskontrolle größtenteils ein manueller Prozess. Ein so genanntes Warenwirtschaftssystem (WWS) bestimmt den aktuellen Bestand aufgrund des alten Bestandes, von Lieferungen und von Abverkäufen. Das System löst manuelle Zählungen in regelmäßigen Abständen aus, oder bei Auffälligkeiten, wie z.B. bei negativem Bestand oder einem langen Zeitraum ohne Abverkäufe. Die Bestandsdaten im WWS sind allerdings aufgrund von Schwund [HD02] und von unverkäuflichen Produkten [Lig02], die beschädigt oder verdorben sind, ungenau. Ferner entspricht die Anzahl von Artikeln im WWS regelmäßig nicht der Anzahl von Artikeln, die tatsächlich zum Verkauf stehen. Dies geschieht aufgrund von fehlplatzierten Artikeln sowie Artikeln, die sich in den Einkaufswägen von Kunden befinden.

Die ungenauen Daten im WWS können dazu führen, dass sich keine Artikel eines Produkttyps in den Verkaufsflächen befinden. Es entstehen so genannte Regallücken (engl. *Out-of-Stocks*), so dass Kunden die gewünschten Artikel nicht kaufen können. Kunden entscheiden sich oft für den Kauf von günstigeren Alternativen oder erwerben den Artikel bei einem anderen Händler. In beiden Fällen verringert sich der Umsatz des Filialbetreibers. Das Optimierungspotential ist groß, denn eine schlechte Warenverfügbarkeit verringert den Umsatz eines Einzelhändlers um ca. 4% [GCB02]. Gründe für die schlechte Warenverfügbarkeit sind hauptsächlich betriebswirtschaftlicher Natur. Das häufige Zählen von Artikeln führt zu genaueren Daten,

bringt jedoch einen höheren Personalaufwand mit sich. Das betriebswirtschaftliche Optimum setzt somit keine optimale Warenverfügbarkeit voraus, vgl. [TF07].

Der Einsatz von RFID erlaubt die kontinuierliche Überwachung der Bestände. Durch genaue Bestandsdaten können Regallücken vermieden werden, so dass der Umsatz steigt. Des Weiteren können Bestellmengen, Lager- und Verkaufsflächen sowie die daraus resultierenden Kosten minimiert werden. Der Personalaufwand wird ebenfalls verringert.

Die **Einhaltung von Planogrammen** ist ein weiterer wichtiger Prozess für den Einzelhandel. Dabei geht es um die Einhaltung von vordefinierten Layoutplänen, die den genauen Ort eines Artikels im Regal spezifizieren. Planogramme verbessern den visuellen Eindruck und erhöhen die Anzahl von Spontankäufen. Zusätzlich optimieren Planogramme die Nutzung der Verkaufsflächen und erhöhen die Warenverfügbarkeit. Planogramme sind wichtig, da eine schnelle und genaue Einhaltung von Planogrammen den Gewinn eines Einzelhändlers um bis zu 8,1% steigern kann [Bis00]. Prinzipiell kann durch den Einsatz von RFID die Einhaltung von Planogrammen verifiziert werden. Der Ort von einem Artikel in einem Smart-Shelf ist identisch mit dem Ort der Antenne, welche den Artikel erfasst. Durch diese genaue örtliche Auflösung können weitere Prozesse optimiert werden, wie das **Finden von Artikeln mit abgelaufenem Mindesthaltbarkeitsdatum**, das **Finden von Artikeln für Kunden** und die **Durchführung von filialinternen Umlagerungen** im Allgemeinen. Beispiele dafür sind Umlagerungen vom Filiallager in die Verkaufsfläche (Nachfüllung), von der Verkaufsfläche ins Filiallager, innerhalb des Filiallagers und innerhalb der Verkaufsfläche.

2.3 Integration von RFID-Daten

Aufgrund der drahtlosen Stromversorgung und Kommunikation weist RFID Charakteristika auf, die bei der Integration von RFID-Daten berücksichtigt werden müssen. Als erstes werden diese Charakteristika vorgestellt. Anschließend wird diskutiert, warum einige dieser Charakteristika Probleme bei der Integration von RFID-Daten verursachen.

C1: Variierende Anzahl erfasster Artikel: Aufgrund von physikalischen Interferenzen, und da Artikel verkauft und nachgefüllt werden, variiert die Anzahl von Artikeln, die erfasst werden, sehr stark [WBB08].

C2: Kontinuierliche Datenströme: Im Einzelhandel werden kontinuierliche Datenströme von RFID-Lesungen erzeugt.

C3: Unvorhersehbare Menge erfasster Artikel: Aufgrund von physikalischen Effekten wie Absorption und Reflektion, kann das Lesefeld einer RFID-Antenne, also das Gebiet, in dem eine Antenne Artikel erfassen kann nicht vorhergesehen werden. Des Weiteren stellen Mengen von erfassten Artikeln im Allgemeinen keine reinen Zufallsstichproben dar [FL04, GS105].

C4: Offene Populationen: Die Anzahl von Artikeln, die sich tatsächlich in einem Regal befinden, ändert sich ständig, da Regale geleert und kontinuierlich aufgefüllt werden.

C5: Große Datenmengen: RFID-Anwendungen im Einzelhandel sind durch eine große Anzahl von Artikeln charakterisiert. Dadurch ergeben sich große Datenmengen, welche die Laufzeit von Schätzverfahren herausfordern.

C6: Artikel weisen ähnliche Lesemuster auf: Artikel desselben Produkttyps besitzen dieselben physikalischen Eigenschaften und weisen daher relativ ähnliche Lesemuster auf. Eine Normalverteilung beschreibt die Anzahl von Lesungen von Artikeln eines bestimmten Produkttyps durch die Antenne, die diese Artikel laut Planogramm erfassen sollte, sehr gut. Dasselbe gilt für die Anzahl von Lesungen durch andere Antennen.

2.3.1 Schwankende Anzahl erfasster Artikel

Für die Bestandskontrolle sind möglichst genaue Daten erwünscht, damit Regallücken vermieden und damit Bestellmengen, Lager- und Verkaufsflächen minimal gehalten werden können. Je ungenauer die Daten, desto größer sind die Sicherheitsbestände, die vorgehalten werden müssen.

RFID-Leser können prinzipiell Artikel in ihrer Umgebung erfassen, allerdings kann nicht sichergestellt werden, dass tatsächlich alle Artikel erfasst werden (Charakteristik C1). Dies geschieht aufgrund der drahtlosen Stromversorgung und Kommunikation. Gegenstände, die Metalle oder Flüssigkeiten enthalten, absorbieren oder reflektieren elektromagnetische Wellen. Dadurch können sich elektromagnetische Wellen überlagern, so dass Funklöcher entstehen. Die Anzahl von Funketiketten, die über das elektromagnetische Feld eines RFID-Lesers mit Strom versorgt werden können, ist beschränkt. Sind viele Funketiketten zu nah beieinander, können einzelne Funketiketten nicht mit Strom versorgt werden [FL04]. Da es sich hierbei um physikalische Effekte handelt, ist nicht zu erwarten, dass eine zukünftige RFID-Technologie diese Probleme beheben wird. Des Weiteren besteht aus betriebswirtschaftlicher Sicht die Forderung nach günstigen und kleinflächigen Funketiketten. Dies beeinflusst die RFID-Lesungen auf eine negative Art und Weise.

Die **RFID-Leserate** wird definiert als die Anzahl von erfassten Artikeln geteilt durch die Anzahl von Artikeln, die sich in der Umgebung eines RFID-Lesers befinden. Die Leserate kann vom RFID-Leser nicht beobachtet werden (Charakteristik C1). Tabelle 2.1 zeigt typische RFID-Leseraten für Produkte aus dem Einzelhandel. Die Leseraten sind unterschiedlich und zum Teil sehr niedrig. Des Weiteren können die Leseraten für Artikel desselben Produkttyps stark schwanken, da die Leserate von der Umgebung abhängt, wie zum Beispiel vom Produkttyp und der Anzahl von Artikeln in der Nähe des RFID-Lesers. Dieses Problem wird in einer Reihe von Experimenten in Abschnitt 3.5.1 weiter analysiert, vgl. dortige Abbildung 3.1.

Aufgrund der schlechten Leserate wird in der Praxis mit der Anzahl von erfassten Artikeln lediglich der Mindestbestand geprüft. Wenn beispielsweise mehr Artikel erfasst werden, als der Mindestbestand vorsieht, ist keine Nachfüllung erforderlich. Ist die Anzahl erfasster Artikel geringer als der Mindestbestand, so ist unklar, ob dies tatsächlich der Fall ist, oder ob es an der schlechten Leserate liegt. Für den Wareneingang werden ebenfalls Schwellenwerte bestimmt [BLHS04]: Wurde beispielsweise ein großer Anteil der Artikel erfasst, die in einer

Tabelle 2.1: Typische Leseraten von RFID [GS105]

Produkttyp	*RFID-Leserate*
Schokoladenmousse	33.0%
Joghurt	37.5%
Duschgel	56.0%
Sonnencreme	67.8%
Pflaumenkuchen	70.0%
Rasierschaum	75.8%
Reiskuchen	95.0%

gelieferten Palette erwartet wurden, dann wird eine vollständige Lieferung angenommen. Ansonsten muss manuell geprüft werden. Durch diesen Umgang mit der schwankenden Anzahl erfasster Artikel entstehen allerdings Kosten durch Personalaufwand, durch größere Sicherheitsbestände, Bestellmengen, Lager- und Verkaufsflächen.

Um der schwankenden Anzahl erfasster Artikel entgegen zu wirken, können Schätzverfahren angewendet werden. Solche Verfahren müssen folgende Eigenschaften von RFID berücksichtigen: Der RFID-Leser produziert kontinuierliche Datenströme (Charakteristik C2). Betrachtet man den Datenstrom innerhalb eines gewissen Zeitfensters als eine Stichprobe für ein Schätzverfahren, so können einzelne Stichproben zu gering sein und das Zusammenführen aller vergangenen Stichproben kann zu einem veralteten Ergebnis führen. Des Weiteren stellt die Menge an erfassten Artikeln im Allgemeinen keine Zufallsstichprobe dar (Charakteristik C3), da die Anzahl von Artikeln in der Umgebung eines RFID-Lesers durch das Kaufverhalten der Kunden beeinflusst wird. Und dieses ist nicht zufällig. Auch nicht, wenn die Artikel desselben Produkttyps betrachtet werden, da Kunden beispielsweise die Artikel in der vordersten Reihe im Regal bevorzugen. Des Weiteren ändert sich die Anzahl von Artikeln, die sich tatsächlich im Regal befinden, sehr oft, da Regale geleert und kontinuierlich aufgefüllt werden (Charakteristik C4).

Da im Einzelhandel eine sehr große Anzahl von Artikeln vorliegt und an vielen Orten sowie in kleinen Zeitabständen RFID-Daten erfasst werden, ergeben sich große Datenmengen (Charakteristik C5). Diese stellen eine Herausforderung für die Laufzeit von Schätzverfahren dar.

2.3.2 Ungenauer Ort der Erfassung

RFID kann prinzipiell eingesetzt werden, um die Einhaltung von Planogrammen zu verifizieren. Entspricht jeder Ort in einem Planogramm einer RFID-Antenne, dann muss lediglich geprüft werden, welche Antenne welche Artikel erfasst. Werden allerdings mehrere RFID-Antennen nah beieinander eingesetzt, so können Artikel von mehreren Antennen gleichzeitig

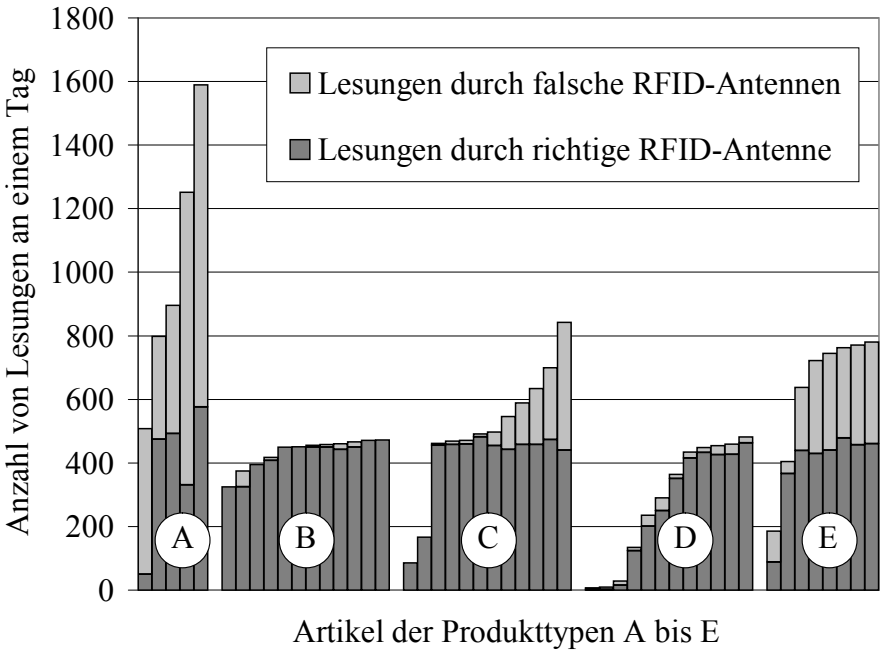

Abbildung 2.5: RFID-Daten aus den Feldversuchen

erfasst werden. Solche Fälle kommen oft vor und es kann nicht entschieden werden, ob diese Artikel das Planogramm einhalten. Werden die RFID-Antennen in größeren Abständen voneinander eingesetzt, so wirkt sich dies negativ auf die Leserate aus. Aus diesem Grund werden solche Installationen in der Praxis vermieden.

Das Lesefeld eines RFID-Lesers ist der Bereich, in dem Artikel erfasst werden können. Es ist nicht möglich, das Lesefeld eines RFID-Lesers zu bestimmen (Charakteristik C3), da sich elektromagnetische Wellen aufgrund von Interferenzen überlagern und sich somit weiter ausbreiten oder sich auslöschen können, so dass Funklöcher entstehen [FL04].

Dieses Problem wurde im Rahmen von Feldversuchen bei einem großen Einzelhändler beobachtet. Typische Ergebnisse sind in Abbildung 2.5 dargestellt. Jeder Balken repräsentiert einen Artikel. Die Ergebnisse zeigen Artikel aus 5 verschiedenen Produkttypen. Da das Planogramm (nicht zu sehen) eine unterschiedliche Anzahl von Artikeln für jeden Produkttyp spezifiziert, ist die Anzahl von Artikeln bzw. Balken in der Abbildung für jeden Produkttyp verschieden. An dieser Stelle werden zwei Bezeichnungen eingeführt, die im Folgenden verwendet werden: Als **richtige Antenne** wird die Antenne bezeichnet, die einen Artikel laut Planogramm erfassen sollte. Die restlichen Antennen, z.B. von benachbarten Orten, werden als **falsche Antennen** bezeichnet. Die dunkelgrauen Balken zeigen die Anzahl von Lesungen durch die richtige Antenne. Der entsprechende hellgraue Balken zeigt die Anzahl von Lesungen durch falsche Antennen. Die Balken sind gestapelt. Die Daten zeigen, warum es schwierig ist zu entscheiden, an welchem Ort ein Artikel platziert ist. Obwohl kein Artikel fehlplatziert war, wurde jeder Artikel mindestens ein Mal von falschen Antennen erfasst. Viele Artikel wurden häufiger durch falsche Antennen als durch die richtige erfasst (Produkt A). Des Weiteren

variiert die Anzahl von Lesungen in einem gewissen Zeitfenster sehr (z.B. Produkt D), da Artikel in Funklöchern liegen können (Charakteristik C1). Aufgrund dieser Probleme werden RFID-Installationen in der Praxis nicht verwendet, um die Einhaltung von Planogrammen zu verifizieren.

Die Daten in der Abbildung zeigen, dass Artikel desselben Produkttyps ähnliche Lesemuster aufweisen (Charakteristik C6). Ein Lesemuster ist die Verteilung der Anzahl von Lesungen durch die richtige und durch falsche Antennen. Es wurde beobachtet, dass eine Normalverteilung die Verteilung der Anzahl von Lesungen gut beschreibt: Wenn x „die Anzahl von Lesungen für einen Artikel“ und $f(x)$ „die Anzahl von Artikeln, die x-Mal erfasst wurden“, beschreibt, dann gilt $f(x) \sim N(\mu, \sigma^2)$. Auf dieser Beobachtung basiert das im Rahmen dieser Arbeit entwickelte Verfahren RPCV. Dabei wird überprüft, ob Artikel, die von mehr als einer RFID-Antenne erfasst wurden, das Planogramm einhalten. Ein weiterer Aspekt, der im Allgemeinen von Schätzverfahren berücksichtigt werden muss, sind die großen Datenmengen, die bei dem Einsatz von RFID im Einzelhandel entstehen (Charakteristik C5). Schätzverfahren müssen also schnell sein und auf eine große Anzahl von Artikeln skalieren.

2.3.3 Komplexe Abfragen auf verteilte und große Datenmengen

Prozesse im Einzelhandel benötigen oft Daten, die auf verschiedenen Rechnern gespeichert sind. Aus drei Gründen ist es schwierig, Datenbankabfragen von solchen Prozessen zu optimieren: Erstens ist die IT-Infrastruktur von Einzelhändlern sehr groß, zweitens besteht sie aus heterogener Hardware und drittens muss sie komplexe Abfragen beantworten.

Infrastruktur: Abbildung 2.6 zeigt die typische IT-Infrastruktur eines großen Einzelhändlers. Ein *Enterprise Resource Planning* (ERP) System unterstützt gemeinsame Prozesse von allen Filialen, wie z.B. Logistik, Einkauf, Stammdaten-Verwaltung oder *Data-Warehousing*. Es wird angenommen, dass der Einzelhändler 100 Filialen besitzt. Jede Filiale betreibt ein eigenes Warenwirtschaftssystem (WWS). Dieses speichert Informationen über bestellte Artikel, Abverkäufe, Bestände usw. Jede Filiale hat ein Kassensystem (engl. *Point-of-Sales*, POS) und zwei RFID-Leser. Das POS speichert die Preise von Produkten und verwaltet alle Kassen in der Filiale. Die RFID-Leser speichern die Daten aller Smart-Shelves im Lager- und im Verkaufsbereich. Bei einem solchen Szenario kann jeder Rechnerknoten eine Sicht materialisieren, sogar ein RFID-Leser.

Hardware: Die IT-Infrastruktur eines Einzelhändlers entwickelt sich im Laufe der Zeit. Aus diesem Grund haben Rechnerknoten verschiedene Ressourcen. Gewisse Rechnerknoten, wie ERP-Systeme und RFID-Leser, sind auf spezifische Aufgaben zugeschnitten. Die Netzwerkbandbreite kann massiv variieren. WWS von großen Filialen sind über schnelle Leitungen mit dem ERP-System verbunden, während kleine Filialen sich über Telefonleitungen einwählen. Innerhalb der Filialen wird Ethernet verwendet.

Arbeitslast: Die Arbeitslast (engl. *Workload*) besteht aus einer hohen Anzahl von komplexen Abfragen und Aktualisierungen, die in verschiedenen Häufigkeiten von verschiedenen Rechnerknoten stammen. Aufgrund von verschiedenen Abverkäufen, Angeboten und Haupt-

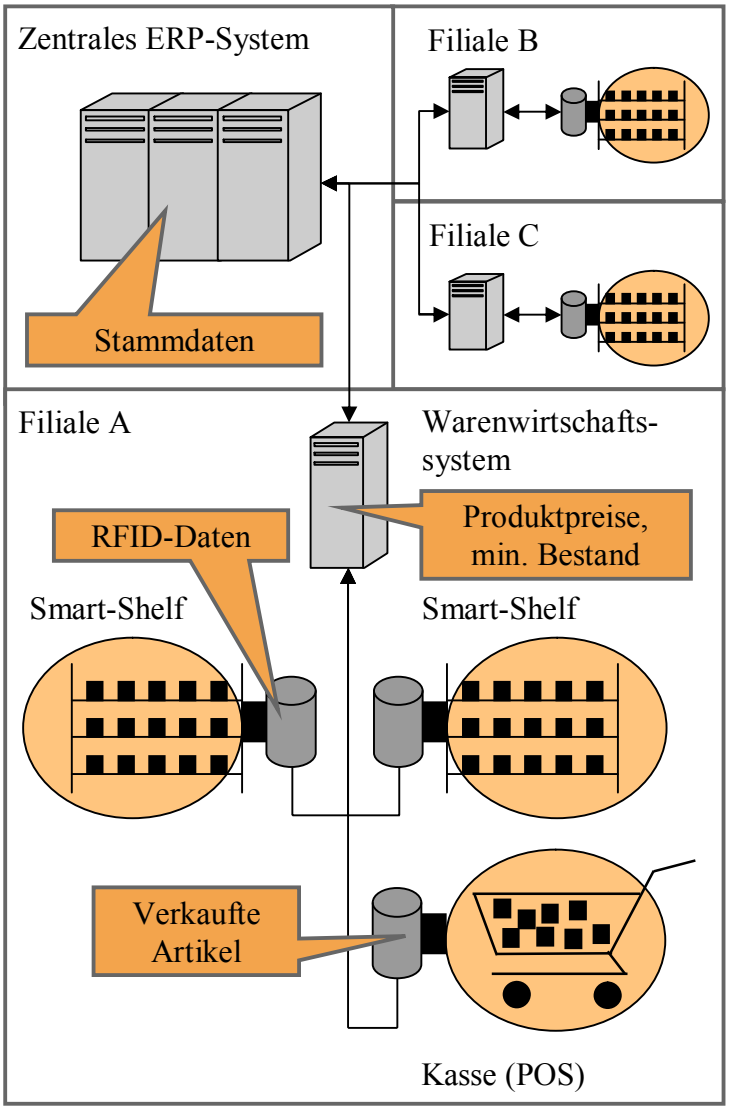

Abbildung 2.6: IT-Infrastruktur eines Einzelhändlers [WBB08]

verkaufszeiten ist die Arbeitslast von jedem Rechnerknoten unterschiedlich. Die Stammdaten und die zusammengefassten Verkaufsdaten der ERP-Systeme werden selten aktualisiert. Das POS und die RFID-Leser erhalten ca. 25.000 Aktualisierungen am Tag. Das WWS führt ca. 9.000 Aktualisierungen am Tag aus.

In der Praxis werden RFID-Daten daher frühzeitig aggregiert [BLHS04]. Das Datenaufkommen kann verringert werden, indem die RFID-Rohdaten, bestehend aus Ort, Zeitstempel und EPC, auf eine Liste mit der Anzahl von Artikeln eines jeden Produkttyps abgebildet werden. Dadurch verringert sich zwar der Rechenaufwand bei der Datenverarbeitung, allerdings auch der Mehrwert, der aus diesen Daten gewonnen werden kann. Beispielsweise können anhand der aggregierten Daten nicht diejenigen Artikel identifiziert bzw. gefunden werden, die ein

abgelaufenes Mindesthaltbarkeitsdatum aufweisen.

2.3.4 Sonstige Probleme bei dem Einsatz von RFID im Einzelhandel

Der Vollständigkeit halber wird an dieser Stelle auf zwei weitere Probleme eingegangen, die bei dem Einsatz von RFID berücksichtigt werden müssen, jedoch nicht Gegenstand dieser Arbeit sind. Es handelt sich dabei um die Wirtschaftlichkeit und um den Datenschutz.

Wirtschaftlichkeit: Während der Einsatz von RFID auf einzelnen Kartons oder Paletten weit verbreitet ist, ist der Einsatz auf einzelnen Artikeln zurzeit auf hochpreisige Produkte beschränkt. Dies ist unter anderem auf den Preis der RFID-Hardware zurückzuführen. Das größte Problem sind die Preise der Funketiketten, da es sich hierbei um variable Kosten handelt. Der Preis eines passiven Funketiketts beträgt im Durchschnitt zwischen 0,20 US$ und 1,00 US$ [AM05]. Für niedrigpreisige Produkte, wie zum Beispiel Milch, sind Funketiketten noch viel zu teuer. Daher sind niedrigpreisige Funketiketten eine Voraussetzung für den flächendeckenden Einsatz von RFID im Einzelhandel. Zur Kostensenkung werden verschiedene Ansätze verfolgt [RNL05, Sar01]. Ein prominentes Beispiel, das die Herstellung von sehr billigen Funketiketten erlaubt, ist das Drucken von elektronischen Schaltungen mit leitfähigen Farben, vgl. [SCFV+08].

Die Wirtschaftlichkeit eines RFID-Einsatzes ist nicht alleine von den Hardwarekosten abhängig, sondern auch von dem Mehrwert, den die Technologie mit sich bringt. Trotz hoher Hardwarekosten kann der Einsatz von RFID wirtschaftlich sein, wenn z.B. die Kosten interner Prozesse stark gesenkt werden können (z.B. [BGH+02]) und/oder der Umsatz gesteigert werden kann (z.B. [ABG+02]). Um die Wirtschaftlichkeit zu bestimmen, können Kosten-Nutzen-Modelle angewendet werden, die allgemein [DBW+08] oder szenariospezifisch [SBCM07] sind. Alternativ können Kosten und Nutzen durch Simulationen analysiert werden. So wird z.B. in [TF07] eine Simulationsstudie vorgestellt, die zeigt bei welchen Parametern (Hardware- und Personalkosten, Preise, Absatz, usw.) sich der Einsatz von RFID im Einzelhandel rentiert.

Datenschutz: Der Einsatz von RFID im Einzelhandel steht und fällt mit der Akzeptanz der Kunden. So konnte eine Kundeninitiative im Jahre 2003 zum Beispiel verhindern, dass ein großes Bekleidungsunternehmen seine Artikel mit RFID ausstattete[1]. Es herrschen Bedenken, dass RFID das Ausspähen der Kunden ermöglichen könnte. Dies soll durch folgendes Beispiel verdeutlicht werden: Ein Kunde kauft einen Artikel, der mit einem Funketikett ausgestattet ist, zum Beispiel Schuhe, bei denen das Funketikett in die Sohle eingearbeitet ist. Dadurch ist dieses nicht sichtbar und lässt sich nicht entfernen. Bezahlt der Kunde die Schuhe mit einer Kredit- oder EC-Karte, könnte der Verkäufer die eindeutige Identifikationsnummer des Artikels mit den Daten der Karte verknüpfen und speichern. Wird der Artikel zu einem anderen Zeitpunkt an einem anderen Ort erfasst, so könnte man anhand der gespeicherten Daten die Person identifizieren.

Es existieren verschiedene Ansätze, um diesem Problem entgegenzuwirken. Der RFID-Standard sieht einen Deaktivierungsbefehl (*Kill*) vor, der durch ein Passwort das Funketikett

[1] http://www.boycottbenetton.com/

permanent deaktiviert [EPC05]. Es könnten z.B. an der Kasse alle Funketiketten deaktiviert werden. Allerdings können dadurch keine RFID-gestützte Prozesse nach dem Verkauf stattfinden, wie Rückgabe von Pfand, Garantiefälle usw. Ein anderer Ansatz sieht vor, dass durch Abreißen eines Streifens auf dem Funketikett die Antenne von dem Chip getrennt wird [KM05]. Dadurch hat der Kunde eine visuelle Bestätigung, dass das Funketikett deaktiviert ist. Allerdings bleiben die Daten auf dem Etikett erhalten, und können bei Bedarf für spätere Prozesse verwendet werden. Ein weiterer Ansatz ist das RSA-Blockertag [JRS03]. Dies ist eine spezielle Hardware, die als Funketikett fungiert, jedoch absichtlich falsche Befehle an den RFID-Leser schickt. Dadurch verhindert das RSA-Blockertag die Lesungen von Funketiketten in seiner Umgebung. So wird zwar die Privatsphäre der Kunden geschützt, allerdings kann das RSA-Blockertag andere RFID-Leser stören, die zum Beispiel in einem Smart-Shelf eingebaut sind und lediglich den Inhalt des Regals erfassen sollen. Eine andere Alternative ist es, Funketiketten mit kryptographischen Fähigkeiten auszustatten, so dass die Identifikationsnummer eines Funketiketts erst nach erfolgreicher Authentifizierung übertragen wird [DLK06, RCT06]. Allerdings sind solche Funketiketten teuer und es ist daher in naher Zukunft kein flächendeckender Einsatz zu erwarten. Des Weiteren verwenden solche Funketiketten oft schwache kryptographische Verfahren [BGS+05].

Kapitel 3

Schätzung der Anzahl von Artikeln

In diesem Kapitel wird ein Verfahren namens TagMark [WBB08] vorgestellt, das die Anzahl von Artikeln in der Umgebung eines RFID-Lesers schätzt. Ein solches Verfahren ist für die Integration von RFID-Daten in Unternehmenssoftware notwendig, da aufgrund von Funklöchern nicht alle Artikel in der Umgebung eines RFID-Lesers erfasst werden können. Allerdings sind solche Schätzungen schwierig zu erhalten da die meisten Schätzverfahren Annahmen treffen, die in typischen RFID-Szenarien nicht gelten, wie zum Beispiel statische Populationen, reine Zufallsstichproben oder Stichproben mit benutzerdefinierten Größen.

TagMark erweitert die so genannte Rückfangmethode aus der Biologie [Seb82], bei der die Anzahl von Individuen einer Population anhand von verschiedenen Stichproben geschätzt wird. Mit TagMark kann die Genauigkeit der Schätzung mittels relativer Konfidenzintervalle definiert werden und es existieren obere Schranken für die Größe der benötigten Stichproben. Das Verfahren kann als Datenbankerweiterung implementiert werden, so dass es nahtlos in existierende Unternehmenssoftware integriert werden kann.

3.1 Die Rückfangmethode aus der Biologie

Die Rückfangmethode (engl. *Mark-Recapture Method*) wird in der Biologie verwendet, um die Anzahl von Individuen in einer Population mit unbekannter Größe N zu schätzen. Die Anzahl von geschätzten Individuen ist $\hat{N}$. Zunächst wird der Population eine Stichprobe der Größe n_1 entnommen. Die Individuen dieser Stichprobe werden markiert, damit sie zu einem späteren Zeitpunkt wiedererkannt werden können, und der Population zurückgegeben. Es wird gewartet, bis sich die Individuen der ersten Stichprobe unter den Individuen der restlichen Population vermischen. Danach wird eine neue Stichprobe der Größe n_2 entnommen. Aus dieser Stichprobe wurden m_2 Individuen bereits in der ersten Stichprobe erfasst und markiert (siehe Tabelle 3.1 für eine Liste der Symbole). Vorausgesetzt der Anteil von markierten Individuen in der zweiten Stichprobe ist eine angemessene Schätzung des Anteils von markierten Individuen in der unbekannten Population, kann $\hat{N}$ wie folgt geschätzt werden [Seb82]:

Tabelle 3.1: Häufig verwendete Symbole

Symbol	*Beschreibung*
A	Relative Abweichung einer Schätzung in Prozent
α	Wahrscheinlichkeit, dass die Abweichung A nicht eingehalten wird
D	Parameter für die Bestimmung der optimalen Stichprobengrößen n_1 und n_2
γ	RFID-Leserate
m_2	Anzahl der markierten Individuen in der zweiten Stichprobe
n_1, n_2	Größe der ersten und zweiten Stichprobe
N	Größe der zu schätzenden Population
$\hat{N}$	Schätzung der Größe der Population
R	Anzahl der nachgefüllten Artikel
w	Größe des Zeitfensters

$$\frac{m_2}{n_2} = \frac{n_1}{\hat{N}} \quad oder \quad \hat{N} = \frac{n_1 n_2}{m_2} \tag{3.1}$$

$\hat{N}$ ist eine angemessene Schätzung von N, falls folgende Annahmen erfüllt sind:

1. Die Population ist geschlossen, d.h. es kommen keine neuen Individuen dazu und keine Individuen verlassen die Population.
2. Alle Individuen können mit derselben Wahrscheinlichkeit Teil der ersten und der zweiten Stichprobe sein.
3. Alle Individuen, die Teil der ersten Stichprobe waren, können in der zweiten Stichprobe identifiziert werden [Seb82].

Folglich benötigt die Rückfangmethode zwei reine Zufallsstichproben. Die Genauigkeit der Schätzung ist abhängig von der Größe beider Stichproben. Ein relatives Konfidenzintervall wird durch die Variablen A und α wie folgt definiert: $\left|\frac{\hat{N}-N}{N}\right|$ stellt die relative Abweichung dar und $(1-\alpha)$ ist die Wahrscheinlichkeit, dass die geschätzte Größe $\hat{N}$ der Population innerhalb der relativen Abweichung A liegt, also $\left|\frac{\hat{N}-N}{N}\right| < A$ [RR64]. Dies wird durch folgende Gleichung beschrieben, wobei $P(\cdot)$ die Wahrscheinlich eines Ereignisses angibt:

$$1 - \alpha \leq P\left(-A < \frac{\hat{N} - N}{N} < A\right) \tag{3.2}$$

Die obige Gleichung kann Anhand der kumulativen Verteilungsfunktion der Normalverteilung umgeformt werden. Die Verteilungsfunktion $\phi(z)$ berechnet die Wahrscheinlichkeit, dass der Wert einer Zufallsvariable größer als z ist. Die notwendige Größe der Stichproben n_1, n_2 hängt somit wie folgt von der Größe N der Population und von dem relativen Konfidenzintervall ab [RR64]:

$$D = \frac{n_1 n_2 (N-1)}{(N-n_1)(N-n_2)} \tag{3.3}$$

$$1-\alpha = \phi\left(\frac{A\sqrt{D}}{1-A}\right) - \phi\left(\frac{-A\sqrt{D}}{1+A}\right) \tag{3.4}$$

Die Hilfsvariable D wurde eingeführt, um die Lesbarkeit der Gleichung zu erleichtern. Es existiert keine geschlossene Formel für die kumulative Verteilungsfunktion. Daher wird D in Gleichung 3.4 durch numerische Verfahren gewonnen. Die notwendige Größe der Stichproben wird berechnet, indem Gleichung 3.3 nach n_1 oder n_2 aufgelöst und der berechnete Wert von D eingesetzt wird.

3.2 Probleme der Rückfangmethode bei RFID-Szenarien

In diesem Abschnitt werden die Parameter der Rückfangmethode für RFID-Szenarien umdefiniert. Prinzipiell ist die Methode für die Schätzung der Anzahl von Artikeln gut geeignet. Allerdings zeigt eine Analyse der Charakteristika von RFID und der Anforderungen des Szenarios, dass die Rückfangmethode angepasst werden muss.

Es sei N die Anzahl von Artikeln, die sich tatsächlich in einem Regal befinden. Die erste Stichprobe enthält n_1 Artikel, die durch den RFID-Leser erfasst wurden. Die zweite Stichprobe enthält n_2 Artikel, die innerhalb eines gewissen Zeitfensters (engl. *Sliding Window*) verkauft wurden. Von allen verkauften Artikeln wurden m_2 Artikel bereits von einem RFID-Leser in der Filiale erfasst. Die ausreichende Größe der Stichproben kann anhand einer groben Schätzung von N durch die Gleichungen 3.4 und 3.3 berechnet werden. Da eine Unterschätzung von N zu kleinen Stichproben führt, muss N überschätzt werden, um das relative Konfidenzintervall einzuhalten. In dem vorgestellten Szenario bietet die Inventurliste eine solche Schätzung, da sie alle Artikel einer Filiale, die noch nicht verkauft wurden, enthält. Da die Inventurliste auch beschädigte und entwendete Artikel enthalten kann, beinhaltet sie immer gleichviele oder mehr Artikel als N.

Betrachtet man das in Kapitel 2 eingeführte RFID-Szenario für den Einzelhandel, so scheint die Rückfangmethode auf den ersten Blick ein geeignetes Schätzverfahren zu sein. Allerdings müssen weitere Charakteristika von RFID berücksichtigt werden, die eine Anpassung und Erweiterung der Rückfangmethode erfordern. Hier werden die relevanten Charakteristika aus Abschnitt 2.3 in den Kontext der Rückfangmethode gestellt:

C2: Kontinuierliche Datenströme: Im Einzelhandel werden kontinuierliche Datenströme von RFID-Lesungen erzeugt, da Kunden Artikel kaufen und Angestellte leere Regale auffüllen. So können einzelne Stichproben zu gering für den Einsatz der Rückfangmethode sein und das Zusammenführen aller vergangenen Stichproben kann zu einem veralteten Ergebnis führen.

C3: Unvorhersehbare Menge erfasster Artikel: Im Allgemeinen stellen Mengen von erfassten Artikeln keine reine Zufallsstichproben dar, da RFID-Lesungen von physikalischen Phänomenen abhängen, die nicht beobachtet werden können [FL04, GS105]. Und die RFID-Lesungen an der Kasse hängen von dem Kundenkaufverhalten ab, da Kunden beispielsweise die Artikel in der vordersten Reihe eines Regals bevorzugen.

C4: Offene Populationen: Im Einzelhandel werden Regale geleert und kontinuierlich aufgefüllt. Für die Rückfangmethode würde dies bedeuten, dass zwischen zwei Stichproben ein großer Teil oder sogar die ganze Population ersetzt wird. Dies ist anders als bei bereits untersuchten Phänomenen, wie zum Beispiel das langsame und ständige Stattfinden von Sterbefällen und Geburten. Somit würde die Nachfüllung eines leeren Regals dem plötzlichen Tod und der sofortigen Neugeburt aller Individuen der Population gleichen.

C5: Große Datenmengen: Durch die große Anzahl von Artikeln im Einzelhandel bringt der Einsatz von RFID große Datenmengen mit sich. Die resultierenden großen Stichproben und Datenmengen stellen eine Herausforderung an die Laufzeit von Schätzverfahren dar.

Zusätzlich müssen Anforderungen der Bestandskontrolle in der Filiale berücksichtigt werden. Folgende Anforderungen wurden zusammen mit einem großen deutschen Einzelhändler ermittelt [BDF+07]. Sie werden mit den Eigenschaften von RFID verglichen, um die nötigen Anpassungen und Erweiterungen der Rückfangmethode zu ermitteln:

Konfidenzintervalle: Die Bestandskontrolle eines Einzelhändlers benötigt verschiedene Konfidenzintervalle. So muss der Bestand der Jahresinventur beispielsweise genauer sein, als der Bestand der regelmäßigen Inventur für die Nachfüllung der Regale. Dies ist aber problematisch: Zum einen benötigt die Rückfangmethode zwei reine Zufallsstichproben, jedoch ist dies bei RFID-Daten nicht gegeben (Charakteristik C3). Zum anderen produzieren RFID-Leser Datenströme von Artikeln (Charakteristik C2), die zu verschiedenen Zeitpunkten erfasst werden. Allerdings kann die Rückfangmethode nicht bestimmen, zu welchem Zeitpunkt die Schätzung gültig ist.

Warenfluss: Der Einsatz von Funketiketten im Einzelhandel hat kontinuierliche Datenströme (Charakteristik C2) und offene Populationen (Charakteristik C4) zur Folge. Allerdings benötigt die Rückfangmethode statische Populationen, die sich zwischen zwei Stichproben nicht verändern.

Skalierbarkeit: Da der Einsatz von Funketiketten im Einzelhandel große Datenmengen (Charakteristik C5) erzeugt, sind die zeitliche und räumliche Komplexität des eingesetzten

Schätzverfahrens wichtig. Dies stellt ein Problem bei der Rückfangmethode dar, da für eine gegebene Genauigkeit die Größe der Stichproben n_1 und n_2 linear von der Größe der Population N abhängig ist. Folglich benötigt die Schätzung einer sehr großen Population sehr große Stichproben.

Robustheit: RFID-gestützte Geschäftsprozesse müssen robust gegen Angriffe wie Diebstahl durch Angestellte sein. Beispielsweise könnte ein Dieb Artikel entwenden, die durch den RFID-Leser momentan nicht erfasst werden, und somit die Eigenschaft von RFID ausnutzen, dass die Mengen von erfassten Artikeln unvorhersehbar (Charakteristik C3) sind. Dies unterscheidet sich von bekannten Problemen der Rückfangmethode, wie zum Beispiel das Sterben von Individuen, da Diebe Schwachstellen des Prozesses angreifen können.

3.3 Verwandte Arbeiten

Zuerst wird auf verwandte Arbeiten eingegangen, bevor TagMark ausführlicher beschrieben wird. TagMark erweitert die Rückfangmethode aus der Biologie, welche die Anzahl von Individuen in einer Population anhand von mehreren Stichproben schätzt. Auf den ersten Blick scheinen allerdings andere Schätzverfahren ebenfalls anwendbar zu sein, um die Anzahl von Funketiketten bzw. Artikeln zu schätzen. Beispiele sind andere Erweiterungen der Rückfangmethode, parametrische Schätzer, statistische Datenbanken, Optimierungen des RFID-Kommunikations-Protokolls und Datenbereinigung (engl. *Data Cleansing*). Allerdings wird in diesem Abschnitt gezeigt, dass Verfahren aus diesen Gebieten nicht für die Schätzung der Anzahl von Artikeln im Einzelhandel geeignet sind.

Erweiterungen der Rückfangmethode: Wie in Abschnitt 3.2 dargestellt, weisen RFID-Daten besondere Charakteristika auf, die eine Anpassung der Rückfangmethode erfordern. Es wurden allerdings viele Verfahren vorgeschlagen, welche die Fähigkeiten der Rückfangmethode erweitern. Das Verfahren von Jolly-Seber [Jol65, Seb65] schätzt die Anzahl von Individuen in offenen Populationen (Charakteristika C4) und kann prinzipiell auch mit kontinuierlichen Datenströmen (Charakteristika C2) umgehen. Allerdings benötigt dieses Verfahren Schätzungen von allen Parametern, welche die Stichproben beeinflussen, wie z.B. Schwund, Kaufverhalten der Kunden und Anteil von fehlplatzierten Artikeln. Es ist sehr zeitaufwändig, Daten zu sammeln, um für jeden Produkttyp solche Schätzungen zu gewinnen. Daher stellt dieses Verfahren keine Alternative zur Rückfangmethode dar. Jackknife-Verfahren [BO78] kombinieren viele Stichproben, um zuverlässige Schätzungen zu berechnen. Sie könnten prinzipiell mit kontinuierlichen Datenströmen (Charakteristika C2) umgehen. Allerdings basieren solche Verfahren auf geschlossene Populationen und alle Stichproben müssen stochastisch unabhängig voneinander sein. RFID-Daten stellen allerdings offene Populationen dar (Charakteristika C4) und aufeinanderfolgende RFID-Lesungen bilden keine Zufallsstichproben (Charakteristik C3).

Parametrische Schätzer: Diese Art von Schätzverfahren verwendet historische Daten, um Schätzungen von Parametern zu berechnen. Beispielsweise schätzen Charikar et

al. [CCMN00] die Anzahl von unterschiedlichen Werten eines Attributs in einer Datenbank, um Abfragen zu optimieren. Die Schätzung basiert auf einer Menge von Zufallsstichproben aus der entsprechenden Tabelle, die in Abhängigkeit einer vordefinierten Wahrscheinlichkeitsverteilung gezogen werden. Dieses Verfahren erfordert Zufallsstichproben. Allerdings stellen erfasste Artikel in RFID-Szenarien keine Zufallsstichproben dar (Charakteristik C3). Aus diesem Grund ist es schwierig, dieses Schätzverfahren auf RFID-Szenarien anzuwenden. Andere parametrische Schätzer, wie z.B. [CDS04, HO91], weisen ähnliche Probleme auf.

Statistische Datenbanken: In statistischen Datenbanken ist jedes Tupel mit der Wahrscheinlichkeit gekennzeichnet, das die Daten des Tupels richtig sind. Statistische Datenbanken könnten beispielsweise die Wahrscheinlichkeit speichern, dass sich ein Artikel im Regal befindet, und die Anzahl von Artikeln mit einer gewissen Genauigkeit schätzen. Ein prominentes Beispiel solcher Datenbanken ist das Projekt Trio [BSHW06]. In diesem Projekt werden Erweiterungen des relationalen Datenbankmodells und von SQL vorgeschlagen, um die Ungewissheit und Abstammung (engl. *Uncertainty* und *Data Lineage*) von Daten zu unterstützen. Das größte Problem bei der Anwendung von Trio und ähnlichen Verfahren [CKP03, DS04, SBHW06] auf RFID-Szenarien ist es, die entsprechenden Wahrscheinlichkeiten zu gewinnen. Bei RFID-Szenarien im Einzelhandel ist die RFID-Leserate im Voraus nicht bekannt, und sie ändert sich sobald sich die Umgebung ändert. Dadurch wären die Wahrscheinlichkeiten in der Datenbank nicht mehr aktuell.

RFID-Kommunikations-Protokoll: Vogt [Vog02] schätzt die Anzahl von Funketiketten in der Umgebung eines RFID-Lesers anhand der Anzahl von Kollisionen während der drahtlosen Kommunikation. Diese Information wird dann verwendet, um die Kommunikation mit den restlichen Funketiketten zu optimieren. Allerdings werden bei diesem Verfahren nur die Funketiketten geschätzt, die mit dem RFID-Leser kommunizieren können. Daher beinhalten die Schätzungen keine Artikel, die sich in Funklöchern befinden. Des Weiteren wird die Anzahl aller Funketiketten unabhängig von dem Produkttyp geschätzt. Dies sind systematische Probleme, die auch für andere Optimierungen des RFID-Kommunikations-Protokolls [KN06, KZBD05] gelten.

Probabilistische Filter: Es existieren verschiedene Verfahren, um RFID-Daten anhand von probabilistischen Filtern zu bereinigen. In [KBS06, XYC+08] werden beispielsweise feste Nebenbedingungen und statistische Daten verwendet, um die Anzahl von Artikeln zu schätzen. Beide Verfahren geben eine Quantifizierung aller möglichen Ergebnisse zurück, d.h. jedes Ergebnis ist mit einer Wahrscheinlichkeit versehen. Der Einsatz solcher Methoden in RFID-Szenarien im Einzelhandel ist aus verschiedenen Gründen schwierig: Es ist unpraktisch, die nötigen statistischen Daten zu erheben, da sich der Ort von Regalen sowie Ort, Typ und Anzahl von Artikeln oft ändern. Des Weiteren ist es nicht möglich, Konfidenzintervalle zu bestimmen. Auch eine Integration in Unternehmenssoftware ist schwierig, da heutige Systeme in der Regel keine Wahrscheinlichkeiten verarbeiten können. Andere Verfahren aus dieser Kategorie erfüllen nicht die Anforderungen aus dem Einzelhandel. [JGF06] setzt beispielsweise voraus, dass alle Stichproben Zufallsstichproben darstellen. An dieser Stelle sei angemerkt, dass solche Methoden von TagMark profitieren könnten. TagMark könnte neue statistische Daten liefern, wie zum Beispiel die Anzahl geschätzter Artikel und die dazugehörigen Wahrscheinlichkeiten.

Viele Funketiketten pro Artikel: [BR07] schlägt vor, mehrere Funketiketten in verschiedenen Ausrichtungen auf jedem Artikel anzubringen. In solchen Szenarien ist die Wahrscheinlichkeit sehr hoch, dass zumindest ein Funketikett nicht in einem Funkloch liegt. Trotzdem liegt die Leserate unter 100%, und, wenn sogar vier Funketiketten auf einem Artikel angebracht sind, liegt die Leserate bei gewissen Aufbauten unter 75%. Des Weiteren sind mehrere Funketiketten aus betriebswirtschaftlicher Sicht nicht praktikabel, da, wie bereits erwähnt, der Preis von einem Funketikett zu hoch für einen Einsatz bei niedrigpreisigen Produkten sein kann.

3.4 TagMark: Eine Rückfangmethode für RFID-Szenarien

In diesem Abschnitt wird TagMark [WBB08] vorgestellt. TagMark ist ein zuverlässiges und skalierbares Schätzverfahren für die Anzahl von Artikel in RFID-Szenarien. Es vereint alle RFID-Lesungen innerhalb eines gewissen Zeitfensters zu einer Stichprobe. So wird beispielsweise jeder Artikel, der innerhalb des Zeitfensters erfasst wird, zu der ersten Stichprobe gezählt.

Die Annahmen von TagMark sind anders als die Annahmen der Rückfangmethode. Kunden kaufen Artikel in einer systematischen Art und Weise und die Anzahl von erfassten Artikeln hängt von unvorhersehbaren physikalischen Phänomenen ab [FL04, GS105]. Somit kann nicht sichergestellt werden, dass es sich um reine Zufallsstichproben handelt. Allerdings liefert TagMark korrekte Ergebnisse, falls folgende Annahmen gelten:

1. Alle Artikel können mit derselben Wahrscheinlichkeit in der ersten Stichprobe erfasst werden.

2. Es existiert keine Korrelation zwischen Funklöchern des RFID-Lesers und dem Kaufverhalten der Kunden.

Annahme 1 erfordert, dass alle Artikel mit derselben Wahrscheinlichkeit von dem RFID-Leser erfasst werden können. Artikel mit defekten Funketiketten oder die *absichtlich* in Funklöchern platziert werden, sind nicht Teil der Population, und somit nicht Teil der Schätzung. Aufgrund des robusten Herstellungsverfahrens und der strengen Qualitätskontrolle haben alle Funketiketten im vorgestellten Szenario dieselben Kommunikationsfähigkeiten.

Annahme 2 besagt, dass das Kaufverhalten der Kunden nicht mit physikalischen Phänomenen korreliert, welche die RFID-Leserate beeinflussen. Ohne diese Annahme wäre das Verhältnis zwischen erfassten und nichterfassten Artikeln in der ersten Stichprobe und den Artikeln in der zweiten Stichprobe nicht repräsentativ für die Population. Die Annahme ist realistisch, da das Bewegen von Artikeln das RFID-Lesefeld in einer unvorhersehbaren Art und Weise verändert, und Kunden gegebenenfalls vorhandene Funklöcher des RFID-Lesers nicht kennen.

TagMark benötigt vier Datenstrukturen:

- *salesHistory:* eine Liste der vergangenen Abverkäufe inklusive Zeitstempel
- *readHistory:* eine Liste der von dem RFID-Leser erfassten Artikel inklusive Zeitstempel
- *replenishHistory:* eine Liste mit Uhrzeit und Anzahl von nachgefüllten Artikeln
- *inventory:* eine Liste mit allen Artikeln einer Filiale, die noch nicht verkauft wurden

```
1:  input Relatives Konfidenzintervall (A, α), readHistory, salesHistory, replenishHistory,
    inventory
2:  for (w = 1 to |readHistory|) do // Bestimme Größe w des Zeitfensters
3:      int n_1 = getCount(readHistory, w)
4:      salesHistory' = removeOutdatedItems(salesHistory, w)
5:      int n_2 = getCount(salesHistory', w)
6:      if n_2 ≥ calculateSampleSize(n_1, A, α, |inventory|) then
        // Prüfe ob Stichproben ausreichend groß sind
7:          int m_2 = getCount(readHistory ∩ salesHistory', w)
8:          int N̂ = calculateEstimate(n_1, n_2, m_2)
9:          for all R in getData(replenishHistory, w) do
            // Korrigiere Schätzung bei Regalnachfüllung
10:             N̂ = N̂ + calculateCorrection(R, w)
11:         end for
12:         break
13:     end if
14: end for
15: output max(N̂, n_1) // Schätzung der Populationsgröße N̂
```

Algorithmus 3.1: TagMark-Algorithmus

TagMark wird in Algorithmus 3.1 beschrieben. Als erstes bestimmt TagMark die notwendige Größe der Stichproben. Die Stichproben hängen von der RFID-Leserate und von dem Kundenkaufverhalten ab und können nicht frei bestimmt werden. Aus diesem Grund beginnt TagMark mit den neuesten Daten sowie der kleinstmöglichen Fenstergröße (Zeile 2) und erhöht diese bis sowohl *readHistory* als auch *salesHistory* genug Artikel enthalten, um das spezifizierte relative Konfidenzintervall einzuhalten (Zeile 6). *getCount(list, w)* ermittelt die Anzahl von Artikeln aus *list*, die innerhalb des Zeitfensters w liegen. *removeOutdatedItems(list, w)* entfernt Artikel aus *list*, die außerhalb des Zeitfensters w zum letzten Mal erfasst wurden. Solche Artikel werden entfernt, da sie sonst die Schätzung verfälschen würden, vgl. Abschnitt 3.4.1. *calculateSampleSize(n_1, A, α, |inventory|)* berechnet die benötigte Größe der zweiten Stichprobe anhand von Gleichung 3.3. Nachdem die Fenstergröße bestimmt wurde, wird die Anzahl vorhandener Artikel $\hat{N}$ geschätzt (Zeile 8). Die Funktion *calculateEstimate(n_1, n_2, m_2)* verwendet Gleichung 3.1, um die Schätzung zu berechnen. Danach prüft TagMark, ob innerhalb

des Zeitfensters die Regale nachgefüllt wurden (Zeile 9). Die Funktion *calculateCorrection(R, w)* wird in Abschnitt 3.4.2 vorgestellt. Nach einer Plausibilitäts-Prüfung wird die endgültige Schätzung zurückgegeben (Zeile 15).

3.4.1 Umgang mit kontinuierlichen Datenströmen

Das in TagMark verwendete Zeitfenster löst das Problem der variierenden RFID-Leseraten bzw. vermeidet die Erzeugung von kleinen Stichproben, welche das spezifizierte relative Konfidenzintervall nicht erfüllen. In TagMark wird die Anzahl von Artikeln geschätzt, die sich am Anfang des Zeitfensters im Regal befanden, d.h. im ältesten Zeitpunkt, an dem Daten innerhalb des Zeitfensters erfasst wurden. Dies kann zu Problemen bei Artikeln führen, die selten verkauft werden, beispielsweise lediglich ein Mal täglich. In Abhängigkeit der Anzahl solcher Artikel in einer Filiale und der RFID-Leserate könnte es mehrere Tage dauern, bis die notwendige Stichprobengröße erreicht wird. In solchen Fällen wäre die Schätzung von TagMark mehrere Tage alt. Dies stellt zwar ein Problem dar, schränkt aber die Nutzbarkeit von TagMark in praktischen Szenarien in der Regel nicht ein. Für eine Inventur ist eine Verzögerung von wenigen Tagen annehmbar, und Artikel, die selten verkauft werden, weisen das geringste Einsparungspotential bei einer Optimierung der Regalverfügbarkeit mittels RFID auf.

Ein weiteres Problem entsteht, wenn ein Artikel verkauft wird, der außerhalb des Zeitfensters aus einem Regal entnommen wurde. Dies kann beispielsweise geschehen, wenn der Einkauf eines Kunden sehr viel Zeit beansprucht. Solche Artikel werden für die zweite Stichprobe nicht berücksichtigt, da sie das Ergebnis verfälschen würden, und da die Schätzung von TagMark zu einem Zeitpunkt gültig ist, an dem solche Artikel bereits aus den Regalen entfernt wurden.

3.4.2 Schätzungen bei offenen Populationen

Im Einzelhandel werden Regale in unregelmäßigen Abständen aufgefüllt, nämlich wenn die Regale leer oder fast leer sind. Der Zeitpunkt und die Anzahl von nachgefüllten Artikeln werden erfasst. Die Nachfüllung bringt zwei Probleme mit sich:

1. Es verletzt die Annahme der Rückfangmethode, dass alle Artikel mit derselben Wahrscheinlichkeit Teil beider Stichproben sein können. Nachgefüllte Artikel werden sofort im Regal erfasst, aber es dauert, bis sie verkauft werden. Erweiterungen der Rückfangmethode sind allerdings auf kleine und vorhersehbare Änderungen der Population ausgelegt (vgl. Charakteristik C4 in Abschnitt 3.2).

2. Nachdem ein Regal nachgefüllt wurde, kann sich die RFID-Leserate ändern. Dies geschieht, weil Objekte in der Umgebung des RFID-Lesers elektromagnetische Wellen absorbieren oder reflektieren können. Somit wäre der Anteil an erfassten Artikeln im Regal anders als der Anteil an markierten Artikeln an der Kasse, d.h. die Stichproben sind nicht mehr repräsentativ für die Population.

Somit müssten für die Rückfangmethode bei Nachfüllung eines Regals neue Stichproben gesammelt werden. Da es lange dauern kann, bis Stichproben mit genügender Größe an der Kasse gesammelt werden, würde die Rückfangmethode für gewisse Zeiträume keine Schätzung liefern können.

TagMark umgeht dieses Problem, indem ein Korrekturparameter wie folgt angewendet wird: Es seien N und N' die Größe der Population vor und sofort nach der Nachfüllung des Regals und R die Anzahl von nachgefüllten Artikeln. Nach der Nachfüllung werden die Stichproben, die innerhalb des Zeitfensters gesammelt wurden, aus zwei Teilen bestehen. Ein Teil der vor und ein Teil der nach der Nachfüllung gesammelt wurde. β wird definiert als die Länge des ersten Teils geteilt durch die komplette Länge des Zeitfensters. So ist beispielsweise sofort nach der Nachfüllung $\beta = 1$.

Als nächstes wird die korrigierte Schätzung $\hat{N}_{corrected}$ für $\beta > 0$ hergeleitet. Es werden zwei Schätzungen $\hat{N}$ und $\hat{N}'$ berechnet, vor und sofort nach der Nachfüllung, anhand der Zeilen 2 bis 8 in Algorithmus 3.1. Insbesondere schätzt TagMark mit $\hat{N} \approx N$ und mit $\hat{N}' \approx N + R$. Der absolute Fehler c wird als $c = N' - \hat{N}'$ definiert und kann wie folgt geschätzt werden:

$$c = (\hat{N} + R) - \hat{N}' \tag{3.5}$$

Man beachte, dass $\hat{N}'$ verschieden von $\hat{N}_{corrected}$ ist, da $\hat{N}'$ voraussetzt, dass die nachgefüllten Artikel durch die Stichproben richtig vertreten werden. Dies ist aber nicht der Fall, da die Stichprobe eine Mischung aus Daten vor und nach der Nachfüllung enthält. Allerdings wird über die Zeit aufgrund neuer Stichproben am Regal und an der Kasse der Anteil β an alten Daten im Zeitfenster geringer. Diese Tatsachen werden genutzt, um die korrigierte Schätzung $\hat{N}_{corrected}$ wie folgt zu berechnen:

$$\hat{N}_{corrected} = \hat{N} + (\beta \cdot c - R) \tag{3.6}$$

TagMark korrigiert die Schätzung anhand einer anderen Schätzung, die eine größere Population voraussetzt, vgl. Gleichung 3.5. Aus diesem Grund beeinflusst die Korrektur das relative Konfidenzintervall der Schätzung. In Gleichung 3.2 wurde das relative Konfidenzintervall über die Variablen A und α definiert. Da diese voneinander abhängig sind, reicht es den Einfluss auf eine der Variablen zu untersuchen. Im Folgenden wird auf die relative Abweichung A fokussiert.

Lemma 3.1. *Werden innerhalb des Zeitfensters Artikel nachgefüllt, so hat die Schätzung $\hat{N}_{corrected}$ eine relative Abweichung von $A + \frac{AR}{N}$.*

Beweis: Der Beweis basiert auf die durch A und α bestimmte Genauigkeit der Rückfangmethode, wie in Abschnitt 3.1 definiert. Sofort nach der Nachfüllung ändert sich die Grundpopulation von N auf $N' = N + R$. Die neue Grundpopulation wird wie folgt in Gleichung 3.2 berücksichtigt: $\hat{N}$ wird durch $\hat{N}_{corrected} + R$ ersetzt, und N durch $N + R$.

$$1-\alpha \leq P\left(-A < \frac{\hat{N}_{corrected} + R - (N+R)}{N+R} < A\right) \tag{3.7}$$

Danach wird die Ungleichung zwischen den Klammern von $P(\cdot)$ mit $\frac{N+R}{N}$ multipliziert, um das relative Konfidenzintervall zu erhalten:

$$1-\alpha \leq P\left(-\left(A+\frac{AR}{N}\right) < \frac{\hat{N}_{corrected} - N}{N} < A+\frac{AR}{N}\right) \tag{3.8}$$

□

3.4.3 Unabhängige statt zufällige Strichproben

RFID-Szenarien können nicht zwei reine Zufallsstichproben garantieren, vgl. Abschnitt 3.2. In diesem Abschnitt wird gezeigt, warum *unabhängig* voneinander gesammelte Stichproben für TagMark ausreichend sind. Zusätzlich wird beschrieben, warum RFID-Szenarien im Einzelhandel solche Stichproben erzeugen.

Lemma 3.2. *Die Schätzung $\hat{N}$ ist korrekt, falls beide Stichproben unabhängig voneinander gesammelt werden.*

Beweis: Zwei Ereignisse X und Y sind unabhängig, falls:

$$P(Y|X) = \frac{P(Y)P(X)}{P(X)} = P(Y) \tag{3.9}$$

$$P(Y|\neg X) = \frac{P(Y)P(\neg X)}{P(\neg X)} = P(Y) \tag{3.10}$$

Die Stichproben der Rückfangmethode resultieren aus zwei Folgen von Bernoulli-Versuchen [Seb82]. Dabei müssen zwei Ereignisse berücksichtigt werden: U bedeutet „ist Teil der ersten Stichprobe“ und V bedeutet „ist Teil der zweiten Stichprobe“. Nach [Seb82] ist die bedingte Wahrscheinlichkeitsverteilung $f(m_2|n_1,n_2)$ von m_2, bei gegebenem n_1 und n_2, binomial, d.h.

$$f(m_2|n_1,n_2) = \binom{n_2}{m_2} p^{m_2}(1-p)^{n_2-m_2} \tag{3.11}$$

wobei

$$p = \frac{n_1}{(N-n_1)k+n_1} \tag{3.12}$$

$$k = \frac{(E[P(V|\neg U)] - E[P(U)\cdot P(V|\neg U)])\cdot E[P(U)]}{E[P(U)\cdot P(V|U)]\cdot(1-E[P(U)])} \tag{3.13}$$

p gibt die Wahrscheinlichkeit an, dass ein Artikel der ersten Stichprobe auch Teil der zweiten Stichprobe ist. Der Erwartungswert wird durch $E[\cdot]$ symbolisiert. k ist der Erwartungswert für den Anteil von Artikel der Population, die Teil der ersten Stichprobe sind (Zähler), geteilt durch den Anteil von markierten Artikeln in der zweiten Stichprobe (Nenner). In [Seb82] wird formal bewiesen, dass die Rückfangmethode eine korrekte Schätzung $\hat{N}$ liefert, genau dann, wenn $k = 1$, d.h. der Anteil an markierten Artikel in der zweiten Stichprobe n_1/N ist. Um Lemma 3.2 zu beweisen, muss jetzt gezeigt werden, dass unabhängig von einander gesammelte Stichproben zu $k = 1$ führen. Anhand der Gleichungen 3.9 und 3.10 kann Gleichung 3.12 wie folgt umgeformt werden:

$$\begin{aligned} k &= \frac{(E[P(V|\neg U)] - E[P(U) \cdot P(V|\neg U)]) \cdot E[P(U)]}{E[P(U) \cdot P(V|U)] \cdot (1 - E[P(U)])} \\ &= \frac{(E[P(V)] - E[P(V)] \cdot E[P(U)]) \cdot E[P(U)]}{E[P(U)] \cdot E[P(V)] \cdot (1 - E[P(U)])} \\ &= \frac{E[P(V)] \cdot E[P(U)] \cdot (1 - E[P(U)])}{E[P(V)] \cdot E[P(U)] \cdot (1 - E[P(U)])} \\ &= 1 \end{aligned} \tag{3.14}$$

□

Es wurde gezeigt, dass TagMark auch ohne reine Zufallsstichproben korrekte Schätzungen liefert, solange die Stichproben unabhängig von einander gesammelt werden, also keine Korrelation zwischen den Funklöchern und den gekauften Artikeln existiert. Wie bereits diskutiert schränkt dies den Einsatz von TagMark nicht ein, da sich Funklöcher oft ändern und von physikalischen Phänomenen abhängen, die die Kunden nicht beobachten können. Dieser Aspekt wird bei den in Abschnitt 3.5 beschriebenen Experimenten weiter analysiert.

3.4.4 Größe der Stichproben in RFID-Szenarien

Bei der Rückfangmethode hängt die Größe der benötigten Stichproben von der Größe der Population N ab. Man betrachte beispielsweise die Stichprobe n_2: Gleichung 3.4 kann mit numerischen Methoden gelöst werden, und Gleichung 3.3 kann wie folgt nach n_2 aufgelöst werden:

$$n_2 = \frac{DN(N - n_1)}{n_1(N - 1) + D(N - n_1)} \tag{3.15}$$

Allerdings sind große Stichproben für RFID-Szenarien im Einzelhandel problematisch. Große Populationen würden einen großen Aufwand beim Sammeln von Stichproben und bei den Berechnungen bedeuten. In diesem Abschnitt wird gezeigt, dass die zweite Stichprobe von TagMark zu einer oberen Schranke konvergiert, die nur von der RFID-Leserate und von dem

relativen Konfidenzintervall abhängt. Dies ist wichtig, da die zweite Stichprobe von verkauften Artikeln abhängt. Eine große zweite Stichprobe könnte das Zeitfenster vergrößern, so dass die Schätzung für einen älteren Zeitpunkt gelten würde.

Lemma 3.3. *Bei einem durch A und α definiertes relative Konfidenzintervall und einer RFID-Leserate von γ konvergiert die benötigte Stichprobengröße gegen*

$$\lim_{N\to\infty} n_2 = \frac{D(1-\gamma)}{\gamma} \tag{3.16}$$

Beweis: In RFID-Szenarien korreliert der Wert von n_1 mit der Größe N durch die RFID-Leserate γ, d.h. $n_1 = \gamma \cdot N$. Das Einsetzen dieser Gleichung in Gleichung 3.15 resultiert in

$$n_2 = \frac{DN(1-\gamma)}{\gamma(N-1)+D(1-\gamma)} \tag{3.17}$$

Der Beweis wird durch Umformen von Gleichung 3.17 erreicht:

$$\begin{aligned} n_2 &= \frac{D(1-\gamma)}{\gamma} \cdot \frac{N + (-1+\frac{D}{\gamma} - D) - (-1+\frac{D}{\gamma} - D)}{N-1+\frac{D}{\gamma}-D} \\ &= \frac{D(1-\gamma)}{\gamma} \cdot \left(1 - \frac{-1+\frac{D}{\gamma}-D}{N-1+\frac{D}{\gamma}-D}\right) \end{aligned} \tag{3.18}$$

Für $N \to \infty$ ergibt sich $\frac{-1+\frac{D}{\gamma}-D}{N-1+\frac{D}{\gamma}-D} = 0$, wodurch Gleichung 3.18 und Gleichung 3.16 zu demselben Ergebnis führen. Bei $\frac{D}{\gamma} - D \geq 1$ handelt es sich um eine obere Schranke, für andere Werte lediglich um einen asymptotischen Grenzwert. □

3.5 Experimente

In diesem Abschnitt werden mehrere Experimente vorgestellt, die eine Intuition über das RFID-Szenario im Einzelhandel geben und zeigen, dass TagMark in der Praxis und mit realen Daten funktioniert. Die Experimente bestätigen, dass die Charakteristika von RFID-Daten den direkten Einsatz von existierenden Schätzverfahren nicht erlauben. Weiterhin werden der Einfluss der Annahmen und die Skalierbarkeit von TagMark untersucht.

TagMark wurde als Erweiterung von SQL in dem relationalen Datenbankmanagementsystem SAP® MaxDB™ implementiert. Die Experimente wurden auf einem Arbeitsplatzrechner (Linux, 2,4GHz x64 CPU) mit der Standardeinstellung von SAP MaxDB [BGMK08] ausgeführt. Es wurde ein Regal aus dem Einzelhandel mit einem RFID-Leser (Intermec IF5) inklusive einer Antenne ausgestattet und mit UHF-Funketiketten des Typs EPC Class 1 Gen2 [EPC06] getestet. In einem vorläufigen Experiment wurde festgestellt, dass die RFID-Antenne maximal 30 sorgfältig platzierte Artikel erfassen kann. Aus diesem Grund wurden 40 Artikel auf dem Regal platziert, um ein anspruchsvolles Szenario zu evaluieren, bei dem sich viele Artikel in Funklöchern befinden. Mit diesem Versuchsaufbau wurden 10 Sequenzen von 40 Kundeninteraktionen gemessen: Jede Minute wurde ein Artikel aus dem Regal entfernt, als verkauft gekennzeichnet und die von dem RFID-Leser erfassten Artikel festgehalten. Somit entspricht die Zeit in den Experimenten der Anzahl von entfernten Artikeln, so dass das Regal nach 40 Minuten geleert war. Danach wurde das Regal nachgefüllt und eine neue Sequenz gestartet. Insgesamt entstanden somit 400 Datensätze.

Diese Vereinfachung des Kaufverhaltens der Kunden ist ausreichend, um die Anwendbarkeit von TagMark auszuwerten. Da TagMark die Datenstrukturen nach Stichproben durchsucht, die das angegebene relative Konfidenzintervall erfüllen, beeinflussen die Zeiten der Entnahme aus dem Regal und des Verkaufs nicht die Qualität der Schätzung. 40 Artikel sind angemessen, um eine Einzelhandelsszenario unter realistischen Annahmen zu evaluieren. In großen RFID-Szenarien kommen viele RFID-Leser mit mehreren Antennen zum Einsatz, allerdings ändert sich dadurch nicht das Verhältnis zwischen erfassten und nichterfassten Artikeln. Experimente in großen RFID-Szenarien würden daher größere Stichproben und somit kleinere Konfidenzintervalle erlauben, wobei sie nicht zu neuen Erkenntnissen führen würden. Des Weiteren beziehen sich die meisten Abfragen auf einzelne Produkttypen und selten werden mehr als 40 Artikel eines Produkttyps in einem Regal gelagert.

3.5.1 Variierende RFID-Leseraten

Als erstes soll eine Intuition über die Qualität von RFID-Daten gegeben werden. Zusätzlich soll damit gezeigt werden, dass es keine direkte Methode gibt (wie z.B. lineare Regression), mittels der die Anzahl von Artikeln in einem Regal mit einer gegebenen Genauigkeit geschätzt werden kann.

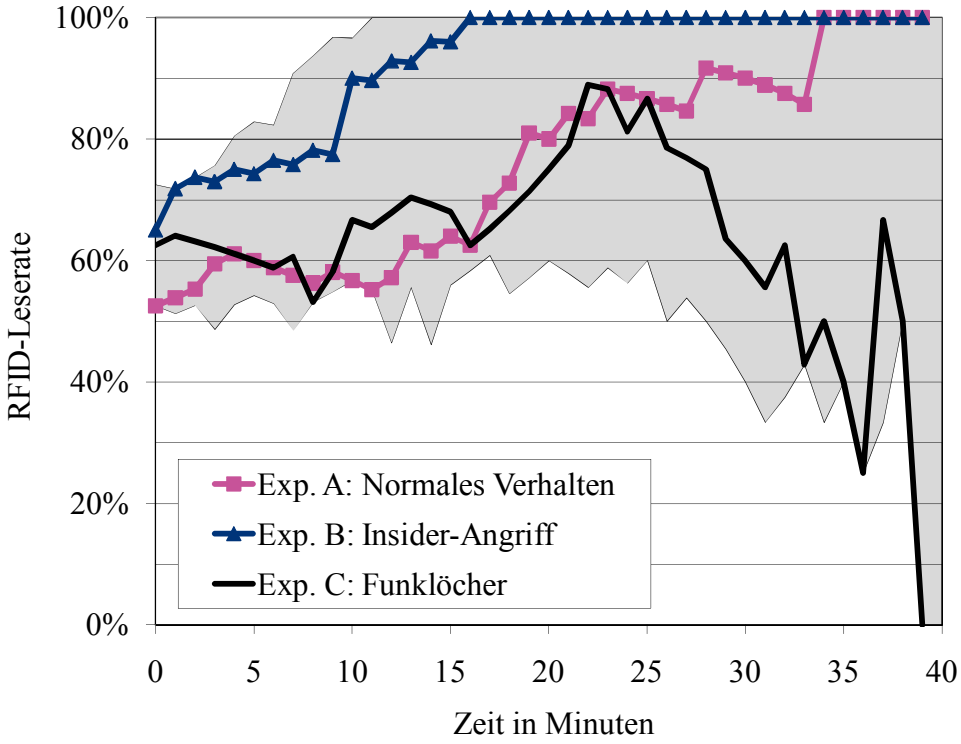

Abbildung 3.1: Verschiedene RFID-Leseraten

Die graue Fläche in Abbildung 3.1 zeigt den Schwankungsbereich der RFID-Leseraten aller Experimente. Die drei Kurven zeigen Beispiele aus drei Sequenzen von Kundeninteraktionen. In den Experimenten A und C wurden die Artikel analog zu normalem Kundenkaufverhalten aus den Regalen entfernt, indem nah am Kunden platzierte Artikel zuerst entnommen wurden. Einziger Unterschied beider Experimente ist, dass die Kunden die Artikel in einer unterschiedlichen Reihenfolge dem Regal entnommen haben. Die RFID-Leserate ist bei beiden Experimenten am Anfang mit ca. 50% niedrig, und steigt mit der Entnahme von Artikeln. In den letzten 13 Minuten sinkt die RFID-Leserate von Experiment C, da aus Zufall viele Artikel in Funklöchern platziert waren. Experiment B zeigt die RFID-Leser bei dem Angriff eines Insiders, der Zugriff auf die RFID-Rohdaten hat und nur nichterfasste Artikel entnimmt. Die RFID-Leserate erreicht nach kurzer Zeit 100%, da nur nichterfasste Artikel entnommen werden.

Die Experimente bestätigen die Erwartungen durch die Literatur [FL04]: Geringe Änderungen bei der Reihenfolge der entnommenen Artikel verursachen sehr unterschiedliche RFID-Leseraten. Aus diesem Grund existiert keine direkte Methode, um mit einer gegebenen Genauigkeit die Anzahl von Artikeln in einem Regal zu schätzen.

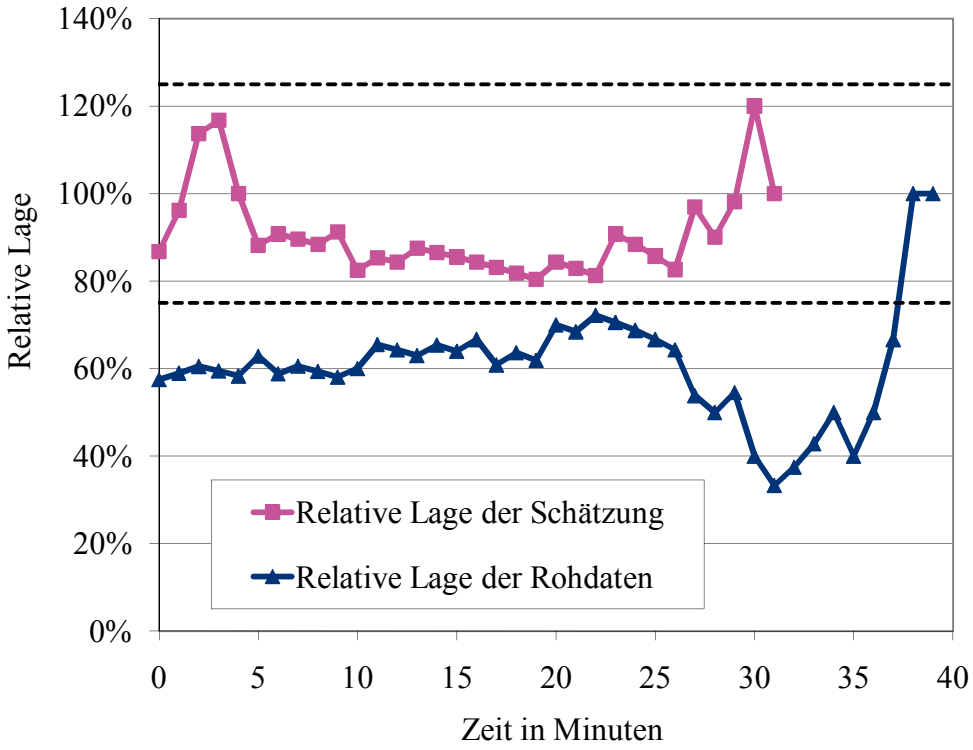

Abbildung 3.2: Beispiel einer Schätzung mit TagMark

3.5.2 Zuverlässigkeit der Schätzung

In diesem Abschnitt wird die Qualität der Schätzung analysiert. Vorerst wird die Nachfüllung der Regale nicht berücksichtigt. Es wird gezeigt, dass TagMark die spezifizierten Konfidenzintervalle in einer realen RFID-Installation bei unterschiedlichen RFID-Leseraten und bei Artikeln, die in verschiedenen Reihenfolgen aus dem Regal entnommen werden, einhält.

Die 400 gesammelten RFID-Datensätze dienen als Eingabe für TagMark. Nach jeder Kundeninteraktion werden zwei Schätzungen berechnet, einmal mit einer relativen Abweichung von $A = 25\%$ und einer Wahrscheinlichkeit von $1 - \alpha = 95\%$, was laut Gleichungen 3.4 und 3.3 einer Fenstergröße von 9 Artikeln entspricht, und einmal eine Schätzung mit den Parametern $A = 18\%$ und $1 - \alpha = 95\%$. Letztere benötigt eine Fenstergröße von 20 Artikeln. Die Schätzungen von TagMark sind korrekt, falls mindestens 95% aller Schätzungen eine relative Abweichung von 25% bzw. 18% aufweisen.

Abbildung 3.2 zeigt ein Experiment im Detail. Die Abbildung zeigt die relative Lage der Anzahl von erfassten und der Anzahl von geschätzten Artikeln zu der Anzahl von Artikeln im Regal. Die gestrichelten Linien markieren die relative Abweichung von 25%. Die Schätzungen von TagMark sind korrekt, falls 95% aller Schätzungen in diesem Intervall liegen. Die X-Achse zeigt die Zeit in Minuten, die der Anzahl von entnommenen Artikeln entspricht, und die Y-Achse zeigt die relative Lage. Da TagMark die Anzahl von Artikeln am Ende des Zeit-

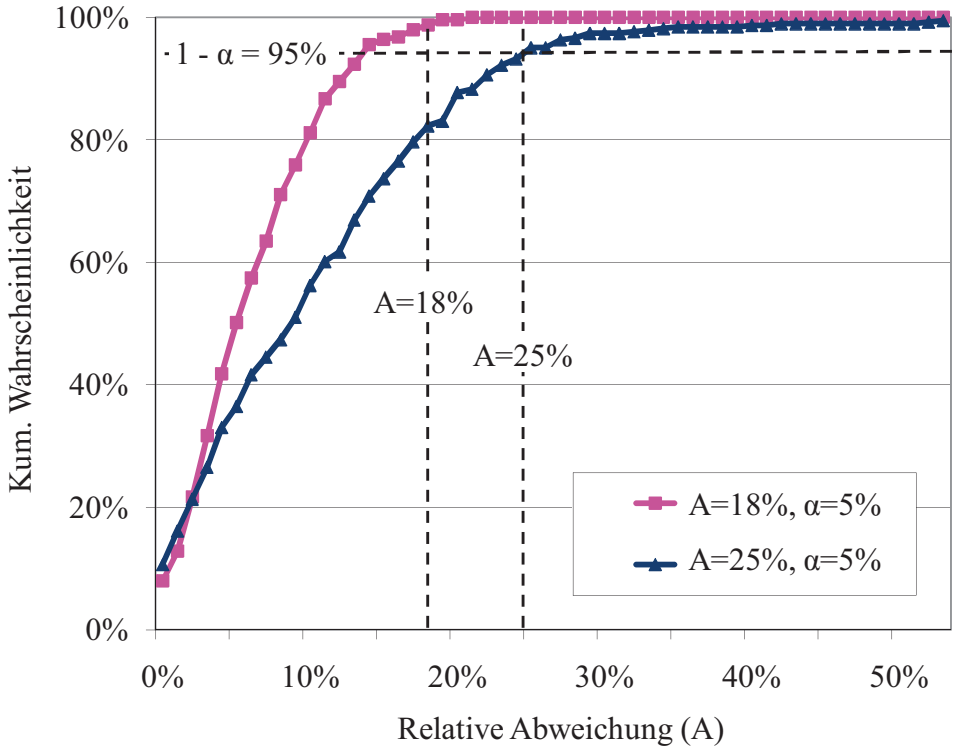

Abbildung 3.3: Kumulierte Werte von TagMark mit verschiedenen Genauigkeiten

fensters schätzt, sind keine Schätzungen für die letzten 9 Minuten vorhanden. Die Abbildung deutet darauf hin, dass TagMark das Konfidenzintervall einhält. Sogar zu Begin des Experiments, als die RFID-Leserate ca. 55% betrug, war die Schätzung korrekt.

Jetzt wird die statistische Signifikanz der Ergebnisse analysiert. Abbildung 3.3 zeigt zwei kumulierte Verteilungsfunktionen für alle Experimente. Der Graph wird gewonnen, indem die Häufigkeiten aller relativen Abweichungen berechnet, sortiert und kumuliert werden. Die X-Achse zeigt die relative Abweichung und die Y-Achse den Anteil der Schätzungen. Beispielsweise zeigt die Abbildung, dass 50% der Schätzungen eine geringere relative Abweichung als 5% bzw. 9% aufweisen. Die vertikalen gestrichelten Linien zeigen die zwei spezifizierten relativen Abweichungen A, und die horizontalen gestrichelten Linien zeigen den Anteil $1 - \alpha = 95\%$, der eine geringere relative Abweichung als A aufweist. Wie aufgrund der analytischen Beweise zu erwarten war, erfüllt TagMark die spezifizierten Konfidenzintervalle: Beide kumulierte Verteilungsfunktionen für die Konfidenzintervalle $(A = 18\%, 1 - \alpha = 95\%)$ und $(A = 25\%, 1 - \alpha = 95\%)$ erreichen die 95%-Marke oberhalb der spezifizierten relativen Abweichungen.

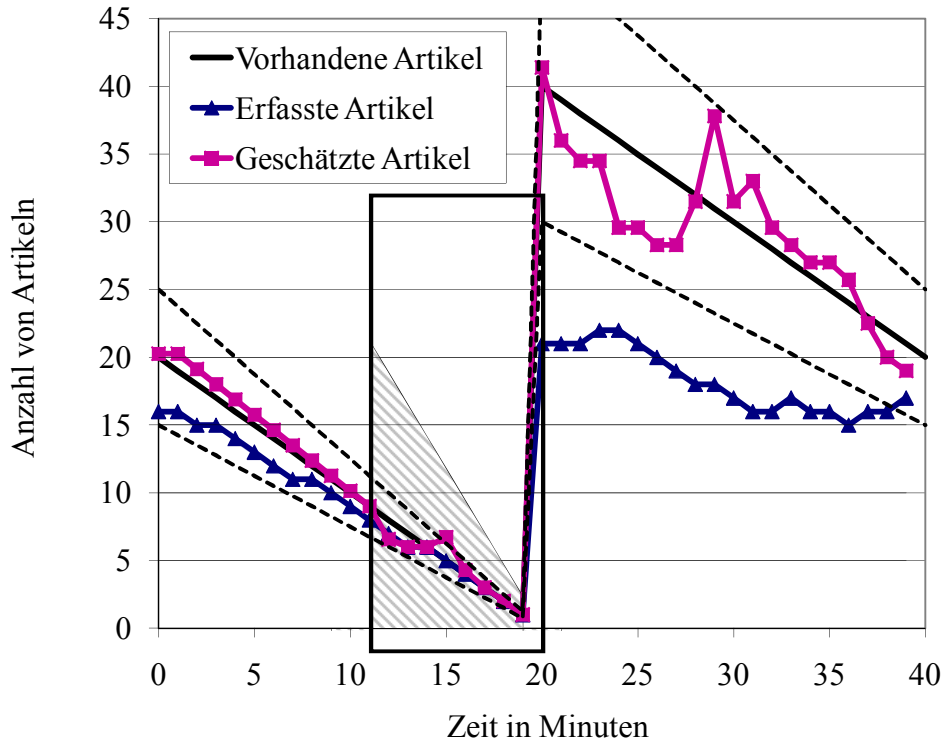

Abbildung 3.4: Beispiel einer Schätzung mit TagMark bei Nachfüllung des Regals

3.5.3 Schätzungen bei Nachfüllung des Regals

In diesem Abschnitt wird geprüft wie gut TagMark funktioniert, wenn ein Regal innerhalb des Zeitfensters nachgefüllt wurde. Wird eine Population N um R Artikel erweitert, so erhöht sich die relative Abweichung um $\frac{AR}{N}$, vgl. Abschnitt 3.4.2. Somit würde die Nachfüllung eines Artikels in einem vollen Regal einen geringen Einfluss auf die Schätzung haben, wobei die komplette Nachfüllung eines leeren Regals den schlimmsten Fall darstellen würde. Dieser Fall wird evaluiert. Der Versuchsaufbau ist wie folgt: Ein Regal ist mit 20 Artikeln gefüllt und analog zu den vorigen Experimenten werden Artikel verkauft. Ist das Regal leer, werden $R = 40$ Artikel aufgefüllt. Es werden dieselben Konfidenzintervalle spezifiziert wie zuvor, $A = 18\%$, $A = 25\%$ und $1 - \alpha = 95\%$. Da die Schätzung von TagMark am Ende des Zeitfensters gültig ist, wird als erstes die Schätzung von $N = 12$ von der Nachfüllung beeinflusst. Die relative Abweichung ändert sich von $A = 18\%$ auf $A + \frac{AR}{N} = 98\%$ und von $A = 25\%$ auf $A + \frac{AR}{N} = 136\%$. Die relative Abweichung ist zwar groß, allerdings ist die Gesamtzahl der Artikel vor der Nachfüllung sehr gering. Dadurch ist der absolute Fehler ebenfalls gering und schränkt die Anwendbarkeit von TagMark in der Praxis nicht ein.

Abbildung 3.4 zeigt das Ergebnis eines typischen Experiments für $A = 25\%$. Die Anzahl von tatsächlich vorhandenen, erfassten und geschätzten Artikeln wird gezeigt. Das Rechteck zeigt die Schätzungen, die durch die Nachfüllung des Regals beeinflusst werden, also bei denen die relative Abweichung $A + \frac{AR}{N}$ beträgt. Die gestrichelten Linien zeigen die relative Abweichung

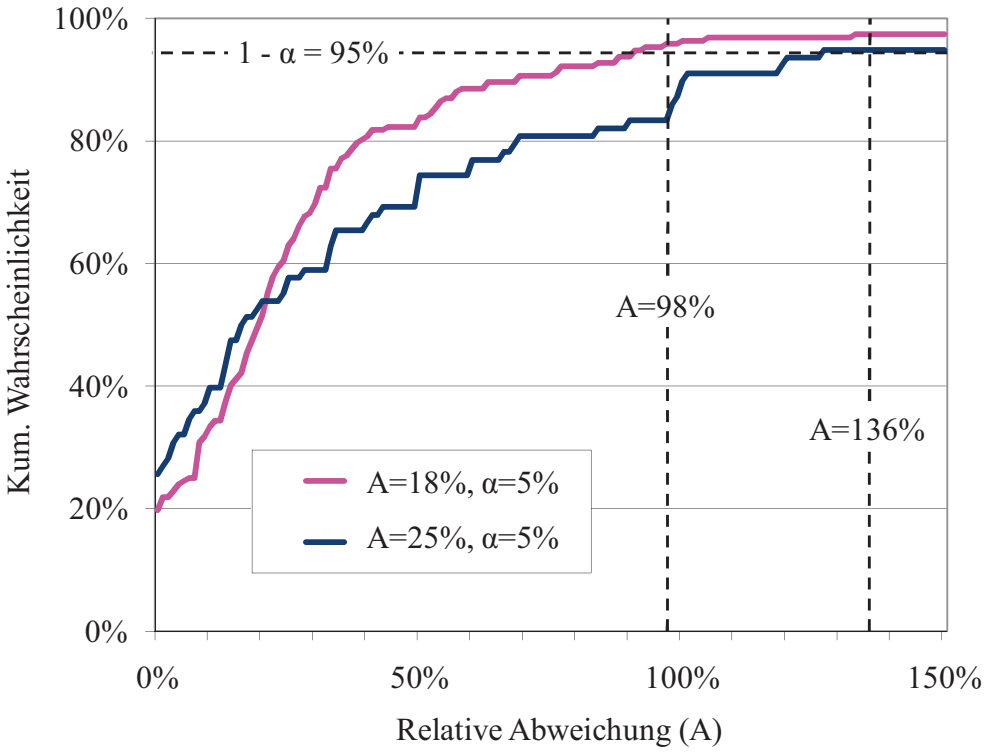

Abbildung 3.5: Kumulierte Werte von TagMark bei Nachfüllung des Regals

von $A = 25\%$ und die graue Fläche die geänderte relative Abweichung von $A = 136\%$. Wie erwartet hält TagMark die avisierten Konfidenzintervalle ein.

Um statistisch signifikante Ergebnisse zu liefern, wird die kumulierte Verteilungsfunktion für die Schätzungen berechnet, die von der Nachfüllung beeinflusst werden. Die Ergebnisse sind in Abbildung 3.5 zu sehen. Die aufgrund der Nachfüllung relaxierten Konfidenzintervalle werden eingehalten.

Das geänderte Konfidenzintervall bei Nachfüllung des Regals hat einen kleinen Einfluss auf die Anwendbarkeit von TagMark. Wird ein fast leeres Regal nachgefüllt, so ist N gering und somit ist der absolute Wert der Abweichung ebenfalls gering. Wird ein fast volles Regal nachgefüllt, so ändert sich die relative Abweichung nur geringfügig.

3.5.4 Angriffe von Insidern

TagMark setzt voraus, dass keine Korrelation zwischen den Funklöchern des RFID-Lesers und dem Kaufverhalten der Kunden existiert. In diesem Abschnitt werden die Auswirkungen auf TagMark untersucht, wenn diese Annahme nicht zutrifft. Aus diesem Grund wird das Szenario evaluiert, bei dem ein Insider mit Zugriff auf die RFID-Rohdaten versucht, TagMark zu täuschen. Der Insider entwendet lediglich Artikel, die nicht von dem RFID-Leser erfasst werden, und hofft somit mit seinem Diebstahl unerkannt zu bleiben.

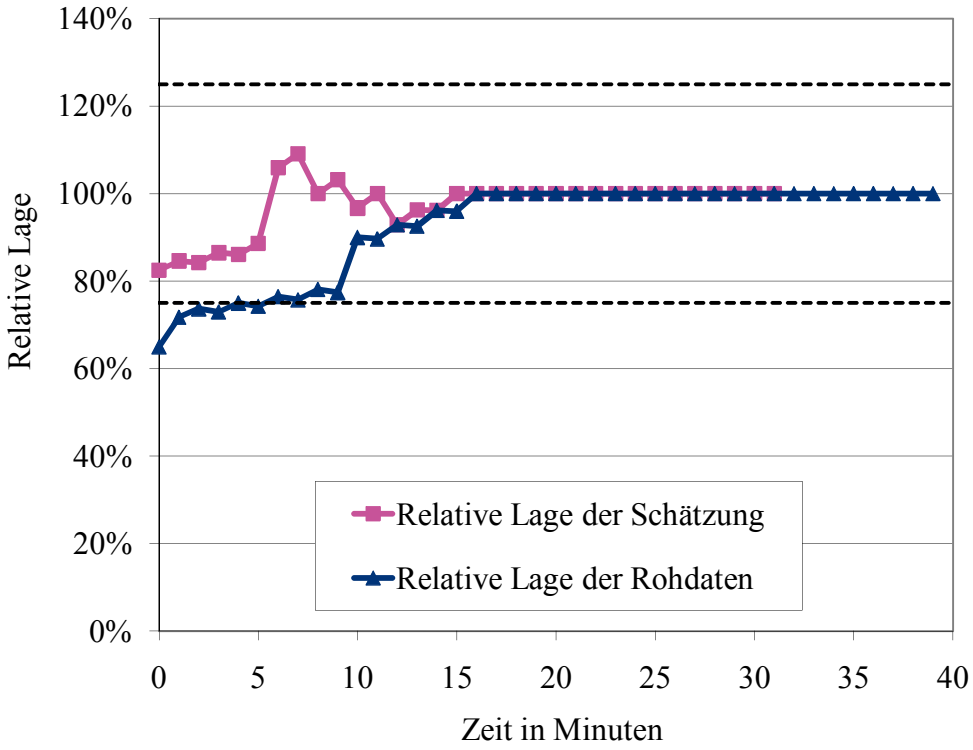

Abbildung 3.6: Beispiel einer Schätzung mit TagMark bei Insider-Angriffen

Diese Art von Angriff wird wie folgt evaluiert: In einem Regal mit 40 Artikeln entfernt ein Insider nach jedem regulären Verkauf einen Artikel, der nicht von dem RFID-Leser erfasst wurde. Dieser Versuchsaufbau wurde ausgewählt, um Änderungen im RFID-Lesefeld zu berücksichtigen. Bei jeder Kundeninteraktion kann sich das RFID-Lesefeld ändern, weshalb ein Insider nicht mehrere Artikel auf einmal entwenden kann.

Es ist zu erwarten, dass TagMark in diesem Experiment die Anzahl von vorhandenen Artikeln unterschätzt. Da nichterfasste Artikel entwendet werden, wird die Stichprobe an der Kasse einen kleinen Anteil an nichterfassten Artikeln enthalten. Allerdings erhöht sich die RFID-Leserate durch die Entwendung von nichterfassten Artikel, und dies hat einen positiven Einfluss auf die Schätzung von TagMark (vgl. Experiment B in Abbildung 3.1).

Abbildung 3.6 zeigt die Ergebnisse eines prominenten Experiments. Die Abbildung zeigt die relative Lage der Anzahl von erfassten und der Anzahl von geschätzten Artikeln zu der Anzahl von Artikeln im Regal. Die gestrichelten Linien markieren die relative Abweichung von 25%. In dem Experiment wurden acht Artikel entwendet bis alle Artikel von dem RFID-Leser erfasst wurden. TagMark konnte das spezifizierte Konfidenzintervall einhalten, obwohl die Anzahl vorhandener Artikel unterschätzt wird. Dieses Ergebnis hängt allerdings von der physikalischen Umgebung ab und kann nicht formell garantiert werden. Aus diesem Grund stellen solche Arten von Attacken ein Problem für TagMark dar. Die restlichen Experimente werden nicht präsentiert, da sie ähnlich verliefen und zu keinen neuen Erkenntnissen führten.

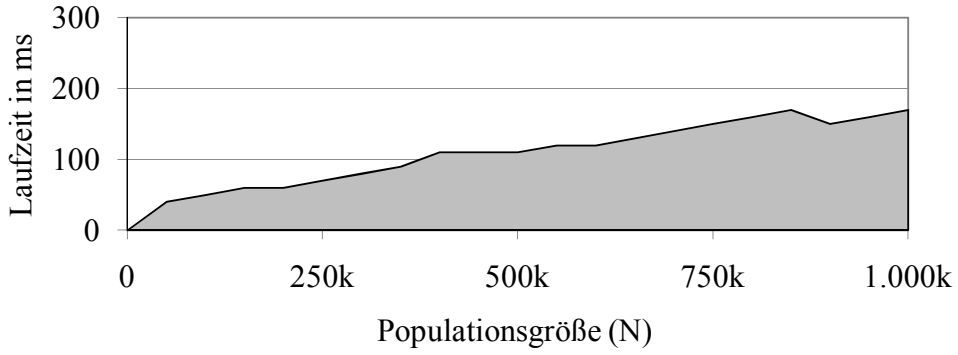

Abbildung 3.7: Laufzeit von TagMark in Abhängigkeit der Populationsgröße N

Bei dem geschilderten Angriff müssen die Artikel einzeln entwendet werden, da sich das Lesefeld nach jeder Kundeninteraktion verändern kann. Man kann Artikeln nicht ansehen, ob sie momentan erfasst werden oder nicht. Daher handelt es sich um ein theoretisches Szenario, das den schlimmsten Fall eines Angriffes darstellt. Obwohl TagMark gegen solche Szenarien keine formellen Garantien bieten kann, scheint es zumindest gegen großflächige Angriffe robust zu sein.

3.5.5 Performanz und Skalierbarkeit

In diesem Abschnitt werden Versuche zu Performanz und Skalierbarkeit vorgestellt, um zu zeigen, dass TagMark in großen Szenarien angewendet werden kann. Es wurden mehrere Tabellen mit synthetischen RFID-Daten erzeugt. Variierte Parameter waren die Größe N der Population, die RFID-Leserate γ und das Konfidenzintervall definiert durch A und α. Um statistisch relevante Daten zu erhalten, wurde jedes Experiment zehnmal wiederholt. Es ist anzumerken, dass TagMark deterministisch ist und somit die Laufzeiten bei einem warmen Cache-Speicher, also wenn sich die notwendigen Daten bereits im Cache-Speicher befinden, gleich sind.

TagMark basiert auf statistischen Berechnungen und nichtverschachtelte Datenbankabfragen mit Zählungen und Gruppierungen. Diese Datenbankabfragen können schnell beantwortet werden. In Abschnitt 3.4.4 wurde zwar gezeigt, dass die Anzahl von nötigen Stichproben konvergiert, allerdings müssen komplette Tabellen abgetastet werden, um die benötigten Datenbankabfragen zu beantworten. Aus diesem Grund ist keine konstante Laufzeit zu erwarten.

Abbildung 3.7 zeigt die durchschnittliche Laufzeit von zehn Experimenten mit den Parametern $1 - \alpha = 95\%$, $A = 10\%$, einer RFID-Leserate von $\gamma = 80\%$ und $n_2 = 0.6N$. Die Größe von N wurde beginnend bei 50.000 in 20 Schritten bis auf $1.000.000$ erhöht. Die gesamte Größe der Tabellen betrug 1,2GB. Die Abbildung zeigt, dass TagMark selbst bei $1.000.000$ Artikeln weniger als 200ms lang rechnet. Versuche mit anderen Parametern lieferten ähnliche Ergebnisse. Die Experimente bestätigen, dass TagMark effizient auf einer relationalen Datenbank implementiert werden kann, und dass es sehr große Datenmengen unterstützt.

3.6 Zusammenfassung

In diesem Kapitel wurde TagMark vorgestellt, ein Verfahren, das die Anzahl von Funketiketten in der Umgebung eines RFID-Lesers schätzt. TagMark erweitert die so genannte Rückfangmethode aus der Biologie, welche die Anzahl von Individuen in einer Population anhand von verschiedenen Stichproben schätzt. Die Genauigkeit der Schätzungen von TagMark kann mittels relativer Konfidenzintervalle bestimmt werden. Des Weiteren existieren obere Schranken für die Größe der benötigten Stichproben. Dies wurde analytisch und experimentell nachgewiesen. Umfassende Experimente mit Daten aus einer RFID-Installation und mit synthetischen Daten haben gezeigt, dass TagMark bei einer Skalierung auf eine große Anzahl von Artikeln schnell ist. Um sicher zu stellen, dass TagMark in der Praxis angewendet werden kann, wurde es unter sowohl realistischen als auch extremen Bedingungen evaluiert. Ein Beispiel für eine solche Situation stellt der Diebstahl durch einen Insider, der nur Artikel entwendet, die der RFID-Leser momentan nicht erfasst, dar.

Kapitel 4

Verifikation von Planogramm-Einhaltung

Im Einzelhandel werden Artikel nach genau definierten Layoutplänen, so genannten Planogrammen, in den Regalen platziert. Prinzipiell kann RFID verwendet werden, um die Einhaltung von Planogrammen zu verifizieren. Entspricht jede RFID-Antenne einem Ort in dem Planogramm, so muss lediglich geprüft werden, welche Antenne einen Artikel erfasst. Allerdings können Funketiketten aufgrund von Reflektionen der elektromagnetischen Wellen von mehreren RFID-Antennen gleichzeitig erfasst werden [FL04]. In solchen Fällen ist es nicht möglich, den genauen Ort eines Artikels bzw. Funketiketts zu bestimmen.

In diesem Kapitel wird ein Verfahren namens *RFID Planogram Compliance Verification* (RPCV) [WBB10] vorgestellt, das entscheidet, ob Artikel, die von mehr als einer RFID-Antenne erfasst wurden, das Planogramm einhalten. Das Verfahren basiert auf der Beobachtung, dass die Anzahl von Lesungen durch eine Antenne für Artikel desselben Produkttyps sehr gut durch eine Normalverteilung beschrieben wird. RPCV stellt jeden Artikel als einen zweidimensionalen Vektor dar. Dieser enthält die Anzahl von Lesungen durch die richtige RFID-Antenne und durch falsche. Für jeden Produkttyp werden diese Daten geclustert. Ein Cluster entspricht dann der Menge von richtig platzierten oder von fehlplatzierten Artikeln. RPCV erzeugt eine Größenordnung weniger falsche Schätzungen als verwandte Arbeiten. Ferner benötigt RPCV weniger RFID-Daten, um gute Schätzungen zu liefern.

4.1 Planogramm-Einhaltung im Einzelhandel

Ein Planogramm spezifiziert den Ort von Produkten auf einem Regalboden. Für jeden Produkttyp wird der Mindestbestand spezifiziert, z.B. „die erste Reihe muss gefüllt sein“. Das Planogramm spezifiziert ebenfalls die Anordnung von Artikeln verschiedener Marken, z.B. erst alle Artikel von Marke A, dann alle Artikel von Marke B. Ebenfalls spezifiziert es die Anordnung von Produkten, also erst Produkt 1 von Marke A, dann das nächste Produkt derselben Marke usw. In dieser Arbeit liegt der Fokus auf Planogrammen, welche die Anzahl von Artikeln eines Typs pro Ort beschreiben.

Planogramme sind wichtig, da eine schnelle und genaue Einhaltung von Planogrammen den Gewinn eines Einzelhändlers um bis zu 8,1% steigern kann [Bis00]. Planogramme verbessern den visuellen Eindruck und die Nutzung der Regalflächen. Ebenso steigern sie die Warenverfügbarkeit, da Regallücken (engl. *Out-of-Stocks*) vermieden werden. Attraktive Layouts erhöhen die Anzahl von Spontankäufen, z.B. indem komplementäre Produkttypen wie Nudeln und Tomatensauce nebeneinander platziert werden. Des Weiteren tendieren Kunden dazu, nur richtig platzierte Artikel einzukaufen; so würde z.B. ein Kunde wahrscheinlich keine Nudeln kaufen, die alleine zwischen Waschmitteln liegen.

Es ist schwierig, Planogramme einzuhalten. Sie sind kompliziert und ändern sich oft, z.B. aufgrund neuer Werbekampagnen oder saisonaler Produkte. Daher ist es schwierig, Nichteinhaltung zu entdecken. Ein anderer Grund dafür ist, dass Regale mit anderen Artikeln aufgefüllt werden können. Wenn zum Beispiel Platz für sechs Artikel in einem Regalboden ist, aber zehn Artikel in einem Karton verpackt sind, könnte ein Angestellter die restlichen vier Artikel irgendwo im Regal ablegen.

RFID kann verwendet werden, um die Einhaltung von Planogrammen zu verifizieren. Ein RFID-Leser steuert viele RFID-Antennen (1:n Relation) an, und er weiß welche Antennen welche Artikel erfassen. Die Antenne, die einen Artikel laut Planogramm erfassen sollte, wird als **richtige Antenne** bezeichnet, andere Antennen als **falsche Antennen**. Mit RFID kann geprüft werden, ob ein Smart-Shelf die richtigen Artikel enthält und ob diese richtig platziert sind. Wenn mehrere RFID-Antennen dicht aneinander eingesetzt werden, können mehrere Antennen denselben Artikel erfassen. Dies geschieht häufig. Der Grund dafür sind physikalische Interferenzen, d.h. es handelt sich um ein grundlegendes Problem, und eine zukünftige RFID-Technologie wird dies nicht umgehen können. Aus diesem Grund kann ein System nicht den Ort von solchen Artikeln bestimmen. Es sei angemerkt, dass es aus Sicht der Datenverarbeitung keinen Unterschied macht, ob Artikel von mehreren Antennen eines RFID-Lesers oder von Antennen mehrerer RFID-Leser erfasst werden. Im Kontext der Planogramm-Einhaltung sind folgende Charakteristika aus Abschnitt 2.3 relevant:

C1: Variierende Anzahl erfasster Artikel: Aufgrund von physikalischen Interferenzen, und, da Artikel verkauft und nachgefüllt werden, variiert die Anzahl von Artikeln, die in einem gewissen Zeitfenster erfasst werden, sehr stark [WBB08].

C3: Unvorhersehbare Menge erfasster Artikel: Aufgrund von physikalischen Effekten wie Absorption und Reflektion, kann das Lesefeld einer RFID-Antenne, also das Gebiet in dem eine Antenne Artikel erfassen kann, nicht vorhergesehen werden.

C5: Große Datenmengen: RFID-Anwendungen im Einzelhandel sind von einer großen Anzahl von Artikeln charakterisiert. Dadurch ergeben sich große Datenmengen, welche die Laufzeit von Schätzverfahren herausfordern.

C6: Artikel weisen ähnliche Lesemuster auf: Eine Normalverteilung beschreibt die Anzahl von Lesungen der Artikel eines bestimmten Produkttyps durch die richtige Antenne sehr gut. Dasselbe gilt für die Anzahl von Lesungen durch falsche Antennen.

4.2 Verwandte Arbeiten

Die Einhaltung von Planogrammen ist eine bekannte Methode, um Abverkäufe zu optimieren [Bis00, BK08]. Allerdings liegen nur wenige Forschungsergebnisse bei der Planogramm-Einhaltung mit Smart-Shelves vor. Decker et al. [DKB03] scheinen die ersten zu sein, die RFID für Planogramm-Einhaltung einsetzen. Um den genauen Ort eines Funketiketts auf dem Regal zu erkennen, verwenden sie mehrere RFID-Antennen pro Artikel, d.h. jedes Smart-Shelf trägt wesentlich mehr Antennen als Artikel. Da eine Filiale im Einzelhandel mehrere Millionen Artikel enthält, ist dieser Ansatz aus betriebswirtschaftlicher Sicht nicht durchführbar. Andere Ansätze, die den Ort von Funketiketten schätzen, die von mehr als einer RFID-Antenne erfasst wurden, werden im Folgenden dargestellt.

Regelbasierte Filter: Verfahren in dieser Kategorie filtern die RFID-Daten innerhalb eines gewissen Zeitfensters basierend auf Regeln [BFHF03, BLHS04, BWL06, BWL$^+$07, FJKR05, JGF06, RDTC06, WL05]. Diese können direkt auf RFID-Datenströme oder auf persistierte Daten angewendet werden. Beispiele von Regeln sind einen Artikel der Antenne zuzuordnen, die den Artikel zuerst erfasst [BWL06, RDTC06] oder die den Artikel zuletzt erfasst [WL05], in der Annahme die neuesten Daten seien korrekt, und den Artikel der Antenne mit den meisten Lesungen zuzuordnen [FJKR05, JGF06]. Im Folgenden werden diese Verfahren entsprechend FIRST, LAST und MOST genannt. Diese Methoden sind schnell, können allerdings viele falsche Schätzungen erzeugen. Wird zum Beispiel MOST auf die Daten von Produkt A in Abbildung 2.5 angewendet, dann werden drei Artikel fälschlicherweise als fehlplatziert klassifiziert, da diese öfters von falschen als von der richtigen Antenne erfasst werden. Da viele kommerzielle Systeme solche Regeln implementieren, z.B. IBM [RDTC06], SAP [BLHS04] und Siemens [BWL06, WL05], werden die Genauigkeit und die Geschwindigkeit dieser Methoden in den Abschnitten 4.4 und 4.5 mit RPCV verglichen.

Probabilistische Filter: Verfahren, die auf probabilistischen Filtern basieren, werden in [JFG08, KBS06, KBS08, XYC$^+$08] vorgeschlagen. Diese Methoden weisen jeder Lesung eine Wahrscheinlichkeit zu, dass diese Lesung richtig ist. Dies geschieht anhand von vordefinierten Einschränkungen und von statistischen Daten. Beispiele von vordefinierten Einschränkungen sind, dass ein Artikel nicht gleichzeitig auf zwei Regalen platziert sein darf, und dass jedes Regal nur eine bestimmte Anzahl von Artikeln halten kann. Statistische Daten sind die Wahrscheinlichkeiten von gewissen Ereignissen, die aus der Historie oder durch manuell erfasste Stichproben gewonnen werden, wie z.B. die Wahrscheinlichkeit, dass sich die Lesefelder von zwei RFID-Antennen überlappen. Allerdings sind solche Daten oft spezifisch sowohl zu dem Ort eines Regals in der Filiale als auch zu den Artikeln im Regal, da Objekte in der Umgebung des RFID-Lesers elektromagnetische Wellen absorbieren oder reflektieren können, und da Artikel von verschiedenen Produkttypen und Regale mit verschiedenem Aufbau unterschiedliche physikalische Eigenschaften aufweisen. Da RFID-Installationen sehr groß sein können, und da der Aufbau der Regale häufig geändert wird, ist es oft nicht praktikabel, solche Stichproben zu sammeln. Daher ist es schwierig, solche Methoden im Einzelhandel tatsächlich einzusetzen. An dieser Stelle sei angemerkt, dass solche Methoden von RPCV profitieren könnten, da RPCV neue statistische Daten liefern könnte. RPCV könnte die

Wahrscheinlichkeit liefern, dass ein bestimmter Artikel richtig oder fehlplatziert ist, basierend auf dem Lesemuster des entsprechenden Produkttyps.

Partikel-Filter: [RLBS08, TSC+09, WKL+08] verwenden Partikel-Filter (engl. *Particle Filter*), um den Ort von Funketiketten zu bestimmen. Partikel-Filter stellen eine Wahrscheinlichkeitsverteilung eines Ereignisses durch generierte Stichproben, so genannte Partikel, dar. Die Stichproben werden im Laufe der Zeit aktualisiert und verwendet, um Schätzungen zu berechnen. Partikel-Filter erzeugen gute Ergebnisse, wenn Funketiketten (oder RFID-Leser) in Bewegung sind, da die Stichproben aktualisiert werden, wenn ein Funketiketten mehrere RFID-Antennen passiert. Allerdings werden im Einzelhandel Funketiketten von mehreren Antennen gleichzeitig erfasst, weil sich die Lesefelder überlappen, und nicht, weil Funketiketten oder Leser in Bewegung sind. Ein Funketikett, das oft durch falsche Antennen erfasst wird, würde somit viele Partikel an dem falschen Ort erzeugen. Dadurch würde der Partikel-Filter schlechte Ergebnisse liefern.

Sonstige Methoden: [TP08, TZP09] haben verschiedene Verfahren studiert, um den Ort von Funketiketten in Bewegung zu schätzen. Verschiedene Algorithmen werden präsentiert, die ein Funketikett einem gewissen Ort zuweisen, nachdem es des Öfteren oder durch mehr als einer Antenne entlang der Lieferkette erfasst wurde. Allerdings sind solche Verfahren nur für Funketiketten in Bewegung anwendbar und lassen sich nicht auf RFID-Szenarien im Einzelhandel anwenden. Des Weiteren können Verfahren zur Schätzung der Anzahl von Artikeln (vgl. Kapitel 3) verwendet werden, um die Variation der Anzahl von Lesungen zu reduzieren. Beispielsweise können, wie bereits erwähnt, mehrere Funketiketten auf einen Artikel angebracht werden [BR07]. Dadurch kann die Variation der Anzahl von Lesungen zwar geringer ausfallen, allerdings ist dies nicht das einzige Problem bei Planogramm-Einhaltung. Artikel mit vielen Funketiketten werden genau so oft oder sogar öfter von falschen Antennen erfasst.

Zusammenfassend ist festzustellen, dass nur die Verfahren FIRST, LAST und MOST direkt für die Verifikation von Planogramm-Einhaltung eingesetzt werden können. Aus diesem Grund werden diese Verfahren in den Abschnitten 4.4 und 4.5 mit RPCV verglichen.

4.3 RPCV: Verifikation von Planogramm-Einhaltung mit RFID

In diesem Abschnitt wird RPCV vorgestellt. RPCV entscheidet, ob Artikel, die von mehr als einer RFID-Antenne erfasst wurden, das Planogramm einhalten. Es ist schnell, skalierbar und es erzeugt eine Größenordnung weniger falsche Schätzungen als verwandte Arbeiten. Des Weiteren benötigt es weniger Daten, um gute Schätzungen zu berechnen; die Größe des benötigten Zeitfensters ist also gering.

RPCV basiert auf der Beobachtung, dass Artikel desselben Produkttyps ähnliche physikalische Eigenschaften und somit relativ ähnliche Lesemuster aufweisen (vgl. Charakteristik C6). Die Intuition für das Verfahren ist wie folgt: Zunächst zählt RPCV für jeden Artikel die Anzahl von Lesungen durch jede Antenne. Wie in den Feldversuchen beobachtet, erfassen RFID-

Leser die Artikel mit verschiedenen Häufigkeiten (vgl. Charakteristik C1), d.h. die Anzahl von Lesungen der einzelnen Artikel kann stark variieren. Um diese Variation zu reduzieren, wird ein Tiefpass-Filter angewendet. Danach werden alle Artikel eines Produkttyps geclustert. Jeder Artikel wird durch einen zweidimensionalen Vektor dargestellt, der die Anzahl von Lesungen durch die richtige und durch falsche RFID-Antennen enthält. Danach wird für jeden Cluster entschieden, ob er fehlplatzierte Artikel repräsentiert.

An dieser Stelle sei angemerkt, dass RPCV mit geringem Aufwand erweitert werden kann, um Produkttypen zu unterstützen, die an mehr als einem Ort platziert werden, z.B. indem solche Produkttypen als verschiedene logische Produkttypen modelliert werden, oder indem statt nur einer richtigen Antenne eine Menge richtiger Antennen angenommen wird.

4.3.1 Der RPCV-Algorithmus im Detail

RPCV benötigt zwei Datenstrukturen:

- $RfidReads$: eine Multimenge mit Tupeln (i, ant) von Artikeln i und der Antenne ant, die diesen Artikel innerhalb eines vordefinierten Zeitfensters erfasst.
- $Planogram$: ein *Map* bei dem die Schlüssel die Produkttypen t sind, und der zu t entsprechende Wert ist die RFID-Antenne ant, welche die Artikel des Typs t erfassen sollte.

Das Zeitfenster bestimmt die Menge an Daten, die RPCV zur Verfügung steht. Die Größe des Fensters ist von dem Einsatzszenario abhängig, d.h. es hängt von den Zeitabständen ab, in denen ein Einzelhändler seine Regale aufräumt, und von der Häufigkeit, mit der die RFID-Leser abgefragt werden. Der Einfluss der Größe des Zeitfensters auf die Qualität und Laufzeit von RPCV wird in Abschnitt 4.5.2 untersucht.

Algorithmus 4.1 beschreibt RPCV. Er iteriert über alle Produkttypen (Zeile 3). Erst bestimmt RPCV welche RFID-Antenne ant jeden Produkttyp laut $Planogram$ erfassen sollte (Zeile 5). Für jeden Artikel zählt RPCV die Anzahl von Lesungen durch die richtige Antenne ($readsR$) und die Anzahl von Lesungen durch falsche Antennen ($readsW$) und fügt diesen zweidimensionalen Vektor zu einer Menge P entsprechend zu t zu (Zeilen 6-10). Die Funktion $project_X$ lässt die Vektoren aus, dessen Stelle nicht X ist. Sie wird verwendet, um Attribute herauszuprojezieren, die in dem aktuellen Schritt nicht verwendet werden, aber in den darauffolgenden Schritten notwendig sind. In Zeile 13 wird Attribut i aus P entfernt, da es nicht beim Clustern verwendet wird. RPCV wendet einen Tiefpass-Filter an, um die Variation der Werte von $readsR$ und $readsW$ zu verringern (Zeile 14). Danach wird ein Clustering-Algorithmus auf die Werte $readsR$ und $readsW$ von jedem Artikel angewendet, um zwei Cluster P_1 und P_2 für jeden Produkttyp zu gewinnen (Zeile 15). Später im Text wird erläutert, warum die Anzahl von Clustern gleich zwei ist. Der erste Parameter der Funktion $Cluster$ in Zeile 15 ist die Menge von Vektoren, die geclustert werden soll, und der zweite Parameter ist Anzahl von Clustern, die gefunden werden sollen. Anschließend entscheidet RPCV, ob ein Cluster fehlplatzierte Artikel repräsentiert. Um dies zu erreichen wird geprüft, ob ein Cluster im Durchschnitt öfter

von falschen Antennen erfasst wurde (Zeilen 16-20). Nachdem RPCV durch alle Produkttypen iteriert hat, wird die Menge von fehlplatzierten Artikeln zurückgegeben. Die Wahl des Clustering-Algorithmus und des Tiefpass-Filters wird in den folgenden Abschnitten dargestellt.

```
 1: input RfidReads, Planogram
 2: MisplacedItems = {}
 3: for all (productType t ∈ Planogram.getKeys()) do
 4:    P = {}
 5:    ant = Planogram.get(t)
 6:    for all (item i ∈ project_1(RfidReads)) do
 7:       if (item i ist von Typ t) then
 8:          readsR = |{(i, a) ∈ RfidReads : a = ant}|
 9:          readsW = |{(i, a) ∈ RfidReads : a ≠ ant}|
10:          P = P ∪ {(i, readsR, readsW)}
11:       end if
12:    end for
13:    P* = project_{2,3}(P)
14:    P* = lowpassFilter(P*)
15:    {P_1, P_2} = Cluster(P*, 2)
16:    for all (cluster P_j) do
17:       if avg(P_j.readsW) > avg(P_j.readsR) then
18:          füge die Elemente aus P, die den Vektoren in P_j entsprechen, in MisplacedItems
             ein
19:       end if
20:    end for
21: end for
22: output MisplacedItems
```

Algorithmus 4.1: RPCV-Algorithmus

Beispiel 4.1 führt ein einfaches Szenario für Planogramm-Einhaltung ein. Dieses Szenario wird in den kommenden Abschnitten verwendet, um RPCV, den Clustering-Algorithmus und den Tiefpass-Filter zu veranschaulichen.

Beispiel 4.1: Abbildung 4.1 zeigt zwei RFID-Antennen und Artikel von zwei verschiedenen Produkttypen in einem zweidimensionalen Raum. Die Produkttypen werden durch Kreise und Quadrate dargestellt. Die graue Fläche zeigt das Lesefeld von Antenne A. Es soll entschieden werden, ob Artikel 1 bis 6, die vom Produkttyp Kreis sind, das Planogramm einhalten. Es wird angenommen, dass das Planogramm besagt, dass diese Artikel unter Antenne B platziert werden sollen. Artikel 1 wird häufig von Antenne A erfasst, Artikel 2 wird häufig von Antennen A und B erfasst und Artikel 3 bis 6 werden häufig von Antenne B erfasst.

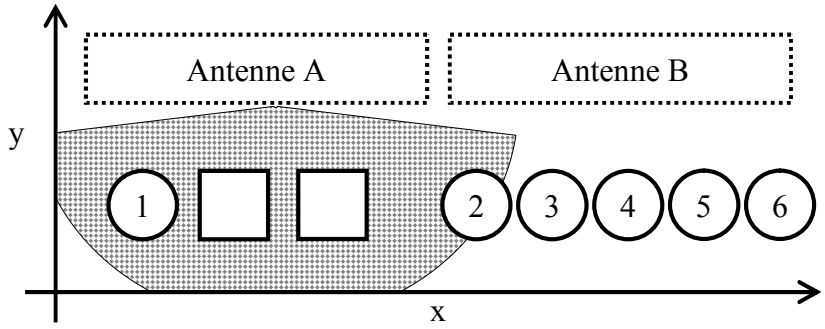

Abbildung 4.1: Beispielszenario für Planogramm-Einhaltung

4.3.2 Clustering mithilfe des Expectation-Maximization-Algorithmus

Artikel desselben Produkttyps haben ähnliche physikalische Eigenschaften, d.h. sie beeinflussen die elektromagnetischen Wellen der RFID-Leser auf eine ähnliche Art und Weise. Daher können Artikel desselben Typs mit einer ähnlichen Wahrscheinlichkeit von falschen Antennen erfasst werden. RPCV nutzt dies aus, indem ein Clustering-Algorithmus verwendet wird. Dieser wird auf das Paar $readsR$ und $readsW$ von jedem Artikel angewendet.

In einer Reihe von vorläufigen Versuchen wurden verschiedene Clustering-Algorithmen ausgewertet [WF05]. Diese Algorithmen wurden in RPCV integriert und es wurde die Anzahl von falschen Schätzungen unter Verwendung der Daten des Feldversuches gezählt. Die Ergebnisse waren wie folgt: *Density-Based Clustering* (11% falsche Schätzungen), *Farthest First Traversal* Algorithmus (10% falsche Schätzungen), *k-Means* (9% falsche Schätzungen) und *Expectation Maximization* (EM, 1% falsche Schätzungen). In anderen Worten, der EM-Algorithmus [DLR77] liefert die besten Ergebnisse für das Einzelhandelsszenario.

1: **input** Tupel $\{(x, y)\}$, Anzahl zu bestimmender Cluster k
2: $\{P_1..., P_k\} = initClusters(\{(x, y)\}, k)$
3: $\{\theta_1..., \theta_k\} = initDistributions(\{P_1, ...P_k\})$
4: **while** (Cluster ändern sich) $\wedge$ (# von Iterationen < Schranke) **do**
5: **for all** $(cluster\ P_j)$ **do**
6: $E_j = estimateExpectedValues(P_j, \theta_j)$ // E-Schritt
7: $\theta_j = estimateParameters(P_j, E_j)$ // M-Schritt
8: $P_j = updateCluster(P_j, \theta_j)$
9: **end for**
10: **end while**
11: **output** $\{P_1, ..., P_k\}$

Algorithmus 4.2: EM-Algorithmus

In Abschnitt 4.4 werden Experimente vorgestellt, bei denen RPCV gut funktioniert und Experimente, bei denen dies nicht der Fall ist. Um die Experimente gut zu verstehen, sind Kenntnisse des EM-Algorithmus erforderlich. Aus diesem Grund wird er in Algorithmus 4.2 vorgestellt.

Der EM-Algorithmus versucht jede Dimension der Eingabedaten durch k Wahrscheinlichkeitsverteilungen abzudecken. Das Ergebnis sind k Cluster und die Wahrscheinlichkeitsverteilung jeder Dimension beschreibt einen Cluster. Da RPCV zweidimensionale Eingabedaten liefert, stellen zwei Wahrscheinlichkeitsverteilungen einen Cluster dar. Der Algorithmus beginnt mit einer initialen Menge von Clustern auf den zweidimensionalen Daten $\{(x, y)\}$ (Zeile 2), die durch die Ausführung weniger Iterationen des k-Means Algorithmus gewonnen wird. Der EM-Algorithmus schätzt die Parameter der Verteilung der initialen Cluster (Zeile 3) und verbessert diese iterativ (Zeilen 4-10). Der Algorithmus iteriert bis die Cluster sich nicht mehr ändern oder bis eine bestimmte Anzahl von Iterationen erreicht wird. Für jeden Cluster und in jeder Iteration wird zuerst der Erwartungswert der Log-Wahrscheinlichkeit der aktuellen Verteilung berechnet (*Expectation*-Schritt, Zeile 6). Anschließend werden die Parameter der Verteilung mit den Erwartungswerten aus dem E-Schritt aktualisiert (*Maximization*-Schritt, Zeile 7). Am Ende gibt der EM-Algorithmus k Cluster zurück. Als Wahrscheinlichkeitsverteilung wird die Normalverteilung verwendet. Sie modelliert die RFID-Daten korrekt und kann schnell berechnet werden, da geschlossene Formeln der Schätzer existieren [DLR77]. Beispiel 4.2 verdeutlicht wie RPCV den EM-Algorithmus verwendet.

Beispiel 4.2: In diesem Beispiel, das Beispiel 4.1 weiterführt, werden die Lesungen von Antenne B betrachtet. Innerhalb des Zeitfensters erfasst Antenne B die Artikel mit folgender Häufigkeit: Artikel 1: 0 Mal, Artikel 2: 9 Mal, Artikel 3 bis 5: 16 Mal, Artikel 6: 25 Mal. Als erstes gruppiert RPCV diese Artikel nach der Anzahl von Lesungen. Dies wird in Abbildung 4.2 gezeigt. Die Balken zeigen die Anzahl von Vorkommnissen von jeder Anzahl von Lesung. Die Abbildung zeigt auch beide Normalverteilungen, mit denen der EM-Algorithmus die Daten abdeckt. Jede Kurve stellt einen Cluster dar. RPCV entscheidet, ob ein Cluster fehlplatzierte Artikel repräsentiert, indem geprüft wird, ob der Cluster im Durchschnitt öfter von falschen Antennen erfasst wurde. Jeder Balken wird der wahrscheinlichsten Verteilung zugeordnet, d.h. zu der Kurve mit dem höchsten Wert auf der X-Position des Balkens. Die Balken mit den Häufigkeiten 0 bis 9 gehören zu einem Cluster, der sich als Cluster von fehlplatzierten Artikeln erweist. Die anderen zwei Balken gehören zu dem anderen Cluster, der sich als Cluster von richtig platzierten Artikeln erweist.

Intern verwendet der EM-Algorithmus Wahrscheinlichkeiten, um die Zugehörigkeit von Artikeln zu Clustern festzulegen. In dem Einzelhandelsszenario liefert diese „weiche“ Zugehörigkeit bessere Ergebnisse als feste Zugehörigkeiten: Während der Iterationen des Clustering-Algorithmus ist die Wahrscheinlichkeit höher, dass sich die Zugehörigkeit eines Artikels zu einem bestimmten Cluster ändert. Dies verringert die Wahrscheinlichkeit, dass der Clustering-Algorithmus sich in einem lokalen Maximum verfängt. Als Beispiel sei Abbildung 4.2 aus Beispiel 4.2 genannt: Der Artikel, der 9 Mal erfasst wurde, gehört mit einer hohen Wahrscheinlichkeit zu dem Cluster von fehlplatzierten Artikeln und mit einer geringen Wahrscheinlichkeit zu dem anderen Cluster.

Wie bereits erwähnt, wird der EM-Algorithmus konfiguriert, um zwei Cluster zu finden, da

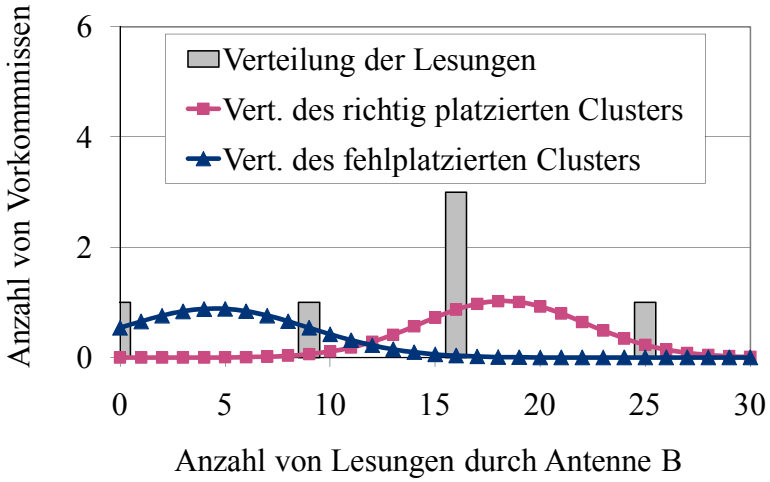

Abbildung 4.2: Verteilung der RFID-Rohdaten

zwischen zwei Arten von Artikeln unterschieden werden soll: Richtig platzierte und fehlplatzierte Artikel. Es wurde mit unterschiedlichen Anzahlen von Clustern experimentiert, wobei zwei Cluster die besten Ergebnisse lieferten. Betrachtet man die RFID-Rohdaten, dann existieren Artikel, die offensichtlich richtig platziert oder fehlplatziert sind, und Artikel, bei denen der tatsächliche Ort schwierig zu ermitteln ist, z.B. weil zwei Antennen einen Artikel sehr oft erfassen. Bei mehr als zwei Clustern kann jede Art solcher Artikel einen eigenen Cluster bilden. Dann kann der Ort solcher „schwierigen“ Artikel nicht bestimmt werden. Mit zwei Clustern wird der EM-Algorithmus dazu gezwungen, solche Artikel entweder dem Cluster von richtig platzierten Artikeln oder dem Cluster von fehlplatzierten Artikeln zuzuordnen. Es sei angemerkt, dass grundsätzlich beide Cluster richtig platzierte oder fehlplatzierte Artikel repräsentieren können, z.B. wenn alle oder keine Artikel fehlplatziert sind. Um solche Situationen zu beherrschen, entscheidet RPCV ob ein Cluster fehlplatzierte Artikel repräsentiert, indem es prüft ob der Cluster im Durchschnitt öfter durch falsche Antennen erfasst wurde.

4.3.3 Reduzierung der Variation der Werte mit einem Tiefpass-Filter

Obwohl die Lesemuster der Artikel desselben Produkttyps tendenziell ähnlich sind, kann die absolute Anzahl von Lesungen einzelner Artikel stark voneinander abweichen. Dies geschieht aufgrund von Funklöchern, und weil Artikel verkauft und nachgefüllt werden (vgl. Charakteristik C1). Dies stellt ein Problem für Clustering-Algorithmen dar. Wie in den Feldversuchen beobachtet, existieren einzelne Artikel, die nur ein Mal erfasst werden, während andere hunderte von Malen erfasst werden.

Um dieses Problem zu bewältigen, wendet RPCV einen Tiefpass-Filter an. Tiefpass-Filter lassen Zahlen mit einem geringen Wert passieren und reduzieren die Werte von großen Zahlen. Ein Tiefpass-Filter für das Einzelhandelsszenario muss die folgenden Anforderungen erfüllen: (1) Er muss die Variation der Anzahl von Lesungen stark verringern und (2) er muss eine

große Anzahl von Artikel unterstützen (vgl. Charakteristik C5). Aufgrund von (2) können nur einfache Filter angewendet werden. Viele mathematische Funktionen können schnell berechnet werden und erfüllen Anforderung (1). In einem ersten Schritt wurden solche Funktionen identifiziert, z.B. Wurzeln, Logarithmen und Sigmoid-Funktionen. In einem nächsten Schritt wurden diese Funktionen in vorläufigen Experimenten mit verschiedenen Parametern evaluiert und auf die Daten der Feldversuche angewendet. Die besten Ergebnisse wurden mit der Quadratwurzel erzielt, da sie die geringste Anzahl von falschen Schätzungen erzeugte. Die Quadratwurzel reduziert deutlich den Wert von großen Zahlen, also von Artikeln mit einer großen Anzahl von Lesungen. Und sie bewirkt geringe Änderungen auf kleine Werte, so dass die Variation der Anzahl von Lesungen verringert wird. Die Funktion $lowpassFilter$ wird in Gleichung 4.1 gezeigt.

$$lowpassFilter(\{(x, y)\}) = \{(\sqrt{x}, \sqrt{y})\} \tag{4.1}$$

Beispiel 4.3 verdeutlicht die Funktionsweise des Tiefpass-Filters. Es zeigt den Einfluss des Filters auf die von dem EM-Algorithmus bestimmten Cluster.

Beispiel 4.3: In Abbildung 4.2 existiert eine große Abweichung bei der Anzahl von Lesungen für jeden Artikel und die Balken haben einen ähnlichen Abstand auf der X-Achse. Der EM-Algorithmus hat Schwierigkeiten solche Daten richtig zu klassifizieren und der Balken bei 9 Lesungen (Artikel 2) ist fälschlicherweise dem Cluster von fehlplatzierten Artikeln zugeordnet. Abbildung 4.3 zeigt das Ergebnis, wenn der Tiefpass-Filter auf die Anzahl von Lesungen angewendet wird. Die Daten sind auf einem geringeren Intervall verteilt, und große Werte liegen auf der X-Achse näher aneinander. Jetzt werden die Daten durch den EM-Algorithmus richtig klassifiziert.

4.4 Experimente mit realen Daten

In den folgenden Abschnitten wird RPCV evaluiert. Dabei wird (1) eine Intuition über die Funktionsweise von RPCV vermittelt, es wird gezeigt, dass (2) RPCV genauere Schätzungen als verwandte Arbeiten liefert, dass (3) es weniger Daten für gute Schätzungen benötigt, (4) es werden Szenarien identifiziert, bei denen RPCV nicht gut funktioniert, und (5) es wird aufgezeigt, dass RPCV bei Schätzungen mit einer großen Anzahl von Artikeln und einer großen Anzahl von RFID-Lesungen schnell ist. RPCV wird mit den Verfahren FIRST, LAST und MOST verglichen, da diese die einzigen Verfahren aus den verwandten Arbeiten sind, die sich direkt auf Planogramm-Einhaltung anwenden lassen. Diese Verfahren teilen Artikel der RFID-Antenne zu, welche die Artikel zuerst, zuletzt oder am häufigsten erfasst hat.

RPCV wurde als Erweiterung von SQL in dem relationalen Datenbankmanagementsystem SAP® MaxDB™ implementiert. Alle Experimente wurden auf einem Arbeitsplatzrechner ausgeführt (Windows, 2GB RAM, 2GHz Doppelkernprozessor, 1 SATA Festplatte). Es wurden

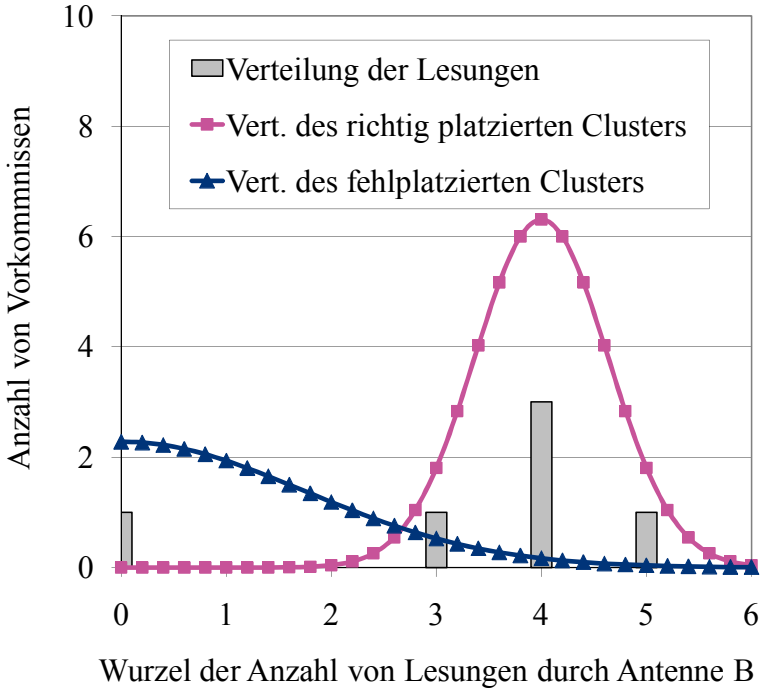

Abbildung 4.3: Verteilung der gefilterten RFID-Daten

die Standardeinstellungen von MaxDB [BGMK08] und von Java (Version 1.6.0_12) verwendet.

RPCV wird mit dem in Abschnitt 2.2.2 vorgestellten Szenario ausgewertet. Es werden Daten aus einer realen RFID-Installation und synthetische Daten verwendet.

4.4.1 Versuchsaufbau der RFID-Installation für reale Daten

Ein Regal aus dem Einzelhandel wurde mit zwei RFID-Lesern (Intermec IF5) ausgestattet. Das Regal stammte von einem großen deutschen Einzelhändler und glich denjenigen Regalen, die in den Filialen eingesetzt wurden. Die Abmessungen des Regals waren 166cm x 110cm x 65cm. Das Regal enthielt vier Regalböden, wobei jeder Regalboden durch zwei RFID-Antennen abgedeckt wurde, vgl. Abbildung 2.4 in Abschnitt 2.2.2. Das Regal wurde mit Artikeln von fünf verschiedenen Produkttypen ausgestattet, die mit UHF-Funketiketten des Typs EPC Class 1 Gen2 [EPC06] bestückt waren. Es werden Artikel von dem Produkttyp ausgewertet, bei dem Schätzungen am schwierigsten sind. Das sind die Artikel mit den schlechtesten Leseraten. Also Artikel, die selten von der richtigen Antenne und oft von naheliegenden Antennen erfasst werden.

Im Durchschnitt enthält ein Regal zwischen 8 und 13 Artikel eines Produkttyps [KZBD05]. Für die Experimente wird ein anspruchsvolles Szenario gewählt, mit viel mehr Artikeln als im Durchschnitt. Es werden 30 Artikel des Produkttyps mit den schlechtesten Lesemustern

verwendet und drei Arten von Szenarien ausgewertet:

1. **Statisches Szenario:** Artikel werden auf verschiedene Art und Weise im Regal angeordnet. Der Anteil an fehlplatzierten Artikeln wird zwischen 0% und 50% variiert. Nachdem alle Artikel einsortiert sind, beginnt die Erfassung der RFID-Daten.

2. **Verkaufs-Szenario:** Artikel werden analog zu dem statischen Szenario angeordnet. Nachdem die Erfassung der RFID-Daten beginnt, werden Artikel, die nicht fehlplatziert sind, nacheinander aus dem Regal entfernt.

3. **Nachfüllungs-Szenario:** Der Aufbau ist wie im Verkaufs-Szenario, aber nachdem alle Artikel verkauft wurden wird das Regal mit neuen Artikeln nachgefüllt.

In dem statischen Szenario gibt es keine Kundeninteraktionen, wie z.B. Abverkäufe oder Nachfüllungen des Regals. Um sicher zu stellen, dass die Kundeninteraktionen in den beiden weiteren Szenarien nicht beeinflusst werden, wurden diese von einer Person ausgeführt, die mit RPCV nicht vertraut war. Dies entspricht dem Stand der Technik, nachdem die Personen, die einen Versuch entwerfen, an diesem nicht teilnehmen dürfen. Beide Szenarien mit Kundeninteraktionen sind anspruchsvoll: Im Verkaufs-Szenario werden die Lesungen von verkauften Artikeln eine große Abweichung zeigen. Und bei dem Nachfüllungs-Szenario werden alle richtig platzierten Artikel einmal ersetzt.

Die RFID-Leser wurden dreimal in der Minute, 10 Minuten lang, abgefragt. Eine Abfrage wird als Lesezyklus bezeichnet. Daher bestand das Zeitfenster aus 30 Lesezyklen. Das Smart-Shelf erzeugte 34.581 Lesungen. Insgesamt wurde der Ort von Artikeln 299 Mal geschätzt.

Dieser Versuchsaufbau wurde gewählt, da er dem Aufbau in den Filialen eines großen deutschen Einzelhändlers entspricht. Es ist allerdings zu erwarten, dass RPCV ebenfalls mit anderen RFID-Installationen funktioniert. Artikel desselben Produkttyps weisen relativ ähnliche Lesemuster auf, weil sie ähnliche physikalische Eigenschaften haben (vgl. Charakteristik C6). Es handelt sich also nicht um eine Eigenheit der RFID-Installation.

4.4.2 Eindruck über die Funktionsweise von RPCV

Dieses Experiment soll eine Intuition über die Funktionsweise von RPCV vermitteln. Es wurde mit Daten aus dem Nachfüllungs-Szenario mit 25% Fehlplatzierungen experimentiert. Die Ergebnisse werden in Abbildung 4.4 dargestellt. Die X-Achse zeigt die Anzahl von Lesungen durch die richtige RFID-Antenne und die Y-Achse die Anzahl von Lesungen durch falsche Antennen. Die dunkelgrauen Kreise repräsentieren richtig platzierte Artikel und die hellgrauen Kreise repräsentieren tatsächlich fehlplatzierte Artikel. Die Fläche der Kreise steht für die Anzahl von Artikeln in der Position in der Abbildung, so steht z.B. der hellgraue Kreis ganz links für zwei Artikel. Artikel oberhalb der Diagonalen wurden öfter von falschen Antennen erfasst als von der richtigen. Beispielsweise repräsentiert der hellgraue Kreis ganz rechts einen

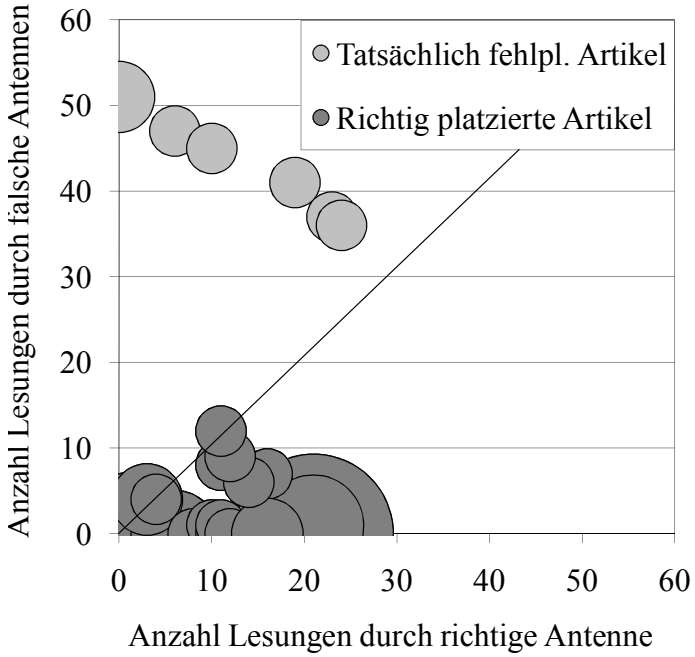

Abbildung 4.4: Nachfüllungs-Szenario, 25% Fehlplatzierungen

Artikel, der 24-mal durch die richtige Antenne und 36-mal durch falsche Antennen erfasst wurde.

Die Artikel, die durch die richtige RFID-Antenne erfasst wurden, zeigen eine geringe Anzahl von Lesungen. Im Durchschnitt sind es ca. 12 Lesungen. Dies geschieht aufgrund der wechselnden Artikel, wegen Abverkäufen und wegen Nachfüllungen des Regals. Fehlplatzierte Artikel werden sehr oft erfasst, da sie nicht verkauft werden. Im Durchschnitt werden sie 44-mal von der Antenne ihres tatsächlichen Ortes erfasst.

RPCV identifiziert einen Cluster von fehlplatzierten Artikeln. Artikel in diesem Cluster weisen im Durchschnitt 44 Lesungen durch falsche Antennen und ca. 8 Lesungen von der richtigen Antenne auf. Des Weiteren identifiziert RPCV einen Cluster von richtig platzierten Artikeln, die im Durchschnitt weniger als eine Lesung durch falsche Antennen und ca. 12 Lesungen durch die richtige Antenne aufweisen.

In diesem Experiment sind die Daten deutlich voneinander getrennt und RPCV erzeugt keine falschen Schätzungen. Das Verfahren MOST hingegen hat Schwierigkeiten in diesem Experiment: 8 richtig platzierte Artikel wurden genau so oft oder sogar öfter von falschen Antennen erfasst. Daher erzeugt MOST 8 falsche Schätzungen. Die Verfahren FIRST und LAST erzeugen 8 bzw. 3 falsche Schätzungen.

Tabelle 4.1: Genauigkeit von RPCV und von verwandten Arbeiten

Verfahren	*Precision*	*Recall*	F_1 *Maß*
RPCV	98,2%	96,4%	97,3%
MOST	63,4%	100,0%	77,8%
FIRST	56,0%	100,0%	71,8%
LAST	38,5%	92,9%	54,5%

4.4.3 Analyse der Genauigkeit mit realen Daten

In dieser Reihe von Experimenten wird die Genauigkeit von RPCV mit der von verwandten Arbeiten verglichen. Dafür werden die Daten der realen RFID-Installation verwendet.

Die Genauigkeit wird mit den häufig verwendeten Metriken *Precision*, *Recall* und dem F_1 Maß gemessen. Diese werden anhand der richtig positiven Schätzungen (engl. *True Positive*, TP), der falsch positiven Schätzungen (engl. *False Positive*, FP) und der falsch negativen Schätzungen (engl. *False Negative*, FN) berechnet. Richtig positive Schätzungen sind fehlplatzierte Artikel, die als solche erkannt werden. Falsch positive Schätzungen sind richtig platzierte Artikel mit falscher Schätzung und falsch negative Schätzungen sind fehlplatzierte Artikel, die der Schätzung nach richtig platziert sind. Precision ist als $\frac{TP}{TP+FP}$ und Recall als $\frac{TP}{TP+FN}$ definiert. Das F_1 Maß ist definiert als $\frac{2 \cdot precision \cdot recall}{precision+recall}$.

Tabelle 4.1 zeigt die Genauigkeiten. Von 299 Schätzungen lag RPCV nur in drei Fällen falsch. Diese werden in der nächsten Reihe von Experimenten untersucht. Das F_1 Maß von RPCV betrug 97%. Das nächst beste Verfahren, MOST, erzeugte 32 falsche Schätzungen. Das F_1 Maß betrug 78%. Folglich erzeugt RPCV eine Größenordnung weniger falsche Schätzungen als verwandte Arbeiten.

4.4.4 Falsche Schätzungen mit realen Daten

In diesem Abschnitt werden die Versuche analysiert, bei denen RPCV falsche Schätzungen produziert. Abbildung 4.5 zeigt einen Versuch aus dem statischen Szenario, bei dem 50% der Artikel fehlplatziert sind. Schätzungen mit diesen Daten sind schwierig, da drei richtig platzierte Artikel (zwei Kreise) sehr oft von falschen Antennen erfasst wurden. Diese Artikel erscheinen auf der Abbildung sehr nahe an den tatsächlich fehlplatzierten Artikeln. Wird RPCV auf diese Daten angewendet, schätzt es einen richtig platzierten Artikel als fehlplatziert (falsch positive Schätzung). In der Abbildung ist dieser Artikel durch ein gestricheltes Rechteck umgeben. Übrigens erzeugt MOST ebenfalls zwei falsch positive Schätzungen. Abbildung 4.6 zeigt einen Versuch aus dem Verkaufs-Szenario, bei dem 50% aller Artikel fehlplatziert sind. In diesem Versuch erzeugt RPCV zwei falsch negative Schätzungen. Beide Artikel sind durch ein gestricheltes Rechteck umgeben. MOST erzeugt ebenfalls zwei falsche Schätzungen bei diesem Experiment.

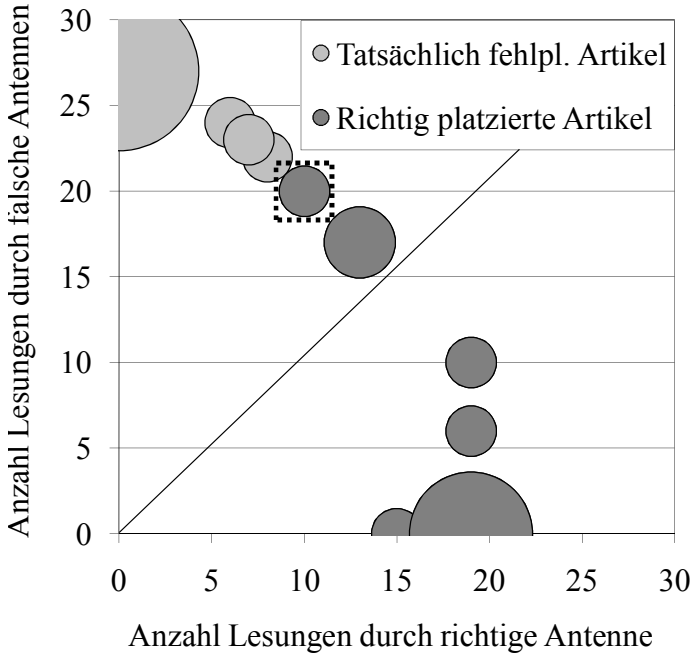

Abbildung 4.5: Statisches Szenario, 50% Fehlplatzierungen

Die falschen Schätzungen in beiden Experimenten sind auf die Funktionsweise des EM-Algorithmus zurückzuführen. Der Algorithmus versucht jede Dimension durch zwei Wahrscheinlichkeitsverteilungen abzudecken, wobei eine Verteilung aus jeder Dimension einen Cluster bildet. Dies soll anhand der Lesungen durch falsche Antennen verdeutlicht werden, siehe Abbildung 4.7. Die X-Achse zeigt die Anzahl von Lesungen durch falsche Antennen und die Y-Achse die entsprechende Anzahl von Vorkommnissen. Die Balken zeigen die Verteilung der RFID-Daten. Der Balken ganz rechts entspricht dem größten Kreis (links oben) in Abbildung 4.6. Jede Kurve zeigt die Normalverteilung von einem Cluster. RPCV gewinnt diese Kurven, indem es die Daten der Abbildung durch zwei Normalverteilungen abdeckt. Der EM-Algorithmus ordnet jeden Artikel der wahrscheinlichsten Normalverteilung zu, d.h. jeder Balken wird der Kurve zugeordnet, die den höchsten Wert an der X-Position des Balkens annimmt. Die Kurven sind verzerrt, da sie aufgrund des Tiefpass-Filters die Quadratwurzel der Werte abdecken. Die falschen Schätzungen aus Abbildung 4.6 sind mit einem gestrichelten Rechteck umgeben. Die Normalverteilung des Clusters der fehlplatzierten Artikel hat einen höheren Mittelwert der Anzahl von Lesungen durch falsche Antennen als die andere Normalverteilung. Sie hat einen sehr hohen Mittelwert und eine sehr geringe Standardabweichung. Die zwei falschen Schätzungen liegen nicht unter dieser Kurve. Der EM-Algorithmus wählt die wahrscheinlichsten Verteilungen, und dies führt zu falschen Schätzungen.

Das eben beschriebene Problem wurde nur in den vorgestellten Experimenten beobachtet. Es scheint also sehr selten vorzukommen. Dennoch sollte das Problem in zukünftigen Arbeiten weiter erforscht werden. Der EM-Algorithmus könnte beispielsweise mit anderen Clustering-Algorithmen kombiniert werden, die ein solches Verhalten nicht aufweisen.

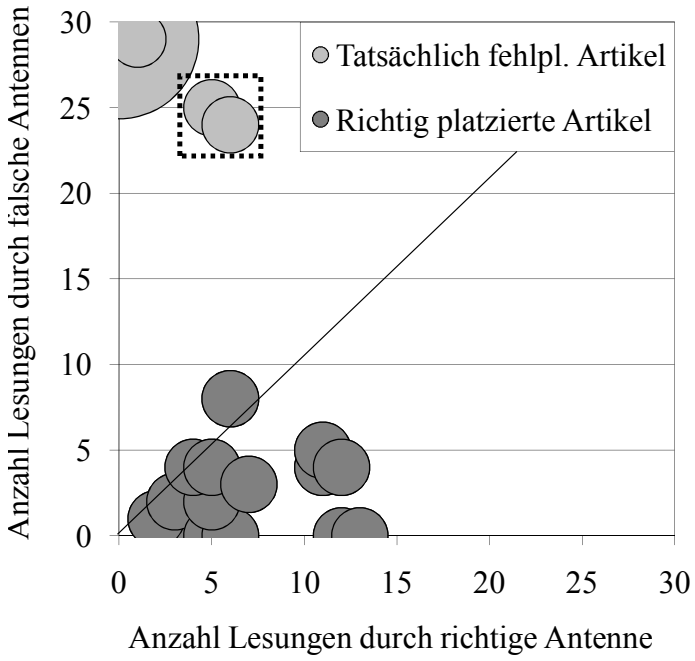

Abbildung 4.6: Verkaufs-Szenario, 50% Fehlplatzierungen

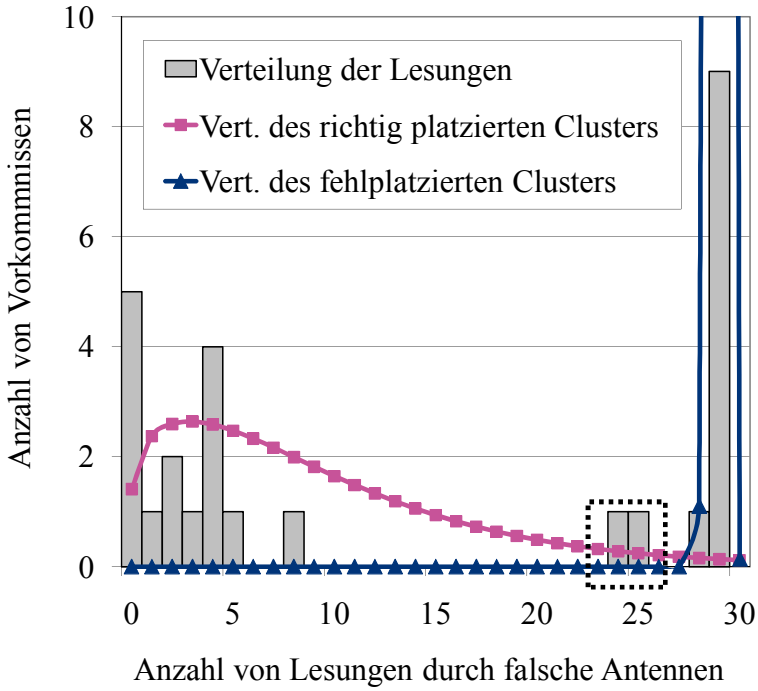

Abbildung 4.7: Verteilung der Lesungen von falschen Antennen

4.5 Experimente mit synthetischen Daten

In diesem Abschnitt werden anhand einer Simulation verschiedene Parameter untersucht, die RPCV beeinflussen. Durch die Simulation können Parameter unabhängig voneinander variiert und extreme Werte getestet werden. Insbesondere werden folgende Parameter variiert: Anzahl von RFID-Lesungen, Anzahl von Artikeln, Anzahl von Artikeln für jedes Lesemuster und die Wahrscheinlichkeiten, dass ein Artikel durch die richtige oder durch falsche RFID-Antennen erfasst wird.

4.5.1 Simulation von RFID-Lesungen für synthetische Daten

Viele Parameter beeinflussen die Anzahl von RFID-Lesungen und sie können nicht analytisch bestimmt werden. Aus diesem Grund werden verschiedene RFID-Lesemuster in einer Simulation generiert. Die Simulation basiert auf Monte-Carlo-Methoden [Kal07] und funktioniert wie folgt: Es werden Lesungen aller Antennen für ein gewisses Zeitfenster simuliert. Dies wird nacheinander mehrmals wiederholt. In jedem Zeitfenster wird mit einer festen Wahrscheinlichkeit p entschieden, ob die richtige Antenne einen Artikel erfasst und mit einer Wahrscheinlichkeit q, ob eine andere Antenne diesen Artikel erfasst. Falls eine Antenne einen Artikel erfasst, wird die Anzahl von Lesungen in diesem Zeitfenster bestimmt, indem eine Zahl aus der Normalverteilung $N(\mu = 0.50; \sigma^2 = 0.25)$ gezogen, mit 100 multipliziert und schließlich auf eine ganze Zahl konvertiert wird. Diese Simulation ist schnell und produziert Daten, die sehr ähnlich zu den Daten der RFID-Installation sind.

Basierend auf Beobachtungen der Daten aus der RFID-Installation werden drei Arten von Lesemustern simuliert: (1) Richtig platzierte Artikel, die oft von der richtigen und selten von falschen Antennen erfasst werden, (2) richtig platzierte Artikel, die oft von der richtigen und von falschen Antennen erfasst werden, und (3) fehlplatzierte Artikel, die oft von falschen Antennen und selten von der richtigen Antenne erfasst werden. Es ist anzumerken, dass Lesemuster (2) auch Artikel enthalten kann, die öfter von falschen als von der richtigen Antenne erfasst wurden, da die Anzahl von Lesungen aus einer Wahrscheinlichkeitsverteilung gezogen wird.

4.5.2 Analyse der Größe des Zeitfensters

In dieser Reihe von Experimenten wird der Einfluss der Größe des Zeitfensters auf die Genauigkeit von RPCV und von verwandten Arbeiten untersucht. Die Parameter der Simulation werden eingestellt, um die Daten der RFID-Installation nachzuahmen. Es wird ein statisches Szenario mit 30 Artikeln simuliert. In dem Szenario ist die Hälfte der Artikel richtig platziert und wird oft von der richtigen Antenne erfasst, ein Viertel der Artikel ist richtig platziert und wird oft von der richtigen und von falschen Antennen erfasst und das restliche Viertel ist fehlplatziert und wird oft von falschen Antennen erfasst. Die Genauigkeit wird nach dem ersten Lesezyklus, nach dem zweiten usw. berechnet, bis 100 Lesezyklen erreicht sind. Es werden die Werte des F_1 Maßes verglichen.

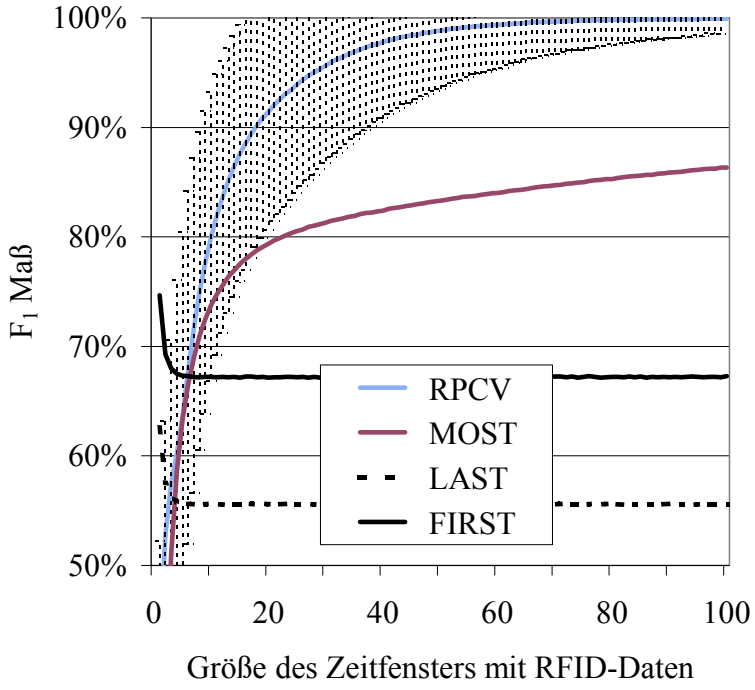

Abbildung 4.8: F_1 Maß für verschiedene Größen des Zeitfensters

Es wird erwartet, dass das F_1 Maß bei den Verfahren FIRST und LAST konstant bleibt, da ein statisches Szenario simuliert wird. Das F_1 Maß von RPCV und MOST sollte sich stabilisieren, da der Anteil von Lesungen durch jede Antenne stabiler wird, je mehr Daten gewonnen werden.

Die durchschnittlichen Ergebnisse von 1.000 Experimenten werden in Abbildung 4.8 gezeigt. Wie erwartet, verhält sich das F_1 Maß von FIRST und LAST konstant. Das F_1 Maß von RPCV und MOST steigt leicht mit der Größe des Zeitfensters an. Die Ergebnisse bei einem Zeitfenster von 30 sind ähnlich zu denen in Tabelle 4.1. Die gestrichelte Fläche zeigt die Standardabweichung von RPCV. Nach 18 Lesezyklen ist das durchschnittliche Ergebnis von RPCV plus der Standardabweichung besser als das zweitbeste Verfahren. Die Standardabweichung der anderen Verfahren wird nicht geplottet, um die Lesbarkeit der Abbildung nicht zu erschweren. Die Standardabweichung von MOST ist annähernd konstant bei 7 Prozentpunkten. Die von FIRST ist konstant bei ca. 11 Prozentpunkten während die von LAST konstant bei ca. 6 Prozentpunkten liegt.

Diese Versuche zeigen, dass RPCV gute Ergebnisse bei einem kleinen Zeitfenster liefert, und dass die Genauigkeit, also das F_1 Maß, schnell 100% erreicht. Aus diesem Grund ist RPCV in Szenarien anwendbar, bei denen Planogramm-Einhaltung in sehr kleinen Zeitabständen gemessen werden muss.

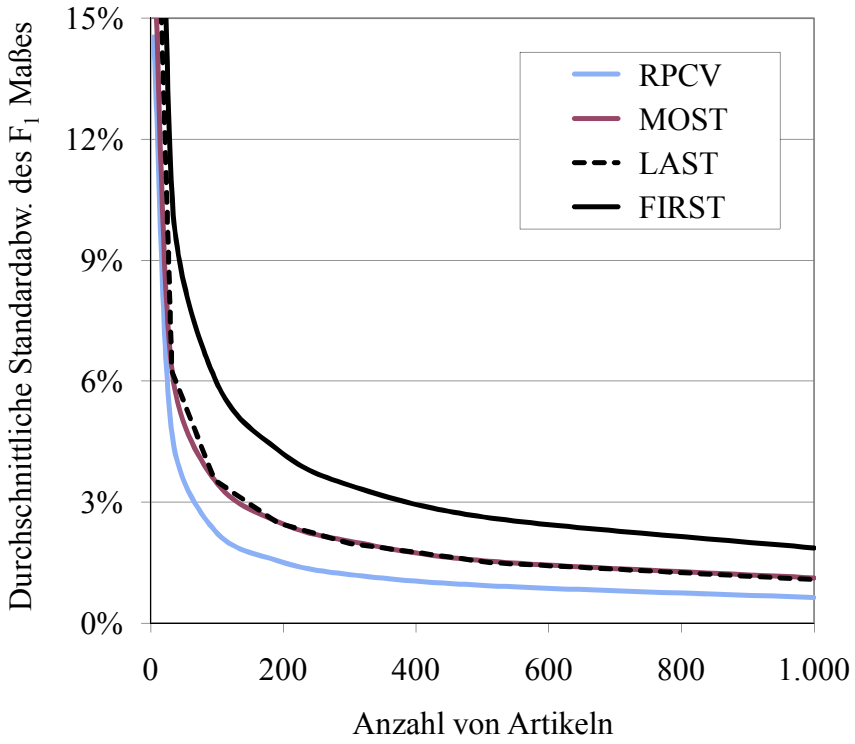

Abbildung 4.9: Standardabweichung vs. Anzahl von Artikeln

4.5.3 Einfluss der Gesamtanzahl von Artikeln auf die Schätzungen

Nun soll analysiert werden, wie die Gesamtanzahl von Artikeln die Schätzungen beeinflusst, und ob eine minimale Anzahl von Artikeln notwendig ist, damit RPCV funktioniert. Obwohl ein Einzelhändler im Durchschnitt zwischen 8 und 13 Artikel eines Produkttyps in einem Regal platziert, wird in dieser Reihe von Experimenten die Anzahl von Artikeln zwischen 4 und 1.000 variiert. Ein komplettes Regal kann weniger als 1.000 Artikel beherbergen, d.h. es handelt sich hier um einen extremen Fall. Für jede Anzahl von Artikeln und für jede Größe des Zeitfensters werden 1.000 Experimente durchgeführt. Die Größe des Zeitfensters wird zwischen 1 und 100 variiert.

Es wird erwartet, dass die Ergebnisse von RPCV nicht von der Anzahl von Artikeln abhängen. Zudem wird erwartet, dass die Standardabweichung steigen wird wenn die Anzahl von Artikeln sinkt, da RPCV weniger RFID-Daten als Eingabe haben wird.

Die Erwartungen werden erfüllt: Die durchschnittlichen Ergebnisse von RPCV variieren nicht mit der Anzahl von Artikeln. Um die Standardabweichung zu analysieren, wird der Durchschnitt von allen Experimenten für jede Anzahl von Artikeln gebildet. Die Ergebnisse sind in Abbildung 4.9 zu sehen. Die Standardabweichung liegt bei ca. 15% für 4 Artikel und sinkt bei einer steigenden Anzahl von Artikeln. Die Standardabweichung von RPCV ist geringer als die von verwandten Arbeiten. Da die durchschnittlichen Ergebnisse von RPCV nicht mit der Anzahl von Artikeln variieren, ist es in Szenarien mit sehr wenigen Artikeln eines Produkttyps

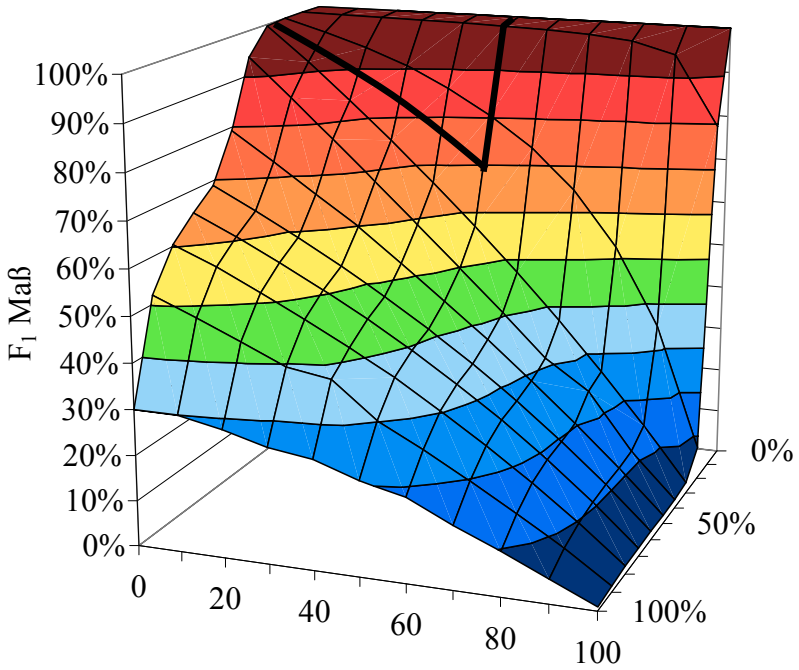

Abbildung 4.10: F_1 Maß im ungünstigsten Fall

anwendbar.

4.5.4 Ungünstigster Fall: Analyse der Lesewahrscheinlichkeit

In dieser Reihe von Experimenten wird die Anzahl von Artikeln in jedem Lesemuster variiert, während die Anzahl von Artikel konstant bei 100 gehalten wird. Des Weiteren werden die Wahrscheinlichkeiten, dass ein Artikel von der richtigen oder von falschen Antennen erfasst wird, getrennt voneinander variiert, d.h. die Werte von p und q werden zwischen 0 und 1 variiert. Ziel dieser Versuche ist es die Situationen zu finden, die RPCV am meisten herausfordern.

Beim Variieren der Anzahl von Artikeln in jedem Lesemuster wurden die schlechtesten Ergebnisse erzielt, als die Anzahl von Artikeln, die sehr oft durch die richtige und durch falsche Antennen erfasst wurden, sehr hoch war. Betrachtet man die Wahrscheinlichkeiten p und q, dass Artikel erfasst werden, gemeinsam, so haben diese einen geringen Einfluss auf die durchschnittliche Qualität von RPCV. Sie beeinflussen jedoch die Standardabweichung, da geringe Wahrscheinlichkeiten einer Lesung in weniger Daten resultieren, ähnlich zu den Versuchen mit kleineren Zeitfenstern.

Als nächstes wurden diese Wahrscheinlichkeiten für die richtige und für falsche Antennen unabhängig voneinander variiert. Es hat sich gezeigt, dass die Differenz zwischen beiden Wahrscheinlichkeiten den größten Einfluss auf die Qualität von RPCV ausübt. Die Ergebnisse werden in Abbildung 4.10 dargestellt. Der Graph zeigt das durchschnittliche Ergebnis von 1.000

Experimenten mit 100 Artikeln und 100 Lesezyklen. Die X-Achse zeigt die Anzahl von Artikeln, die oft durch die richtige und durch falsche Antennen erfasst wurde. Diese werden als „schwierige Artikel“ bezeichnet. Für diese Artikel wird die Wahrscheinlichkeit, dass ein Artikel durch die richtige Antenne erfasst wird, auf $p = 50\%$ fixiert, und die Wahrscheinlichkeit q, dass ein Artikel durch falsche Antennen erfasst wird, wird in der Y-Achse variiert. Die restlichen Artikel werden gleichermaßen zwischen den richtig platzierten Artikeln aufgeteilt, die oft von der richtigen Antenne erfasst werden, und fehlplatzierten Artikeln, die oft von falschen Antennen erfasst werden.

Im Allgemeinen liegt das F_1 Maß von RPCV unter folgenden Umständen über 70%: Weniger als die Hälfte aller Artikel wird oft durch die richtige und durch falsche Antennen erfasst. Und die Wahrscheinlichkeit q, dass Artikel durch falsche Antennen erfasst werden, beträgt die Hälfte von p, also 25% in der Abbildung. Die fetten Linien zeigen dieses Intervall in der Abbildung. Das Intervall entspricht den üblichen Eigenschaften von RFID-Lesern, d.h. Parameter, unter denen RPCV nicht gut funktioniert, entsprechen unrealistischen Eigenschaften von RFID-Lesern.

4.5.5 Performanz und Skalierbarkeit

In diesem Abschnitt werden die Performanz und Skalierbarkeit von RPCV untersucht, indem die Laufzeit von RPCV mit der Laufzeit von verwandten Arbeiten verglichen wird. In dieser Reihe von Experimenten werden Datenbanken mit verschiedenen Größen verwendet. Obwohl ein Einzelhändler im Durchschnitt zwischen 8 und 13 Artikel eines Produkttyps in einem Regal platziert, wird in dieser Reihe von Experimenten mit 100 Artikeln experimentiert. Die Anzahl von Lesungen für jeden Artikel wird zwischen 10 und 100.000 variiert. Somit hat die größte Tabelle in dieser Reihe von Experimenten 10.000.000 Zeilen. Abbildung 4.11 zeigt diese Ergebnisse. Jede Zahl zeigt den Durchschnitt aus 20 Durchläufen. Die Laufzeit von RPCV liegt zwischen 140ms und 160ms. Obwohl RPCV langsamer als verwandte Arbeiten ist, ist es trotzdem schnell bei Schätzungen mit einer sehr großen Anzahl von Lesungen. Des Weiteren benötigt RPCV weniger Daten, um gute Schätzungen zu erzeugen, und produziert eine Größenordnung weniger falsche Schätzungen als verwandte Arbeiten.

In der nächsten Reihe von Experimenten wird die Gesamtzahl von Lesungen in jeder Datenbank konstant gehalten und die Anzahl von Artikeln wird zwischen 100 und 1.000 variiert. Es soll gemessen werden, wie RPCV in extremen Situationen skaliert, da es sehr unwahrscheinlich ist, dass ein Einzelhändler mehr als 100 Artikel eines Produkttyps in einem Regal platziert. Die Ergebnisse werden in Abbildung 4.12 präsentiert. Die Laufzeiten von FIRST und LAST betragen ca. 100ms und werden durch eine wachsende Anzahl von Artikel nicht beeinflusst. Bei 100 RFID-Lesungen ist die Laufzeit von MOST ca. 125ms und wächst annähernd linear auf 182ms bei 1.000 Lesungen. RPCV verhält sich ebenfalls annähernd linear: 124ms für 100 Lesungen und 291ms für 1.000 Lesungen. Die Tatsache, dass sich das Wachstum annähernd linear verhält, wird durch weitere Daten untermauert, die zu Gunsten einer besseren Lesbarkeit in der Abbildung nicht dargestellt werden.

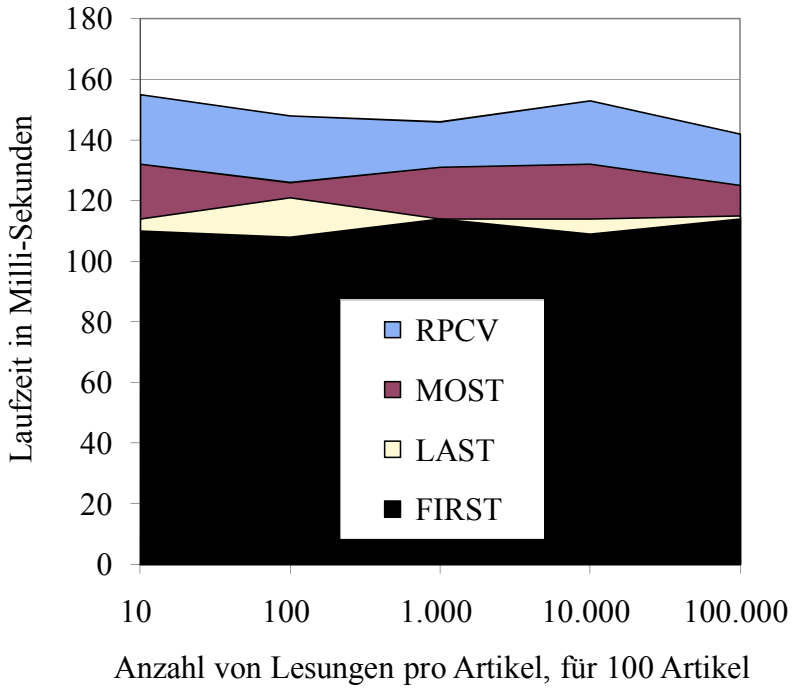

Abbildung 4.11: Laufzeit in Abhängigkeit der Anzahl von Lesungen

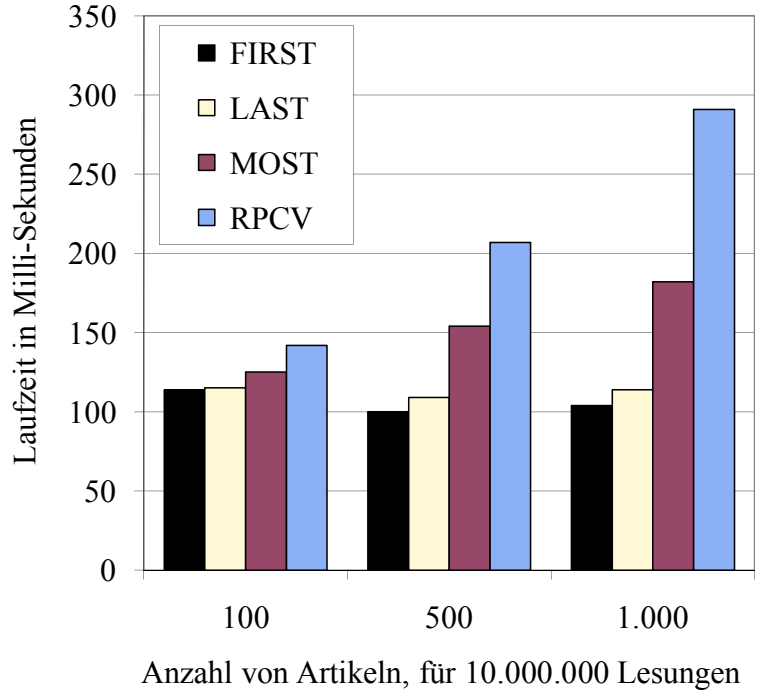

Abbildung 4.12: Laufzeit in Abhängigkeit der Anzahl von Artikeln

4.6 Zusammenfassung

In diesem Kapitel wurde das Verfahren RPCV vorgestellt. Es entscheidet, ob Artikel, die von mehr als einer RFID-Antenne erfasst wurden, sich an dem richtigen Ort in einem Regal befinden. RPCV basiert auf der Beobachtung, dass Artikel desselben Produkttyps die elektromagnetischen Wellen der RFID-Antennen auf ähnliche Art und Weise beeinflussen und daher ähnliche Lesemuster aufweisen. RPCV betrachtet alle Artikel eines Produkttyps gemeinsam und entscheidet anhand eines Clustering-Algorithmus, welche Artikel richtig und welche fehlplatziert sind. Experimente mit einer RFID-Installation und mit synthetischen Daten zeigen, dass RPCV eine Größenordnung weniger falsche Schätzungen als verwandte Arbeiten liefert, dass es schnell ist und weniger RFID-Daten benötigt, um gute Schätzungen zu liefern.

Kapitel 5

Abfrageoptimierung mittels materialisierter Sichten

Durch den Einsatz von RFID fallen sehr große Datenmengen an, da durch die automatisierte Erfassung Daten an mehreren Orten und in kleineren Zeitabständen erfasst werden können. Neue Arten der Datenverarbeitung sind erforderlich, um solche Datenmengen in Unternehmenssoftware effizient zu verarbeiten und somit integrieren zu können.

In diesem Kapitel wird ein Verfahren vorgestellt [WBHB09], um komplexe Abfragen auf verteilte und große Datenmengen zu optimieren. Die Optimierung erfolgt durch die Wahl von geeigneten materialisierten Sichten (MS). Eine MS ist ein (Zwischen-)Ergebnis einer oder mehrerer Abfragen, das vorberechnet und gespeichert wird. Gespeicherte Ergebnisse können verwendet werden, um zukünftige Abfragen schneller zu beantworten. Allerdings müssen MS aktualisiert werden, wenn sich die unterliegenden Relationen ändern, wobei eine Vielzahl von Parametern berücksichtigt werden muss. Da die Wahl von geeigneten MS NP-vollständig ist [Gup97], wird das Problem durch die Verwendung mehrere Heuristiken gelöst, die den Lösungsraum so stark verkleinern, dass ein genetischer Algorithmus angewendet werden kann. Obwohl das Verfahren für den Einsatz von RFID im Einzelhandel entwickelt wurde, ist es allgemeingültig und kann für eine sehr breite Kategorie weiterer Probleme verwendet werden.

5.1 Materialisierte Sichten in verteilten Szenarien

Materialisierte Sichten (MS) sind eine bekannte Methode, um komplexe Datenbankabfragen zu optimieren. MS können die Performanz eines Datenbankmanagementsystems (DBMS) erhöhen, indem die Neuberechnung von teuren Datenbankoperationen vermieden wird. In verteilten DBMS können (Zwischen-)Ergebnisse in der Nähe des abfragenden Knotens materialisiert werden, und die Datenübertragung im Netzwerk verringern. Allerdings müssen MS aktualisiert werden, wenn sich die unterliegenden Relationen ändern, wobei viele Einschränkungen berücksichtigt werden müssen, wie z.B. die Ressourcen der einzelnen Knoten. Daher

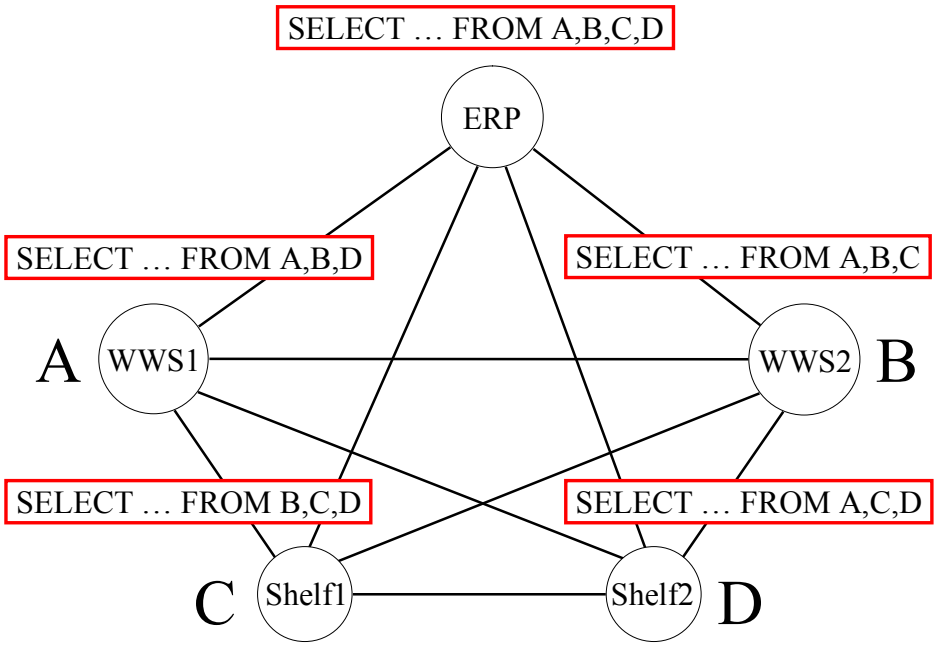

Abbildung 5.1: Einfaches Beispielszenario einer verteilten Datenbank

stellt die Wahl von geeigneten MS in verteilten Szenarien ein kompliziertes Problem dar. Das folgende Beispiel soll dies verdeutlichen:

Beispiel 5.1: Wie in Kapitel 2 dargestellt, möchte ein Einzelhändler RFID-Technologie einsetzen. Abbildung 5.1 spiegelt in einem einfachen Szenario die IT-Infrastruktur dieses Einzelhändlers wieder. Das Szenario besteht aus verschiedenen Rechnerknoten: Ein *Enterprise Resource Planning* (ERP) System, zwei Warenwirtschaftssysteme (WWS), die Tabellen A und B speichern, und zwei *Smart-Shelves* mit den Tabellen C und D. ERP, WWS und die Smart-Shelves speichern unterschiedliche Daten. Eine typische Abfrage in einem solchen Szenario verknüpft Daten von verschiedenen Rechnerknoten, z.B. „Verknüpfe den Produktkode der blauen Hosen aus dem ERP mit den Inventurdaten von allen Filialen in einem Radius von 20km, und zähle die Anzahl von Tupeln, bei denen das Attribut *verkauft* gleich *nein* ist“. Viele Parameter müssen bei der Optimierung solcher Abfragen berücksichtigt werden. Falls die Tabellen oft aktualisiert werden, sollten keine Sichten materialisiert werden. Bei schnellen Netzwerkverbindungen sollte das zentrale ERP die Tabellen der WWS materialisieren. Ansonsten könnten die WWS die Produktkodes des ERP materialisieren, usw.

Verfahren zur Wahl von geeigneten MS für eine gegebene Menge an Abfragen existieren seit längerem für zentrale DBMS [ACK+04, ACN00, BKS00, BDD+98, SDN98, YKL97, ZY99, ZLGD07, ZZL+04]. Diese Verfahren basieren auf monotonen Kostenmodellen. Allerdings entstehen beim Berücksichtigen von Aktualisierungen in verteilten Szenarien nicht-

monotone Kostenmodelle [BL03, GM99], so dass solche Verfahren sich nicht direkt auf verteilte Szenarien abbilden lassen. Alle Verfahren zur Wahl von MS in verteilten Szenarien, die dem Autor bekannt sind, treffen restriktive Annahmen, um monotone Kostenmodelle zu erreichen oder die Komplexität zu verringern, wie z.B. keine Aktualisierungen berücksichtigen [BL03, Gup97, HZ96, YGYL05] oder nur einen abfragenden Rechnerknoten erlauben [Gup97, GM99, HZ96, JGL07, LZB+07]. Verwandte Arbeiten werden im folgenden Abschnitt im Detail diskutiert.

Zusammenfassend lässt sich sagen, dass keine Verfahren existieren, um zu bestimmen, *welche Sicht* auf *welchem Knoten* materialisiert werden soll in *verteilten DBMS* mit *heterogenen Rechnerknoten* mit verschiedenen Einschränkungen bzgl. Rechenleistung, IO-Geschwindigkeit, und Netzwerkbandbreite, bei dem jeder Rechnerknoten verschiedene *Abfragen und Aktualisierungen* in *verschiedenen Häufigkeiten* stellt. Dieses Problem wird als *verteiltes Sichtenauswahlproblem* bezeichnet.

Dieses Problem ist aus verschiedenen Gründen schwierig: Erstens, ist das verteilte Sichtenauswahlproblem bekanntlich NP-vollständig [Gup97]. Zweitens, resultiert die Berücksichtigung von Aktualisierungen in verteilten Szenarien mit vielen heterogenen Knoten in nichtmonotone Kostenmodelle. Aus diesem Grund können keine Greedy-Algorithmen angewendet werden, die optimale Lösungen für Teilprobleme bestimmen, da der Vorteil der Materialisierung einer einzelnen Sicht von den bisherigen MS abhängen kann. Drittens, wächst die Anzahl der möglichen Sichten, die materialisiert werden können, exponentiell mit der Anzahl von Rechnerknoten und Abfragen und mit der Anzahl an Spalten, Verknüpfungsprädikate, Gruppierungen und Tabellen, die in jeder Abfrage referenziert werden. Aufgrund des sehr großen Lösungsraums können *Brute-Force*-Methoden, wie z.B. *Backtracking*, sowie biologisch-motivierte Ansätze, wie Ameisenalgorithmen oder genetische Algorithmen, nicht direkt angewendet werden. Des Weiteren müssen in der Praxis folgende nichtfunktionale Anforderungen erfüllt werden:

1. **Robuste Ergebnisse bei ungenauen Kosten:** Datenbankoptimierungen basieren auf Schätzungen der Kosten, die durch die Verarbeitung der Daten entstehen, wie z.B. Kosten für das Laden einer Tabelle, für Projektionen, für Verknüpfungen, Gruppierungen etc. Die Schätzung kann von den tatsächlichen Kosten abweichen, weshalb geeignete MS, trotz eines vereinfachten oder ungenauen Kostenmodells, gewählt werden müssen.

2. **Flexible Kostenmodelle:** Die IT-Infrastruktur im unternehmerischen Umfeld verfolgt verschiedene Optimierungsziele, wie z.B. Dienstgüte, Ausfallsicherheit oder Effizienz, und manche Rechnerknoten können für gewisse Arten von Abfragen maßgeschneidert sein. Aus diesem Grund muss ein Verfahren zur Auswahl von geeigneten MS mit verschiedenen Kostenmodellen funktionieren, ohne Einschränkungen wie Linearität oder Monotonie vorauszusetzen.

5.2 Verwandte Arbeiten

Die verwandten Arbeiten im Gebiet der Abfrageoptimierung mittels materialisierter Sichten lassen sich in drei Szenarien unterteilen: (1) Ein Rechnerknoten speichert alle Tabellen und Sichten werden auf diesem Rechnerknoten materialisiert (nichtverteiltes Szenario), (2) Tabellen sind auf verschiedene Rechnerknoten verteilt, Sichten werden auf einem Rechnerknoten materialisiert (semiverteiltes Szenario) und (3) Tabellen und Sichten können auf beliebigen Rechnerknoten im Netzwerk liegen (verteiltes Szenario). Das dritte Szenario entspricht dem Problem, das in diesem Kapitel bearbeitet wird.

Nichtverteiltes Szenario: Die Wahl von materialisierten Sichten für zentralisierte Szenarien ist ein gut erforschtes Problem. In solchen Szenarien ist die Speicherkapazität in der Regel der limitierende Faktor. [ACN00] beschreibt eine Heuristik für die automatische Wahl von materialisierten Sichten und Indizes für zentralisierte Datenbanken, ohne Aktualisierungen zu berücksichtigen. Es werden ein Modell und ein Algorithmus vorgeschlagen, um „teure“ Tabellen auszuwählen. Dieser Algorithmus wurde im Rahmen dieser Arbeit angepasst und verwendet, um teure Tabellen für verteilte Szenarien zu finden, vgl. Abschnitt 5.3.1. Erweiterungen des Verfahrens aus [ACN00] beinhalten horizontale Partitionierung [ACK+04] und die ausschließliche Materialisierung von häufig abgefragten Spalten [ZLGD07]. All diese Verfahren iterieren durch Teilmengen von Kandidaten für MS und bestimmen die finale Menge von MS aus optimalen Lösungen von Teilproblemen. Dies ist nur möglich, wenn die MS nach einem monotonen Kostenmodell bewertet werden. Ein monotones Kostenmodell ist ein Modell, in dem die Kostenersparnis durch die Materialisierung von zwei Sichten immer kleiner oder gleich der Summe der Kostenersparnis für die Materialisierung der einzelnen Sichten getrennt voneinander ist. Werden allerdings Aktualisierungskosten berücksichtigt, so könnte eine Sicht anhand einer anderen Sicht aktualisiert werden. Dann ist keine Monotonie gegeben, da die Kostenersparnis durch die Materialisierung beider Sichten größer als die Summe der Kostenersparnis für die Materialisierung der einzelnen Sichten wäre.

Es existieren Verfahren, die Aktualisierungskosten für zentralisierte Datenbanken berücksichtigen: [YKL97] beschreibt einen Algorithmus, um MS aus existierenden Ausführungsplänen (engl. *Query Plan*) zu gewinnen. Da erst die Ausführungspläne und dann die MS bestimmt werden, wird lediglich der Ausführungsplan optimiert und nicht das komplette System. Vergleichbare Algorithmen weisen ebenfalls Einschränkungen auf, die zu schlechten Lösungen führen können. Beispielsweise wird in [BPT97] nur eine repräsentative Untermenge aller Abfragen und Aktualisierungen berücksichtigt. In [ZZL+04] werden Kandidaten für MS durch Erzeugung mehrerer Ausführungspläne gewonnen, die nach „teuren“ Teilgraphen durchsucht werden.

Viele Verfahren wurden für *Data-Warehouses* entwickelt, die für spezifische Aktualisierungs-Strategien, Anforderungen an die Speicherkapazität oder mehrdimensionalen Datensätzen zugeschnitten sind: [BKS00] beschreibt, wie der Speicherplatz zwischen MS und Indizes automatisch aufgeteilt werden kann. [SDN98] schlägt eine Heuristik vor, die entscheidet, welche aggregierte Sichten für mehrdimensionale Datensätze vorberechnet werden sollen. [BDD+98] zeigt wie MS inkrementell aufgrund von Log-Daten aktualisiert werden können. In [MRSR01]

wird ein Verfahren für die Wartung von MS vorgestellt, das gemeinsame Teilausdrücke verschiedener MS ausnutzt. [ZLE07] entwickelt einen Algorithmus für eine „faule“ Wartung von MS, bei der MS nicht sofort aktualisiert werden müssen. Dadurch sinken die Aktualisierungskosten und Aktualisierungen können kumuliert werden. Verfahren, die entscheiden welche MS für jede Abfrage verwendet werden soll, werden in [BDD+98, PL00] präsentiert.

[ZY99] verwendet einen genetischen Algorithmus, um MS aus einem Genom auszuwählen, das einen kompletten Ausführungsplan für mehrere Abfragen aus einem Data-Warehouse kodiert. Allerdings kann dieses Verfahren nicht auf verteilte Szenarien erweitert werden, da der Ausführungsplan spezifisch für einen bestimmten Rechnerknoten ist und der Lösungsraum in einem verteilten Szenario zu groß wäre, um direkt durch einen genetischen Algorithmus evaluiert zu werden.

Semiverteiltes Szenario: Viele Ansätze wählen MS in semiverteilten Szenarien. [HZ96] stellt ein Modell vor, um materialisierte und virtuelle Sichten für verteilte Systeme zu bestimmen, bei denen die Sichten von einem Mediator gespeichert und gewartet werden. Das Modell berechnet die Antwortzeit von Abfragen, die Aktualität der MS und die Systemlast aufgrund von Parametern wie Abfragehäufigkeiten, Komplexität der Abfragen, Aktualisierungshäufigkeiten und Verzögerungen durch das Netzwerk. [Gup97] schlägt eine Heuristik vor, um MS in einem Data-Warehouse zu wählen. Die Heuristik basiert auf Sicht-Graphen. Die Autoren zeigen, dass das Problem NP-vollständig ist, und deren Lösung mindestens 63% des optimalen Nutzens erreicht. In [GM99] wird diese Lösung erweitert, um MS unter der Einschränkung zu wählen, das die Wartungszeit der MS minimal ist, statt der üblichen Einschränkung der begrenzten Speicherkapazität. Dies scheint das erste Verfahren zu sein, das eine solche Einschränkung berücksichtigt. Das Problem der Sichtenwahl wurde ebenfalls in föderierten Datenbanken untersucht: [LZB+07] analysiert welche Sichten in *Front-End*-Datenbanken und [JGL07] welche Sichten in *Back-End*-Datenbanken materialisiert werden sollen. Ein selbst-optimierender Algorithmus für Datenplatzierung (engl. *Data Placement*) für parallele Datenbanken wird in [LKO+00] vorgeschlagen. In [WOL99] wird ein Verfahren für die inkrementelle Wartung von MS in semiverteilten Szenarien präsentiert.

Verteiltes Szenario: Es existieren wenige Arbeiten, welche die Wahl von MS in verteilten Szenarien erforschen. [BL03] scheint die erste Veröffentlichung zu sein, die dieses Problem angeht. Die Konzepte werden für Data-Warehouses dargestellt, müssten aber ebenfalls auf relationale Datenbankmanagementsysteme anwendbar sein. Auf einem Rechnerknoten kann jeder Kandidat für eine MS in einem direkten azyklischen Graph eingeordnet werden, der zeigt welche Sichten von anderen Sichten ableitbar sind. Dieser Graph, der als Aggregationsgitter (engl. *Aggregation Lattice*) bezeichnet wird, wird schließlich nach guten MS durchsucht. Allerdings werden in diesem Verfahren nicht die Last der einzelnen Knoten und die Netzwerkübertragung berücksichtigt, was in verteilten Szenarien wichtig ist. Des Weiteren werden keine Aktualisierungen berücksichtigt, da dies ein nichtmonotones Kostenmodell zur Folge hätte. [YGYL05] verfolgt einen ähnlichen Ansatz für Data-Warehouses, der ebenfalls auf Graphen basiert. In der Veröffentlichung wird betont, dass keine effiziente Lösung für die Wahl von MS in verteilten Szenarien existiert.

Tabelle 5.1: Häufig verwendete Symbole

Symbole des Szenarios	
$f(q), f(u)$	Häufigkeit einer Abfrage oder Aktualisierung
N	Menge von Rechnerknoten
Q	Menge von Abfragen
U	Menge von Aktualisierungen

Symbole der Algorithmen	
C	Menge aller Kandidaten für MS
$cost_{query}(x, c_1, n_1, c_2, n_2, \cdots, c_i, n_i)$	Kosten einer Abfrage oder Aktualisierung x, wenn Sicht c_w auf Rechnerknoten n_w materialisiert wird
$cost_{table}(x, t)$	Kosten von Tabelle t in einer Abfrage oder Aktualisierung x
Q_{ts}	Menge von Abfragen, welche die Teilmenge von Tabellen $ts \in TS$ enthält
s_{pop}	Größe der Population des genetischen Algorithmus
TS	Finale Menge von Tabellen-Teilmengen aus dem Tabellen-Selektions-Algorithmus
θ	Schwellenwert zur Bestimmung von relevanten Teilmengen von Tabellen
θ_{expTab}	Schwellenwert für „teure“ Tabellen
θ_{sim}	Schwellenwert für die Ähnlichkeit von Populationen
U_{ts}	Menge von Aktualisierungen, welche die Teilmenge von Tabellen $ts \in TS$ enthält

5.3 Wahl von materialisierten Sichten für verteilte DBMS

In diesem Abschnitt wird ein Verfahren vorgestellt, um das verteilte Sichtenauswahlproblem zu lösen. Es wird von einem verteilten DBMS ausgegangen, bestehend aus einer Menge N an Rechnerknoten, einer Menge Q an Abfragen, einer Menge U an Aktualisierungen und deren respektiven Häufigkeiten f, vgl. Tabelle 5.1. Der Lösungsraum ist sehr groß: Es sei m die Anzahl möglicher MS. Dann existieren $2^{m \cdot |N|}$ mögliche Lösungen für welche Sichten auf welchen Knoten materialisiert werden sollen. Des Weiteren kann m sehr groß sein: Für jede Abfrage $q \in Q$ können MS für alle Zusammensetzungen aus Spalten und Prädikaten in q erzeugt werden, also für jede Abfrage existieren mindestens $(2^{|Spalten|} - 1) \cdot (2^{|Prädikate|} - 1)$ MS. Verknüpfungsprädikate und Gruppierungen können m weiter vergrößern.

Das hier vorgestellte Verfahren verwendet Hintergrundwissen über Datenbankoptimierungen, um minderwertige MS aus dem Lösungsraum zu entfernen, so dass das Problem durch einen

genetischen Algorithmus [BBM93] gelöst werden kann. Die zu berücksichtigenden Spalten und Prädikate werden eingeschränkt, indem die syntaktische Ähnlichkeit von Abfragen berücksichtigt wird und auf „teure“ Tabellen fokussiert wird. Verfahren für die Wahl von MS bestehen üblicherweise aus drei Schritten: Auswahl von „teuren“ Tabellen, Erstellung von vielversprechenden MS und Auswahl der zu materialisierenden Sichten, vgl. [ACN00]. Das vorgestellte Verfahren hält sich an diese Struktur, schlägt aber neuartige Lösungen für die letzten zwei Schritte vor:

1. **Selektion von Tabellen:** Dieser Schritt analysiert die Arbeitslast (engl. *Workload*) des DBMS und wählt Teilmengen von Tabellen aus, die einen wesentlichen Einfluss auf die Ausführungskosten dieser Arbeitslast haben. Die Arbeitslast besteht aus allen Abfragen, Aktualisierungen und den entsprechenden Häufigkeiten. Für diesen Schritt werden die Metriken eines Verfahrens für zentrale DBMS [ACN00] angepasst, um verteilte Szenarien zu unterstützen.

2. **Generierung von Kandidaten:** Basierend auf den Teilmengen von Tabellen werden Kandidaten für MS erstellt. In diesem Schritt wird die Arbeitslast syntaktisch analysiert. Die Abfragen werden in Aggregationen, Verknüpfungsprädikate und sonstige Prädikate aufgeteilt. Schließlich werden Tabellen, Operatoren und Prädikate rekombiniert, um Kandidaten für MS zu erstellen. Die besten Kandidaten werden anhand eines Kostenmodells ausgewählt.

3. **Wahl der MS:** In diesem Schritt wird ein genetischer Algorithmus angewendet, um zu wählen (1) welche Kandidaten von MS tatsächlich materialisiert werden und (2) auf welchen Rechnerknoten.

Kostenmodell: Das vorgestellte Verfahren benötigt Kostenschätzungen von einem Kostenmodell. Wie bereits im vergangenen Abschnitt erwähnt, soll das Verfahren nicht von einem bestimmten Kostenmodell oder von Restriktionen wie Linearität oder Monotonie abhängen. Es kann auf jedes Kostenmodell aufgebaut werden, das folgende zwei Metriken anbietet:

$cost_{query}(x, c_1, n_1, c_2, n_2, \cdots, c_i, n_i)$ schätzt die Kosten einer Abfrage oder Aktualisierung x, wenn i Sichten vorhanden sind und jede Sicht c_w auf dem Rechnerknoten n_w materialisiert wird. Eine Sicht kann auch auf mehreren Rechnerknoten materialisiert werden. Die Angaben der Sichten c und der Rechnerknoten n ist optional. Werden beide Parameter ausgelassen, schätzt die Funktion die Kosten ohne Sichten und/oder Rechnerknoten zu betrachten.

$cost_{table}(x, t)$ schätzt die Kosten für den Zugriff auf eine einzelne Tabelle t bei einer Abfrage oder Aktualisierung x, also die Kosten für das Einlesen, für die Selektion und um Daten zu versenden.

Das Kostenmodell, das in der Auswertung zum Einsatz kommt, wird in Abschnitt 5.4.2 beschrieben.

5.3.1 Schritt 1: Selektion von Tabellen

In einem ersten Schritt werden diejenigen Tabellen ausgewählt, die einen hohen Einfluss auf die Laufzeit der Abfragen haben, z.B. weil sie eine hohe Kardinalität haben, weil sie oft abgefragt werden oder weil sie auf langsamen Rechnerknoten gespeichert sind. Zu diesem Zeitpunkt enthält der Lösungsraum $(2^x - 1)$ mögliche Kombinationen von relevanten Tabellen, wobei x die Anzahl von verschiedenen Tabellen darstellt, die von den Abfragen referenziert werden. Ein naiver Ansatz wäre es, durch jede Teilmenge von Tabellen zu iterieren und die relevanten Teilmengen von Tabellen zu filtern. Aber dieser Ansatz ist aufgrund der hohen Anzahl verschiedener Möglichkeiten nicht praktikabel. Stattdessen werden für das Verfahren die Metriken eines existierenden Algorithmus [ACN00] für die Selektion von Tabellen auf verteilte Szenarien erweitert.

Dieser Algorithmus funktioniert wie folgt (vgl. Algorithmus 5.1): Es wird von $i = 1, 2, ...$ bis zur Anzahl von referenzierten Tabellen iteriert. In jeder Iteration wird eine Menge S_i aus Teilmengen von Tabellen mit i Tabellen erzeugt und eine grobe Schätzung der Kostenersparnis berechnet, die durch die Materialisierung dieser Teilmengen von Tabellen zustande kommen würde (Zeilen 4-13). Falls die Ersparnis einer Teilmenge von Tabellen geringer als der Schwellenwert θ ist, wird die Teilmenge von Tabellen verworfen (Zeilen 7-12). Zum Schluss verwendet der Algorithmus eine genaue Metrik für die Bestimmung der Kostenersparnis, um minderwertige Teilmengen von Tabellen zu verwerfen (Zeile 14). Eine Evaluierung dieses Algorithmus und der Wahl seiner Parameter ist in [ACN00] zu finden.

```
 1: input Schwellenwert θ
 2: S_1 = {ts | ts ist eine Teilmenge von Tabellen mit Größe 1, und TS–Cost(ts) ≥ θ}
 3: i = 1
 4: while i < max(Anzahl Tabellen in einer Abfrage) and |S_i| > 0 do
 5:    i = i + 1
 6:    S_i = {}
 7:    G = {ts | ts ist eine Teilmenge von Tabellen mit Größe i, ∃s ∈ S_{i−1}, mit s ⊂ ts}
 8:    for all ts ∈ G do
 9:       if TS–Cost(ts) ≥ θ then
10:          S_i = S_i ∪ {ts}
11:       end if
12:    end for
13: end while
14: S = S_1 ∪ S_2 ∪ ... ∪ S_{max(Anzahl Tabellen in einer Abfrage)}
15: TS = {ts | ts ∈ S, TS–Weight(ts) ≥ θ}
16: output TS
```

Algorithmus 5.1: Tabellen-Selektions-Algorithmus

Der Algorithmus benötigt einen Schwellenwert θ für die Kosten und zwei Metriken: $TS–Weight(ts)$ berechnet die genauen Kosten für die Abfragen, welche die Teilmenge von Tabellen ts verwendet. Und $TS–Cost(ts)$ liefert eine grobe Schätzung der Kosten, wobei $TS–$

$Weight(ts) \geq TS\text{–}Cost(ts)$. Beide Metriken lassen sich wie folgt von dem vorgestellten allgemeinen Kostenmodell ableiten:

$$TS-Cost(ts) = \sum_{q \in Q_{ts}} cost_{query}(q) * f(q) \tag{5.1}$$

Q_{ts} ist die Menge der Abfragen, die Tabellen in ts referenzieren. $cost_{query}(q)$ berechnet die Kosten einer Abfrage q, wenn keine Sichten materialisiert werden, und $f(q)$ ist die Häufigkeit der Abfrage q.

In [ACN00] wird $TS\text{–}Weight(ts)$ definiert als die Summe der Kosten aller Abfragen, gewichtet durch die Kardinalität der Tabellen. Allerdings ist diese Definition für verteilte Szenarien nicht geeignet, da die Kardinalität nicht der einzige Parameter ist, der die Laufzeit in verteilten Szenarien beeinflusst. Daher wird $TS\text{–}Weight(ts)$ anhand von $cost_{query}(q)$ berechnet, um die Performanz der Rechnerknoten zu berücksichtigen, welche die Tabellen speichern:

$$TS-Weight(ts) = \sum_{q \in Q_{ts}} \left(cost_{query}(q) * f(q) * \frac{\sum_{t \in ts}(cost_{table}(q,t))}{\sum_{t \in T_q}(cost_{table}(q,t))} \right) \tag{5.2}$$

Q_{ts} ist die Menge der Abfragen, die Tabellen in ts referenzieren, T_q ist die Menge an nötigen Tabellen, um Abfrage q zu beantworten, und $f(q)$ ist die Häufigkeit der Abfrage q. $cost_{table}(q,t)$ berechnet die Kosten der Tabelle t in q, also die Kosten für Einlesen, Selektion und Datenversand. Diese Kosten sind spezifisch für den Knoten, der t hält. Diese Definition beinhaltet die Kardinalität der Tabellen, die Häufigkeit der Abfragen und die Performanz der beteiligten Rechnerknoten. Es ist anzumerken, dass die Werte von $TS\text{–}Cost(ts)$ und $TS\text{–}Weight(ts)$ bei einer steigenden Kardinalität von ts fallen, da weniger Abfragen alle Tabellen in ts verwenden.

Beispiel 5.2: Um die Funktionsweise des Tabellen-Selektions-Algorithmus zu verdeutlichen, wird der Algorithmus aus dem einfachen Szenario aus Abbildung 5.1 angewendet. Der Algorithmus beginnt mit den Teilmengen von Tabellen, die nur eine Tabelle enthalten, also $S_1 = \{\{A\}, \{B\}, \{C\}, \{D\}\}$, und iteriert durch Teilmengen mit zwei, drei und vier Tabellen. Es seien θ die Kosten, die entstehen, wenn eine Teilmenge von Tabellen von mindestens drei Abfragen verwendet wird. Dann ist das Ergebnis des Algorithmus $TS = \{\{A\}, \{B\}, \{C\}, \{D\}, \{A, B\}, \{A, C\}, \{A, D\}, \{B, C\}, \{B, D\}, \{C, D\}\}$.

5.3.2 Schritt 2: Generierung von Kandidaten für materialisierte Sichten

In diesem Schritt werden vielversprechende Kandidaten für MS anhand der Abfragen und der zuvor identifizierten Teilmengen von Tabellen ts generiert. Im Folgenden werden die Tabellen in ts als *Basistabellen* bezeichnet. Existierende Verfahren, wie z.B. [ZZL+04], gewinnen Kandidaten für MS durch Erzeugung mehrerer Ausführungspläne, die nach „teuren“ Teilgraphen

durchsucht werden. Die Ausführungspläne können aus Multi-Query-Optimierung [MRSR01, ZZL+04] oder durch Verschmelzung mehrerer Ausführungspläne [Gup97, GM99] gewonnen werden. Allerdings benötigen diese Methoden viele Aufrufe des Abfrageoptimierers, was bei komplexen Szenarien zeitintensiv ist. Des Weiteren wächst die Anzahl möglicher Alternativen exponentiell mit der Anzahl von Rechnerknoten, Operatoren und Tabellen. Daher erzeugen solche Verfahren sehr große Mengen von Kandidaten in komplexen Szenarien.

In dieser Arbeit wird ein anderer Ansatz verfolgt, der jede Teilmenge von Tabellen ts getrennt betrachtet und die dazugehörigen Abfragen syntaktisch in Selektion, Verknüpfungsprädikate, Gruppierungen und Aggregationen zerlegt. Die Kandidaten werden in zwei Schritten generiert: Im ersten Schritt werden zwei Basiskandidaten erzeugt. Ein Basiskandidat materialisiert die Verknüpfung aller Tabellen in ts und unterstützt die Schnittmenge aller Prädikate aller Abfragen in Q_{ts}, die Tabellen in ts referenzieren. Der andere Basiskandidat unterstützt zusätzlich Gruppierungen und Aggregationen. Im zweiten Schritt werden spezialisierte Kandidaten aus den Basiskandidaten erzeugt. Der Gedanke hinter dieser Vorgehensweise ist wie folgt: Die Verknüpfung aller Tabellen und die Schnittmenge aller Prädikate erzeugen einen Kandidaten mit einer sehr hohen Kardinalität. Wird dieser Kandidat in mehrere spezialisierte Kandidaten aufgeteilt, die nur wenige Tabellen und Prädikate unterstützten, kann das Abfragen und Aktualisieren dieser Kandidaten effizienter werden, da sie eine geringere Kardinalität aufweisen. Der Ansatz für die Spezialisierung der Kandidaten basiert auf einem Binärbaum, in dem die Prädikate des Basiskandidaten kontinuierlich geteilt werden. Der Basiskandidat unterstützt alle Abfragen in Q_{ts} und ist die Wurzel des Baums, dessen Blätter spezialisierte Kandidaten darstellen, die nur Tabellen und Prädikate einer einzelnen Abfrage unterstützen.

Erzeugung von Basiskandidaten

Es wird zwischen zwei Arten von Kandidaten unterschieden: *Allgemeine Basiskandidaten* und *aggregierte Basiskandidaten*. Allgemeine Basiskandidaten können von allen Abfragen in Q_{ts} verwendet werden. Aggregierte Basiskandidaten können nur von Abfragen mit denselben Aggregationen und denselben Selektionen auf die aggregierte Spalten verwendet werden. Für jede Teilmenge von Tabellen wird ein allgemeiner Basiskandidat und falls möglich, ein aggregierter Basiskandidat erzeugt. Die Kandidaten bestehen aus den folgenden Komponenten:

1. Der **allgemeine Basiskandidat** enthält:
   - Basistabellen: Die Verknüpfung aller Tabellen in ts.
   - Prädikate: Selektionen und Verknüpfungsprädikate, die für alle Abfragen in Q_{ts} gemeinsam sind und Tabellen in ts referenzieren. Falls Abfragen verschiedene Intervalle für dieselbe Spalte auswählen, wird die Vereinigung der Intervalle verwendet.
   - Projektionen: Alle Spalten in ts, die im Ergebnis einer Abfrage enthalten, oder für die Berechnung von Verknüpfungen, Gruppierungen oder Aggregationen von Abfragen in Q_{ts} notwendig sind.

2. Der **aggregierte Basiskandidat** beinhaltet den allgemeinen Kandidat und folgendes:

   - Gruppierungen: Jede Spalte in ts, die in einer Abfrage in Q_{ts} gruppiert wird, oder für die Berechnung von anderen Prädikaten notwendig ist, wird gruppiert.
   - Aggregationen: Jede Aggregation, die in einer Abfrage in Q_{ts} vorkommt und sich auf eine Tabelle in ts bezieht.

Beispiel 5.3 vermittelt einen Eindruck, wie Basiskandidaten erzeugt werden.

Spezialisierung von Kandidaten

Anstatt jeweils einen allgemeinen Basiskandidaten und einen aggregierten Basiskandidaten zu verwenden, kann es effizienter sein, mehrere spezialisierte Basiskandidaten mit sehr selektiven Prädikaten oder Aggregationen zu materialisieren. Solche MS weisen eine geringere Kardinalität auf und können deswegen effizienter abgefragt und aktualisiert werden. Anhand des allgemeinen Basiskandidats $c_{ts,Q}$ aus Beispiel 5.3 soll dies verdeutlicht werden: Der Basiskandidat unterstützt die Abfragen q_1 bis q_5. Ein Kandidat, der nur die Abfragen q_2, q_4, q_5 unterstützt, hat eine kleinere Kardinalität, da seine Prädikate selektiver sind: $\sigma_{\text{A.a1 = B.b1} \wedge \text{A.a2 > 100}}$. Ein Kandidat, der nur Abfrage q_5 unterstützt, würde folgende Prädikate enthalten: $\sigma_{\text{A.a1 = B.b1} \wedge \text{A.a2 > 100} \wedge \text{B.b2 = 10}}$.

Jetzt werden spezialisierte Kandidaten für MS erzeugt, die der genetische Algorithmus zusammen mit den Basiskandidaten auswerten soll. Ein naiver Ansatz wäre es, Kandidaten für alle Permutationen der Prädikate zu erzeugen. Dies ist allerdings nicht praktikabel, da die Anzahl von Möglichkeiten exponentiell mit der Anzahl von Prädikaten wächst. Der vorgestellte Ansatz nutzt die Ähnlichkeit von Abfragen. Da alle Abfragen in Q_{ts} dieselben Basistabellen in ts verwenden, können viele Abfragen Prädikate auf denselben Spalten enthalten. Aus diesem Grund sollte es möglich sein, gewisse Gruppen von Abfragen zu finden, die von einer sehr selektiven MS profitieren. Beispielsweise sind die Abfragen q_2, q_4, q_5 ähnlich, da sie von einer MS mit dem Prädikat $\sigma_{\text{A.a2 > 100}}$ und der Projektion $\pi_{\text{A.a1, A.a2}}$ profitieren. Die Ähnlichkeit zweier Abfragen wird definiert als die Kardinalität der Schnittmenge der Ergebnisse beider Abfragen. Andere Metriken für die Ähnlichkeit können ebenfalls angewendet werden.

Alle Prädikate der Abfragen in Q_{ts} werden in einem Binärbaum in Abhängigkeit von deren Ähnlichkeit organisiert, vgl. Beispiel 5.4. Die Wurzel wird durch einen Basiskandidaten gebildet, der die Werte aller Abfragen abdeckt, wobei die Blätter spezialisierte Kandidaten für einzelne Abfragen enthalten. Aus diesem Grund müssen nur $(2 \cdot |Q_{ts}| - 1)$ Kandidaten berücksichtigt werden. Der Binärbaum wird verwendet, um Kandidaten zu wählen, welche die Kosten der Ausführung der Abfragen minimieren. Allerdings dürfen MS nicht vergessen werden, die die Aktualisierung anderer MS vereinfachen. Beispielsweise können Zwischenergebnisse materialisiert werden, die für die Aktualisierung mehrerer MS hilfreich sind. Solche so genannte Stützkandidaten für MS werden in einem nachfolgenden Schritt erzeugt.

(1) Binärbaum aufbauen: Um einen Binärbaum aufzubauen, werden ein Basiskandidat und alle Abfragen in Q_{ts} der Wurzel des Baumes zugeordnet. Ausgehend von der Wurzel werden

Beispiel 5.3: Gegeben seien die Abfragen $Q = \{q_1, q_2, q_3, q_4, q_5\}$, die Tabellen A, B und C referenzieren:

q_1 : „SELECT A.a1, A.a2, B.b2, C.c2
FROM A, B, C,
WHERE A.a1 = B.b1 AND A.a1 = C.c1“

q_2 : „SELECT A.a1, A.a2, B.b2
FROM A, B
WHERE A.a1 = B.b1 AND A.a2 > 100“

q_3 : „SELECT A.a1, A.a2, A.a3, B.b2
FROM A, B
WHERE A.a1 = B.b1“

q_4 : „SELECT A.a1, SUM(B.b3)
FROM A, B
WHERE A.a1 = B.b1 AND A.a2 > 120
GROUP BY A.a1“

q_5 : „SELECT A.a1, A.a2, B.b2
FROM A, B
WHERE A.a1 = B.b1 AND A.a2 > 100 AND B.b2 = 10“

Eine Teilmenge von Tabellen $ts \subseteq TS$, die in dem vorherigen Schritt aus allen in Q verwendeten Tabellen gewonnen wurde, ist $ts = \{\text{A}, \text{B}\}$. Der allgemeine Basiskandidat ($c_{ts,Q}$) und der aggregierte Basiskandidat ($c'_{ts,Q}$) für die Teilmenge von Tabellen und Abfragen in Q sind:

$c_{ts,Q}$: „SELECT A.a1, A.a2, A.a3, B.b2, B.b3
FROM A, B
WHERE A.a1 = B.b1“

$c'_{ts,Q}$: „SELECT A.a1, SUM(B.b3)
FROM A, B
WHERE A.a1 = B.b1
GROUP BY A.a1“

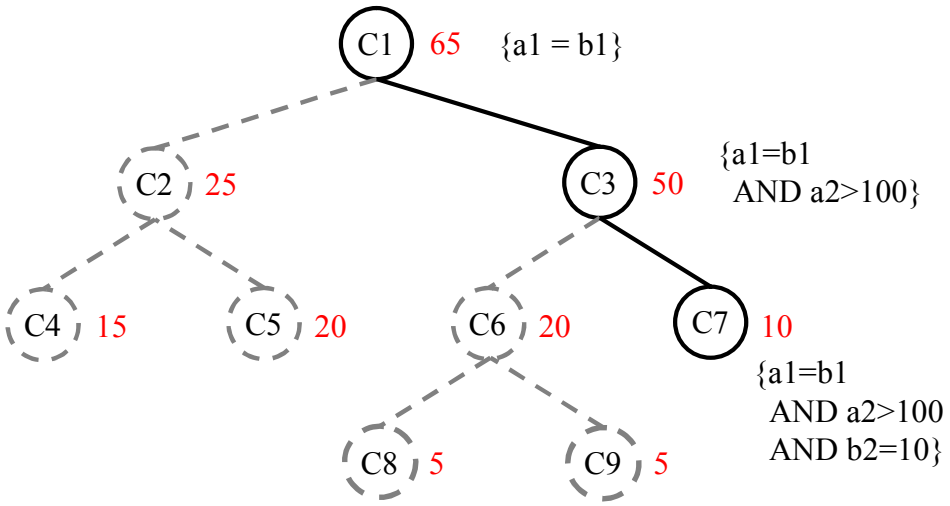

Abbildung 5.2: Binärbaum für die Abfragen aus Beispiel 5.3

Beispiel 5.4: Der Binärbaum in Abbildung 5.2 unterstützt die Basiskandidaten der Abfragen q_1 bis q_5 aus Beispiel 5.3. Kandidat C1 stellt den Kandidat $c_{ts,Q}$ aus Beispiel 5.3 dar. Das Beispiel zeigt, wie die Abfragen aufgrund ihrer Ähnlichkeit aufgeteilt werden. Die erste Teilung trennt Abfragen mit Selektionsprädikaten (q_2, q_4, q_5) von denen ohne (q_1, q_3). Die Beschriftungen neben den Kreisen zeigen die Kostenersparnis, die durch den Kandidat in jedem Knoten des Baumes hervorgerufen wird. Als Beispiel sei der Kandidat C2 genannt, der eine Kostenersparnis von 25 erzeugt. Die Kandidaten C4 und C5 des Kindknotens von C2 unterstützen dieselben Abfragen, deren Kostenersparnis jedoch mit $15 + 20$ höher ist als die von C2. Aus diesem Grund wird Kandidat C2 entfernt und C4 und C5 werden beibehalten. Die finale Menge von Kandidaten ist C3, C4 und C5.

Kindknoten erzeugt, indem die Menge von Abfragen des Vaterknotens auf die Kindknoten der Art aufgeteilt wird, dass jeder Knoten eine Menge von Abfragen mit ähnlichen Prädikaten und somit ähnlichen Ergebnissen enthält. Für jeden Kindknoten wird ein neuer Kandidat generiert, der für diese kleinere Menge an Abfragen spezialisiert ist. Als Ähnlichkeitsmaß wird die Kardinalität der Schnittmenge der Ergebnisse der Abfragen verwendet. Für jeden Knoten werden zwei Kindknoten erzeugt, bis die Blätter nur noch eine Abfrage unterstützen. Für einen Basiskandidaten, der $|Q_{ts}|$ Abfragen unterstützt, werden $2 \cdot |Q_{ts}| - 1$ Kandidaten generiert.

(2) Minderwertige Kandidaten entfernen: Nachdem der Binärbaum mit Kandidaten aufgebaut wurde, werden minderwertige Kandidaten entfernt. Als Kriterium wird die Netto-Kostenersparnis r_{net} verwendet. Diese entspricht der Höhe der Kosten, die durch die Materialisierung eines Kandidats c gespart werden können. Dieser Wert berechnet sich aus der Differenz der minimalen Aktualisierungskosten $cost_{upd}$ und der maximalen Kostenersparnis r_{max}:

$$r_{net}(c) = r_{max}(c) - cost_{upd}(c) \tag{5.3}$$

$$r_{max}(c) = \sum_{q \in Q_{ts}} f(q) * \max_{n \in N} (cost_{query}(q) - cost_{query}(q, c, n)) \tag{5.4}$$

$$cost_{upd}(c) = \sum_{u \in U_{ts}} f(u) * \min_{n \in N} (cost_{query}(u, c, n) - cost_{query}(u)) \tag{5.5}$$

Die maximale Kostenersparnis eines Kandidaten c ergibt sich aus der Summe der maximalen Kostenersparnis jeder Abfrage $q \in Q_{ts}$, d.h. für jede Abfrage q wird die Kostenersparnis berechnet als wäre c auf allen Knoten materialisiert (Gleichung 5.4). Die minimalen Aktualisierungskosten werden analog berechnet (Gleichung 5.5), wobei U_{ts} die Menge an Aktualisierungen auf die Menge von Tabellen ts angibt. Es werden die minimalen Aktualisierungskosten verwendet, da Aktualisierungen viel teurer als das Replizieren von Tabellen sind. Das Verfahren kann also eine Sicht auf dem Knoten mit den geringsten Aktualisierungskosten aktualisieren und die Sicht an die restlichen Knoten replizieren.

Die Knoten in jeder Ebene des Binärbaums werden beginnend bei der untersten Ebene (umgekehrte Breitensuche) durchlaufen. Dabei wird die Netto-Kostenersparnis von jedem Kandidaten berechnet. Falls die Summe der Kostenersparnis der Kandidaten der Kindknoten kleiner ist als die Kostenersparnis des Kandidaten des Vaterknotens, werden die Kindknoten entfernt. Andernfalls wird der Vaterknoten entfernt. In diesem Schritt verliert der Baum seine binäre Eigenschaft. Dieser Schritt ist abgeschlossen, wenn der Baum nur aus einer Ebene besteht.

(3) Stützkandidaten für MS hinzufügen: Die Kandidaten, die in den letzten beiden Schritten erzeugt wurden, verringern lediglich die Kosten von Abfragen. Es kann allerdings MS geben, welche die Aktualisierungskosten anderer MS senken. Beispielsweise könnte eine Sicht ein komplexes Zwischenergebnis mehrerer Sichten materialisieren: Eine MS, welche die Tabellen A, B und C verknüpft, und eine MS, die die Tabellen A, B und D verknüpft, könnten beide von einer MS über die Verknüpfung der Tabellen A und B profitieren. Dadurch müsste bei einer Änderung von A oder B deren Verknüpfung nur einmal berechnet werden. In diesem Schritt werden Kandidaten für solche MS erzeugt. Diese Kandidaten werden als Stützkandidaten bezeichnet.

Für jeden Kandidat, der zwei oder mehr Tabellen unterstützt, werden teure Basistabellen identifiziert. Diese sind Tabellen, die hohe Kosten bei der Aktualisierung einer Sicht verursachen. Die Stützkandidaten sollen solche Tabellen nicht abdecken, da diese sich häufig ändern oder teure Aktualisierungen verursachen. Eine Tabelle t wird als teuer für einen Kandidaten c gewertet, falls folgende Ungleichung gilt:

$$\sum_{u_t \in U_t} cost_{query}(u_t, c) > \left(\sum_{u \in U} cost_{query}(u, c) \right) * \theta_{expTab} \tag{5.6}$$

$cost_{query}(u, c)$ berechnet die Aktualisierungskosten des Kandidaten c für die Aktualisierung u. U ist die Menge von Aktualisierungen, die eine Basistabelle von c ändert, U_t ist die Menge von Aktualisierungen, die Tabelle t ändert, und θ_{expTab} ist ein Schwellenwert.

Für jeden Kandidat c werden die teuren Basistabellen aus der Menge von unterstützten Basistabellen entfernt. Aus dieser neuen Menge von Basistabellen und aus den unterstützten Abfragen wird ein Stützkandidat c_{sup} gebildet. Anschließend werden die Aktualisierungskosten von c mit und ohne c_{sup} verglichen. Falls die Kosten verringert werden, wird c_{sup} der finalen Menge von Kandidaten hinzugefügt.

Algorithmus für die Generierung von Kandidaten

An dieser Stelle wird beschrieben, wie die in diesem Kapitel vorgestellten Konzepte in einem Algorithmus vereinbart werden, vgl. Algorithmus 5.2. Der Algorithmus erzeugt eine Menge von Kandidaten für MS aus einer Menge von Teilmengen von Tabellen TS, einer Menge von Abfragen Q und einer Menge von Aktualisierungen U. Für jede Teilmenge von Tabellen $ts \in TS$ wird die Menge an Abfragen Q_{ts} und Aktualisierungen U_{ts} gewonnen, die ts enthalten (Zeilen 5, 6). Aus diesen Mengen werden Basiskandidaten erzeugt (Zeile 7). Für jeden Kandidaten werden mithilfe eines Binärbaums spezialisierte Kandidaten erzeugt (Zeile 10), wobei nur die besten Kandidaten beibehalten werden (Zeile 11). Danach werden zusätzliche Stützkandidaten erzeugt, die Aktualisierung anderer Kandidaten erleichtern (Zeile 13). Am Ende wird die resultierende Menge von Kandidaten zurückgegeben (Zeile 14).

```
1:  input Menge von Teilmengen TS, Menge von Abfragen Q und von Aktualisierungen U
2:  C_base = {}
3:  C_res = {} // Initialisiere Ergebnismenge
4:  for all (ts ∈ TS) do // Generiere Basiskandidaten
5:      Q_ts = selectReferencingQueries(ts)
6:      U_ts = selectReferencingUpdates(ts)
7:      C_base = C_base ∪ generateBaseCandidates(ts, Q_ts, U_ts)
8:  end for
9:  for all (c_base ∈ C_base) do // Für jeden Basiskandidat
10:     tree = buildTree(c_base) // Baue Binärbaum für Basiskandidat
11:     C_res = C_res ∪ cutTree(tree) // Entferne minderwertige Kandidaten
12: end for
13: C_res = C_res ∪ generateSupportingCandidates(C_res) // Füge Stützkandidaten hinzu
14: output C_res
```

Algorithmus 5.2: Algorithmus zur Erzeugung von Kandidaten

5.3.3 Schritt 3: Auswahl von materialisierten Sichten

In diesem Schritt werden die Kandidaten für MS, die im letzten Schritt gewonnen wurden, evaluiert. Insbesondere wird bestimmt, (1) welche Kandidaten materialisiert werden und (2)

auf welchen Knoten sie materialisiert werden. Da der Lösungsraum trotz der vorgestellten Heuristiken weiterhin exponentiell wächst, können nicht alle möglichen Lösungen evaluiert werden. Aus diesem Grund wird ein genetischer Algorithmus angewendet.

Genetische Algorithmen [BBM93] gehören zur Klasse von probabilistischen Optimierungsverfahren. Der Algorithmus organisiert mögliche Lösungen (*Individuen*) in *Populationen*. Neue Individuen werden durch zufällige *Mutationen* und durch *Rekombinationen* von Individuen erzeugt. Die nächste *Generation* besteht aus Individuen, die aufgrund ihrer Qualität (*Fitness*) *selektiert* wurden. Um das verteilte Sichtenauswahlproblem mit einem genetischen Algorithmus zu lösen, werden alle möglichen Konfigurationen in binäre Allokationsmatrizen kodiert. Jede Allokationsmatrix spezifiziert, welche Kandidaten für MS (Spalten) auf welchen Rechnerknoten (Zeilen) materialisiert werden. Der genetische Algorithmus wird für das verteilte Sichtenauswahlproblem verbessert, indem (1) die initiale Population mithilfe von Datenbankwissen generiert und (2) die Selektion auf das Problem zugeschnitten wird. Dadurch konvergiert der genetische Algorithmus für das Problem schneller als im Normalfall [EHM99].

Initiale Population

Obwohl der Lösungsraum in den vorherigen Schritten verkleinert wurde, indem vielversprechende Kandidaten für MS ausgewählt wurden, wächst die Anzahl von möglichen Lösungen weiterhin exponentiell mit der Anzahl von Rechnerknoten, auf denen Sichten materialisiert werden können. Im Normalfall beginnen genetische Algorithmen mit einer zufällig ausgewählten Population. Wenn die initiale Population allerdings nahe dem Optimum ist, wird der genetische Algorithmus in wenigen Iterationen konvergieren, da er eine große Anzahl von minderwertigen Lösungen nicht durchlaufen muss. Aus diesem Grund wird folgende Heuristik angewendet: *Ein Kandidat c wird bevorzugt auf einem Knoten materialisiert, der entweder eine oder mehr Basistabellen von c speichert (reduziert Aktualisierungskosten) oder eine Abfrage stellt (reduziert Abfragekosten)*. Es sei darauf hingewiesen, dass durch Mutationen weiterhin Sichten auf Rechnerknoten materialisiert werden können, die weder eine Basistabelle speichern noch Abfragen stellen.

Nach dieser Heuristik wird eine Menge N_c von vielversprechenden Rechnerknoten für jeden Kandidat c bestimmt:

$$N_c = \{n \mid n \in N, (\exists t, t \in ts_c, n \in N_t) \ \vee (\exists q, q \in Q_c, n = origin(q))\} \tag{5.7}$$

ts_c sind die Basistabellen von c, N_t ist die Menge von Rechnerknoten, die Tabelle t speichert, Q_c ist die Menge von Abfragen, die von c unterstützt wird, und $origin(q)$ liefert den Rechnerknoten, der die Abfrage q stellt.

Selektionsfunktion

Genetische Algorithmen setzen neue Populationen zusammen, indem sie die fittesten Individuen der vorigen Population auswählen und Mutationen und Rekombinationen hervorrufen.

Für das verteilte Sichtenauswahlproblem bedeutet dies, dass die MS nach deren Kostenersparnis ausgewählt werden. Allerdings kann dies dazu führen, dass die Population aus sehr ähnlichen Individuen besteht. Somit würde der genetische Algorithmus mehrere ähnliche Individuen parallel auswerten. Um die Anzahl von Iterationen zu verringern und die Population zu diversifizieren, wird bei der Selektion die Ähnlichkeit von Individuen berücksichtigt. Die Ähnlichkeit $s(i_1, i_2)$ von zwei Individuen i_1, i_2 wird als die Summe von identischen Bits in deren Allokationsmatrizen definiert:

$$s(i_1, i_2) = \left(\sum_{i=1}^{|N|} \sum_{j=1}^{|C|} x_{i,j} \right) / (|N| * |C|) \tag{5.8}$$

$$x_{i,j} = \begin{cases} 0, & \text{falls } A_1[i][j] \neq A_2[i][j] \\ 1, & \text{falls } A_1[i][j] = A_2[i][j] \end{cases} \tag{5.9}$$

A_1, A_2 sind die jeweiligen Allokationsmatrizen von i_1 und i_2, N die Menge an Rechnerknoten und C die Menge an Kandidaten für MS. Die durchschnittliche Ähnlichkeit eines Individuums i zu allen Individuen in der Population P wird wie folgt berechnet:

$$avgSimilarity(i, P) = \left(\sum_{i' \in P} s(i, i') \right) / \, |P| \tag{5.10}$$

```
1:  input Population P, Populationsgröße s_pop, Ähnlichkeits-Schwellenwert θ_sim
2:  P_sort = sortPopulationByFitness(P) // Sortiere Individuen absteigend nach Fitness
3:  P_next = {} // Initialisiere die nächste Population
4:  for all (individual i in P_sort) do
5:     if (avgSimilarity(i, P_next) < θ_sim) then
6:        P_next = P_next ∪ i // Füge Individuum i zu P_next hinzu
7:     end if
8:     if (|P_next| = s_pop) then
9:        break // Verlasse Schleife
10:    end if
11: end for
12: output P_next
```

Algorithmus 5.3: Selektionsfunktion

Die Selektionsfunktion $select(P, s_{pop}, \theta_{sim})$ wird in Algorithmus 5.3 beschrieben. Eingaben des Algorithmus sind die aktuelle Population P, die aus der Auswahl vergangener Individuen und aus mutierten und rekombinierten Individuen besteht, die maximale Größe der Population s_{pop} und ein Schwellenwert θ_{sim} für die Ähnlichkeit. Als erstes werden die Individuen in P absteigend nach deren Fitness sortiert (Zeile 2). Beginnend bei dem fittesten Individuum wird

durch alle Individuen iteriert (Zeile 4). Dabei wird geprüft, ob die durchschnittliche Ähnlichkeit eines Individuums zur nächsten Population kleiner als der Schwellenwert θ_{sim} ist (Zeile 5, Gleichung 5.10), und falls dies der Fall ist, wird das Individuum der nächsten Population hinzugefügt (Zeile 6). Der Algorithmus terminiert, wenn die maximale Größe der nächsten Population erreicht wurde (Zeile 8) oder wenn alle Individuen geprüft wurden. Danach wird die nächste Population zurückgegeben.

Die Wahl des Schwellenwertes θ_{sim} stellt einen Kompromiss zwischen Laufzeit und Qualität der Ergebnisse dar. Durch einen kleinen Schwellenwert werden kleine Populationen (P_{next}) erzeugt, die schnell ausgewertet werden können. Ein großer Schwellenwert hingegen resultiert in einer großen Anzahl von alternativen Lösungen und könnte daher bessere Ergebnisse herbeiführen. In den Experimenten wurde ein guter Kompromiss erreicht, wenn der Schwellenwert $(2 \cdot |C|)$ betrug.

Algorithmus für die Selektion von MS

```
1:  input Mutationswahrscheinlichkeit p_m, Rekombinationswahrscheinlichkeit p_c
2:  P_anc = generateInitialPopulation() // Generiere initiale Population
3:  while (Abbruchkriterium nicht erfüllt) do
4:      P_new = {}
5:      while (i < |P_anc|) do // Für jedes Individuum in P_anc
6:          if random() < p_m then // Mutiere i-tes Individuum und füge es P_anc hinzu
7:              P_new = P_new ∪ mutate(P_anc, i)
8:          end if
9:          if (random() < p_c) then
            // Rekombiniere i-tes mit zufälligem Individuum und füge Nachkomme P_anc hinzu
10:             j = (random() * |P_anc|)
11:             P_new = P_new ∪ crossover(P_anc, i, j)
12:         end if
13:         i = i + 1
14:     end while
15:     P_suc = select((P_anc ∪ P_new), s_pop, θ_sim)
16:     P_anc = P_suc
17: end while
18: output getBestIndividual(P_suc)
```

Algorithmus 5.4: Genetischer Algorithmus

Algorithmus 5.4 beschreibt, wie die Heuristiken in einen genetischen Algorithmus integriert wurden. Zuerst generiert der Algorithmus eine initiale Population (Zeile 2). Danach durchläuft der Algorithmus eine Schleife und berechnet neue Populationen bis das Abbruchkriterium erreicht ist (Zeile 3). Das Abbruchkriterium wird definiert als ($genCount \geq genThreshold$), wobei $genThreshold$ die maximale Anzahl von Iterationen ohne signifikante Verbesserungen angibt und $genCount$ solche Iterationen zählt. Neue Populationen werden durch Mutation

(Zeile 6), Rekombination (Zeile 11) und Selektion (Zeile 15) der Vorgänger-Population gebildet. In der nächsten Iteration der Schleife wird die neue Population zur Vorgängerpopulation (Zeile 16). Am Ende gibt $getBestIndividual(P_{anc})$ das fitteste Individuum zurück.

5.4 Experimente

In diesem Abschnitt wird die vorgestellte Lösung für das verteilte Sichtenauswahlproblem evaluiert, indem verschiedene Szenarien simuliert werden, bestehend aus einer Menge von Abfragen, einer Arbeitslast und einem verteilten Aufbau von Rechnerknoten mit verschiedenen Ressourcen. Genetische Algorithmen können keine optimalen Lösungen garantieren, allerdings zeigen die Experimente, dass die Lösungen „gut“ sind und von einem Datenbankadministrator nachvollzogen werden können. Die Ergebnisse des vorgestellten Verfahrens werden mit einem *Brute-Force*-Algorithmus verglichen, der den kompletten Lösungsraum nach der optimalen Lösung durchsucht. Des Weiteren wird gemessen, wie das Verfahren mit verfälschten Kostenschätzungen umgeht. Zudem wird gezeigt, dass die Variation der Ergebnisse sehr gering ist. Zum Schluss wird dargestellt, wie das Verfahren sehr großen Szenarien mit bis zu 400 Rechnerknoten, 1.000 Tabellen und ca. 3.000 Abfragen und Aktualisierungen bewältigt.

Die Wahl der Parameter des genetischen Algorithmus steht nicht im Fokus dieser Arbeit. Die Experimente wurden mit folgenden Parametern durchgeführt: 64 Individuen pro Population, einer Rekombinations- und Mutationswahrscheinlichkeit von jeweils 25% bzw. 15% und einem Abbruch, falls die letzten vier Iterationen eine geringere Kostenersparnis als 0,1% erzielten. Eine ausführliche Analyse der Parameter von genetischen Algorithmen kann in [EHM99] gefunden werden.

5.4.1 Versuchsaufbau

Für die Versuche wurden die Algorithmen aus Abschnitt 5.3 in Java implementiert. Es wurde ein verteiltes Datenbankszenario inklusive Rechnerknoten, Datenbanktabellen und einem kostenbasierten Datenbankoptimierer simuliert. Das Kostenmodell für den Optimierer wird in dem folgenden Abschnitt beschrieben.

Der Optimierer generiert Ausführungspläne für die Abfragen in dem verteilten Datenbankszenario, d.h. der Optimierer entscheidet welche MS in welchen Abfragen verwendet werden. Des Weiteren berechnet der Optimierer die Abfrage- und Aktualisierungskosten, die von den beschriebenen Algorithmen verwendet werden. Die Implementierung des Optimierers basiert auf bekannten Verfahren [Cha98, ML86], die für MS und für verteilte Szenarien erweitert wurden. Der Optimierer folgt dem Paradigma der dynamischen Programmierung und berechnet Ausführungspläne aus einer eingeschränkten Menge von Operatoren. Insbesondere werden Selektionen, Projektionen, Verknüpfungen und Gruppierungen (SPJG) ohne verschachtelte Unterabfragen unterstützt. Um einen verteilten Ausführungsplan zu berechnen, bestimmt der Optimierer die Reihenfolge der Verknüpfungen und welche Rechnerknoten welche Operatoren ausführen. Falls MS vorhanden sind, prüft der Optimierer, ob Zwischenergebnisse aus

einer MS gewonnen werden können. Der Optimierer schätzt die Kardinalität der Zwischenergebnisse basierend auf der Annahme, dass die Werte unabhängig und gleichverteilt sind. Für Aktualisierungen wird von einer sofortigen und inkrementellen Aktualisierung ausgegangen.

Es wird mit einem einfachen und mit einem realen Szenario experimentiert.

Einfaches Szenario: Es wird ein einfaches Szenario verwendet, wie in Abbildung 5.1 beschrieben, um die Funktionsweise des Algorithmus zu verdeutlichen. Das Szenario besteht aus fünf Rechnerknoten mit denselben Eigenschaften hinsichtlich Rechenleistung, IO-Geschwindigkeit und Netzwerkbandbreite. Alle Tabellen haben dieselben Struktur, Kardinalität und Häufigkeit der Aktualisierungen.

Reales Szenario: Dieses Szenario basiert auf dem in Abschnitt 2.3.3 eingeführten Einzelhandelsszenario. Das Szenario besteht aus einem ERP-System und variiert zwischen 20 und 100 Filialen. Für jede Filiale werden dem Szenario 4 Rechnerknoten, 10 Tabellen, 15 Abfragen und 14 Aktualisierungen hinzugefügt. Im Gegensatz zu dem einfachen Szenario sind die Arbeitslast und die Eigenschaften von jedem Knoten unterschiedlich. Das Szenario und die Testdaten stammen mit Ausnahme der RFID-Daten, die aufgrund eines durchgeführten Feldversuches geschätzt wurden, von einem großen Einzelhändler.

5.4.2 Kostenmodell

Für die Experimente wurde ein zu [ML86] ähnliches Kostenmodell verwendet, das auf verteilte Szenarien erweitert wurde. Für jeden relationalen Operator wurde die einfachste Implementierung angenommen, wie z.B. sequentielle Tabellendurchläufe und verschachtelte Schleifen bei Verknüpfungen. Das Kostenmodell setzt konsistente Daten voraus, d.h. falls die Daten in einer MS nicht aktuell sind, müssen diese vor eine Abfrage aktualisiert werden. Die Kosten einer Abfrage werden berechnet als die Summe der benötigten relationalen Operatoren. Dazu werden die folgenden Größen verwendet: Die Rechenleistung (CPU) und die IO-Geschwindigkeit (IO) von jedem Knoten sowie die Netzwerkbandbreite $NET_{N1,N2}$ zwischen zwei Rechnerknoten $N1$ und $N2$. Des Weiteren wird auf Katalogdaten zurückgegriffen: $iCard$ ist die Kardinalität einer Tabelle, iW die gesamte Breite eines Tupels, also die Anzahl von Bytes in einer Zeile und $compW$ ist die aufsummierte Breite der Spalten, die durch Prädikate abgefragt werden. $aggW$ und $groupW$ sind die aufsummierten Breiten der Spalten, die aggregiert oder gruppiert werden. Diese Größen werden wie folgt in das Kostenmodell integriert:

Selektion: Tabelle laden, inklusive Projektion und Selektion.

$$cost_{sel} = (IO * iW * iCard) + (CPU * compW * iCard)$$

Verknüpfung: Verschachtelte Schleife, Evaluierung der Verknüpfungsprädikate.

$$cost_{join} = (CPU * compW * iCard_1 * iCard_2)$$

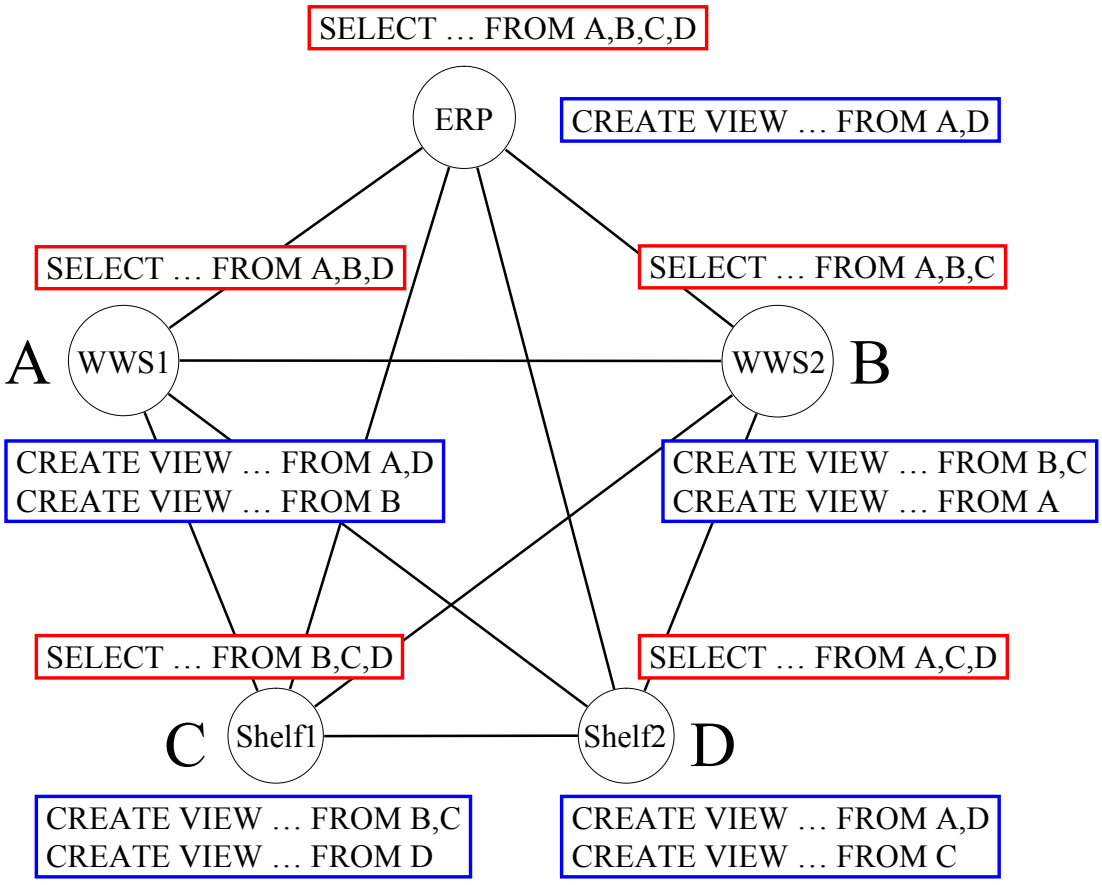

Abbildung 5.3: Materialisierte Sichten für das einfache Szenario

Gruppierung, Aggregation: Sortierung der Eingabe und sequentieller Vergleich oder Aggregation.
$cost_{agg} = (CPU{*}groupW{*}iCard{*}log_2(iCard))+(CPU{*}iCard{*}(groupW+aggW))$

Daten versenden: Kosten für das Versenden von Daten.
$cost_{net} = (NET_{N1,N2} * iW * iCard)$

5.4.3 Eindruck über die Funktionsweise

Mit dieser Reihe von Experimenten mit dem einfachen Szenario soll gezeigt werden, dass die vorgestellten Verfahren „gute“ MS generieren, die intuitiv vernünftig sind. Abbildung 5.3 zeigt die MS, die für dieses Szenario für die gegebene Arbeitslast und Menge von Abfragen generiert werden. Die Kanten symbolisieren die Netzwerkverbindungen und die Knoten symbolisieren die Rechnerknoten. Da alle Rechnerknoten dieselben Eigenschaften und alle Tabellen dieselbe Kardinalität haben sowie mit derselben Häufigkeit aktualisiert werden, wird erwartet, dass alle Rechnerknoten ihre Abfragen aus lokalen Tabellen und MS berechnen, um

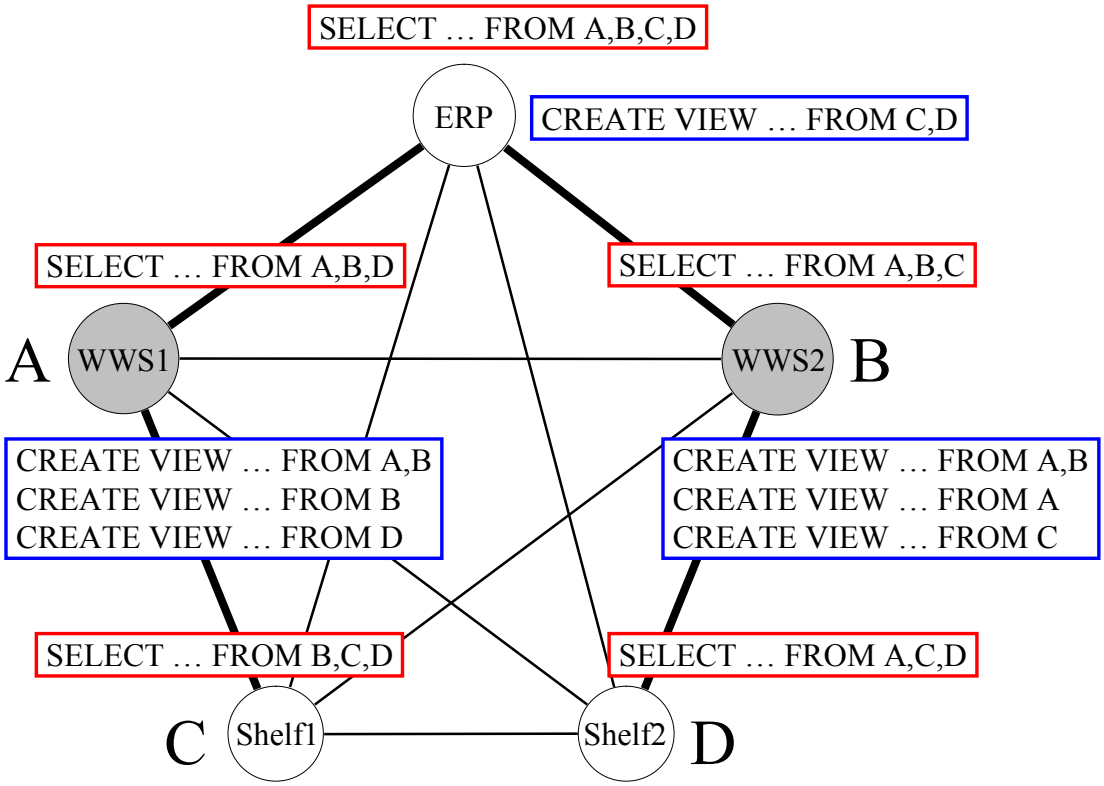

Abbildung 5.4: MS nach Änderung der Rechnerknoten für das einfache Szenario

somit Netzwerkübertragungskosten zu sparen. Die Abbildung zeigt, dass diese Erwartung erfüllt wird. Die vorgeschlagenen MS sind auf alle Knoten verstreut. Es werden MS über ein bzw. zwei Tabellen für die Rechnerknoten WWS1, WWS2, Shelf1 und Shelf2 vorgeschlagen und eine MS über zwei Tabellen auf dem Rechnerknoten ERP. Diese Aufstellung erlaubt den Rechnerknoten WWS1, WWS2, Shelf1 und Shelf2 ihre Abfragen ohne direkten Datentransfer über das Netzwerk zu beantworten. Es ist anzumerken, dass die MS aus den Tabellen A und D nur ein Mal berechnet und dann auf andere Rechnerknoten repliziert wird. Aufgrund der Symmetrie des Szenarios, und da der genetische Algorithmus nichtdeterministisch ist, existieren Lösungen mit gleicher Qualität, wie z.B. eine Sicht auf B und C statt auf A und D auf dem Rechnerknoten ERP zu materialisieren.

Anschließend werden die Kosten für CPU und IO auf den WWS-Rechnerknoten und deren Netzwerkkosten zu dem ERP und zu den Smart-Shelves um 50% verringert. Falls das vorgestellte Verfahren gut funktioniert, sollten mehr Sichten auf diesen Knoten materialisiert werden, da sie schnelle Berechnungen durchführen und schneller Daten übertragen können. Abbildung 5.4 zeigt, dass das Verfahren wie erwartet funktioniert: Sichten, die zuvor auf den Rechnerknoten Shelf1 und Shelf2 materialisiert waren, sind jetzt bei den Rechnerknoten WWS1 und WWS2 angesiedelt. Die fettgedruckten Linien in der Abbildung stellen die tatsächlichen Kommunikationskosten dar, z.B. haben WWS1 und WWS2 schnelle Verbindungen zu dem ERP, aber die Verbindung zwischen WWS1 und WWS2 ist langsam.

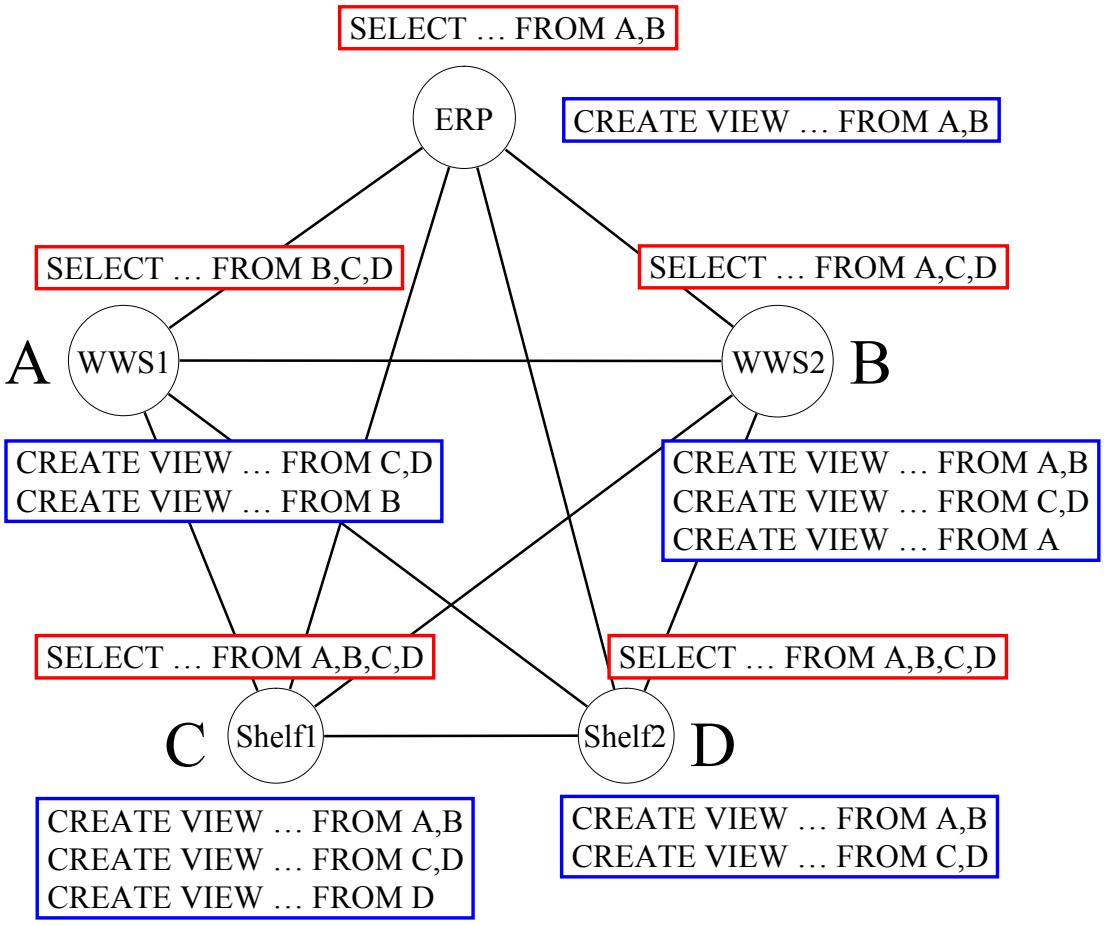

Abbildung 5.5: MS nach Änderung der Arbeitslast für das einfache Szenario

Nun werden die ursprünglich gleichen Kosten wieder eingestellt und die Arbeitslast wird geändert. Die Anzahl von abgefragten Tabellen wird in Shelf1 und Shelf2 von drei auf vier erhöht und bei dem ERP von vier auf zwei gesenkt. Die Knoten WWS1 und WWS2 fragen jetzt drei externe Tabellen ab, statt zwei externe und eine interne Tabelle. Es wird erwartet, dass in dieser Einstellung die lokale Berechnung von Abfragen bevorzugt wird, da alle Rechnerknoten dieselben Eigenschaften besitzen. Die Änderung der Arbeitslast soll eine Verschiebung der Sichten auf die Smart-Shelves hervorrufen. Die Ergebnisse sind in Abbildung 5.5 dargestellt. Verglichen mit dem ersten Versuch dieser Reihe können jetzt alle Abfragen aus lokalen Daten beantwortet werden. Aufgrund der Symmetrie des Szenarios existieren wieder mehrere Lösungen mit derselben Qualität, z.B. Tabelle C auf Shelf2 materialisieren statt D auf Shelf1.

5.4.4 Variationen der Ergebnisse

Genetische Algorithmen sind nichtdeterministisch, daher variiert die Qualität der ausgewählten MS bei mehreren Durchläufen mit derselben Eingabe. Allerdings wird aufgrund der vielen

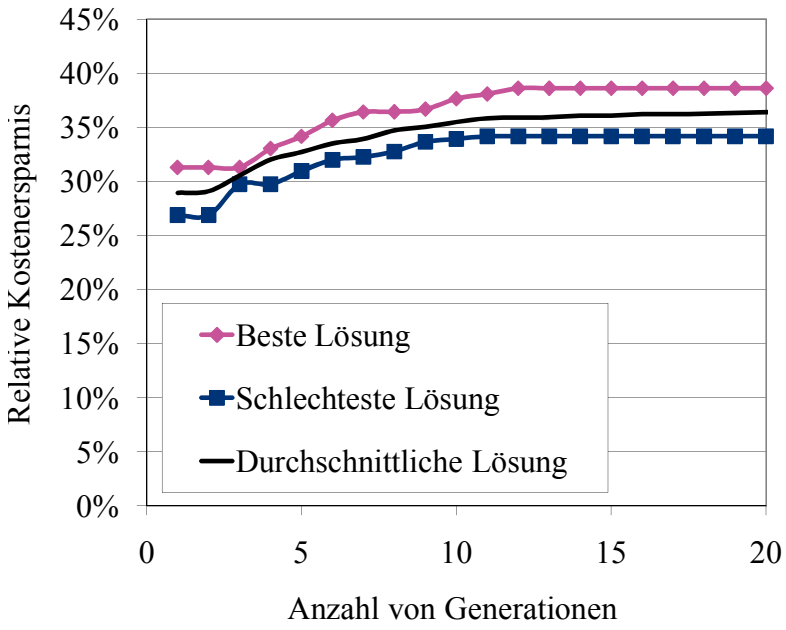

Abbildung 5.6: Variation der relativen Kostenersparnis

Heuristiken, die minderwertige Ergebnisse frühzeitig entfernen, davon ausgegangen, dass die Ergebnisse des genetischen Algorithmus vergleichbar gute Ergebnisse liefern. Um diese Annahme zu verifizieren, wurden zehn Versuche mit dem realem Szenario durchgeführt, bestehend aus 20 Filialen und insgesamt 80 Rechnerknoten. Dabei wurde in jeder Generation die relative Kostenersparnis gemessen. Die relative Kostenersparnis r wird definiert als die Differenz der ursprünglichen Kosten (c_0) und der Kosten bei dem Einsatz von MS (c_m), normalisiert nach den Kosten ohne MS: $r = (c_0 - c_m)/c_0$.

Abbildung 5.6 zeigt die Ergebnisse dieser Reihe von Versuchen. Jeder Punkt auf einer der Kurven stellt eine Generation des genetischen Algorithmus dar. Für jede Generation werden die beste, die schlechteste und die durchschnittliche relative Kostenreduktion der zehn Versuche gezeigt. Das Abbruchkriterium des genetischen Algorithmus wurde für diese Versuche gelockert, damit alle Versuche für deutlich mehr Generationen als notwendig laufen. Es werden nur die ersten 20 Generationen abgebildet, da eine größere Anzahl von Generationen zu keiner neuen Erkenntnis führt. Die Laufzeit von jedem Versuch lag unter 4 Minuten.

Die Versuche bestätigen, dass die angewendeten Heuristiken effektiv sind: Der genetische Algorithmus konvergiert nach sehr wenigen Generationen. Des Weiteren weisen die besten und die schlechtesten Ergebnisse eine Abweichung von weniger als 2% von dem durchschnittlichen Wert auf. Die Versuche zeigen auch, dass die Ergebnisse stabil sind. Bereits nach einer Generation werden MS vorgeschlagen, die zu einer signifikanten Kostenersparnis führen.

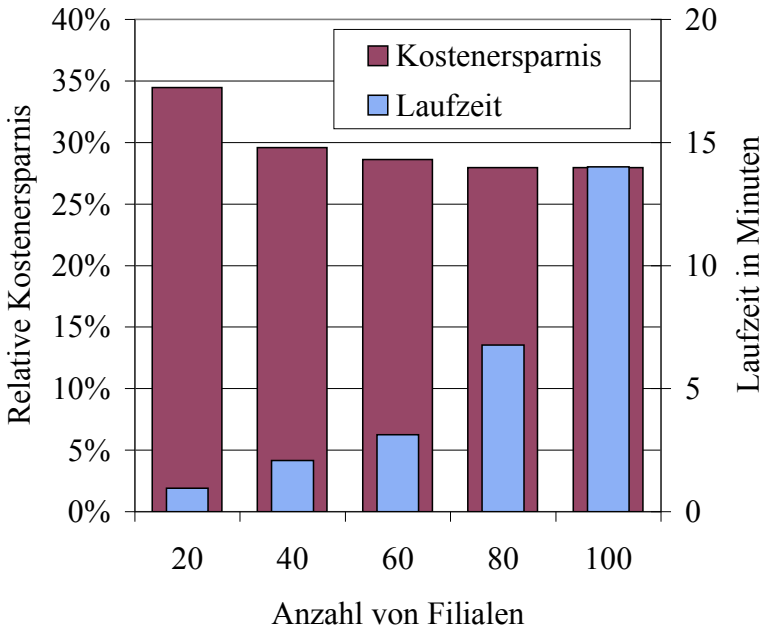

Abbildung 5.7: Laufzeit und Kostenersparnis in Abhängigkeit der Szenariogröße

5.4.5 Kosteneinsparungen und Laufzeit

In dieser Reihe von Versuchen wird analysiert, wie die vorgestellten Verfahren in großen Szenarien funktionieren. Es werden Versuche mit dem realen Szenario durchgeführt und es werden sowohl die Laufzeit als auch die relative Kostenersparnis gemessen. Um den Einfluss der Größe des Szenarios auf das vorgestellte Verfahren zu messen, wird die Anzahl von Filialen zwischen 20 und 100 variiert. Für jede Szenariogröße wurden zehn Versuche durchgeführt und die durchschnittlichen Ergebnisse berechnet. Es wird erwartet, dass die Größe des Szenarios einen geringen Einfluss auf die Qualität der vorgeschlagenen MS hat, und dass die Laufzeit trotz einer sehr hohen Anzahl von Filialen niedrig bleiben wird.

Die Ergebnisse werden in Abbildung 5.7 gezeigt. Alle Versuche zeigen eine Kostenersparnis von mindestens 28%. Die Kostenersparnis sinkt geringfügig mit der Größe des Szenarios. Obwohl die Laufzeit signifikant mit der Größe des Szenarios zunimmt, ist sie trotzdem gering. Das Szenario mit 100 Filialen, das mehr als 400 Rechnerknoten mit mehr als 1.000 Tabellen beinhaltet, wird in ca. 14 Minuten berechnet. Wie erwartet bestätigen die Versuche, dass die Größe des Szenarios die Qualität der Ergebnisse nur geringfügig beeinflusst. Des Weiteren wurde gezeigt, dass sich die vorgestellten Verfahren auf sehr große verteilte Szenarien anwenden lassen.

5.4.6 Optimalität

Genetische Algorithmen können nicht garantieren, dass eine optimale Lösung gefunden wird. Allerdings kann gezeigt werden, dass die vorgestellten Algorithmen „gute“ Lösungen produ-

zieren.

In diesem Versuch werden die Ergebnisse des vorgestellten Algorithmus mit der optimalen Lösung verglichen. Dafür wurde ein einfacher *Brute-Force*-Algorithmus implementiert, der die optimale Lösung findet, indem alle möglichen Lösungen durchsucht werden, also alle möglichen MS und alle möglichen Rechnerknoten zur Materialisierung. Da dies sehr zeitintensiv ist, wurde das reale Szenario auf den ERP-Rechnerknoten und auf zwei Filialen reduziert, also neun Rechnerknoten und 24 Tabellen.

Nachdem das vorgestellte Verfahren und der Brute-Force-Algorithmus angewendet wurden, wird die Qualität beider Lösungen verglichen. In diesem Szenario liefert das vorgestellte Verfahren eine Lösung, die sehr nahe am Optimum liegt. Die relative Kostenersparnis der vorgeschlagenen MS ist weniger als 0,01% schlechter als die relative Kostenersparnis der optimalen Lösung. Allerdings beträgt die Laufzeit des genetischen Algorithmus lediglich 30 Sekunden, während die Suche durch den kompletten NP-vollständigen Lösungsraum 10 Stunden dauerte.

5.4.7 Robustheit der Ergebnisse

Die Auswahl von MS basiert auf Kostenschätzungen durch ein Kostenmodell. Sind diese Schätzungen schlecht, so kann sich dies auf die Qualität der Lösung auswirken. Wenn beispielsweise die Rechenleistung eines Rechnerknotens überschätzt wird, könnten viele Sichten auf einem langsamen Rechnerknoten materialisiert werden. Aus diesem Grund müssen praktische Lösungen für das verteilte Sichtenauswahlproblem trotz ungenauer Kostenschätzungen gute Lösungen liefern.

Mit dieser Reihe von Experimenten wird der Einfluss von ungenauen Kostenschätzungen auf die Qualität der Lösung analysiert. Das reale Szenario wird erneut verwendet. Es wird ein Kostenmodell als 100% akkurat definiert, d.h. es wird angenommen, die Kosten werden korrekt modelliert. Dann werden die Kosten von CPU, IO und NET einzeln variiert.

Die Ergebnisse sind in Abbildung 5.8 zusammengefasst. Die X-Achse zeigt die relative Erhöhung und Verringerung der Kostenparameter während die Y-Achse die relative Kostenersparnis darstellt. Falls die Kosten für CPU 75% überschätzt oder 50% unterschätzt werden, wird die Qualität der Lösung schlechter. Die Modellierung der IO-Kosten beeinflusst die Qualität der Lösung, falls sie um mehr als 50% unterschätzt wird. Eine falsche Schätzung der NET-Kosten hat einen sehr geringen Einfluss auf die Lösung. Dies lässt sich durch die Heuristik für die initiale Population erklären: Es werden MS auf Knoten platziert, die Abfragen stellen oder Tabellen speichern, um die Netzwerkkosten zu minimieren, auch wenn diese Kosten null sind. Es sei darauf hingewiesen, dass trotz stark abweichender Kostenschätzung in keinem Fall MS vorgeschlagen wurden, die höhere Kosten verursachen als sie einsparen. Aus diesem Grund ist das vorgestellte Verfahren robust gegen ungenaue Kostenschätzungen.

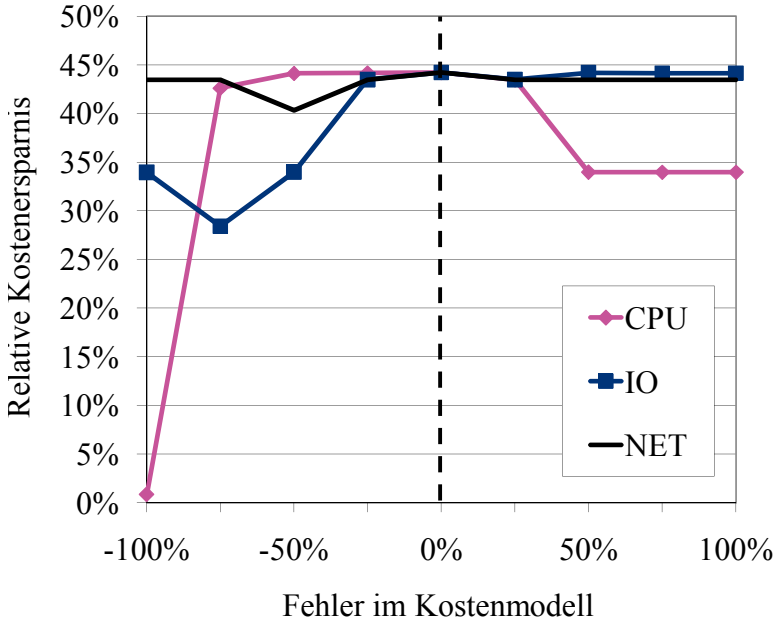

Abbildung 5.8: Einfluss der Ungenauigkeit des Kostenmodells auf das Verfahren

5.5 Zusammenfassung

In diesem Kapitel wurde ein Verfahren vorgestellt, um geeignete materialisierte Sichten (MS) für verteilte Datenbankszenarien zu wählen. MS sind (Zwischen-)Ergebnisse einer oder mehrerer Abfragen, die vorberechnet und gespeichert werden. Dadurch können zukünftige Abfragen schneller beantwortet und somit komplexe Abfragen sowohl auf verteilte als auch große Datenmengen, die im Einzelhandel typisch sind, optimiert werden. Allerdings müssen MS aktualisiert werden, wenn sich die unterliegenden Relationen ändern. Das Problem, MS für verteilte Szenarien zu wählen, ist NP-vollständig. Aus diesem Grund wurde folgender Ansatz verfolgt: Durch die Anwendung mehrerer Heuristiken wird der Lösungsraum derart verkleinert, dass ein genetischer Algorithmus angewendet werden kann. Experimente mit synthetischen Daten mit bis zu 400 Rechnerknoten haben gezeigt, dass die vorgeschlagene Menge von MS für Datenbankadministratoren intuitiv ist, dass die Ergebnisse eine geringe Variation aufweisen und dass das Verfahren schnell sowie skalierbar ist. Des Weiteren ist es robust gegen verfälschte Parameter, die für die Optimierung verwendet werden, wie z.B. die Leistung der Rechnerknoten.

Kapitel 6

Zusammenfassung und Ausblick

Durch den Einsatz von Radiofrequenzidentifikation (RFID) können Bestände und Warenbewegungen automatisiert erfasst werden. Dadurch lassen sich Geschäftsprozesse in verschiedenen Branchen optimieren. Ein vielversprechendes Einsatzszenario ist der Einzelhandel. Heute ist die Datenerfassung manuell, sie geschieht selten und ist fehlerbehaftet. Dies kann dazu führen, dass sich keine Artikel eines Produkttyps in den Verkaufsflächen befinden. Artikel können auch fehlplatziert sein, so dass sie sich zwar in der Filiale befinden, aber von Kunden nicht gekauft werden können. Aus diesem Grund liegt die Warenverfügbarkeit in der Filiale zwischen 90% und 93% [ECR03]. Das Optimierungspotential ist groß, da die schlechte Warenverfügbarkeit den Umsatz eines Einzelhändlers um ca. 4% verringert [GCB02].

Im Rahmen eines Pilotprojektes mit einem großen deutschen Einzelhändler wurden Teile des Sortiments einer Filiale mit RFID ausgestattet, um die Machbarkeit und Vorteile von RFID zu bewerten. Auf technischer Ebene ergaben sich allerdings Probleme, welche die Integration von RFID-Daten in Unternehmenssoftware erschwerten: Wenn RFID-Leser Artikel erfassen, treten Störungen aufgrund von Absorptionen und Reflektionen von elektromagnetischen Wellen ein. Aus diesem Grund konnten (1) nicht alle Artikel in der Umgebung eines RFID-Lesers erfasst werden, und (2) Artikel wurden oft von mehreren RFID-Lesern bzw. Antennen gleichzeitig erfasst, so dass der Ort dieser Artikel nicht bestimmt werden konnte. Ferner entstanden (3) durch den Einsatz von RFID große Datenmengen, die in komplexen Abfragen über verteilte Daten effizient verarbeitet werden mussten. In dieser Arbeit wurden Verfahren vorgestellt, um diese Probleme zu lösen.

TagMark [WBB08] ist ein Verfahren, das die Anzahl von Artikeln in der Umgebung eines RFID-Lesers schätzt. Es erweitert ein statistisches Verfahren aus der Biologie, das die Anzahl von Individuen in einer Population anhand von verschiedenen Stichproben schätzt. TagMark kann die Genauigkeit der Schätzungen mittels relativer Konfidenzintervalle bestimmen. Des Weiteren ist die Größe der benötigten Stichproben durch obere Schranken begrenzt. Dies wurde analytisch hergeleitet und bewiesen. Umfassende Experimente mit einer RFID-Installation und mit synthetischen Daten zeigen, dass TagMark in realistischen Szenarien und unter extremen Bedingungen anwendbar ist. Es ist schnell und skaliert auf eine große Anzahl von Artikeln.

Mit dem Verfahren *RFID Planogram Compliance Verification* (**RPCV**) [WBB10] kann entschieden werden, ob Artikel, die von mehr als einer RFID-Antenne erfasst wurden, sich an dem richtigen Ort innerhalb eines Regals befinden. Dieser Ort wird in vordefinierten Layoutplänen, so genannten Planogrammen, spezifiziert. RPCV basiert auf der Beobachtung, dass Artikel desselben Produkttyps relativ ähnliche Lesemuster aufweisen. RPCV clustert alle Artikel desselben Produkttyps und entscheidet, ob sie richtig oder fehlplatziert sind. Experimente mit einer RFID-Installation und mit synthetischen Daten zeigen, dass RPCV eine Größenordnung weniger falsche Schätzungen als verwandte Arbeiten liefert, dass es schnell ist und weniger RFID-Daten benötigt, um gute Schätzungen zu liefern.

Durch die **Wahl von geeigneten materialisierten Sichten** [WBHB09] werden komplexe Abfragen auf verteilte und große Datenmengen optimiert. Materialisierte Sichten (MS) sind (Zwischen-)Ergebnisse einer oder mehrerer Abfragen, die vorberechnet und gespeichert werden. Diese Ergebnisse können verwendet werden, um zukünftige Abfragen schneller zu beantworten. Allerdings müssen MS aktualisiert werden, wenn sich die unterliegenden Relationen ändern. Da die Wahl von geeigneten MS NP-vollständig ist [Gup97], wird das Problem durch die Verwendung mehrerer Heuristiken gelöst, die den Lösungsraum so stark verkleinern, dass ein genetischer Algorithmus angewendet werden kann. Das Verfahren ist schnell und skalierbar: Es berechnet beispielsweise eine gute Lösung für ein großes Szenario mit 400 Rechnerknoten in ca. 15 Minuten.

Die in dieser Arbeit vorgestellten Verfahren lösen die wichtigsten Probleme bei der Integration von RFID in Unternehmenssoftware. Es ist zu erwarten, dass sie sofort in der Praxis umgesetzt werden können, da sie nicht nur in realistischen Szenarien, sondern auch unter extremen Bedingungen evaluiert wurden.

6.1 Ausblick auf Folgearbeiten

Der Einsatz von RFID ist ein breit gefächertes Thema, dem viele Forschungsfragen unterliegen. Eine Auswahl dieser Forschungsfragen wurde im Rahmen dieser Arbeit untersucht. Die Ergebnisse bilden einen Ausgangspunkt für viele Folgearbeiten, wie nachfolgend beispielhaft aufgeführt:

Kombination von RPCV und TagMark mit anderen Verfahren: Viele Schätzverfahren für die Anzahl von Funketiketten und für den Ort eines Funketiketts verwenden statistische Daten als Eingabe. Beispiele solcher Verfahren sind in [JFG08, KBS06, KBS08, XYC+08] zu finden. Diese Verfahren könnten die Daten von TagMark und RPCV verwenden, um bessere Schätzungen zu gewinnen. Verfahren, die sowohl die Anzahl als auch den Ort von Funketiketten schätzen, würden beide Eingaben in ihrem Modell kombinieren, so dass sowohl die geschätzte Anzahl als auch der Ort in sich konsistent wären. Offen bleibt allerdings die Frage, ob sich solche Verfahren im Einzelhandel anwenden lassen, da sie, wie bereits erwähnt, nicht alle notwendigen Anforderungen erfüllen. Ferner ist es unpraktisch, die restlichen statistischen Daten zu erheben, da sich der Ort von Regalen sowie Ort, Typ und Anzahl von Artikeln oft ändert.

Vorausschauende Wahl von materialisierten Sichten: Das in dieser Arbeit vorgestellte Verfahren zur Wahl von MS für verteilte Szenarien ist als eine Wartungsarbeit konzipiert. In regelmäßigen Abständen wird der Algorithmus aufgerufen, um eine Menge von MS zu finden, welche die Kosten der beobachteten Abfragen und Aktualisierungen minimiert. Tritt jedoch eine schnelle Änderung der Art und Häufigkeiten der Abfragen und Aktualisierungen ein, wird der Mehrwert der gegenwärtig gewählten MS geringer und neue MS werden erst bei der nächsten Ausführung des Algorithmus erstellt. Allerdings sind solche Änderungen der Abfragen im Einzelhandel vorhersehbar. Sie treten beispielsweise vor größeren Feiertagen wie Ostern oder Weihnachten auf. Für die Bestelloptimierung werden viele Quellen berücksichtigt, um die Nachfrage der Kunden vorherzusagen [HW08]. Auf ähnliche Art und Weise könnten die Art und Häufigkeiten der Abfragen sowie Aktualisierungen geschätzt werden, um geeignete Sichten im Voraus zu bestimmen.

Einfluss der genaueren Daten auf die Wirtschaftlichkeit: Bei der Betrachtung der Wirtschaftlichkeit eines Einsatzes von RFID wird im Voraus anhand eines Modells berechnet, ob sich der Einsatz der Technologie rentiert. In solchen Modellen werden in der Regel die RFID-Leserate, da die Größe von Sicherheitsbeständen und Lagerflächen von der Genauigkeit der Daten abhängt, sowie viele weitere Kennzahlen, die Umsatz und Kosten beeinflussen, berücksichtigt. Allerdings kann mit Verfahren wie TagMark die Genauigkeit der RFID-Daten erhöht werden. Daher wäre es sinnvoll, Schätzverfahren bei der Betrachtung der Wirtschaftlichkeit zu berücksichtigen. Beispielsweise könnte TagMark auf die Parameter des Modells angewendet werden, um die durch TagMark gewonnene Genauigkeit der Daten einkalkulieren zu können. Viele Modelle (z.B. [SBCM07, TF07]) enthalten bereits die notwendigen Parameter für TagMark, wie die geschätzte Leserate im Regal sowie an der Kasse, die Anzahl von Artikeln im Regal, die Anzahl von Abverkäufen usw. Analog könnte RPCV bei Prozessen berücksichtigt werden, die von der Genauigkeit der örtlichen Auflösung abhängen.

Unternehmensübergreifende Optimierung von Geschäftsprozessen: In dieser Arbeit wurde der Einsatz von RFID betrachtet, um Geschäftsprozesse innerhalb eines Unternehmens zu optimieren. Ist ein Artikel mit RFID ausgestattet, fallen allerdings Daten entlang der kompletten Lieferkette an. Durch die Verknüpfung dieser Daten können Prozesse entlang der kompletten Lieferkette optimiert werden. Beispiele solcher Prozesse stellen unter anderem die Bestellung, Optimierung von Losgrößen, Erkennung von Produktfälschungen und Chargenrückverfolgung dar. Allerdings befürchten Unternehmen, dass durch die Freigabe von Informationen sensible Unternehmensdaten bekannt werden könnten. Hierzu zählen beispielsweise sowohl die Lieferanten als auch die Vertriebskanäle des Unternehmens. Aus diesem Grund sind neuartige Sicherheitsmechanismen erforderlich, um solche Szenarien zu implementieren. Des Weiteren müssen viel größere Datenmengen verarbeitet werden, da für jeden Artikel Daten bei mehreren Unternehmen anfallen.

Tabellen- und Algorithmenverzeichnis

Tabellenverzeichnis

Algorithmenverzeichnis

Abbildungsverzeichnis

Abkürzungsverzeichnis

CPU ..... Hauptprozessor (*Central Processing Unit*)
DBMS .... Datenbankmanagementsystem (*Database Management System*)
EM ..... *Expectation-Maximization*
EPC ..... Eletronischer Produktkode (*Electronic Product Code*)
ERP ..... *Enterprise Resource Planning*
GHz ..... Gigahertz
GTIN ..... *Global Trade Item Number*
HF ..... Hochfrequenz (*High Frequency*)
IO ..... Eingabe/Ausgabe (*Input/Output*)
kHz ..... Kilohertz
LF ..... Langwelle (*Low Frequency*)
MHz ..... Megahertz
MS ..... Materialisierte Sicht(en)
NP ..... Nichtdeterministisch polynomielle Zeit
PDA ..... *Personal Digital Assistant*
POS ..... Verkaufsort (*Point-of-Sales*)
RAM ..... Hauptspeicher (*Random Access Memory*)
RFID ..... Radiofrequenzidentifikation (*Radio Frequency Identification*)
RPCV .... *RFID Planogram Compliance Verification*
SATA ..... *Serial ATA (Advanced Technology Attachment)*
SGTIN ... *Serialized Global Trade Item Number*
SPJG ..... Selektion, Projektion, Verknüpfung (*Join*) und Gruppierung
SQL ..... *Structured Query Language*
UHF ..... Ultra-Hochfrequenz (*Ultra High Frequency*)
WWS ..... Warenwirtschaftssystem

Literaturverzeichnis

[ABG+02] ALEXANDER, Keith ; BIRKHOFER, Garry ; GRAMLING, Kathryn ; KLEINBERGER, Herb ; LENG, Stephen ; MOOGIMANE, Dhaval ; WOODS, Maurice: Focus on Retail: Applying Auto-ID to Improve Product Availability at the Retail Shelf / Auto-ID Center White Paper. 2002. – Forschungsbericht 1, 8, 17

[ACK+04] AGRAWAL, Sanjay ; CHAUDHURI, Surajit ; KOLLAR, Lubor ; MARATHE, Arun ; NARASAYYA, Vivek ; SYAMALA, Manoj: Database Tuning Advisor for Microsoft SQL Server 2005. In: *Proceedings of VLDB'04*, 2004 66, 68

[ACN00] AGRAWAL, Sanjay ; CHAUDHURI, Surajit ; NARASAYYA, Vivek R.: Automated Selection of Materialized Views and Indexes in SQL Databases. In: *Proceedings of VLDB'00*, 2000 66, 68, 71, 72, 73

[AM05] ASIF, Zaheeruddin ; MANDVIWALLA, Munir: Integrating the Supply Chain with RFID: A Technical and Business Analysis. In: *Communications of the Association for Information Systems* (2005) 17

[AMSW06] ANKE, Jürgen ; MÜLLER, Jens ; SPIESS, Patrik ; WEISS FERREIRA CHAVES, Leonardo: A Service-Oriented Middleware for Integration and Management of Heterogeneous Smart Items Environments. In: *Proceedings of 4th MiNEMA Workshop*, 2006

[BBM93] BEASLEY, David ; BULL, David R. ; MARTIN, Ralph R.: An overview of genetic algorithms: Part 1, fundamentals. In: *University Computing* (1993) 71, 80

[BDD+98] BELLO, Randall G. ; DIAS, Karl ; DOWNING, Alan ; JAMES J. FEENAN, Jr. ; FINNERTY, James L. ; NORCOTT, William D. ; SUN, Harry ; WITKOWSKI, Andrew ; ZIAUDDIN, Mohamed: Materialized Views in Oracle. In: *Proceedings of VLDB'98*, 1998 66, 68, 69

[BDF+07] BEER, Michael ; DECKER, Christian ; FAHL, Harald ; FISCHER, Manfred ; GIERLING, Wolfgang ; KELLER, Kirsten ; LANG, Hartmut ; MORITZ, Tobias ; SCHAFFNER, Nicole ; SPANACHI, Florin ; STEINBERGER, Klaus ; WEISS FERREIRA CHAVES, Leonardo: Projekt LoCostix - AP 3.1 Geschäftsprozesse: Identifikation und Anforderungen. 2007. – Meilensteindokument 10, 22

[BFHF03] BRUSEY, James ; FLOERKEMEIER, Christian ; HARRISON, Mark ; FLETCHER, Martyn: Reasoning about uncertainty in location identification with RFID. In: *Proceedings of RUR Workshop*, 2003 43

[BGH+02] BOUSHKA, Michael ; GINSBURG, Lyle ; HABERSTROH, Jennifer ; HAFFEY, Thaddeus ; RICHARD, Jason ; TOBOLSKI, Joseph: Auto-ID on the Move: The Value of Auto-ID Technology in Freight Transportation / Auto-ID Labs White Paper. 2002. – Forschungsbericht 17

[BGMK08] BÖGELSACK, André ; GRADL, Stephan ; MAYER, Manuel ; KRCMAR, Helmut: *SAP MaxDB Administration*. Harlow : SAP Press, 2008. – ISBN 978–3–89842–730–2 32, 51

[BGS+05] BONO, Steve ; GREEN, Matthew ; STUBBLEFIELD, Adam ; JUELS, Ari ; RUBIN, Avi ; SZYDLO, Michael: Security Analysis of a Cryptographically-Enabled RFID Device. In: *Proceedings of USENIX Security Symposium*, 2005 18

[Bis00] BISHOP, Willard: Documenting the Value of Merchandising / National Association for Retail Merchandising Service. 2000. – Forschungsbericht 11, 42, 43

[BK08] BAI, Ruibin ; KENDALL, Graham: A Model for Fresh Produce Shelf-Space Allocation and Inventory Management with Freshness-Condition-Dependent Demand. In: *Informs Journal on Computing* (2008) 43

[BKS00] BELLATRECHE, Ladjel ; KARLAPALEM, Kamalakar ; SCHNEIDER, Michel: On efficient storage space distribution among materialized views and indices in data warehousing environments. In: *Proceedings of CIKM'00*, 2000 66, 68

[BL03] BAUER, Andreas ; LEHNER, Wolfgang: On solving the view selection problem in distributed data warehouse architectures. In: *Proceedings of SSDBM'2003*, 2003 67, 69

[BLHS04] BORNHÖVD, Christof ; LIN, Tao ; HALLER, Stephan ; SCHAPER, Joachim: Integrating automatic data acquisition with business processes: Experiences with SAP's Auto-ID Infrastructure. In: *Proceedings of VLDB'04*, 2004 12, 16, 43

[BO78] BURNHAM, Kenneth P. ; OVERTON, Walter S.: Estimation of the size of a closed population when capture probabilities vary among animals. In: *Biometrika* (1978) 23

[BPT97] BARALIS, Elena ; PARABOSCHI, Stefano ; TENIENTE, Ernest: Materialized Views Selection in a Multidimensional Database. In: *Proceedings of VLDB'97*, 1997 68

[BR07] BOLOTNYY, Leonid ; ROBINS, Gabriel: The Case for Multi-Tag RFID Systems. In: *Proceedings of WASA'07*, 2007 25, 44

[BSHW06] BENJELLOUN, Omar ; SARMA, Anish D. ; HALEVY, Alon ; WIDOM, Jennifer: ULDBs: databases with uncertainty and lineage. In: *Proceedings of VLDB'06*, 2006 24

[BWL06] BAI, Yijian ; WANG, Fusheng ; LIU, Peiya: Efficiently filtering RFID data streams. In: *Proceedings of CleanDB Workshop*, 2006 43

[BWL+07] BAI, Yijian ; WANG, Fusheng ; LIU, Peiya ; ZANIOLO, C. ; LIU, Shaorong: RFID Data Processing with a Data Stream Query Language. In: *Proceedings of ICDE'07*, 2007 43

[CCMN00] CHARIKAR, Moses ; CHAUDHURI, Surajit ; MOTWANI, Rajeev ; NARASAYYA, Vivek: Towards estimation error guarantees for distinct values. In: *Proceedings of PODS'00*, 2000 24

[CDS04] CHAUDHURI, Surajit ; DAS, Gautam ; SRIVASTAVA, Utkarsh: Effective use of block-level sampling in statistics estimation. In: *Proceedings of SIGMOD'04*, 2004 24

[Cha98] CHAUDHURI, Surajit: An Overview of Query Optimization in Relational Systems. In: *Proceedings of PODS'98*, 1998 83

[CKP03] CHENG, Reynold ; KALASHNIKOV, Dmitri V. ; PRABHAKAR, Sunil: Evaluating Probabilistic Queries over Imprecise Data. In: *Proceedings of SIGMOD'03*, 2003 24

[DBW+08] DECKER, Christian ; BERCHTOLD, Martin ; WEISS FERREIRA CHAVES, Leonardo ; BEIGL, Michael ; RÖHR, Daniel ; RIEDEL, Till ; BEUSTER, Monty ; HERZOG, Thomas ; HERZIG, Daniel: Cost-Benefit Model for Smart Items in the Supply Chain. In: *Proceedings of Internet of Things'08*, 2008 17

[DKB03] DECKER, Christian ; KUBACH, Uwe ; BEIGL, Michael: Revealing the Retail Black Box by Interaction Sensing. In: *Proceedings of ICDCSW'03*, 2003 8, 43

[DLK06] DUC, Dang N. ; LEE, Hyunrok ; KIM, Kwangjo: Enhancing Security of EPCGlobal Gen-2 RFID against Traceability and Cloning / Auto-ID Labs White Paper. 2006. – Forschungsbericht 18

[DLR77] DEMPSTER, Arthur P. ; LAIRD, Nan M. ; RUBIN, Donald B.: Maximum-likelihood from incomplete Data via the EM Algorithm. In: *Journal of the Royal Statistical Society, Series B 39* (1977) 47, 48

[DS04] DALVI, Nilesh ; SUCIU: Efficient query evaluation on probabilistic databases. In: *Proceedings of VLDB'04*, 2004 24

[ECR03] ECR EUROPE: Optimal Shelf Availability: Increasing Shopper Satisfaction at the Moment of Truth. 2003. – Forschungsbericht 1, 8, 93

[EHM99] EIBEN, Ágoston E. ; HINTERDING, Robert ; MICHALEWICZ, Zbigniew: Parameter Control in Evolutionary Algorithms. In: *IEEE Transactions on Evolutionary Computation* (1999) 80, 83

[EPC05] EPCGLOBAL INC.: Specification for RFID Air Interface: EPC Radio-Frequency Identity Protocols Class-1 Generation-2 UHF RFID Protocol Communications at 860MHz - 960MHz, Version 1.1.0. 2005. – Standard 18

[EPC06] EPCGLOBAL INC.: EPC Tag Data Standards, Version 1.3. 2006. – Standard 7, 32, 51

[Fin03] FINKENZELLER, Klaus: *RFID Handbook: Fundamentals and Applications in Contactless Smart Cards and Identification*. John Wiley & Sons, Inc., 2003. – ISBN 978–0–47084–402–1 1, 5, 6

[FJKR05] FRANKLIN, Michael J. ; JEFFERY, Shawn R. ; KRISHNAMURTHY, Sailesh ; REISS, Frederick: Design Considerations for High Fan-in Systems: The HiFi Approach. In: *Proceedings of CIDR'05*, 2005 43

[FL04] FLOERKEMEIER, Christian ; LAMPE, Matthias: Issues with RFID Usage in Ubiquitous Computing Applications. In: *Proceedings of Pervasive'04*, 2004 2, 6, 11, 12, 14, 22, 25, 33, 41

[Fuj08] FUJITSU: Fujitsu Develops World's First 64KByte High-Capacity FRAM RFID Tag for Aviation Applications. 2008. – Pressemitteilung 7

[GCB02] GRUEN, Thomas W. ; CORSTEN, Daniel S. ; BHARADWAJ, Sundar: Retail Out of Stocks: A Worldwide Examination of Extent, Causes, and Consumer Responses. 2002. – Forschungsbericht 1, 10, 93

[GM99] GUPTA, Himanshu ; MUMICK, Inderpal S.: Selection of Views to Materialize Under a Maintenance Cost Constraint. In: *Lecture Notes in Computer Science*, 1999 67, 69, 74

[GS105] GS1 FRANCE: RFID for logistic applications - Tests results. EPCglobal Inc., France Lab, 2005. – Forschungsbericht 11, 13, 22, 25, 97

[GSH07] GAUKLER, Gary M. ; SEIFERT, Ralf W. ; HAUSMAN, Warren H.: Item-Level RFID in the Retail Supply Chain. In: *Production and Operations Management* (2007) 1, 8

[Gup97] GUPTA, Himanshu: Selection of Views to Materialize in a Data Warehouse. In: *Proceedings of ICDT'97*, 1997 3, 65, 67, 69, 74, 94

[HD02] HOLLINGER, Richard C. ; DAVIS, Jason L.: National Retail Security Survey 2001 / Department of Sociology and the Center for Studies in Criminology and Law, University of Florida. 2002. – Forschungsbericht 10

[HO91] HOU, Wen-Chi ; OZSOYOGLU, Gultekin: Statistical estimators for aggregate relational algebra queries. In: *ACM Transactions on Database Systems* (1991) 24

[HW08] HANKE, John E. ; WICHERN, Dean W.: *Business Forecasting*. Prentice Hall, 2008. – ISBN 978–0–13230–120–6 95

[HWM06] HARDGRAVE, Bill C. ; WALLER, Matthew ; MILLER, Robert: RFID's Impact on Out of Stocks: A Sales Velocity Analysis / University of Arkansas. 2006. – Forschungsbericht 1

[HZ96] HULL, Richard ; ZHOU, Gang: Towards the Study of Performance Trade-offs Between Materialized and Virtual Integrated Views. In: *Proceedings of VIEWS'96*, 1996 67, 69

[JFG08] JEFFERY, Shawn R. ; FRANKLIN, Michael J. ; GAROFALAKIS, Minos: An adaptive RFID middleware for supporting metaphysical data independence. In: *The VLDB Journal* (2008) 43, 94

[JGF06] JEFFERY, Shawn R. ; GAROFALAKIS, Minos ; FRANKLIN, Michael J.: Adaptive cleaning for RFID data streams. In: *Proceedings of VLDB'06*, 2006 24, 43

[JGL07] JIANG, Haifeng ; GAO, Dengfeng ; LI, Wen-Syan: Exploiting Correlation and Parallelism of Materialized-View Recommendation for Distributed Data Warehouses. In: *Proceedings of ICDE'07*, 2007 67, 69

[Jol65] JOLLY, George M.: Explicit Estimates from Capture-Recapture Data with Both Death and Immigration-Stochastic Model. In: *Biometrika* (1965) 23

[JRS03] JUELS, Ari ; RIVEST, Ronald L. ; SZYDLO, Michael: The Blocker Tag: Selective Blocking of RFID Tags for Consumer Privacy. In: *Proceedings of ACM Conference on Computer and Communications Security*, 2003 18

[Kal07] KALOS, Malvin H.: Monte Carlo methods in the physical sciences. In: *Proceedings of WSC'07*, 2007 57

[KBS06] KHOUSSAINOVA, Nodira ; BALAZINSKA, Magdalena ; SUCIU, Dan: Towards correcting input data errors probabilistically using integrity constraints. In: *Proceedings of MobiDE'06*, 2006 24, 43, 94

[KBS08] KHOUSSAINOVA, Nodira ; BALAZINSKA, Magdalena ; SUCIU, Dan: PEEX: Extracting Probabilistic Events from RFID Data. In: *Proceedings of ICDE'08*, 2008 43, 94

[KM05] KARJOTH, Günter ; MOSKOWITZ, Paul: Disabling RFID Tags with visual confirmation: Clipped Tags are silenced / IBM Research Report. 2005. – Forschungsbericht 18

[KN06] KODIALAM, Murali ; NANDAGOPAL, Thyaga: Fast and reliable estimation schemes in RFID systems. In: *Proceedings of MobiCom'06*, 2006 24

[KOW10] KERSCHBAUM, Florian ; OERTEL, Nina ; WEISS FERREIRA CHAVES, Leonardo: Privacy-Preserving Computation of Benchmarks on Item-Level Data Using RFID. In: *Proceedings of WiSec'10*, 2010

[KZBD05] KROHN, Albert ; ZIMMER, Tobias ; BEIGL, Michael ; DECKER, Christian: Collaborative Sensing in a Retail Store Using Synchronous Distributed Jam Signalling. In: *Proceedings of Pervasive'05*, 2005 24, 51

[Lig02] LIGHTBURN, Anne: Unsaleables Benchmark Report / Joint Industry Unsaleables Steering Committee, Food Distributors International / Food Marketing Institute / Grocery Manufacturers of America. 2002. – Forschungsbericht 10

[LKO+00] LEE, Mong L. ; KITSUREGAWA, Masaru ; OOI, Beng C. ; TAN, Kian-Lee ; MONDAL, Anirban: Towards self-tuning data placement in parallel database systems. In: *Proceedings of SIGMOD'00*, 2000 69

[Loe05] LOEBBECKE, Claudia: RFID Technology and Applications in the Retail Supply Chain: The early Metro Group Pilot. In: *Proceedings of Bled eConference*, 2005 8

[LZB+07] LI, Wen-Syan ; ZILIO, Daniel C. ; BATRA, Vishal S. ; ZUZARTE, Calisto ; NARANG, Inderpal: Load balancing and data placement for multi-tiered database systems. In: *Data & Knowledge Engineering* (2007) 67, 69

[ML86] MACKERT, Lothar F. ; LOHMAN, Guy M.: R* optimizer validation and performance evaluation for local queries. In: *Proceedings of SIGMOD'86*, 1986 83, 84

[MRSR01] MISTRY, Hoshi ; ROY, Prasan ; SUDARSHAN, S. ; RAMAMRITHAM, Krithi: Materialized view selection and maintenance using multi-query optimization. In: *Proceedings of SIGMOD'01*, 2001 68, 74

[PL00] POTTINGER, Rachel ; LEVY, Alon Y.: A Scalable Algorithm for Answering Queries Using Views. In: *Proceedings of VLDB'00*, 2000 69

[RCT06] RIEBACK, Melanie R. ; CRISPO, Bruno ; TANENBAUM, Andrew S.: The Evolution of RFID Security. In: *IEEE Pervasive Computing* (2006) 18

[RDTC06] RAO, Jun ; DORAISWAMY, Sangeeta ; THAKKAR, Hetal ; COLBY, Latha S.: A deferred cleansing method for RFID data analytics. In: *Proceedings of VLDB'06*, 2006 43

[RLBS08] RÉ, Christopher ; LETCHNER, Julie ; BALAZINKSA, Magdalena ; SUCIU, Dan: Event queries on correlated probabilistic streams. In: *Proceedings of SIGMOD'08*, 2008 44

[RNL05] RAO, K. V. Seshagiri ; NIKITIN, Pavel V. ; LAM, Sander F.: Antenna Design for UHF RFID Tags. In: *IEEE Transactions on Antennas and Propagation* (2005) 17

[RR64] ROBSON, Douglas. S. ; REGIER, Henry A.: Sample Size in Petersen Mark-Recapture Experiments. In: *Transactions of the American Fisheries Society* (1964) 20, 21

[Sar01] SARMA, Sanjay: Towards the 5 cent Tag / Auto-ID Labs White Paper. 2001. – Forschungsbericht 17

[SBCM07] SOUNDERPANDIAN, Jayavel ; BOPPANA, Rajendra V. ; CHALASANI, Suresh ; MADNI, Asad M.: Models for Cost-Benefit Analysis of RFID Implementations in Retail Stores. In: *IEEE Systems Journal* (2007) 17, 95

[SBHW06] SARMA, Anish D. ; BENJELLOUN, Omar ; HALEVY, Alon ; WIDOM, Jennifer: Working Models for Uncertain Data. In: *Proceedings of ICDE'06*, 2006 24

[SCFV+08] SUBRAMANIAN, Vivek ; CHANG, Josephine B. ; FUENTE VORNBROCK, Alejandro de l. ; HUANG, Daniel C. ; JAGANNATHAN, Lakshmi ; LIAO, Frank ; MATTIS, Brian ; MOLESA, Steven ; REDINGER, David R. ; SOLTMAN, Daniel ; VOLKMAN, Steven K. ; ZHANG, Qintao: Printed Electronics For Low-Cost Electronic Systems: Technology Status and Application Development. In: *Proceedings of IEEE European Solid-State Device Research Conference*, 2008 17

[SDN98] SHUKLA, Amit ; DESHPANDE, Prasad ; NAUGHTON, Jeffrey F.: Materialized View Selection for Multidimensional Datasets. In: *Proceedings of VLDB'98*, 1998 66, 68

[Seb65] SEBER, George A. F.: A Note on the Multiple-Recapture Census. In: *Biometrika* (1965) 23

[Seb82] SEBER, George A. F.: *The Estimation of Animal Abundance and Related Parameters*. Blackburn Press, 1982. – ISBN 978–1–93066–555–2 19, 20, 29, 30

[Sto48] STOCKMAN, Harry: Communication by Means of Reflected Power. In: *Proceedings of the I.R.E*, 1948 5

[TF07] THIESSE, Frédéric ; FLEISCH, Elgar: Zum Einsatz von RFID in der Filiallogistik eines Einzelhändlers. In: *Proceedings of Wirtschaftsinformatik'07*, 2007 11, 17, 95

[TP08] TU, Yu-Ju ; PIRAMUTHU, Selwyn: Reducing false reads in RFID-embedded supply chains. In: *Journal of Theoretical and Applied Electronic Commerce Research* (2008) 44

[TSC+09] TRAN, Thanh ; SUTTON, Charles ; COCCI, Richard ; NIE, Yanming ; DIAO, Yanlei ; SHENOY, Prashant: Probabilistic Inference over RFID Streams in Mobile Environments. In: *Proceedings of ICDE'09*, 2009 44

[TZP09] TU, Yu-Ju ; ZHOU, Wei ; PIRAMUTHU, Selwyn: Identifying RFID-embedded objects in pervasive healthcare applications. In: *Decision Support Systems* (2009) 44

[Vog02] VOGT, Harald: Efficient Object Identification with Passive RFID Tags. In: *Proceedings of Pervasive'02*, 2002 24

[WAMM06] WEISS FERREIRA CHAVES, Leonardo ; ANKE, Jürgen ; MOREIRA SÁ DE SOUZA, Luciana ; MÜLLER, Jens: Service Lifecycle Management Infrastructure for Smart Items. In: *Proceedings of MidSens'06*, 2006

[Wan06] WANT, Roy: An Introduction to RFID Technology. In: *IEEE Pervasive Computing* (2006) 5, 6

[WBB08] WEISS FERREIRA CHAVES, Leonardo ; BUCHMANN, Erik ; BÖHM, Klemens: TagMark: Reliable Estimations of RFID Tags for Business Processes. In: *Proceedings of KDD'08*, 2008 2, 11, 16, 19, 25, 42, 93, 99

[WBB10] WEISS FERREIRA CHAVES, Leonardo ; BUCHMANN, Erik ; BÖHM, Klemens: Finding Misplaced Items in Retail by Clustering RFID Data. In: *Proceedings of EDBT'10*, 2010 2, 41, 94

[WBHB09] WEISS FERREIRA CHAVES, Leonardo ; BUCHMANN, Erik ; HUESKE, Fabian ; BÖHM, Klemens: Towards Materialized View Selection for Distributed Databases. In: *Proceedings of EDBT'09*, 2009 3, 65, 94

[WF05] WITTEN, Ian H. ; FRANK, Eibe: *Data Mining: Practical Machine Learning Tools and Techniques*. 2005. – ISBN 978–0–12088–407–0 47

[WK08] WEISS FERREIRA CHAVES, Leonardo ; KERSCHBAUM, Florian: Industrial Privacy in RFID-based Batch Recalls. In: *Proceedings of InSPEC'08*, 2008

[WK10] WEISS FERREIRA CHAVES, Leonardo ; KERSCHBAUM, Florian: Security and Privacy in Track & Trace Infrastructures. In: BOUÇA, Duarte (Hrsg.) ; GAFAGNÃO, Amaro (Hrsg.): *Agent-Based Computing*. Nova Science Publishers, 2010. – ISBN 978–1–60876–684–0

[WKL+08] WELBOURNE, Evan ; KHOUSSAINOVA, Nodira ; LETCHNER, Julie ; LI, Yang ; BALAZINSKA, Magdalena ; BORRIELLO, Gaetano ; SUCIU, Dan: Cascadia: A System for Specifying, Detecting, and Managing RFID Events. In: *Proceedings of MobiSys'08*, 2008 44

[WL05] WANG, Fusheng ; LIU, Peiya: Temporal management of RFID data. In: *Proceedings of VLDB'05*, 2005 43

[WN10] WEISS FERREIRA CHAVES, Leonardo ; NOCHTA, Zoltán: Printed Organic Smart Labels: Breakthrough Towards the Internet of Things? In: RANASINGHE, Damith C. (Hrsg.) ; SHENG, Quan Z. (Hrsg.) ; ZEADALLY, Sherali (Hrsg.): *Unique Radio Innovation for the 21st Century*. Springer, 2010. – ISBN 978–3–642–03461–9

[WOL99] WANG, Hui ; ORLOWSKA, Maria ; LIANG, Weifa: Efficient refreshment of materialized views with multiple sources. In: *Proceedings of CIKM'99*, 1999 69

[XYC+08] XIE, Junyi ; YANG, Jun ; CHEN, Yuguo ; WANG, Haixun ; YU, Philip S.: A Sampling-Based Approach to Information Recovery. In: *Proceedings of ICDE'08*, 2008 24, 43, 94

[YGYL05] YE, Wei ; GU, Ning ; YANG, Genxing ; LIU, Zhenyu: Extended Derivation Cube Based View Materialization Selection in Distributed Data Warehouse. In: *Proceedings of WAIM'05*, 2005 67, 69

[YKL97] YANG, Jian ; KARLAPALEM, Kamalakar ; LI, Qing: Algorithms for Materialized View Design in Data Warehousing Environment. In: *Proceedings of VLDB'97*, 1997 66, 68

[ZLE07] ZHOU, Jingren ; LARSON, Per-Ake ; ELMONGUI, Hicham G.: Lazy maintenance of materialized views. In: *Proceedings of VLDB'07*, 2007 69

[ZLGD07] ZHOU, Jingren ; LARSON, Per-Ake ; GOLDSTEIN, J. ; DING, Luping: Dynamic Materialized Views. In: *Proceedings of ICDE'07*, 2007 66, 68

[ZY99] ZHANG, Chuan ; YANG, Jian: Genetic Algorithm for Materialized View Selection in Data Warehouse Environments. In: *Proceedings of DaWaK'99*, 1999 66, 69

[ZZL+04] ZILIO, Daniel C. ; ZUZARTE, Calisto ; LOHMAN, Guy M. ; PIRAHESH, Hamid ; GRYZ, Jarek ; ALTON, Eric ; LIANG, Dongming ; VALENTIN, Gary: Recommending Materialized Views and Indexes with IBM DB2 Design Advisor. In: *Proceedings of ICAC'04*, 2004 66, 68, 73, 74

Lebenslauf

Daten zur Person

Name:	Leonardo Weiss Ferreira Chaves
Geburtsdatum:	29.05.1981
Geburtsort:	Rio de Janeiro, Brasilien

Ausbildung

Feb. 1986 – Sep. 2000:	Deutsche Schule in Rio de Janeiro, abgeschlossen mit der Allgemeinen Hochschulreife (Abitur).
Okt. 2000 – Okt. 2005:	Studium der Informatik (Diplom) an der Universität Karlsruhe (TH).
Apr. 2006 – Mai 2010:	Promotion an der Universität Karlsruhe (TH), Institut für Datenorganisation und Programmstrukturen (Prof. Böhm).

Berufliche Erfahrungen

Aug. 2004 – Okt. 2004:	Praktikant bei EvoBus GmbH in Portugal.
Jan. 2006 – Jul. 2009:	Doktorand bei SAP Research in Karlsruhe.
Aug. 2009 – heute:	Projektleiter bei SAP Research in Karlsruhe.

Veröffentlichungen auf Workshops

Mai 2006	ANKE, Jürgen ; MÜLLER, Jens ; SPIESS, Patrik ; WEISS FERREIRA CHAVES, Leonardo: A Service-Oriented Middleware for Integration and Management of Heterogeneous Smart Items Environments. In: *Proceedings of 4th MiNEMA Workshop*, 2006
Nov. 2006	WEISS FERREIRA CHAVES, Leonardo ; ANKE, Jürgen ; MOREIRA SÁ DE SOUZA, Luciana ; MÜLLER, Jens: Service Lifecycle Management Infrastructure for Smart Items. In: *Proceedings of MidSens'06*, 2006
Sep. 2008	WEISS FERREIRA CHAVES, Leonardo ; KERSCHBAUM, Florian: Industrial Privacy in RFID-based Batch Recalls. In: *Proceedings of InSPEC'08*, 2008

Veröffentlichungen in Buchkapiteln

Jun. 2010 WEISS FERREIRA CHAVES, Leonardo ; KERSCHBAUM, Florian: Security and Privacy in Track & Trace Infrastructures. In: BOUÇA, Duarte (Hrsg.) ; GAFAGNÃO, Amaro (Hrsg.): *Agent-Based Computing*. Nova Science Publishers, 2010. – ISBN 978–1–60876–684–0

Jul. 2010 WEISS FERREIRA CHAVES, Leonardo ; NOCHTA, Zoltán: Printed Organic Smart Labels: Breakthrough Towards the Internet of Things? In: RANASINGHE, Damith C. (Hrsg.) ; SHENG, Quan Z. (Hrsg.) ; ZEADALLY, Sherali (Hrsg.): *Unique Radio Innovation for the 21st Century*. Springer, 2010. – ISBN 978–3–642–03461–9

Veröffentlichungen auf Konferenzen

Mrz. 2008 DECKER, Christian ; BERCHTOLD, Martin ; WEISS FERREIRA CHAVES, Leonardo ; et al.: Cost-Benefit Model for Smart Items in the Supply Chain. In: *Proceedings of Internet of Things'08*, 2008.

Aug. 2008 WEISS FERREIRA CHAVES, Leonardo ; BUCHMANN, Erik ; BÖHM, Klemens: TagMark: Reliable Estimations of RFID Tags for Business Processes. In: *Proceedings of KDD'08*, 2008.

Mrz. 2009 WEISS FERREIRA CHAVES, Leonardo ; BUCHMANN, Erik ; HUESKE, Fabian ; BÖHM, Klemens: Towards Materialized View Selection for Distributed Databases. In: *Proceedings of EDBT'09*, 2009.

Mrz. 2010 WEISS FERREIRA CHAVES, Leonardo ; BUCHMANN, Erik ; BÖHM, Klemens: Finding Misplaced Items in Retail by Clustering RFID Data. In: *Proceedings of EDBT'10*, 2010.

Mrz. 2010 KERSCHBAUM, Florian ; OERTEL, Nina ; WEISS FERREIRA CHAVES, Leonardo: Privacy-Preserving Computation of Benchmarks on Item-Level Data Using RFID. In: *Proceedings of WiSec'10*, 2010.

Jun. 2010 WEISS FERREIRA CHAVES, Leonardo ; DECKER, Christian: A Survey on Organic Smart Labels for the Internet-of-Things. In: *Proceedings of INSS'10*, 2010.