Diplomarbeit

zum Thema

Analyse von Bestelldaten im Hinblick auf Taxonomien

Igor Kuferstein

Diplomarbeit
am Fachbereich Informatik
der Universität Dortmund

9. Mai 2005

Betreuer:
Prof. Dr. Katharina Morik
Dipl.-Inform. Timm Euler

Diplomarbeit

zum Thema

Analyse von Bestelldaten im Hinblick auf Taxonomien

Autor: Igor Kuferstein

Betreuer:

Prof. Dr. Katharina Morik
Dipl.-Inform. Timm Euler

1. Einleitung

In modernen Zeiten gewinnen die intelligenten Methoden der Analyse von elektronischen Verkaufsplattformen immer mehr an Bedeutung. Die Verbreitung solcher Systeme führt dazu, dass ihre Komplexität weiter steigt. Andererseits entstehen aber gerade durch die immer größer werdende Akzeptanz dieser Systeme mehr Möglichkeiten, größere Datenmengen zur Durchführung ihrer wissenschaftsbasierten Analyse zu bekommen. Viele Unternehmen, die die so genannte „electronic-Comerce“ betreiben, sind an solcher Analyse zwecks Verbesserung und Weiterentwicklung ihrer Systeme interessiert. Viele bekannte Online-Kaufhäuser setzen auf wissenschaftlich ausgearbeitete Ansätze bei der Analyse ihrer Systeme. Ein solcher vielfach bewährter Ansatz besteht in der Analyse der Bestelldaten, die bei der Nutzung eines Bestellsystems über einen längeren Zeitraum gesammelt werden. Dabei handelt es sich um große Datenmengen, deren manuelle Verarbeitung natürlicherweise nicht möglich ist. Vielmehr soll gerade die Größe der Datenmenge bei den intelligenten, rechnergestützten Verarbeitungsmethoden der Wissensentdeckung sogar ein Vorteil bieten. Sehr oft werden die Bestelldaten einer Untersuchung unterzogen, deren Ziel die Entdeckung von Zusammenhängen zwischen den bestellten Produkten ist, insbesondere der zeitlichen oder räumlichen. Manchmal, aber viel seltener, werden die semantischen Zusammenhänge der Produkte gesucht. Mit anderen Worten, man sucht die Produkte, die „zusammenhängend“ sind, oder Gruppen von Produkten bilden. Manchmal sind die Produkte bereits in solche Gruppen organisiert. Manchmal fehlt aber diese Information. In Warenwirtschaftsystemen, in denen die Produktgruppen vorhanden sind, können die Produktgruppen wiederum in größere Gruppen organisiert sein. Diese Zuordnung kann im Prinzip viele Stufen haben. Das bedeutet, dass eine hierarchische Struktur der Gruppierung vorliegt, oder eventuell gesucht wird. Besteht in dieser Hierarchie eine systematische Einordnung und Zuordnung der Unterstrukturen zu Oberstrukturen, so spricht man von einer Taxonomie, oder von einer hierarchischen Ordnung. Die Aspekte der Analyse von Bestelldateien im Bezug auf solche Taxonomien werden in dieser Arbeit untersucht.

Welche Fragen werden in der Arbeit diskutiert und wie ist die Arbeit aufgebaut?

1.1. Ziele

Der Schwerpunkt der Analyse in dieser Arbeit ist die Entdeckung von häufigen Itemsets, Assoziationsregeln und insbesondere von so genannten „generalisierten“ Assoziationsregeln[1]. Deshalb wird das Thema der Entdeckung von Assoziationsregeln und häufigen Itemsets behandelt, die eine weit verbreitete Analysemethode in der Bestelldatenanalyse darstellt. Dieses Thema wird zunächst auf dem „hierarchielosen“ Niveau eingeführt und später im Bezug auf Taxonomien vertieft. Um an dieser Stelle den Begriff „generalisierte“ Regeln kurz vorzustellen, kann man sagen, dass es solche Regeln sind, die mit Einbezug von Taxonomien erzeugt werden.

Die generalisierten Assoziationsregeln und ihre Bedeutung werden unter verschiedenen Blickwinkeln diskutiert. Unterschiedliche Konzepte der Generalisierungen und Ansätze zu Definition und Entdeckung der generalisierten Regeln werden vorgestellt. Ein

[1] Definitionen der Begriffe s. in Kapitel 3.1.1

Bewertungsmaß der Generalisierung im Zusammenhang mit verschiedenen Gruppierungsmöglichkeiten wird untersucht.

Ferner wird ein Verfahren entwickelt, das auf der Basis von häufigen Itemsets eine Bildung von neuen oder Überprüfung von vorhandenen Taxonomien ermöglicht. Die Aspekte der Regelentdeckung bei parallel vorhandenen Taxonomien von verschiedenen Attributen werden untersucht. Es wird versucht, die Rolle der Taxonomien und die Vor- und Nachteile der Analyse mit und ohne Taxonomien zu diskutieren. Die Schwierigkeiten, neue Erkenntnisse und zu beachtenden Aspekte werden diskutiert. Die theoretischen Überlegungen werden größtenteils mit praktischen Experimenten ausprobiert.

Zusammengefasst kann man die wesentlichen Fragen, die in der Arbeit behandelt werden, so formulieren:

1. Was ist der aktuelle Wissensstand auf dem Gebiet der Bestelldatenanalyse und insbesondere der Entdeckung von häufigen Itemsets und Assoziationsregeln im Hinblick auf Taxonomien?
2. Welche Arten von generalisierten Regeln werden unterschieden und welche Ansätze gibt es bei ihrer Entdeckung?
3. Wie können die Regeln bewertet und verbessert werden?
4. Welche Vorteile bzw. Nachteile bieten die Taxonomien in Zusammenhang mit Entdeckung und Analyse der Regeln?
5. Wie können parallel vorhandene Taxonomien benutzt werden?
6. Wie können Hierarchien überprüft oder neu gebildet werden?
7. Wie können die Ergebnisse der Analyse praktisch eingesetzt werden?

1.2. Aufbau

Zuerst wird das Informationssystem beschrieben, das einerseits die Datenquelle für diese Arbeit darstellt, und andererseits als Integrationsplattform für die praktische Experimente und Weiterentwicklung dienen soll. Das Informationssystem wurde von der Firma Intensis GmbH entwickelt, deren kurze Beschreibung sowie ein Überblick über das Informationssystem und seine Bestellsystem-Komponente im Anfangskapitel gegeben werden. Einige Überlegungen im Hinblick auf den Einsatz der Ergebnisse der Arbeit werden gesammelt.

Danach folgt ein theoretischer Teil, in dem mehrere wichtige Arbeiten auf dem Gebiet der Entdeckung von Assoziationsregeln im Überblick vorgestellt werden. Verschiedene Konzepte und Ansätze für die Entdeckung der generalisierten Assoziationsregeln werden dabei diskutiert. Der für die Entdeckung der Assoziationsregeln meist angewandte Algorithmus „Apriori“ wird erklärt. Er wird als Kernalgorithmus für den praktischen Teil der Arbeit fungieren. Am Anfang dieses theoretischen Teils werden fachliche Begriffe und Definitionen eingeführt. Danach werden die Ansätze verschiedener Autoren diskutiert. Dabei wird jeder Ansatz in mehreren Teilen vorgestellt, die eine kurze Motivation, den Ansatz selbst und die abschließende Diskussion beinhalten. Die Besonderheiten einiger Ansätze werden in expliziten Unterkapiteln vorgestellt. Darauf folgend werden die Aspekte der Interessensmaße der gefundenen Regeln und deren Filterungsmöglichkeiten vorgestellt.

Nach dem größeren theoretischen Teil folgt ein Kapitel, das eine Planung und einführende Beschreibung der praktischen Experimente beinhaltet. Es wird festgelegt, welche Experimente durchgeführt werden können und was diese bezwecken. Anschließend folgt ein Abschnitt, der über die durchgeführten Experimente berichtet und deren Ergebnisse analysiert.

Die Arbeit schließt eine Zusammenfassung und Ausblick ab, wo rückblickend die Ergebnisse der Arbeit vorgestellt und Überlegungen für weitere mögliche Entwicklung gemacht werden.

2. Systembeschreibung

2.1. Allgemeine Kurzbeschreibung des Intensis I2S Systems

Die Intensis GmbH, Dortmund, ist ein Unternehmen, das sich hauptsächlich mit der Entwicklung interaktiver Informationssysteme für mittelständische Unternehmen und Konzerne beschäftigt. Diese Systeme beinhalten einen zentralen Kern, der nahezu bei allen realisierten Projekten als Standardteil des Informationssystems eingesetzt wird, die sogenannten „Standard Services“, sowie weitergehende, speziell für Kundenwünsche „maßgeschneiderte“ Services, die sogenannten „Professional Services“. Die Struktur eines solchen Projektes zeigt Abbildung 1:

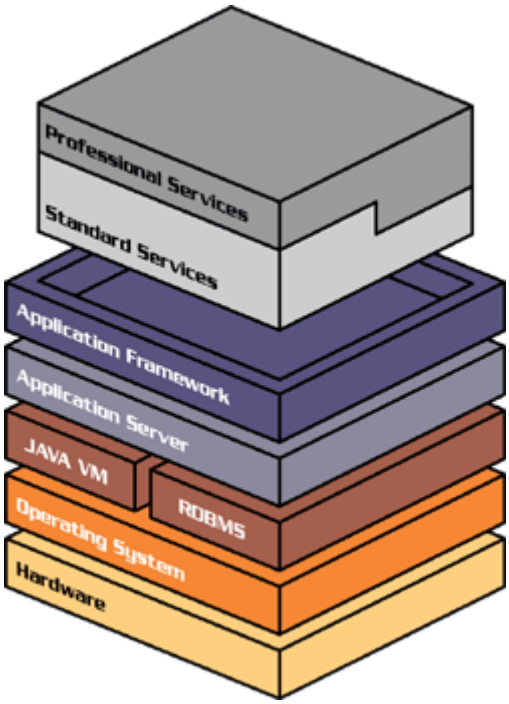


Abbildung 1 Struktur des I2S Systems

Die unteren Basis-Schichten des Systems sind die Hardware und das Betriebssystem. Darüber sind Java Virtual Machine und eine relationale Datenbank angesiedelt. Auf diese stützt sich ein Applikationsserver mit einem Framework. Das System, das den Namen „I2S“[2] trägt, greift auf die Funktionalitäten und Dienste der darunter liegenden Schichten über das Framework zu. Auf der Benutzerseite erfolgt der Zugriff auf die Funktionalität des Systems völlig ohne ein spezielles Clientprogramm und wird über einen konventionellen Web-Browser realisiert. Die Standardservices sind im Wesentlichen ein Content Management System. Dieses ist sehr komplex, aber flexibel ausgelegt und wird mit Hilfe einer speziellen Applikation, des sogenannten Redaktionssystems, mit diversen Inhalten, im Wesentlichen aber mit Dokumenten, gefüllt. Die Dokumente, aber auch andere Inhalte, werden bestimmten Menüs zugeordnet. Die Menüs sowie die Dokumente haben ein sehr kompliziertes und

[2]Der Name des Systems “I2S” entstand als Abkürzung von „individuality to standard”.

mehrstufig differenziertes Konzept der Zugriffsrechte und Sichten. Außerdem werden viele Dokumente mehrsprachig im System verwaltet. Dieses System kann von einem Unternehmen genutzt werden, das beispielsweise mehrere hundert Mitglieder und Standorte umfasst. Dabei kann jedes Mitglied sehr viele Benutzer haben. Die Mitglieder sind in verschiede Gruppen unterteilt. Die Benutzer sind bestimmten Profilen zugeordnet. Es können verschiedene Netze angelegt und bestimmten Mitgliedern zugeordnet werden. Die Menüs können für bestimmte Mitglieder, Mitgliedergruppen, Netze, Profile oder einzelne Benutzer „freigeschaltet“ oder auch „nicht zugänglich“ gemacht werden.

Die kundenspezifisch entwickelten Applikationen sind die so genannten Professional Services, die ins System integriert sind und über die gleiche Rechte- und Menüstruktur wie die Dokumente verwaltet werden. Die Professional Services können unterschiedlichen Zwecken dienen. Zum Beispiel kann eine Lieferantendatenbank ein solcher Professional Service sein, die dem Benutzer eine komfortable und sehr differenzierte Suche nach Lieferanten ermöglicht. Eine andere mögliche Applikation kann beispielsweise Umsätze und Konditionen zwischen verschiedenen Lieferanten und Mitgliedern erfassen und verwalten. Ein elektronisches Bestellsystem, das sogenannte Lagerbestellsystem, das bei einem der Kunden im Einsatz ist, ist eine e-Commerce[3] Komponente und stellt ein weiteres Beispiel eines Professional Services dar.

2.2. e-Commerce Komponente bei NIS: Lagerbestellsystem

Das Lagerbestellsystem ist ein Bestandteil des „Nordwest Informationssystems“ (NIS) und seit ca. 2,5 Jahren in Betrieb. Das NIS wurde von der Intensis GmbH für die NORDWEST Handel AG entwickelt und 2001 in Betrieb genommen.

Die NORDWEST Handel AG agiert europaweit als Einkaufs- und Serviceunternehmen für 480 Gesellschafter mit 750 Standorten auf dem Markt der Bauelemente, Haustechnik, Werkzeuge, Stahl- und Eisenwaren. Der zentral fakturierte Jahresumsatz 2003 lag bei 2.068,2 Mio. Euro, der Außenumsatz bei 5.907 Mio. Euro. NORDWEST ist das Bindeglied zwischen den angeschlossenen Fachgroßhändlern und den Herstellern. Das Kerngeschäft besteht in der Zentralregulierung, im Strecken- und Lagergeschäft. Daneben werden Dienstleistungen in den Bereichen Marketing, Betriebsberatung, Logistik und Finanzen angeboten.

Das Lagerbestellsystem ist ein interaktives Bestellsystem, das unter anderem über Bestandsdaten des NORDWEST-Zentrallagers informiert und eine differenzierte Suche mit vielen Suchkriterien und deren Verknüpfungen sowie eine komfortable Bestellung von

[3] Eine der möglichen Definitionen für e-Commerce wird in [Microsoft Encarta, 2005] gegeben:
„E-Commerce, auch Electronic Commerce, aus dem Englischen stammender Begriff für „elektronischer Handel”, also die elektronische Unterstützung und Abwicklung von Geschäftsprozessen über das Internet. Allgemein lassen sich vier Kommunikationsebenen von E-Commerce unterscheiden:

1. Unternehmer mit Unternehmer (B2B, business to business)
2. Einzelhändler mit Privatkunden (B2C, business to consumer)
3. Unternehmen mit öffentlichen Einrichtungen (B2PA, business to public administration)
4. Konsumenten mit öffentlichen Einrichtungen (C2PA, consumer to public administration)

Dem E-Commerce werden der gesamte Geschäftsprozess, von Werbung, Geschäftsanbahnung und -abwicklung bis hin zum Kundenservice … zugerechnet.“
In dieser Arbeit wird mit dem Begriff „e-Commerce“ die Kommunikationsebene 1. dieser Definition gemeint.

Artikeln ermöglicht. Umfang und Funktionalität des Systems sind ziemlich breit und komplex, wobei nur ein Teil davon für die vorliegende Arbeit relevant sein wird. Die mit dem System erzeugbaren „Warenkörbe“, die auf einen Knopfdruck zu einer Bestellung werden, die bereits ausgeführten und protokollierten Bestellungen samt ihren Daten sowie die Suchfunktion werden im Folgenden von zentraler Bedeutung sein.

Wie entsteht im NIS Lagerbestellesystem eine Bestellung? Der simpelste Ablauf ist wie folgt:

Der Kunde führt entsprechend seinem Bedarf eine Suche der benötigten Artikel nach bestimmten Suchkriterien durch, die einzeln oder kombiniert verwendet werden können (Fall 1), oder gestaltet die Suche über Katalog (Fall 2). Die Suchmaske zeigt Abbildung 2. Bei der Katalogsuche navigiert der Benutzer einfach durch die vorgegebene Hierarchie.

Abbildung 2 Suchmaske

Abbildung 3 Katalogsuche

Die oberste Ebene der Hierarchie, nämlich die Warengruppen, ist auf der Abbildung 3 zu sehen. Diese werden in Produktgruppen unterteilt, denen dann die einzelnen Artikel zugeordnet sind.

Die Suche ergibt eine Liste, die alle Artikel enthält, bei denen die eingegebenen Suchkriterien zutreffen (im Fall 1), oder die Artikel, die die unterste Ebene der Hierarchie bilden, über die der Benutzer sich bewegt hat(im Fall 2). Aus dieser Liste (Abbildung 4) kann der Benutzer die einzelnen für ihn interessant erscheinenden Artikel detailliert betrachten (Abbildung 5), oder sie direkt in einen neuen oder früher gespeicherten virtuellen Warenkorb legen (Abbildung 6). Den Warenkorb kann man bearbeiten, mit diversen speziellen Optionen versehen, speichern oder direkt bestellen. Das ist der grobe Ablauf eines Bestellprozesses. Es gibt noch viele weitere Zusatzmöglichkeiten und Optionen im System, auf die hier nicht detailliert eingegangen wird, da sie zu komplex und für diese Arbeit nicht relevant sind. Hier werden nur einige davon erwähnt. Zum Beispiel kann man die früher ausgeführten Bestellungen ansehen und gegebenenfalls reanimieren (sozusagen „klonen“). Eine andere interessante Option ist, die Artikel zwischen verschiedenen Warenkörben zu bewegen. Es ist sogar möglich, eine Datei hochzuladen, aus der dann automatisch ein Warenkorb erstellt wird.

Lagerbestellsystem

1 2 3

Anzahl der insgesamt gefundenen Artikel: 44

Artikel-Nr.	Artikelbezeichnung	EK	PE	Bestand	FB	VPE	Menge
2000209007	2-TASTEN-MINI-HANDSENDER,HSM 2, 868,3 MHZ	33,00	1	16,0	BE	1,0	
2000209008	2-TASTEN-MIKRO-HANDSENDER,HSE 2, 868,3 MHZ	33,00	1	10,0	BE	1,0	
2000209021	ANTRIEBSKOPF SUPRAMATIC E,868,3 MHZ	199,00	1	13,0	BE	1,0	
2000209022	ANTRIEBSKOPF PROMATIC,868,3 MHZ	135,00	1	10,0	BE	1,0	
2000209170	NOW HANDSENDER 868 MHZ	29,00	1	0,0	BE	1,0	
3000255160	RAUCHMELDEZENTRALE RMZ-K	281,68	1	2,0	BB	1,0	
3000262440	STEUERUNG MODI 2650	60,20	1	5,0	BB	1,0	
3000262445	STEUERUNG TROLL COMFORT	54,95	1	12,0	BB	1,0	
3000262446	STEUERUNG TROLL STANDARD	35,50	1	5,0	BB	1,0	
3000262447	STEUERUNG TROLL ECO	24,95	1	2,0	BB	1,0	
3000262455	STEUERUNG F. ROHRMOTOREN,CLIP	60,50	1	4,0	BB	1,0	
3000262456	SONNEN/DAEMMERUNGSMO 3770	25,00	1	3,0	BB	1,0	
3000262457	SONNEN/DAEMMERUNGSSE 3720	15,20	1	5,0	BB	1,0	
3000262460	IR-HANDSENDER	32,40	1	0,0	BB	1,0	
4000353501	KLEBEPISTOLE PATTEX PXP2K,230V,KOFFER,6 PATR.12 MM,SUPERMATIC 200 PLUS	36,85	1	5,0	WA	1,0	

Warenkorb Nr. Neu ··› Hinzufügen ··› Neue Suche

Anzahl der insgesamt gefundenen Artikel: 44

Abbildung 4 Suchergebnisliste

Produktgruppe: "Clip"

Steuerung für Rohrmotoren "CLIP"

programmierbare Steuerung für motorbetriebene Rollläden problemlose Integration in das bestehende Schalterprogramm mit Installationsgehäuse, 3-Draht-Technik ohne Null-Leiter weiß Zufallsgenerator Tippbetrieb Sonnen- und Dämmerungsmodul mit IR-Empfänger nachrüstbar

Ausführung: S+D Modul 3770

Artikel-Nr.:	3000262456	EK:	25,00
Artikelbezeichnung:	SONNEN/DAEMMERUNGSMO 3770	PE:	1
		VPE:	1,00
FB:	Beschläge	Mengeneinheit:	Stück
EAN:		Empf. VK:	36,25
		Bestand:	**0,0**

Fenster schließen

Abbildung 5 Details zu einem Produkt

Warenkorb Nr. 23955

Kundenauftrags-/Warenkorb-Nr.: Zeichen:

Versandart: LKW (Standardtour) Versand-Datum:

Lieferoptionen: keine

Neutrale Lieferscheine:
Schriftliche Bestätigung:
Lieferung nur komplett:
Ohne Rückstands-Notierung:

Hinweis: bitte benutzen Sie als Dezimaltrenner bei der Mengeneingabe ein Komma, falls nötig, und keinen Punkt!
Falls Sie aus Versehen eine negative Zahl eingeben, wird nur der absolute Betrag als gültige Eingabe übernommen.

Status	Artikel-Nr.	Artikelbezeichnung	EK	Menge	Gesamtpreis	Kd-Art-Nr.	Löschen	Verschieben
●	5000622272	HOLZSCHR.571-4.6 ZN 8X30,6KT-KOPF	1,92	2,0	0,04			
●	5000622274	HOLZSCHR.571-4.6 ZN 8X40,6KT-KOPF	2,11	2,0	0,04			
●	5000622208	HOLZSCHR.571-4.6ZN 8X 55,6KT-KOPF	3,09	2,0	0,06			
●	5000622210	HOLZSCHR.571-4.6ZN 8X 65,6KT-KOPF	3,28	2,0	0,07			
●	5000622280	HOLZSCHR.571-4.6 ZN 8X90,6KT-KOPF	3,96	2,0	0,08			
●	5000622321	HOLZSCHR.571-4.6ZN 12X280,6KT-KOPF	35,59	2,0	0,71			
●	5000622338	HOLZSCHR.571-4.6ZN 16X160,6KT-KOPF	32,35	2,0	0,65			

Folgenden Artikel zum Warenkorb zusätzlich hinzufügen

Artikel-Nr.	Menge	Kd-Art-Nr.

Gesamtbetrag 1,65

Information:
Hier können Sie Ihre Mengeneingaben ändern, einen Warenkorb speichern, einzelne Positionen aus dem Warenkorb entfernen oder den Warenkorb komplett löschen

Verwerfen Speichern

Bestellen

Abbildung 6 Warenkorb

Das System ist kein sehr großes Bestellsystem, das mit Systemen wie „www.amazon.de“ oder ähnlichen mithalten könnte. Es werden täglich lediglich 50 bis 130 Bestellungen ausgeführt, wobei ein Wachstum der Nutzung natürlich sehr erwünscht ist. Seit Inbetriebnahme des Lagerbestellsystems bis zum jetzigen Zeitpunkt (01.04.2005) sind insgesamt ca. 60.000

Bestellungen ausgeführt worden. Zur Auswahl stehen ca. 25.000 Artikel. Diese können von jedem Benutzer, der auf das Lagerbestellsystem zugreifen und die Bestellfunktion nutzen darf, in beliebigen Kombinationen bestellt werden.

Dieses Effizienz dieses Systems soll mit Hilfe der „einfachen“ und der „generalisierten“ Assoziationsregeln[4] gesteigert werden. Was darunter zu verstehen ist, wird in Kapitel 2.3 diskutiert. Aber zunächst sollen hier noch Artikelattribute sowie Kundenattribute beschrieben werden, die später von Bedeutung sein können.

2.2.1. Artikelattribute

Die Artikel in diesem Lagerbestellsystem haben viele Attribute. Manche sind nur für die interne Verarbeitung interessant, manche könnten aber eventuell für die spätere Verwendung bei Hierarchienbildung und Regelentdeckung von Bedeutung sein (siehe Kapitel 4.3 und 5.4). Nachfolgend sind von den letztgenannten einige wichtige aufgelistet und kurz erklärt:

- Artikelnummer (Artikel-ID): Die eindeutige Artikelidentifikationsnummer. Zwar ist diese 10-Stellig, aber 6 Stellen der Nummer sind immer genug, um eine eindeutige Identifizierung des Artikels zu ermöglichen. Deshalb enthalten die Bestelldateien und später die daraus erzeugten Transaktionen im Sinne von Assoziationsregelentdeckung nur die 6-Stellige Artikelnummer.
- Artikelbezeichnung: Ein textueller Bezeichner für einen Artikel, oder kurz der Artikelname.
- Einkaufspreis: Dieser Preis bildet die Basis für spätere Preisberechnungen.
- Verkaufspreis: Der für Wiederverkäufer empfohlene Verkaufspreis.
- Diverse Teuerungszuschläge: Diese werden ebenfalls für die Preisbildung verwendet.
- Preiseinheit: Anzahl der Artikel, für die der Preis angegeben wird, z. B. 1, 100, 1000.
- Mengeneinheit: Einheit der Menge, wie Stück, Kilo, Sack, Tonne etc., die bei den Mengeneingaben gemeint sind.
- Verpackungseinheit: Anzahl der Artikel in einer Verpackung.
- Besonderheiten: Spezielle technische und andere Besonderheiten des Artikels.
- Zusätzliche Artikelbezeichnungen: Es sind für den Bedarfsfall zwei vorgesehen.
- Artikelbeschreibung: Ausführlichere Artikelbeschreibung.
- Produktgruppe: Eine Sammelbezeichnung für die Gruppe der ähnlichen Artikel. Sie dient der Hierarchiebildung, und zwar der Bildung der zweiten Stufe der Hierarchie; die Artikel sind immer einer Produktgruppe zugewiesen.
- Fachbereich: Zugehörigkeit zu einem Fachbereich. Alle Produkte sind einem Fachbereich zugewiesen. Eine andere Hierarchie könnte parallel auch über dieses Attribut gebildet werden. Die Fachbereiche sind im Wesentlichen die Warengruppen, die die Oberbegriffe für die Produktgruppen sind. Die Warengruppen werden auch für die Hierarchiebildung verwendet.
- Abbildungsdatei: Bei den meisten Artikeln ist für die Detaildarstellung eine Bilddatei vorhanden. Oft ist dieselbe Abbildung bei mehreren Artikeln zu finden. Das bedeutet,

[4] Diese Begriffe werden in späteren Kapiteln definiert und ausführlicher erklärt (Für grundlegende Begriffe und Definitionen s. Kapitel 3.1.1).

dass die Artikel sehr ähnlich sind und sich nur in kleinen Details unterscheiden, also höchstwahrscheinlich zu derselben Gruppe gehören.

2.2.2. Kundenattribute

Die Käufer, die im System bestellen, sind keine privaten, sondern gewerblichen Kunden. Deshalb ist das Kaufverhalten der Kunden anders als bei Privatkäufern. Es ist deswegen unwichtig, welcher Benutzer, also Mitarbeiter einer Firma, die im System bestellen darf, die Bestellung aufgibt. Bei einer Bestellung ist immer die ganze Firma als Kunde zu sehen. Die Firmen, als Mitglieder im System genannt, haben viele Attribute. Folgende davon könnten aber interessant sein:

- Mitgliedsnummer (Mitglieds-ID): Identifizierungsnummer des Mitglieds, 15-stellig.
- Name: Der Mitgliedsname; es gibt noch zusätzliche, kurze Namen.
- PLZ: Die Postleitzahl ist ein passendes Attribut, um eine Unterteilung der Mitglieder möglich zu machen.
- Stadt.
- Land.

Die Mitglieder gehören außerdem bestimmten Mitgliedergruppen an. Die Informationen können für die Hierarchiebildung über die Kundenattribute benutzt werden. Insbesondere sind die Stadt, die PLZ und die Gruppenzugehörigkeit für die Gruppierung der Kunden interessant (siehe Kapitel 4.3 und 5.4).

2.2.3. Hierarchien

Wie bereits oben kurz beschrieben, gehören Artikel zu bestimmten Produktgruppen. Diese Produktgruppen wiederum gehören zu bestimmten Warengruppen. Somit ist eine Hierarchie gegeben, die aus drei Stufen besteht. Die unterste Stufe sind Artikel, die mittlere machen die Artikelgruppen aus und die höchste besteht aus Warengruppen. Parallel dazu gibt es eine Zugehörigkeit der Artikel zu sogenannten Fachbereichen. Diese sind den Warengruppen sehr ähnlich. Der Unterschied liegt nur in der Anzahl der Warengruppen und der Fachbereiche (es gibt etwa dreimal soviel Warengruppen wie Fachbereiche). Die Unterteilung in die Warengruppen ist also etwas feiner.

Die Warengruppen kann man auf der Abbildung 3 oben sehen. Die Fachbereiche sind auf der Abbildung 7 sichtbar. Vergleicht man die beiden Abbildungen, so sieht man die Gemeinsamkeiten und Unterschiede.

Abbildung 7 Fachbereiche

2.2.4. Bestelldateien

Die Bestellungen werden sowohl in der Datenbank, als auch als Dateien im System festgehalten. Allerdings werden die Bestelldaten aus der Datenbank gelöscht, sobald der Kunde seine bereits ausgeführte Bestellung, die er immer noch in der protokollierten Form betrachten kann, selber löscht. In diesem Fall sind die Bestelldaten nur als Dateien vorhanden. Diese Dateien werden im System (theoretisch) „für immer“ festgehalten und werden gar nicht gelöscht. Deshalb löhnt es sich nicht, den Datenbankdatenbestand an Bestellungen zu verarbeiten, da er auf jeden Fall nicht vollständig und sehr dynamisch ist. Deswegen werden in der vorliegenden Arbeit die Bestelldateien benutzt bzw. weiter verarbeitet. Für jede Bestellung existiert genau eine Datei.

Eine Bestelldatei enthält pro Artikel eine Zeile (s. Beispiel unten), die unter Anderem folgende Informationen beinhaltet:

- Artikelnummer (6-Stellig)
- Menge
- Mitglieds-ID (diese ist natürlich in allen Zeilen einer Datei gleich)
- Spezielle Verpackungs-, Versand- und Lieferungsanweisungen
- Vermerke und Zeichen
- Bestelldatum

Für die Arbeit sind zu mindest Artikelnummer und eventuell Mitgliedsnummer sowie Bestelldatum relevant. Um die benötigten Informationen zu gewinnen, müssten die Dateien geparst werden.

Hier ist als Beispiel ein Teil des Inhaltes einer Bestelldatei dargestellt:

000025076 03001130504	**81853000001000000100**	**V0413**
000025076 03001130504	**81012300000500000050**	**V0413**
000025076 03001130504	**81022300000500000050**	**V0413**
000025076 03001130504	**81664500000600000060**	**V0413**
000025076 03001130504	**85500700000500000050**	**V0413**

Aus diesen Daten werden Zeile für Zeile die Daten einer einzelnen Transaktion gewonnen.

2.3. Potenzielle Vorteile beim Einsatz der Assoziationsregeln und generalisierten Assoziationsregeln. Integrations- / Nutzungsmöglichkeit.

Um die Verbesserungspotenziale auszuschöpfen, muss man sie zunächst entdecken.

Dabei ist das Hauptziel eines jeden Unternehmens natürlich die Gewinnmaximierung. Da der Gewinn vom Umsatz abhängt, will man versuchen, den Umsatz zu steigern.

In dem konkreten Fall der NORDWEST Handel AG macht das Lagerbestellsystem noch keinen entscheidenden Umsatzanteil aus. Man will versuchen, das Potenzial dieses Systems und die Vorteile des „Online-Geschäftes“ auszunutzen, die unter anderem darin bestehen, dass Kunden direkt angesprochen und auf bestimmte Waren aufmerksam gemacht werden können. Die genauere und differenzierte Untersuchung des Kaufverhaltens von Kunden ermöglicht ziemlich genaue Vorhersagen über die für die Kunden potenziell interessanten Artikel oder Artikelgruppen, die dann den Kunden vorgeschlagen werden können. Andererseits ist ein Kunde sich vielleicht über bestimmte Artikel bzw. Artikelkategorien oder -gruppen gar nicht bewusst und kann darauf hingewiesen werden, dass er in diesen Bereichen bisher noch keine Kaufaktivitäten getätigt hat. Die Kaufempfehlungen können bei der Gestaltung von Newsletter berücksichtigt werden. Hat der Kunde einen Newsletter abonniert, wird er die Kaufempfehlungen mit großer Wahrscheinlichkeit auch lesen. Dieses ist eine gute Werbemöglichkeit, bei der solche Artikel vorgeschlagen werden, die den Kunden wahrscheinlich interessieren werden.

Eine andere Optimierungsmöglichkeit eröffnet sich, wenn strukturierte und geordnete Kaufinteressen von Kunden vorliegen. So kann die Angebots- oder Sonderangebotsgestaltung an die Assoziationsregeln angepasst werden. Außerdem kann die Lagerhaltung bzw. die Nachbestellung der Artikel besser organisiert und geplant werden. Die bei der Bildung der Assoziationsregeln berechneten Supportwerte[5] geben dem Manager auch einen guten Überblick bei der ABC-Analyse[6] des Angebotsprogramms. So kann die Artikelpalette optimiert werden, indem die Artikel mit kleinem Support eventuell aus dem Programm herausgenommen und dadurch diverse Fixkosten erspart werden. Die Schwellgrenze „kleiner Support“ ist für den Manager ein frei definierbarer Wert.

Einen weiteren sehr wichtigen Nutzen sollen die generalisierten Assoziationsregeln bringen, die den Hauptaspekt dieser Arbeit ausmachen werden. Über die gefundenen generalisierten Regeln verschiedener Art wird die Betrachtung der Kundeninteressen auf einem höheren

[5] Definition zum Support s. in Kapitel 3.1.1

[6] Die ABC-Analyse ist die praktische Anwendung der Pareto-Verteilung im Rahmen betriebswirtschaftlicher Analysen, ein Verfahren, um wichtige Artikel zu identifizieren. Dabei erfolgt eine Einteilung in unterschiedliche Klassen. Ziele: das "Wesentliche" vom "Unwesentlichen" trennen, Rationalisierungschwerpunkte setzen, unwirtschaftliche Anstrengungen vermeiden, die Wirtschaftlichkeit steigern
Die gängige Aufteilung sieht die Bildung jeweils einer A-, B- und C-Klasse vor, woher das Verfahren seinen Namen hat. Das Einsatzgebiet der ABC-Analyse ist vielfältig; so werden Kunden nach ihrem Umsatzanteil, Produkte nach ihren Verkaufszahlen bzw. ihrer Drehgeschwindigkeit oder Lieferanten nach ihrem Einkaufsvolumen klassifiziert. Aber auch in der Lagerhaltung werden mit Hilfe dieses Verfahrens A-, B- und C-Plätze identifiziert (s.[Wikipedia]).

Abstraktionsniveau möglich. Außerdem werden neue Gruppierungen der Artikel möglich, die wiederum zum besseren Verständnis des Kundenkaufverhaltens beitragen (s. dazu Kapitel 4.3 bzw. 5.9) und eine Verbesserung bzw. Verfeinerung der vorhandenen Gruppenbildung ermöglichen.

3. Literaturüberblick

In diesem Kapitel werden mehrere wichtige Arbeiten auf dem Gebiet der Entdeckung von Assoziationsregeln im Überblick dargestellt. Insbesondere werden verschiedene Ansätze für die Entdeckung der generalisierten Assoziationsregeln diskutiert. Der für die Assoziationsregelentdeckung meist angewandte Algorithmus „Apriori“ wird erklärt. Er wird als Kernalgorithmus für den praktischen Teil der Arbeit fungieren. Die konzeptuellen Gedanken, die für den praktischen Teil der Arbeit relevant sind und seine theoretische Grundlage bilden sollen, werden hier gesammelt.

In verschiedenen Literaturquellen werden viele Begriffe mit gleicher oder ähnlicher Bedeutung unterschiedlich bezeichnet. Um einen einheitlichen Formalismus im weiteren Verlauf der Arbeit zu ermöglichen, werden die grundlegenden Begriffe und Definitionen nach der zunächst kurzen Vorstellung der Assoziationsregeln eingeführt.

3.1. Assoziationsregeln

Das Problem der Entdeckung von Assoziationsregeln ist seit längerer Zeit bekannt und wurde erstmals in [Agrawal et. al. 1993] behandelt. Was sind die Assoziationsregeln? Eine sehr allgemeine Formulierung wäre: „Assoziationsregeln sind ein Modell der Abhängigkeiten zwischen verschiedenen Ereignissen“.

Gegeben sei eine große Datenbank mit Verkaufstransaktionen. Jede Transaktion besteht aus Artikel (Items), die von einem Kunden in ein und demselben Einkaufsvorgang gekauft wurden. Eine Regel ist dann ein Ausdruck der Form $X \Rightarrow Y$, wobei X und Y Artikelmengen in den Transaktionen sind. Intuitiv versteht man die Bedeutung einer solchen Regel als folgende Aussage: „Die Transaktionen, die die Artikelmengen X enthalten, enthalten auch Artikelmengen Y.“

Nun soll ein Beispiel die Vorstellung von Assoziationsregeln im Vorfeld ermöglichen, bevor diese weiter unten viel ausführlicher betrachtet werden.

Angenommen, dass folgende Aussage gegeben ist: „Kunden, die ein Notebook gekauft haben, haben auch eine Notebook-Maus und eine Notebook-Tasche gekauft“. Das ist eine mögliche Assoziationsregel, die allerdings (noch) nicht genau besagt, wie oft die Notebooks insgesamt gekauft wurden und bei wie vielen Kunden, die eins gekauft haben, diese Aussage zutrifft. Diese quantitativen Informationen über die Regeln werden zusätzlich zu den häufigen Itemsets und den Regeln durch einen Algorithmus gefunden, der weiter unten dargestellt wird. Dieser Algorithmus heißt „Apriori“. Der Algorithmus generiert die signifikanten Assoziationsregeln zwischen den Artikeln in der Datenbank anhand der vorhandenen Transaktionen.

3.1.1. Grundlegende Begriffe und Definitionen

Für die späteren Beschreibungen und Erklärungen müssen jetzt einige Begriffe eingeführt und definiert werden.

Sei **gegeben**:

$\mathcal{I}$ eine Menge aller potenziell möglichen Elemente (Items).

D die Menge der Transaktionen über $\mathcal{I}$

Definition 1

Eine Menge $X = \{i_1, ..., i_k\} \subseteq \mathcal{I}$ wird **Itemset**, oder ein

k-Itemset genannt, wenn es k Elemente enthält.

Man sagt, eine Transaktion T=(*tid*, *I*) mit Elementen aus $\mathcal{I}$ „**unterstützt**" (im Original „supports", s. [Agrawal et al., 1993]) ein Itemset $X \subseteq \mathcal{I}$, wenn sie dieses Itemset enthält: $X \subseteq I$, wobei T ein Tupel, *tid* ein Transaktions-Identifikator und *I* die Menge aller Elemente (Items) in dieser Transaktion sind.

Definition 2

Der **Cover** von einem Itemset *X* in *D* besteht aus der Menge der Transaktions-Identifikatoren der Transaktionen in *D*, die *X* unterstützen:

$$cover(X, D) := \{tid \mid (tid, I) \in D, X \subseteq I\}$$

Definition 3

Der **Support** von einem Itemset *X* ist die Anzahl der Transaktionen in *cover* von X in D

$$support(X, D) := |cover(X, D)|$$

Mit anderen Worten, der Support ist die Anzahl der Transaktionen, die das Itemset *X* enthalten.

Definition 4

Die **Frequenz** (im Original „frequency", s. [Agrawal et al., 1993]), oder der **relative Support** von einem Itemset *X* in *D*, ist die Wahrscheinlichkeit, dass das Itemset *X* in der Transaktion $T \subseteq D$ vorkommt.[7]

[7]Dabei wird sehr oft in der Literatur genau diese Definition der Frequenz für die Definition des Begriffs „Support" benutzt. Der einzige Unterschied von der oberen Definition von Support ist, dass der Support oben die absolute Anzahl der Transaktionen bedeutet, die das Itemset beinhalten, während die Frequenz, oder der Support bei manchen Autoren, die relative Anzahl von solchen Transaktionen darstellt, gemessen an der gesamten Anzahl von Transaktionen.

$$frequency(X,D) := P(X) = \frac{support(X,D)}{|D|}$$

Definition 5

Ein Itemset ist **häufig** (im Original „frequent"), wenn sein Support eine vorgegebene Untergrenze für Support, den **minimum-Support** (oder **minsup**), nicht unterschreitet. Wenn es mit dem absolutem Support ausgedrückt wird, sei er als σ_{abs} bezeichnet, dann

$0 \le \sigma_{abs} \le |D|$,

wenn es mit dem relativen Support, bezeichnet als σ_{rel}, ausgedrückt ist, dann

$0 \le \sigma_{rel} \le 1$,

Aus den oberen Definitionen ist es ersichtlich, dass $\sigma_{abs} = \sigma_{rel} \cdot |D|$ ist. In der Algorithmusbeschreibung unten wird ständig der absolute Support gemeint, deshalb wird der Index „abs" weggelassen und der minsup mit σ bezeichnet.

Die **Menge aller häufigen Itemsets** *F* ist dann wie folgt definiert:

Definition 6

$F(D,\sigma) := \{X \subseteq \mathcal{I} \mid \text{support}(X,D) \ge \sigma\}$

Es wird der **Begriff** der **Itemset Mining** eingeführt:

Seien eine Menge der möglichen Items $\mathcal{I}$, Transaktionsdatenbank *T* über $\mathcal{I}$ und der minsup σ gegeben. Finde $F(D,\sigma)$, oder verkürzt *F*, sowie den jeweiligen Support von den häufigen Itemsets.

Jetzt kann man **den Begriff der Assoziationsregel** genauer definieren:

Definition 7

Eine **Assoziationsregel** (im klassischen Sinne) ist ein Ausdruck der Form $X \Rightarrow Y$, wo *X* und *Y* die Itemsets sind, und $X \cap Y = \{\}$. So eine Regel bedeutet, dass wenn eine Transaktion einen Itemset *X* enthält, enthält sie auch einen Itemset *Y*.

Dabei bezeichnet man *X* als **Body** und *Y* als **Head**.[8]

[8] Bemerkung: in der Literatur werden oft die Begriffe „**Hypothese"** oder **„Voraussetzung"** anstatt „**Body",** und „**Konklusion"** oder **„Folgerung"** anstatt „**Head"** verwendet. Die Äquivalenz dieser Begriffe ist nur im Kontext der Assoziationsregeln zu verstehen, da sie in der Logik nicht immer gegeben ist.

Definition 8

Der **Support** einer Assoziationsregel ist der Support von $X \bigcup Y$ in D.

Analog ist die **Frequenz**, oder der **relative Support** der Regel definiert:

Definition 9

$$Support_{rel} = \frac{support(X \bigcup Y)}{|D|}$$

Die Assoziationsregel ist dementsprechend häufig, wenn ihr Support den Minsup-Wert σ erreicht. Auch hier kann sowohl der absolute, als auch relative Support verwendet werden.

Definition 10

Die **Confidence** (manchmal auch als "**Accuracy**" bezeichnet) einer Assoziationsregel $X \Rightarrow Y$ in D ist die bedingte Wahrscheinlichkeit, dass Y in einer Transaktion vorkommt, wenn diese Transaktion X enthält.

$$confidence(X \Rightarrow Y, D) \coloneqq P(Y \mid X) = \frac{support(X \bigcup Y, D)}{support(X, D)}$$

Die Regel heißt **confident**, wenn der Wert $P(Y \mid X)$ den gegebenen **minimum-Confidence** (**minconf**)-Wert γ erreicht.

Sei $P(X)$ die Wahrscheinlichkeit, dass alle Elemente aus X in einer Transaktion T enthalten sind. Dann ist der $Support(X \Rightarrow Y) = P(X \bigcup Y)$ und die $Confidence(X \Rightarrow Y) = P(X \mid Y)$.

Definition 11

Seien D und $\mathcal{I}$ wie oben definiert, σ der Minsup-Wert und γ der Minconf-Wert. Eine Ansammlung der häufigen und confidenten Assoziationsregel kann dann wie folgt definiert werden

$R(D, \sigma, \gamma) \coloneqq \{X \Rightarrow Y \mid X, Y \subseteq \mathcal{I}, X \bigcap Y = \{\},$
$X \bigcup Y \in F(D, \sigma), confidence(X \Rightarrow Y, D) \geq \gamma\},$

Dann heiße die Aufgabe der **Assoziationsregelentdeckung**:

Finde $R(D, \sigma, \gamma)$ bei gegebenen D, $\mathcal{I}$, σ, γ.

Außerdem ist man bei dieser Aufgabe natürlich auch an den quantitativen Größen der jeweiligen Regel interessiert: den Werten von Support und Confidence.

Nach der genauen Definition dieser quantitativen Größen, die oben noch fehlten, hat eine Assoziationsregel die Form:

Body $\Rightarrow$ **Head** [Support, Confidence].

und das obige Beispiel am Anfang des Kapitels 3.1 könnte jetzt genauer formuliert werden:

„75% der Kunden, die ein Notebook gekauft haben, haben auch eine Notebookmaus und eine Notebooktasche gekauft. Dabei wurde in 2% aller Käufe ein Notebook gekauft“. Kurz ausgedruckt sehe die Regel so aus:

„Notebook $\Rightarrow$ Notebookmaus, Notebooktasche“[0,02; 0,75]. Hierbei wurde der Support relativ angegeben. Genau solche Regeln liefert der folgende Algorithmus.

3.1.2. Apriori

Dieser Algorithmus wurde in [Agrawal et. al. 1993] vorgestellt. Das Hauptanwendungsgebiet von Apriori ist die Warenkorbanalyse. Der Algorithmus dient dazu, die so genannten häufigen Artikelmengen (oder im Original, „**frequent itemsets**“) zu entdecken und auf deren Basis die Assoziationsregeln zu bilden, die häufig auftretende Muster im Kaufverhalten der Kunden widerspiegeln. Und obwohl mit häufigen Itemsets meistens häufige Artikelmengen gemeint werden, können es im Prinzip beliebige Elemente sein. So wird z.B. in [Fung et.al. 2000] eine Methode für das hierarchische Clustering[9] der Textdokumente anhand der häufigen Itemsets, die mit Apriori gefunden werden, vorgeschlagen. Dabei agieren hier die Textwörter als Items. Das ist ein Beispiel dafür, dass Apriori auch bei anderen Datamining-Problemen Anwendung findet.

Der Algorithmus erhält als Eingabe die Tabelle mit Transaktionen und zwei zusätzliche Eingabeparameter: minimum-Support σ (oder kurz, minsup) und minimum-Confidence γ (oder kurz, minconf), die oben im Kapitel 3.1.1 definiert wurden. Als Ausgabe liefert der Algorithmus die häufigen Itemsets und die Assoziationsregeln, die den Eingabeparametern entsprechen. Bevor der Algorithmus genauer beschrieben wird, soll die folgende **Idee** erklärt werden, die dem Apriori-Algorithmus zugrunde liegt:

Hat eine Artikelmenge einen hinreichenden Support, so hat auch jede ihrer Teilmengen mindestens den gleichen oder noch größeren Support. Oder anders ausgedrückt, wenn eine Artikelmenge M keinen hinreichenden Support hat, dann hat auch keine Artikelmenge, die diese Menge M als Teilmenge enthält, einen hinreichenden Support. Eigenschaften der Art „wenn A eine Eigenschaft hat, dann auch jede beliebige Menge, die in A enthalten ist“ werden Monotonie-Eigenschaften genannt und können oft zur Effizienzsteigerung verwendet werden. Die Richtigkeit der Idee ist leicht zu sehen. Angenommen, eine Artikelmenge $A = \{a_1,...,a_n\}$ hat Support *s*. Das bedeutet, es gibt mindestens *s* Datensätze, die alle Artikel $a_i, i = 1,...,n$ enthalten. Insbesondere enthalten diese Datensätze natürlich auch jede Teilmenge von A. Also hat auch jede Teilmenge von A einen Support von mindestens *s*. Mit dieser Idee können nun die gesuchten Artikelmengen systematisch aufgebaut werden, beginnend mit den einelementigen Artikelmengen über die zweielementigen usw., bis der vorgegebene minimale Support unterschritten wird oder bis ein anderes Kriterium[10] erfüllt ist.

[9] Zitat aus [Fung et.al. 2000]: *“…The intuition of our clustering criterion is that there are some frequent itemsets for each cluster (topic) in the document set, and different clusters share few frequent itemsets. A frequent itemset is a set of words that occur together in some minimum fraction of documents in a cluster. Therefore, a frequent itemset describes something common to many documents in a cluster. We use frequent itemsets to construct clusters and to organize clusters into a topic hierarchy…”*

[10] Ein anderes Kriterium kann z.B. die (vorgegebene) maximale Länge der häufigen Itemsets sein.

Der **Apriori**-Algorithmus in der einfachsten Form sieht so aus:

Eingabe: Transaktions-Daten, minsup, minconf.

Berechne alle häufigen Itemsets mit Support ≥ minsup und alle Confidenten Regeln mit Confidence ≥ minconf

Teil 1:

Berechne alle häufigen Itemsets mit Support ≥ minsup

Teil 2:

Berechne aus diesen die Regeln mit Confidence ≥ minconf

Ausgabe: die häufigen Itemsets und die Assoziationsregeln.

Der Einfachheit halber wird angenommen, dass Items innerhalb der Transaktionen und die später generierten Itemsets lexikographisch sortiert vorliegen (wenn dies am Anfang nicht der Fall ist, muss natürlich eine Extra-Routine über die Transaktionen laufen).

Der formale Ablauf des Algorithmus wird anhand einer Pseudocode-Beschreibung betrachtet:[11]

(vgl. [Goethals, 2003]):

Teil 1, Apriori Itemset Mining

Input: D, σ //(wie oben definiert)

Output: $F(D, \sigma)$) //(wie oben definiert)

1: $C_1 := \{\{i\} \mid i \in \mathcal{I}\}$ // Starte mit einem Element

2: $k := 1$

3: **while** $C_k \neq \{\}$ **do**

4: // Berechne die Supportwerte aller kandidierenden Itemsets („candidate itemsets“)

5: **for all** *transactions (tid , I)* $\in$ D **do**

6: **for all candidate itemsets** $X \in C_k$ **do**

7: **if** $X \subseteq I$ **then**

8: $X.support++$

9: **end if**

10: **end for**

11: **end for**

12: // Extrahiere alle häufigen Itemsets

[11] Vorbemerkung zur Notation: $X[i]$ bedeutet das *i*-te Element aus dem Itemset *X* anhand der Lexikographischen Sortierung. Der *k*-Präfix eines Itemsets *X* ist das *k*-Itemset, d.h. ein Itemset der Größe *k*: $\{X[1],...,X[k]\}$

13: $F_k := \{X \mid X.support \geq \sigma\} \mid X \in C_k\}$

14: //Generiere neue candidate itemsets (die sog. „Candidate Generation"-Methode)

15: **for all** $X,\ Y \in F_k, X[i] = Y[i]$ **for** $1 \leq i \leq k$ **- 1, and** X**[**k**]** $< Y$**[**k**] do**

16: $I = X \cup \{Y[k]\}$

17: **if** $\forall J \subset I, \mid J \mid =$ **k :** $J \in F_k$ **then**

18: $C_{k+1} := C_{k+1} \cup I$

19: **end if**

20: **end for**

21: $k++$

22: **end while**

Der Algorithmus berechnet iterativ mit einer Tiefensuche im Suchraum aller möglichen Itemsets die kandidierenden Itemsets („candidate itemsets") C_{k+1} der Größe k+1, angefangen bei k=0 (Zeile 1). Ein Itemset ist ein Kandidat für ein häufiges Itemset, wenn alle seine Subsets häufig sind. Hier kommt die Grundidee von Apriori zum Tragen. Angefangen wird mit C_1, das alle Items von $\mathcal{I}$ enthält. Irgendwann mit einem bestimmten k sind alle Itemsets der Größe k+1 generiert. Das geschieht in zwei Schritten. Im ersten Schritt („join step") werden die Itemsets die Menge F_k mit sich selbst vereinigt. Die Vereinigung der Itemsets $X \cup Y \in F_k$ wird generiert, wenn X und Y den gleichen *k*-1-Präfix haben. (Zeilen 15-20). Im zweiten Schritt („prune step") wird überprüft, ob alle k-elementigen Teilmengen von $X \cup Y$ in F_k sind, und wenn ja, wird $X \cup Y$ in C_{k+1} eingefügt (Zeilen 17-18). Der Support des jeweiligen Itemset wird berechnet, indem alle Transaktionen darauf geprüft werden, ob das Itemset in ihnen vorkommt, und wenn dies der Fall ist, wird er hoch gezählt. Erreicht ein Itemset den minsup-Parameter σ, wird es in die Menge der häufigen Itemsets eingefügt (Zeilen 5-13).

Wenn alle häufigen Itemsets berechnet sind, können nun die häufigen und „confidenten" Regeln generiert werden. Das geschieht im Teil 2, der dem Teil 1 sehr ähnlich ist.

Teil 2, Apriori - Association Rule mining

Input: D, σ, γ //(wie oben definiert)

Output: $R(D, \sigma, \gamma)$ //(wie oben definiert)

1: **Compute** $F(D, \sigma)$ // führe **Teil 1** aus und finde **alle häufigen Itemsets**.

2: $R := \{\}$

3: **forall** $I \in F$ **do**

4: $R := R \cup I \Rightarrow \{\}$

5: $C_1 := \{\{i\} \mid i \in I\};$

6: $k := 1$

7: **while** $\mathbf{C_k} \neq \{\}$ **do**

8: //Extrahiere alle Heads der confidenten Assoziationsregeln

9: $H_k := \{X \in C_k \mid confidence(I \setminus X \Rightarrow X, D) \geq \gamma$

10: //Generiere neue kandidierenden Heads

11: **forall** $X, Y \in H_k, X[i] = Y[i]$ **for** $1 \leq i \leq k$-1**, and** $X[k] < Y[k]$ **do**

12: $I = I \cup \{Y[k]\}$

13: **if** $\forall J \subset I, \mid J \mid = k : J \in H_k$ **then**

14: $C_{k+1} := C_{k+1} \cup I$

15: **end if**

16: **end for**

17: $k++$

18: **end while**

19: // Kumuliere alle Assoziationsregeln

20: $R := R \cup \{I \setminus X \Rightarrow X \mid X \in H_1 \cup ... \cup H_k\}$

21: **end for**

Zunächst werden alle häufigen Itemsets mit dem Teil 1 gefunden. Dann wird jeder dieser Itemsets *I* in zwei kandidierenden Teilmengen zerlegt: Kandidat-Head X und Kandidat-Body $Y=I \setminus X$. Der Prozess startet mit Y={}, dass die immer mit 100% Confidence geltende Regel $I \Rightarrow \{\}$ bedeutet (Zeile 4). Danach produziert der Algorithmus iterativ die kandidierenden Heads C_{k+1} der Größe k+1, angefangen bei k=0 (Zeile 5). Ein Head ist nur dann ein Kandidat, wenn alle seine Untermengen bereits als confidente Regeln bekannt sind. Der Prozess der Generierung von Kandidat-Heads ist genau der gleiche wie bei der Generierung der häufigen Itemsets im Teil 1(hier Zeilen 11-16). Um die Confidence eines Kandidat-Heads *Y* zu berechnen, wird der Support von *I* und *X* aus *F* extrahiert. Alle Heads, die in confidenten Regeln resultieren, werden in H_k eingefügt (Zeile 9). Am Ende enthält R alle confidenten Regeln.

Nachdem der Apriori-Algorithmus beschrieben wurde, wird im Folgenden der Begriff der Generalisierten Assoziationsregeln erläutert.

3.2. Generalisierte Assoziationsregeln

Hier wird das Thema der generalisierten Assoziationsregeln ausführlicher diskutiert. Dieses Thema wurde von vielen Autoren behandelt. Die Konzepte und Ansätze der Autoren sind unterschiedlich und haben zum Teil auch unterschiedliche Zwecke. Einige bedeutende Arbeiten auf diesem Gebiet sollen hier vorgestellt werden. Die Aspekte, die für weitere Anwendung im Rahmen der Diplomarbeit interessant sind und zumindest ansatzweise ausprobiert werden könnten, werden hier beleuchtet.

Allen Arbeiten gemeinsam ist, dass versucht wird, die Regeln auf einem abstrakteren Niveau als die einfachen Regeln zu finden. (Zum Beispiel, wenn man an die reellen Daten aus dem System denkt, wäre eventuell eine Regel wie „wenn Bohrer 3,5mm, Bohrer 5,0mm und Bohrer 6,0 mm gehärtet gekauft werden, dann auch Bohrer 7,5mm…“ nicht sehr interessant.)

Interessanter wäre etwa die Regel „wenn Produkte aus Produktgruppe „Bohrer“ und aus Warengruppe „Beschläge“ gekauft werden, dann auch Produkte aus Produktgruppe „Bohrmaschinen“, Produktgruppe „Büroausstattung“ und Warengruppe „Arbeitsschutz “. Um diese Regel genauer zu verstehen, werden zunächst die Taxonomien und die Arten der Regel vorgestellt.

3.2.1. Taxonomien, Crosslevel- , Multiplelevel- und Multidimensionale Regel

Bei den meisten Arbeiten auf dem Gebiet der generalisierten Regel wird der Begriff „Taxonomie**“** benutzt. Was ist eine Taxonomie? Eine kurze Erklärung wäre:

eine **Taxonomie** ist eine „is a“-Hierarchie. Mit anderen Worten, stellt eine Taxonomie eine hierarchische Ordnung dar.

Der Begriff der Taxonomie kann so definiert werden:

Definition 12

Mit **Taxonomie** bezeichnet man ein Modell, das Begriffe oder Objekte eines Themengebietes oder Objektdomäne in hierarchische Beziehung setzt und klassifiziert. Die **Taxonomie** ist die Einteilung von Dingen.

Zum Beispiel seien Artikel gegeben, die in Artikelgruppen unterteilt werden. Diese wiederum werden in Warengruppen unterteilt. Jeder Artikel gehört zu einer Artikelgruppe und jede Artikelgruppe ist ein Teil einer Warengruppe. Somit wird die „is a“-Hierarchie klar. Es können allerdings auch mehrere Taxonomien gleichzeitig existieren. Angenommen es gibt neben der vorhandenen Hierarchie noch eine Preisgruppenunterteilung. Das wäre dann die zweite Taxonomie. In solchen Fällen kann man die mehrfachen Taxonomien mit Hilfe eines DAG („directed acyclic graph“) als eine Taxonomie modellieren, deshalb wird weiter immer eine einfache Taxonomie angenommen.

Welche Arten von Regeln werden gesucht, wenn man über die generalisierten Regeln spricht? Verschiedene Autoren meinen verschiedene, oder zumindest teilweise verschiedene Arten von Regeln. All diese Arten von Regeln kann man in zwei Kategorien unterteilen:

Die so genannten **Cross-Level-** und die **Multiple-Level**-Regeln.

Die Vertreter der ersten Art beinhalten Elemente aus beliebigen Stufen der Hierarchie, bei der zweiten Art von Regeln werden sie für jede Stufe für sich gesucht. Manchmal werden die generalisierten Regeln einfach als **multidimensional** bezeichnet. Dabei werden eigentlich immer noch die Cross-Level-Regeln gemeint. Man kann sich denken, dass die Multiple-Level-Regeln ein spezieller Fall der Cross-Level-Regeln sind. Der Unterschied wird in den folgenden Kapiteln deutlicher.

3.2.2. Entdeckung der generalisierten Regel nach Agrawal und Srikant

Zunächst soll die wohl bekannteste Arbeit auf dem Gebiet vorgestellt und diskutiert werden: [Agrawal und Srikant, 1995]

3.2.2.1. Motivation

Gegeben sei eine große Datenbank von Transaktionen, wo jede Transaktion eine Artikelmenge (Itemsets) beinhaltet. Auf den Artikeln ist eine Taxonomie definiert. Gesucht sind Assoziationen zwischen den Elementen (Items) beliebiger Stufen dieser Taxonomie.

Bemerkung: Man muss an dieser Stelle noch einmal die beliebig mögliche Kombination der Taxonomie-Stufen unterstreichen (also die „Cross-Level"-Regeln).

Zum Beispiel sei eine Taxonomie gegeben, die besagt, dass Jacken der Kategorie „Oberbekleidung" angehören und die Oberbekleidung der Oberkategorie „Kleidung" angehört. Somit ist eine Jacke ein Oberbekleidungstück und eine Kleidung. Angenommen, es gibt die Regel: „Menschen, die Oberbekleidung kaufen, kaufen auch Schuhe". Diese Regel kann gültig sein, auch wenn die beiden Regeln „Menschen, die Jacken kaufen, kaufen auch Schuhe" und „Menschen, die Kleidung kaufen, kaufen auch Schuhe" nicht gelten.

Die Regeln, die auf Basis von Hierarchien entdeckt bzw. gebildet werden, werden **Generalisierte Assoziationsregeln** genannt. Der Begriff wird weiter unten noch genauer spezifiziert.

Abgesehen von dem bereits erwähnten höheren Abstraktionsniveau der generalisierten Regeln soll zumindest noch ein möglicherweise bestehendes Problem der oben beschriebenen „einfachen" Regeln mit den generalisierten Regeln adressiert werden: Während bei der Regelentdeckung auf der untersten Hierarchieebene meistens der Item- und Itemset-Support zu klein bleibt und deshalb kaum Regeln finden ließ, können die Itemsets der höheren Hierarchiestufen einen größeren Supportwert erreichen. Dadurch werden mehr Regeln entdeckt.

3.2.2.2. Ansatz

In [Agrawal und Srikant, 1995] wird ein Ansatz vorgestellt, solche generalisierten Regeln zu entdecken, der eigentlich dem naiven Ansatz entspricht: Es wird zunächst der gleiche Algorithmus verwendet, der bei der Entdeckung der einfachen, d. h. nicht generalisierten Assoziationsregel (AR) verwendet wird: Apriori „Basic". Danach wird der Algorithmus noch modifiziert und laufzeitoptimiert und es werden weitere Varianten von Apriori diskutiert, die eine bessere Laufzeitperformance erzielen. Die Eingabedaten für den Algorithmus werden aber geändert: die Transaktionen werden mit den Hierarchieinformationen erweitert, also angereichert. D.h., zu jedem Artikel werden seine Vorfahren aus höheren Hierarchiestufen in die Transaktionen übernommen, wobei das mehrfache Vorkommen der gleichen Elemente aus den oberen Hierarchiestufen in einer Transaktion nicht zugelassen wird. Außerdem wird ein Interessen-Maß[12] eingeführt, mit dessen Hilfe die Anzahl der redundanten Regeln (nach Angaben der Autoren) sich um bis zu 60% verringern lässt.

[12] Bemerkung: dieses Interessen-Maß wird in einem separaten Kapitel 3.3.1 weiter behandelt.

Die Taxonomien über den Artikeln liegen in den meisten Fällen bereits vor.

In Abbildung 8 ist ein Beispiel einer Taxonomie dargestellt:

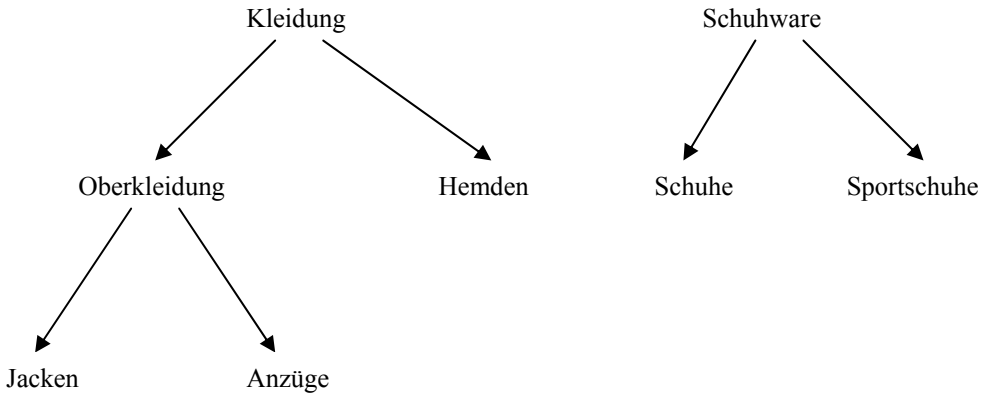


Abbildung 8 Beispiel einer Artikeltaxonomie

Angenommen, es wurden die folgenden beiden „einfachen[13]“ Regeln gefunden:

„Jacken $\Rightarrow$ Sportschuhen“ und „Anzüge $\Rightarrow$ Sportschuhe“. Mit dieser Taxonomie könnte man zum Beispiel versuchen, aus diesen beiden Regeln die Regel:

„Menschen, die Oberbekleidung kaufen, tendieren dazu, Sportschuhe zu kaufen“ zu folgern. Denkt man an den Support dieser Regel, so soll man sich klar sein, dass der Support der Regel „Oberbekleidung $\Rightarrow$ Sportschuhe“ nicht zwingend die Summe der Supportwerte der Regeln „Jacken $\Rightarrow$ Sportschuhe“ und „Anzüge $\Rightarrow$ Sportschuhe“ sein muss: es kann sein, dass manche Menschen in der gleichen Transaktion Jacken, Anzüge und Sportschuhe kaufen. Es könnte auch sein, dass die Regel „Oberbekleidung $\Rightarrow$ Sportschuhe“ gültig ist, aber die Regeln „Jacken $\Rightarrow$ Sportschuhe“ und „Kleidung $\Rightarrow$ Sportschuhe“ nicht gelten. Die erste erreicht nicht den Minimum-Support, die letzte den Mininmum-Confidence-Wert

Definition 13

Eine **Generalisierte Assoziationsregel** ist eine Implikation der Form $X \Rightarrow Y$, wobei $X \subseteq I$ und $Y \subseteq I$, $X \subseteq I = \emptyset$, und kein Element in Y ist Vorfahre von einem oder mehreren der Elemente in X. Der Sinn der letzten Restriktion ist klar: die Regeln der Form

„$x \Rightarrow Vorfahre(x)$“ sind immer erfüllt und haben die Confidence 100%, also sind redundant.[14] In [Agrawal und Srikant, 1995] sagen die Autoren, dass diese Regeln generalisiert sind, weil sie Elemente aus jedem Level der Taxonomie enthalten können. (Vgl. [Agrawal et al., 1993], wo solche Regeln noch nicht unterstützt werden und Elemente nur aus der untersten Hierarchieebene in den Regeln vorkommen können.)

[13] Mit dem Begriff „Einfache“ Regeln werden die herkömmlichen Assoziationsregeln gemeint, die ohne Taxonomien erzeugt werden.

[14] Die oben in Kapitel 3.1.1 eingeführten Definitionen der Parameter Support und Confidence bleiben auch hier gültig.

Also werden generalisierte Regel gesucht, deren Support und Confidence die vom Benutzer eingegebenen Mindestwerte erreichen. Dabei können allerdings auch redundante Regeln gefunden werden. Später wird untersucht, wie man solche erkennen und herausfiltern kann.

Beispiel:

sei eine Taxonomie von der Abbildung 8 gegeben.

Es sei $\mathcal{I}$ ={ Hemd, Jacke, Anzug, Sportschuhe, Schuhe } eine Menge aller gekauften Elemente (Items), minsup=30% (d.h. 2 aus insgesamt 6 Transaktionen, s. Abbildung 9) und minconf=60%.

Man kann sehen, dass die Regeln "Anzug $\Rightarrow$ Sportschuhe" und „Jacke $\Rightarrow$ Sportschuhe" nicht den genügenden Support haben, wobei aber die Regel „Oberbekleidung $\Rightarrow$ Sportschuhe" den minimalen Support erreicht.

Transaktionen

Transaktion-ID	Artikel (Items)
100	Hemd
200	Jacke, Sportschuhe
300	Anzug, Sportschuhe
400	Schuhe
500	Schuhe
600	Jacke

Itemsets

Itemsets	Support
{Jacke}	2
{Oberbekleidung}	3
{Kleidung}	4
{Schuhe}	2
{Sportschuhe}	2
{Schuhware}	4
{Oberbekleidung, Sportschuhe}	2
{Kleidung, Sportschuhe}	2
{Oberbekleidung, Schuhware}	2
{Kleidung, Schuhware}	2

Regeln

Regeln	Support	Confidence
Oberbekleidung $\Rightarrow$ Sportschuhe	33%	66,6%
Oberbekleidung $\Rightarrow$ Schuhware	33%	66,6%
Sportschuhe $\Rightarrow$ Oberbekleidung	33%	100%
Sportschuhe $\Rightarrow$ Kleidung	33%	100%

Abbildung 9 Transaktionstabelle, häufige Itemsets und entsprechende Regeln.

Aus den Daten in den Tabellen kann man folgende Beobachtungen machen:

a) Wenn eine Menge (Itemset) {*x,y*} den minimum-Support-Wert erreicht, dann auch die Mengen $\{\hat{x}, y\}$, $\{x, \hat{y}\}$ und $\{\hat{x}, \hat{y}\}$, wobei $\hat{x}$ einen Vorfahren von *x* bezeichnet. Jedoch, auch wenn der minimum-Support in diesem Fall von allen drei Regeln $x \Rightarrow \hat{y}$, $x \Rightarrow \hat{y}$ und $\hat{x} \Rightarrow \hat{y}$ erreicht wird, kann nur die erste garantiert die Minimum-Confidence haben, die restlichen können, müssen aber nicht die Minimum-Confidence haben.

b) Der Support eines Elements in der Taxonomie ist nicht gleich der Summe der Supporte seiner Kinder, weil z. B. mehrere seine Kinder in der gleichen Transaktion enthalten sein können.

Fazit: Man kann nicht die generalisierten Regeln mit Elementen aus höheren Hierarchiestufen direkt aus den Regeln mit Elementen der untersten Hierarchiestufe ableiten.

In ihrer Arbeit stellen die Autoren 3 Algorithmen vor, die zur Lösung dieser Aufgabe eingesetzt werden können. Der erste ist der „Basic Apriori", der genau so abläuft wie bereits oben beschrieben. Die zwei weiteren Algorithmen, „Cumulate" und „EstMerge" dienen der Laufzeitoptimierung und werden hier nicht weiter betrachtet, da die zu verarbeitende Datenmenge keine Perfomance-Probleme bereitet. Der „Basic-Apriori" wird aber für die späteren Experimente als Grundstein verwendet.

Für die Anwendung des Apriori werden zunächst die Transaktionen erweitert. Und zwar werden zu jedem Element in einer Transaktion *T* alle seine Vorfahren in die Transaktion geschrieben. Dabei wird zusätzlich geprüft, ob diese bereits in der Transaktion vorhanden sind, und falls ja, werden diese nicht erneut in die Transaktion eingetragen. D. h., es werden keine Elemente (doppelt oder gar mehrfach) eingetragen. Jedes Element erscheint dann in der erweiterten Transaktion T' einmal und, die Transaktion enthält alle seine Vorfahren.

Der erste Schritt des Algorithmus berechnet den Support der einelementigen Itemsets, d. h. den Support der Elemente, die jetzt sowohl von der untersten als auch von allen anderen Hierarchieebenen stammen können.

Ein Schritt *k* besteht aus zwei Phasen. In der ersten Phase werden die häufigen Itemsets L_{k-1}, die im Schritt k-1 gefunden wurden, für die Generierung der Menge der Kandidatitemsets C_k benutzt. Danach werden die Transaktionen gescannt und der Support der Kandidaten in C_k wird berechnet. Für die schnelle Supportberechnung braucht man eine Möglichkeit, die Transaktionen schnell darauf zu prüfen, ob sie einen Kandidaten des C_k enthalten. Dafür können für die Speicherung z. B. die speziellen Datenstrukturen Hashtrees verwendet werden, die in [Agrawal und Srikant, 1994] beschrieben sind. Andere Implementierungsvarianten verwenden andere effiziente Datenstrukturen: T-Trees, Tries oder Prefix-Trees. (Zu den letzteren siehe z.B. [Goethals, 2003])

Der **Basic Apriori** Algorithmus sieht so aus:

L_1:={häufige 1-Itemsets}
k=2 // k ist die Pass-, oder Schrittnummer
while ($L_{k-1} \neq \emptyset$) **do**
begin
    C_k :=neue Kandidaten der Größe *k*, die aus L_{k-1} generiert werden
    forall transactions $t \in D$ **do**
    begin
        Füge alle Vorfahren von jedem in *t* enthaltenen Element in *t* ein,
        lasse keine Duplikate zu.
        Inkrementiere den Zähler von allen Kandidaten in C_k, die in *t* enthalten sind.
    end
    L_k :=Alle Kandidaten in C_k mit min-Support
    k:=k+1;
end
Antwort: =$\bigcup L_k$;

Diesen Ansatz von [Agrawal und Srikant, 1995] wird in Experimenten in leicht modifizierter Form verwendet[15].

Zusammenfassend kann man den Ansatz so formulieren: Die Methode zur Entdeckung der einstufigen Assoziationsregeln wird auf die Entdeckung der Assoziationsregeln in mehrstufigen hierarchischen Strukturen angewandt.

3.2.2.3. Diskussion

Dieser Ansatz wird später noch in Experimenten untersucht. Man kann aber feststellen, dass bei diesem Ansatz unter anderem auch unerwünschte Effekte entstehen. Beispielsweise kann der größere Support nur von den höheren Hierarchiestufen erreicht werden. D. h. wenn man die Regeln auf unteren Stufen der Hierarchie finden will, soll der minimum-Support möglichst klein gewählt werden. Folglich werden auf den hohen oder mittleren Stufen viele uninteressante Regeln gefunden. Außerdem gibt noch ein von den Autoren gar nicht besprochenes Problem, das nachfolgend beschrieben wird.

Angenommen, eine Produktgruppe *P* besteht aus zehn Artikeln: {A,B,C,D,E,F,G,H,I,J}. Diese werden aber nur zu einem Teil gekauft, sagen wir, nur zwei davon: A und B. Alle anderen wurden niemals gekauft. Sei *Q* eine andere Produktgruppe, die folgende Artikel beinhaltet: {S,T,U}. Angenommen, es gibt folgende einfache Regel: A, B$\Rightarrow$S. Bei einer Generalisierung nach dem oberen Ansatz gibt es in jedem Fall eine Regel $P \Rightarrow Q$, da eine Gruppe auf jeden Fall nicht kleineren Support-Wert haben kann als ihre Artikel. Wenn man

[15] Bemerkung: Die Erweiterung der Itemsets mit Vorfahren wird nicht innerhalb des Algorithmuslaufes, jeweils für die k-Itemsets, durchgeführt, wie in [Agrawal und Srikant, 1996] vorgeschlagen, sondern noch vor dem Algorithmusstart für alle Elemente der Transaktionen. Anschließend werden die dabei entstehenden redundanten Regeln ausgefiltert.

genau überlegt, versteht man, dass die Regel $P \Rightarrow Q$ eine „OR"-verknüpfte Verknüpfung aller Artikel aus den jeweiligen Artikelgruppen bedeutet:

$A \vee B \vee C \ldots \vee \ldots \vee J \Rightarrow S \vee T \vee U$. Die Artikel C, D,…,J wurden aber überhaupt nicht gekauft und werden eventuell auch nie gekauft. Deshalb ist es vielleicht nicht korrekt, eine so allgemeine Aussage zu treffen, dass „wenn *irgendein Produkt aus der Produktgruppe P* gekauft wird, auch ein Produkt aus der Gruppe *Q* gekauft wird". (Natürlich mit entsprechenden Support- und Confidence-Werten). Denn allein A und B haben zu dem Support der Gruppe *P* beigetragen, so dass diese in die Regel einfloss. Hier muss man eventuell einen zusätzlichen Parameter finden, der die Generalisierung der Regeln in dieser Hinsicht steuert. Deshalb wird versucht, diesen Aspekt im praktischen Teil der Arbeit zu berücksichtigen.

3.2.3. Entdeckung der generalisierten Regel nach Han und Fu

3.2.3.1. Motivation

Eine etwas andere, oder genauer genommen erweiterte Sichtweise auf das Problem wird in [Han und Fu, 1999] vorgestellt. Dabei wird ein Top-Down-Ansatz verfolgt. Während beim oben beschriebenen Ansatz die minimum-Support-Werte und der minimum-Confidence-Wert für alle Hierarchiestufen gleich bleiben, schlagen Han und Fu vor, verschiedene minimum-Support- und eventuell minimum-Confidence-Schwellen für verschiedene Hierarchiestufen zu verwenden. Die Benutzung eines einzigen kleinen minimum-Supportwertes wird zur Entdeckung vieler uninteressanten Regel führen, während bei der Benutzung eines großen minimum-Support- Wertes, der für alle Stufen gleich benutzt wird, eventuell interessante Regel verloren gehen. Diese Nachteile wollen die Autoren vermeiden. Darin besteht der entscheidende Unterschied zum Ansatz von Agrawal und Srikant.

Und zwar sollen auf niedrigeren Stufen niedrigere Werte gewählt werden. Die Methode wird als sogenannte *„progressive deepening method"*, zu Deutsch: progressive Vertiefung.

3.2.3.2. Ansatz

Die Methode ist eine Erweiterung des Apriori Algorithmus. Dabei werden zunächst die häufigen Itemsets auf dem obersten Level der Hierarchie gefunden. Dann wird die Suche „progressiv vertieft", also auf den niedriger liegenden Levels weitergeführt. In der Methode wird angenommen, dass man nur die Nachkommen untersuchen soll, deren Vorfahren „häufig" sind. Dies ist offensichtlich: wenn ein Element relativ selten vorkommt, wird sein Nachkommen höchstens genau so selten oder noch seltener vorkommen und ist deswegen uninteressant.

Seien hier die Definitionen von *Itemset A*, *Support* $\sigma(A/S)$ und *Confidence* $\varphi(A \Rightarrow B/S)$ von oben weiterhin gültig. Eine Erweiterung der Definitionen ist nur für die Begriffe „häufig", minimum-Support und minimum-Confidence Werte notwendig:

Definition 14

ein Set A ist häufig („*large*“ im Original, vgl. [Han und Fu, 1999]) in S auf der Hierarchiestufe l, wenn der Support von A nicht kleiner als für diese Stufe vorgegebener min-Support –Wert (σ'_l) ist.

Es wird der Begriff der „*starken*“ Regel eingeführt.

Definition 15

Für Set S ist die Regel $A \Rightarrow B$ *stark* (*„strong“* im Original, vgl. [Han und Fu, 1999]), wenn alle Vorfahren von jedem Element in A und B, (d.h. korrespondierende Elemente auf den höher liegenden Stufen), wenn es solche gibt, häufig auf ihren eigenen (korrespondierenden) Stufen sind, „ $A \wedge B$ “ häufig (auf der aktuell betrachteten Stufe) ist, und die Confidence von „ $A \wedge B$ “ mindestens so groß wie min-Confidence auf der aktuell betrachteten Stufe ist.

Die Betrachtung der Nachkommen von „nichthäufigen“ Itemsets aus den höheren Stufen entfällt somit, da diese in keinem Fall den minimum-Support erreichen können und die zu untersuchende Itemset-Menge wird kleiner. Die Ergebnismenge wird bei diesem Ansatz stärker als beim Ansatz von Agrawal und Srikant eingeschränkt.

Die Autoren bieten auch einen Ansatz, wie man eine Hierarchie bilden kann, falls sie nicht vorgegeben ist. Jeder Artikel besitzt mehrere Attribute in der Datenbanktabelle: z.B. ID, Preis, Bezeichnung, Beschreibung, etc. Dann kann man die Daten in der Datenbank beispielsweise so gruppieren, dass die zu betrachtenden Attribute der Artikel (also alle Spalten) in allen Zeilen bis auf die ID des Elements, gleich sind und nur in den Artikel-IDs sich unterscheiden. Die Gesamtheit solcher Artikel-IDs wird zu einem Element (z.B. einem kommaseparierten String) zusammengefasst und als Eintrag für die neu gebildete Spalte verwendet. Alle anderen Spalteneinträge bleiben unverändert, und es bleibt nur eine Zeile. D.h., die Artikel werden so zu einer Artikelgruppe zusammengefasst, bei der alle Attribute für alle beinhalteten Artikel gleich sind und nur die ID des jeweiligen Artikels sich von den anderen unterscheidet. Die Taxonomie kann dann so gebildet werden, dass die Attribute (oder die Spalten in der Tabelle) die gruppierenden Merkmale sind. Jede Gruppe wird dann bei der Regelentdeckung als atomares Element behandelt. Wenn es z.B. Attribute „category“, „content“ und „brand“ gibt, können Gruppen gebildet werden, zu denen die Artikel gehören, die die gleichen Werte für diese Attribute besitzen (vgl. mit Gruppierung von Psaila/Lanzi in Kapitel 3.2.5)

Wie wird bei der der Entdeckung von *starken* ARs vorgegangen?

Angenommen, gegeben sind die Transaktionen, die Lebensmittel beinhalten, und es existiert eine Taxonomie über diese.

Der Prozess beginnt auf dem Top-Level der Hierarchie und. Z. B. kann man nach Regeln mit min-Support=5% und min-Confidence=50% auf dem höchsten, also dem allgemeinsten Level suchen. Man entdeckt beispielsweise 1-elementige Itemsets (mit entspr. Support in Klammern) „Brot“(25%), „Obst“(30%), …, 2-elementige Itemsets „Obst, Brot“(19%), etc., und ein Set von starken Regeln, wie „Brot $\Rightarrow$ Obst(76%)“.

Auf dem 2-ten Level, sei der min-Support 2% und die min-Confidence 40%. Man findet z.B. 1-elemntiges Itemset „Birnen“(10%), „Weißbrot“(15%),… und 2-elementiges Itemset „Birnen, Weißbrot“ (60%),…“ etc. Der Prozess wiederholt sich weiter auf niedrigeren Levels so lange, bis keine weiteren häufigen Itemsets gefunden werden können

Die Methode für die Entdeckung der „Multiple-Level“ Assotiation Rules[16]

Für die leichtere Verarbeitung wird die Hierarchie-Information kodiert, anstatt die originale Transaktionstabelle zu benutzen. Ein kodierter String, der die Position in der Hierarchie repräsentiert, braucht nur wenige Bits. Man benutzt eine Ziffern-Sequenz. Z.B. die Sequenz „112“ für „Vollmilch der Firma Ja“ könnte so entstanden sein:
Die erste „1“ repräsentiert „Milch“ auf dem Level 1, die zweite „1“ „Vollmilch“ auf dem Level 2 und die letzte „2“ „Firma Ja“ auf dem Level 3. Also: Eine Stufe der Hierarchie entspricht einer Position in der Sequenz; der Wert an einer Stelle *k* kodiert das Element auf seiner *k*-ten Hierarchiestufe.

Hier folgt die kurze Beschreibung der Entdeckung von häufigen Itemsets:

Algorithmus ML_T2L1
Finde Multiple-Level-häufige Itemsets für Entdeckung der starken (im Original „strong“, vgl. [Han und Fu, 1999] Multiple-Level- Assoziationsregel in der Transaktionen-Datenbank.

Eingabe:

1. $\tau[1]$: das der Hierarchie Information entsprechend kodierte und aufgabenrelevante Set von Transaktionen in Format (*TID, Itemset*), in dem jedes Element die kodierte Hierarchieinformation enthält.
2. min-Support Schwellwert (*minsup[l]*) für jedes Hierarchielevel.

Ausgabe:

Multiple-Level-häufige Itemsets.

Methode:

Top-Down, progressiver Vertiefungsprozess, der die häufigen Itemsets auf verschiedenen Hierarchielevels folgendermaßen findet:

Starte auf Level 1, entdecke für jeden Level die häufigen k-Itemsets, $L[l, k]$, und die häufigen Itemsets, $LL[l]$ (für alle k) wie folgt:

[16] Bemerkung: Zu beachten ist, dass hier die „multiple-level“- und nicht die „cross-level“-Regeln gesucht werden.

```
for (l := 1; L ≠ ∅ and l < max_level; l++) do
{
    if l = 1 then
    {
       L[l,1] := get_frequent_itemsets([1], l);
       τ[2] := get_filtered_t_table (ττ[1]L[1,1]);
    }
    else L[l,1] := get_frequent_itemsets(τ(τ[2l);
    end if
    for(k := 2; L[l,k - 1] ≠ ∅ ; k++) do
    {
        C_k := apriori_gen(L[l,k − 1]);
        //Kandidaten generierung von Apriori - Algorithmus
        foreach transaction t ∈ τ[2] do
        {
        C_t := gen_subsets(C_k , t);
        foreach canditdate c ∈ C_t do c.support++;
        }
    L[l,k] := {c ∈ C_k | c.support ≥ minsup[l]}
    }
    end for
    LL[l] := ⋃_k L[l,k];
}
end for
```

Entsprechend dem oberen formalen Algorithmus, sieht die Methode folgendermaßen aus:

Auf dem Level 1 werden die 1-elementigen Itemsets *L*[*l*,1] aus der kodierten Transaktionen-Tabelle *τ*[1] mit „*get_frequent_itemsets(τ*[1],*l);*“ gewonnen. Auf jedem anderen Level *l* werden die aus *τ*[2] (der gefilterten Transaktionen-Tabelle) mit der Methode „*get_frequent_itemsets(τ*[2],*l);*“ gewonnen. Bei *l* >2 werden nur die Items aus *L*[*l* - 1,1] beim herausfinden der 1-Itemsets *L*[*l*,1] aus *τ*[2] betrachtet. Die Transaktionen werden gescannt; wenn eine Transaktion *t* ein Item *i* enthält, wird der Support von *i* hochgezählt. Nach dem Scannen werden alle Items mit Support kleiner als *minsup*[*l*] herausgefiltert.

Die Inhalt der gefilterten Transaktionen-Tabelle *τ*[2] wird über die Funktion „*get_filtered_t_table (τ*[1], *L*[1,1]*);*“ gewonnen, die *L*[1,1] als Filter nutzt, um folgendes herauszufiltern: 1) alle Items, die nicht häufig aus Level 1 sind, und 2) alle Transaktionen, die keine häufigen Itemsets enthalten.

Die k-Itemsets mit k>1 werden auf Level *l* wie folgt berechnet:

Generiere die Kandidatenmenge $L[l,k\text{-}1]$ mit Kandidat-Generierungsroutine vom Apriori (gen_candidate)

Für jede Transaktion t in $\tau[2]$, inkrementiere den Support der Itemsets, die in Kandidatenmenge C_k enthalten sind. Füge alle solchen Itemsets zusammen mit ihrem Support zu $L[l,k]$ hinzu, falls sie den $minsup[l]$ erreichen.

Die Menge der häufigen Itemsets auf dem Level l, $LL[l]$ ist die Vereinigung aller $L[l,k]$ für alle k.

Nachdem die häufigen Itemsets $LL[l]$ gefunden worden sind, kann man die Assoziationsregeln basiert auf min-Confidence für jedes Level $minconf[l]$ mit dem Apriori-Verfahren berechnen.

Die oben beschriebene Methode berechnet die Regeln auf jedem Level für sich, also die Multiple-Level-Regeln. Man kann aber an den Cross-Level-Regeln interessiert sein. Wie bereits oben erwähnt, sind das solche Assoziationsregeln, deren Elemente auf der linken, auf der rechten, oder auf beiden Seiten zu unterschiedlichen Hierarchieebenen gehören. Um diese Regeln zu finden, braucht das oben beschriebene Verfahren eine kleine Modifizierung.

Zunächst werden die gleichen min-support- und minconfidence-Werte für alle Levels verwendet. Wenn die häufigen k-itemsets für k>1 generiert sind, werden Elemente auf allen Levels zusammengefügt, wobei aber solche Itemsets, die zusammen mit Elementen ihre Nachkommen beinhalten, herausgefiltert werden. Dabei transponiert sich das Verfahren zum Verfahren von Agrawal und Srikant.

3.2.3.3. Diskussion

Während in [Agrawal und Srikant, 1995] die Entdeckung der generalisierten „Cross-Level“ Regeln eingeführt wurde, wurde in [Han und Fu, 1999] eine Methode für die Entdeckung von „Multiple-Level“-Regeln vorgestellt. Die unterschiedlichen Supportwerte für unterschiedliche Hierarchiestufen ermöglichen, dass auf niedrigeren Stufen einen niedrigeren Minsup-Wert vorgegeben wird. Das erhöht die Chance für die Itemsets auf den niedrigeren Hierarchiestufen, als „häufig“ entdeckt zu werden. Gleichzeitig werden keine „unnötigen“ und im Vorfeld als „nicht häufig“ bekannten Itemsets der niedrigeren Stufen betrachtet, weil ihre „Eltern“ schon „nicht häufig“ sind. Aber das Problem, das bereits bei Diskussion des Ansatzes von Srikant und Agrawal angesprochen wurde, nämlich die Rechtfertigkeit der Generalisierung und des Einsetzens von Eltern für die Kinder in die generalisierten Regeln auch dann, wenn es vielleicht zu fälschlicher Vorstellung über die Zusammenhänge bzw. Zusammenkäufe führt, wird hier auch nicht gelöst.

Der Ansatz von Han und Fu wird im praktischen Teil der Arbeit ebenso wie der Ansatz von Agrawal und Srikant weiter untersucht.

3.2.4. Ansatz von Li und Sweeney

In **[**Li und Sweeney, 2004] wird ein Problem der Entdeckung von “**robusten Regeln**“ eingeführt. Solche Regel sind aussagekräftigere mehrdimensionale Assoziationsregeln.

Die Attribute werden mit ihren Hierarchien betrachtet, d. h. auch hier werden die Taxonomien für die Bildung der generalisierten Regeln benutzt. Unterschiedliche Kombinationen der Attributgeneralisierungen werden betrachtet, um ein Maximum an Informationen, die die Regeln ausdrücken, zu gewinnen. Die von den Autoren verwendeten Daten sind aber in Wirklichkeit keine Transaktionen im Sinne von „zusammengekauften Artikeln“ wie z. B. bei Srikant und Agrawal. Es wird eine Datenbanktabelle benutzt, die eine Datensammlung über bestimmte Personen und ihre Eigenschaften beinhaltet. Das sind solche Eigenschaften wie Geburtsdatum, Registrierdatum, Wohnort, Geschlecht, Parteiangehörigkeit etc. Diese Eigenschaften sind bei allen Datensätzen immer vorhanden, d. h. die Datensätze bestehen immer aus der gleichen Anzahl von Spalten, deren Werte immer belegt sind.

Die Autoren führen den Begriff „robuste Regel“ ein und bilden einen Generalisierungsbaum („GenTree“).[17]

3.2.4.1. Motivation

Das Finden der bedeutungsvollen Regeln für die Menschen, die mit deren Hilfe Entscheidungen treffen, erfordert nicht nur die quantifizierbare Bestätigung der Regel durch Support und Confidence, sondern auch die Möglichkeit, die Regeln in Form von semantischen Termen zu gewinnen, die für diese Menschen nutzbar sind. So kann die Entdeckung der generalisierten Assoziationsregeln als ein Problem der Suche in einem vordefinierten Raum der potenziellen Regeln formuliert werden, wobei die Regeln gesucht werden, die zu den gegebenen Problemstellungen semantisch am besten passen. Dabei soll die Eigenschaft „robust“ die Aussagekraft dieser Semantik unterstützen.

3.2.4.2. Ansatz

Ähnlich wie bei Srikant und Agrawal werden Hierarchien benutzt. Die Autoren nennen sie Attribut-Hierarchien, da sie in den Beispielen der Transaktionen keine Artikel, sondern mehrere Eigenschaften von Menschen benutzen, die sie als Attribute bezeichnen. Prinzipiell macht es aber keinen Unterschied, ob die Artikel, oder die Attribute in den Transaktionen enthalten sind. Die Autoren nutzen eine andere Schreibweise und benutzen „*“ als Wildcards, um zu zeigen, dass die Attribute an den Stellen unterschiedliche Werte annehmen können. Semantisch sind diese Hierarchien auch Taxonomien, ähnlich wie die Taxonomien von Srikant und Agrawal, und haben die Form „...more general than…“, durch die die Größe des Suchraumes für die potenziellen Assoziationsregeln, der dem Lernenden zu Verfügung steht, bedeutend erweitert wird. Die Regeln können dann (gemischt) mit den Werten aus verschiedenen Hierarchieebenen unterschiedliche Konzepte der Generalisierung repräsentieren. Die Autoren bezeichnen diese Konzepte als multidimensionale generalisierte Assoziationsregel.

[17] Bemerkung: Der Generalisierungsbaum wird hier nur vorgestellt, jedoch nicht weiter verwendet. Die Autoren schlagen auch einen Algorithmus für die Konstruktion des Baumes vor, der wegen seiner Komplexität hier aber nicht betrachtet wird. Jedoch stellt der Generalisierungsbaum einen Interessanten Ansatz für eine Hierarchiebildung aus vorhandenen Daten dar.

Beispiele: seien einige Regeln gegeben:

(A) „Menschen, die in PLZ 02139-Gebiet leben, tendieren dazu, Demokraten zu sein“ (Support 19%, Confidence 57,9%)

(B) „Frauen, die in Cambridge (021**) leben und in den Siebziegern (197**/**/**) registriert sind, tendieren dazu, Demokraten zu sein. (Support 2,6%, Confidence 83,0%)

(C) „Weiße Republikaner, die in PLZ 15213-Gebiet leben und ein Haus besitzen, tendieren dazu, Frauen ohne Kinder zu sein“. (Support 2,1%, Confidence 55,4%)

Die multidimensionale generalisierte Assoziationsregel (A) ist die traditionelle, mit den Werten aus der niedrigsten Stufe der Hierarchie ausgedrückte AR. Die Regeln (B) und (C) haben auch Termen aus höheren Stufen der Hierarchien. Die Tabelle, aus der die Werte für die Regeln (A) und (B) stammen, hatte die Attribute {5-Ziffer PLZ, Geschlecht, Registrierungsdatum (Jahr/Monat/Tag), Partei}. Die Tabelle für die Regel (C) hatte zusätzlich noch {Rasse/Volkstum, eigenes Haus, Anzahl von Kindern}.

In diesen Beispielen sind Unterschiede sowohl in Semantik als auch in Support und Confidence zu sehen. Betrachtet man die folgenden Beispiele, so sieht man, dass während die Support- und Confidence-Werte bei allen Regeln gleich sind, die Regeln jedoch semantische Unterschiede haben:

(D): „Menschen, die in PLZ 1521*-Gebiet leben, tendieren dazu, im Jahr 1965 (1965/**/**)“ registriert zu sein“

(E): „Menschen, die in PLZ 152**-Gebiet leben, tendieren dazu, im August 1965 (1965/08/**) registriert zu sein“

(F): „Menschen, die in PLZ 152**-Gebiet leben, tendieren dazu, im PLZ-1521-Bezirk zu leben und im August 1965 (1965/08/**) registriert zu sein“.

Die Regel (F) ist die so genannte „**robuste Regel**“. Sie hat am meisten generalisierten Ausdruck im Body und am meisten spezialisierten Ausdruck im Head.

Die Interpretation der Assoziationsregeln bei Menschen ist ähnlich der „if-then“-Implikation. In der mathematischen Implikation $p \Rightarrow q$ ist p die Hypothese und q die Konklusion. Eine Assoziationsregel (geschrieben „Body $\Rightarrow$ Head“), die die Hypothese (Body) mit einer großen Anzahl von potenziellen Elementen und die Konklusion (Head) eng spezifiziert hat, kann als mehr nützlich oder mehr interessant betrachtet werden. Die Begründung ist folgende: Die breit ausgedrückte Hypothese expandiert den Gültigkeitsbereich der Subjekte der Hypothese, und die spezifisch ausgedruckte Konklusion liefert exakte Informationen über diese Subjekte. D. h., gesucht sind die Regeln, die möglichst allgemeinen (generalisierten) Body und möglichst präzisen (spezialisierten) Head haben. Diese werden „robuste Regeln“ genannt.

Eine robuste Regel hat sowohl quantifizierbare Maße, wie Support und Confidence, als auch semantische Nebenbedingungen. Der Body einer Regel ist mehr generalisiert als der Body einer anderen, wenn seine Terme aus dem gleichen oder einem höheren Hierarchielevel stammen, als die Terme im Body der anderen, und/oder er hat weniger Terme als der Body der anderen. Ähnlich, der Head einer Regel ist spezieller ausgedrückt als der Head einer anderen Regel, wenn seine Terme aus dem gleichen oder niedrigeren Hierarchielevel

stammen als die Terme im Head der anderen, und/oder er hat mehr Terme. Bei den gleichen Support- und Confindence-Tupel unter mehreren Regeln, ist die Regel mit dem am meisten generalisiert ausgedrückten Body und am meisten spezialisiert ausgedrückten Head die robuste Regel. Eine Menge von Regeln beinhaltet mindestens eine robuste Regel. Es sind auch mehrere möglich.

Für ein gegebenes Attribut A aus der Tabelle D wird eine **Generalisierungshierarchie der Domänen** DGH_A („domain generalisation hierarchy“) für eine Menge von Funktionen $f_h : h = 0,...,k-1$ definiert, so dass:

$$A_0 \xrightarrow{f_0} A_1 \xrightarrow{f_1} ... \xrightarrow{f_{k-1}} A_k$$

$$A = A_0 \text{ und } |A_k| = 1.$$

Die DGH_A wird über $\bigcup_{h=0}^{k} A_h$ definiert.

Die A_h-s sind die Domäne der Attribute auf jeweiliger Hierarchiestufe und die f_h-s die Funktionen, die die lineare Ordnung auf A_h-s, bestimmen. Dabei ist A_0 das minimale Element in der Basisdomäne, und A_k das maximale Element. Es ist die Einelementigkeit von A_k gefordert, um es zu ermöglichen, dass alle Werte zu einem einzigen Wert generalisiert werden können. Weil die generalisierten Werte anstelle der mehr spezifischen Werte benutzt werden, ist es wichtig, dass alle Domänen in der Hierarchie semantisch kompatibel sind.

Gegeben sei die Generalisierungshierarchie der Domänen DGH_A für ein Attribut A.

Wenn $v_i \in A_i$ *und* $v_j \in A_j$, sagen wir $v_i \le v_j$ dann und nur dann, wenn

$i \le j$ und $f_{j-1}(...f_i(v_i)...) = v_j$

Das definiert die partielle Ordnung „$\le$“ auf $\bigcup_{h=0}^{k} A_h$.

Eine solche Beziehung impliziert die Existenz einer **Generalisierungshierarchie der Werte für ein Attribut** A: VGH_A („value generalizaton hierarchy“). Es sind auch unterschiedliche DGH_A für ein und dasselbe Attribut möglich. Aus Gründen der Einfachheit wird diese Situation aber nicht betrachtet. Für eine gegebene Tabelle $D(A_1, A_2, ..., A_m)$ und die DGH_A-s $1 \le i \le m$, die Menge der möglichen Generalisierungen der Werte in D ist die Generalisierungshierarchie $GH_D = DGA_{A1} \times ... \times DGA_{Am}$, also ein Kartesisches Produkt der DGH_A-s. GH_D definiert einen Verband, dessen minimales Element den Wert in der gleichen Domäne wie D hat. Jeder Knoten in diesem Verband repräsentiert eine eindeutige Kombination der Generalisierungskonzepte für die Regeln in D. Die Anzahl der Knoten in dem Verband ist das Produkt der Größen den Attributhierarchien:

$$\prod_{i=1}^{m} |DGH_i| := |DGH_1| \cdot |DGH_2| \cdot ... \cdot |DGH_m|$$

Man kann die Knoten, die durch die Menge der Generalisierungsfunktionen (f_1**,...,** f_m) verbunden werden, als Paare definieren. Dabei ist die Gesamtanzahl der Paare im Verband die Anzahl der möglichen multi-dimensionalen „Cross-Level"-Regeln, die repräsentiert werden können. Die Anzahl der insgesamt möglichen Regeln ist größer und wird durch die Anzahl der Werte in der Datenbanktabelle gegeben.

Für die weiteren Erläuterungen muss die Notation des Konzepts der gerichteten azyklischen Graphen eingeführt werden. (DAG, „Directed Acyclic Graph"). Ein Knoten wird durch kleine Buchstaben, z. B. *x* bezeichnet. Wenn es eine Kante zwischen *x* und *y* gibt, sagt man, dass die Knoten *x* und *y* „verbunden" sind (unabhängig von der Richtung der Kante). Man bezeichnet einen Knoten *p* als „Vater" von *c* und *c* als „Kind" von *p*, wenn es eine Kante gerichtet von *p* nach *c* gibt. Man sagt, „*p* ist ein Vorfahre von *c*" und „*c* ist ein Nachkomme von *p*", wenn es einen Weg im DAG von *p* nach *c* gibt.

(Angepasste) **Regeldefinition** nach Li und Sweeny:

Definition 17

Es sei $A = \{A_1,..., A_m\}$ eine Menge der Attribute in einer relationalen Datenbank *D*, wo jedes A_1 eine assoziierte Hierarchie hat, deren VGH ein DAG ist.

Dann ist eine „multi-dimensionale generalisierte Assoziationsregel" eine Implikation der Form

$X \Rightarrow Y$, wo gilt : $X = \{v_1^x, v_2^x,..., v_m^x\}$, $Y = \{v_1^y, v_2^y,..., v_m^y\}$, und v_i^x *und* v_i^y sind Werte in der A_i-s Hierarchie. Es ist gefordert, dass v_i^y kein Vorfahre von v_i^x ist, d. h., $v_i^y < v_i^x$ für alle $1 \le i \le m$.[18]

X ist dann der Body und Y der Head der Regel. Support und Confidence werden wie im klassischen Sinne definiert.

Traditionelle Assoziationsregeln sind ein spezieller Fall dieser Definition, meinen die Autoren.[19]

Z. B., die traditionelle Regel der Form $\{v_1^x\} \Rightarrow \{v_2^y\}$ ist äquivalent mit der zur oberen Definition passenden Regel der Form

$\{v_1^x, *_2, *_3 ..., *_m\}$, $Y = \{v_1^y, *_2, *_3 ..., *_m\}$, wo $v_1^x = v_1^y$ und jedes $*_i$ das höchste Level der Generalisierung in der A_i's Hierarchie repräsentiert, d. h. Wurzel in der VGH_{Ai} ist.

Der Vorteil einer solchen Definition ist die Möglichkeit, interessantere generalisierte Regeln auszudrücken. Sei folgendes Beispiel:

Es sei A={5-Ziffer-PLZ, Registrierungsdatum (JJ/MM/DD), Status}. Angenommen, es gibt eine Regel {021**, 1996/**/**,*} $\Rightarrow$ {021**,1996/08/**, aktiv} mit 89,9% Confidence und 4,5% Support. Diese Regel lernt sowohl „in der Breite", d. h. vom „nichts wissen" zum „etwas wissen" über ein Attribut („* $\Rightarrow$ aktiv" für Attribut „Status"), als auch in die Tiefe, d. h. vom „etwas wissen" zum „etwas mehr spezifisch wissen" über ein Attribut

[18] Wie man sieht, haben die Regeln nach dieser Definition eine gleiche Anzahl von Stellen *m* auf beiden Seiten

[19] Dabei soll nicht zwangsläufig die feste Stellenanzahl auf beiden Seiten der Regel gewährleistet werden.

(„1996/**/** $\Rightarrow$ 1996/08/**“ für Attribut „Registrierungsdatum“), was der Form „Vorfahre(x) $\Rightarrow$ x“ entspricht. Die letzte Art des Lernens „in die Tiefe“ könnte man nicht mit traditionellen AR betreiben.

Zu Veranschaulichung sei eine Attributwerte-Tabelle für 2 Attribute wie in Abbildung 10 gegeben. Aus diesen Werten wird für jedes Attribut eine VGH gebildet, die auf den weiteren Abbildungen (Abbildung 11 und Abbildung 12) zu sehen sind. Werden die beiden VGHs verbunden, so entsteht ein Generalisierungsbaum, der in der Abbildung 13 dargestellt ist.

Nachfolgend wird der Generalisierungsbaum beschrieben.

3.2.4.3. Generalisierungsbaum

Definition 18

Ein Generalisierungsbaum (GenTree) ist ein DAG, der multi-dimensionale Generalisierungs-Beziehungen zwischen allen Datentupeln in der relationalen Datenmenge der hierarchisierten Attribute repräsentiert und die Kriterien der Vollständigkeit („completeness“) und Exaktheit („conciseness“) erfüllt.

Es gibt zwei Typen von Knoten im GenTree: „Blätter“ („leaves“) und „Nichtblätter“ (non-leaves“). Jedes Blatt repräsentiert ein korrespondierendes Datentupel. Jedes „Nichtblatt“ präsentiert dagegen die multi-dimensionale Generalisierungsform und die Menge aller Datentupeln, die zu dieser Generalisierungsform generalisiert werden können. Die Wurzel („root“) ist ein spezieller „Nichtblatt“-Knoten, der die allgemeinste Generalisierung aller Attribute und die Menge aller Datentupeln darstellt.

Notation

Mit *Form(x)* wird die korrespondierende multi-dimensionale Generalisierungsform (oder der Ausdruck wenn x ein Blatt ist) bezeichnet, die x darstellt. Ein Beispiel ist *Form(x)*=(ab*, 1*) in der Abbildung 11.

$Form(x)_i$ bezeichnet den Wert des *i*-ten Attributes in *Form(x)*.

Tuples(x) bezeichnet die Menge der Tupeln, die mit *Form(x)* generalisiert oder dargestellt werden können.

Definitionen im Generalisierungsbaum

Definition 19

Es wird definiert:

$Form(x)_i < Form(y)_i$ und $Form(y)_i > Form(x)_i$ dann und nur dann,

wenn die $Form(y)_i$ „genereller“ als die $Form(x)_i$ ist. D.h., es existiert ein Pfad in VGH von $Form(x)_i$ zu $Form(y)_i$;

$Form(x) = Form(y)$ dann und nur dann, wenn $Form(x)_i = Form(y)_i$ für alle $1 \le i \le m$; (dabei ist *m* die Anzahl der Attribute)
$Form(x) < Form(y)$ (und $Form(y) > Form(x)$) dann und nur dann, wenn
$Form(x)_j \le Form(y)_j$ für alle $1 \le i \le m$, und $Form(x)_j < Form(y)_j$ für mindestens ein $j, 1 \le j \le m$;
*x ist e*in *Vorfahre* von *y* (und *y* ist ein *Nachkomme* von *x*) dann, wenn $Form(x) > Form(y)$, und *x* ist *Vater* von *y* (und *y* ist *Kind* von *x*) wenn diese direkt verbunden sind.
Die *Wurzel* ist der Vorfahre von allen anderen Knoten und die Blätter sind nie Vorfahren.

ATTRIBUT 1	ATTRIBUT 2
aba	11
aab	11
abb	10
abb	11
aaa	11

Abbildung 10 Tabelle der Attributwerte

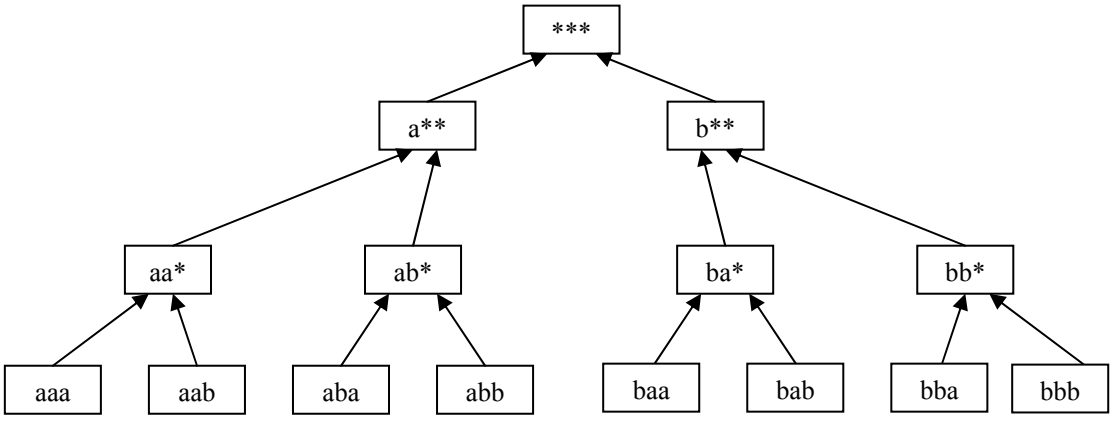


Abbildung 11 Hierarchie (VGH) des Attributes 1

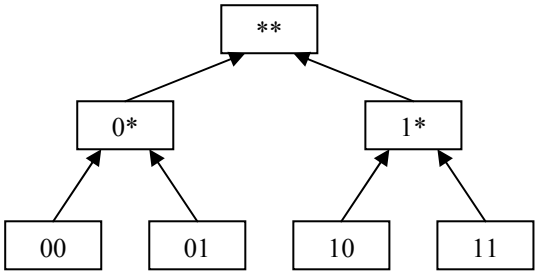


Abbildung 12 Hierarchie (VGH) des Attributes 2

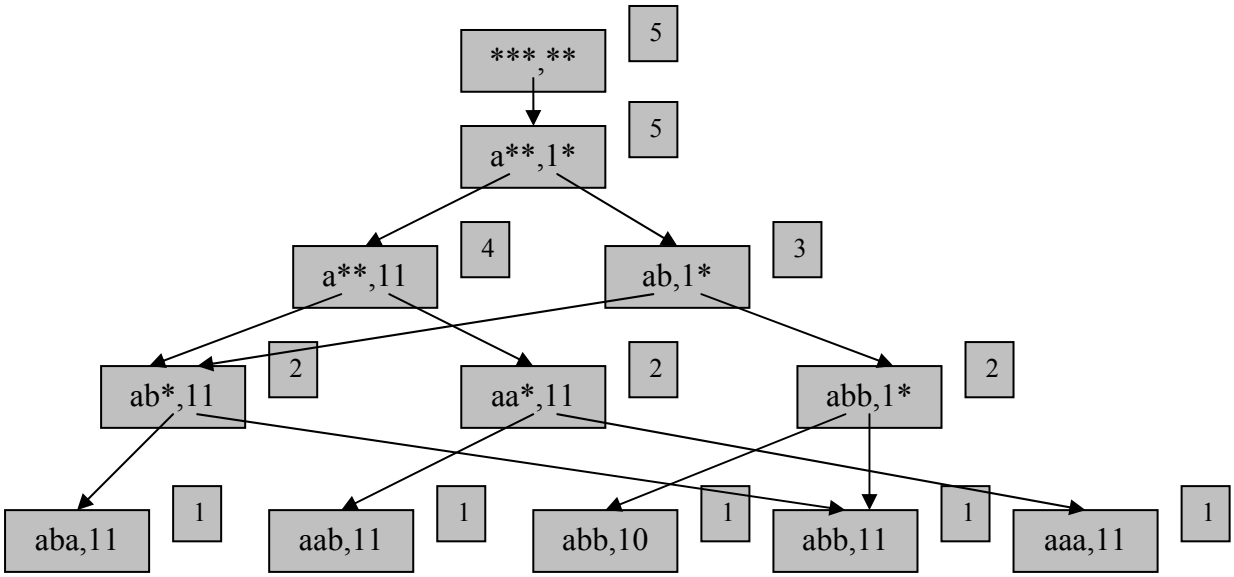


Abbildung 13 Der GenTree, aufgebaut anhand der Tabelle *D* und VGH's [20]

3.2.4.4. Diskussion

Im Gegensatz zu anderen Autoren versuchen Li und Sweeney eine andere, erweiterte Bedeutung den generalisierten Assoziationsregeln zu geben: Sie sollen „in die Tiefe" lernen und möglichst allgemeine Hypothesen zu möglichst konkreten Konklusionen schlussfolgern.

Solche Regeln nennen die Autoren „robust". Die Autoren suchen direkt nur diese und keine anderen Regeln, ohne sie aus der gesamten Menge der Regeln filtern zu müssen. Sie bilden keine allgemein mögliche Struktur (im Gegensatz zu dem unten beschriebenen Psaila-und-Lanzi-Ansatz) und benutzen nur die gegebenen Daten für die Hierarchie- bzw. GenTree-Bildung.

Das Konzept des Generalisierungsbaumes ist eine originale Betrachtungsweise der Hierarchien und stellt eine interessante Methode der Hierarchiebildung dar. Hier ist der Ansatz interessant, mehrere Taxonomien zu verwenden, die auf der obersten Stufe als die „allgemeinste" Generalisierung zusammengefasst werden, d. h. eine gemeinsame Hierarchie bilden. Durch die vorher bekannte und feste Anzahl von Attributen, (und somit Hierarchien), haben die Regeln immer feste und auf beiden Seiten gleiche Anzahl der Elemente. Das kann man sowohl als Vorteil als auch als Nachteil interpretieren. Zum Beispiel wenn man diese Sichtweise auf die Daten aus dem zu behandelnden System überträgt, könnte man die Elemente der obersten Hierarchiestufe, also die Warengruppen, von denen es im System 38 gibt, als unabhängige Hierarchien betrachten, und die Regeln würden immer 38 Stellen auf jeder Seite haben. Das wäre natürlich Unsinn, weil die meisten Stellen auf beiden Seiten den gleichen Wert haben würden, abgesehen davon, dass die Regeln zu lang und kaum noch lesbar wären. Man könnte sich die Werte aber auf beiden Seiten, die gleich sind, „gekürzt" denken, dann hätte man die viel kürzeren Regeln, die wirklich nur die unterschiedlichen

[20] Die Tuppeln in der Tabelle erscheinen als Blätter im GenTree. Die mit dem Knoten assoziierte Zahl zeigt, wie viele Tupeln sind mit ihm repräsentiert, d.h. |Tupeles(x)|

Werte auf beiden Seiten stehen hätten und bei denen die rechte Seite der Regeln eine „detailliertere“ Darstellung der linken Seite wäre (um robuste Regeln zu haben). Aber an dieser Stelle sollte man sich den wichtigen Unterschied merken, der die Abbildung der Daten aus dem Bestellsystem auf die von den Autoren benutzen Daten nicht ganz ermöglicht: Angenommen, jeder Artikel wäre ein Attribut in der Darstellung von Autoren. Bei den Bestellungen müssen gar nicht (und werden auch eher sehr selten) die Artikel aus allen möglichen Warengruppen zusammen in jeder Transaktion vorkommen, wohingegen bei den Daten, die die Autoren benutzen, immer Attributwerte aus aller Attributhierarchien in jedem Datensatz (was einer unseren Transaktion entspräche) vorhanden sind.

Das Ziel, die robusten Regeln auf den im System vorhandenen Daten zu finden, kann aber auch durch Filtern erreicht werden (s. Kapitel 5.11.3.1), obgleich mit etwas Performanzeinbußen: bei genauer Untersuchung jeder gefundenen Regeln kann festgestellt werden kann, zu welcher Hierarchieebene ein Element gehört und welche Elemente auf welcher Seite einer Regel stehen. Wird bei solcher Untersuchung einer Regel festgestellt, dass die im Body stehenden Elemente aus den höheren Hierarchiestufen mit Elementen aus der untersten Stufe im Head spezialisiert werden, kann die Regel als „robust“ markiert werden.

3.2.5. Ansatz von Psaila und Lanzi

3.2.5.1. Motivation

Der Artikel [Psaila und Lanzi, 2000] beschreibt eine analytische Methode zur Unterstützung der Assoziationsregelentdeckung in Datenbanken bei der Benutzung von expliziten und impliziten Hierarchien in den Daten. Es wird versucht, Generalisierung über mehrere vorhandene Hierarchien zu erstellen und dabei interessante Regeln zu finden. Die Ausprägungen der Hierarchien werden miteinander kombiniert und untersucht. Wie gut ist eine Generalisierung? Um den Grad der Generalisierung bzw. ihre Nützlichkeit zu bewerten, führt die Methode eine passende Metrik ein, die den Autoren bei ihren Experimenten effektiv erschien.

Die Entdeckung startet zunächst mit Suchanfragen auf hohem Granularitätslevel (also wird mit den am wenigsten generalisierten Hierarchiestufen angefangen) und setzt die Generalisierung über die Hierarchien schrittweise solange fort, bis die neuen Suchen in höheren Stufen der Hierarchien weniger sinnvoll sind als die früheren. Es wird kein spezieller Algorithmus vorgeschlagen. Lediglich die methodische Unterstützung wird geboten. Es wird außerdem ein Konzept der sogenannten Meta-Patterns eingeführt. Die Abstraktion von Algorithmuswahl sehen die Autoren als einen Vorteil.

3.2.5.2. Ansatz, Metapatterns

Eine wichtige Frage bei der Entdeckung von ARs ist: „An welcher Art von Generalisierung sind wir eigentlich interessiert?“ Jeder Autor versucht diese Frage unter eigenem Blickwinkel zu beantworten. Die Autoren Psaila und Lanzi versuchen eine methodische Antwort auf diese Frage geben mit dem Begriff der „Metapatterns“ für die Assoziationsregelentdeckung (oder speziellen „mining-quiries“) zu geben.

Definition 20

Ein **Metapattern** für Assoziationsregeln ist ein Tupel

p:(T,g,m,s,c), wo *T*, *g*, *m*, *s* und *c* die Parameter des Metapatterns sind.

T ist die Faktentabelle, die die zu analysierenden Daten enthält. Die Notation *Schema(T)* bedeutet die Menge der Attribute (Spalten) der Tabelle.

m ist das Attribut der Regel (oder das „mined attribute"), d. h. das Attribut, an dem die Regel entdeckt werden sollte. Wenn V_m eine Domäne des Attributes *m* ist, dann assoziiert die Regel die Werte aus V_m.

Genauer, assoziiert eine gegebene Regel ***r*** die Teilmengen *B* und *H* aus V_m :

$r : B \Rightarrow H$, d. h., $B \subset V_m, H \subset V_m, B \bigcap H = \emptyset$. Die Größe („size") der Regel ist die Anzahl der Werte im Body und im Head, d. h. $size(r) = |B| + |H|$.

g ist das Gruppierungsattribut, das zeigt, wie die Regeln die Regelmäßigkeiten ausdrucken. Die Faktentabelle ist logisch in Gruppen aufgeteilt, die immer den gleichen Wert für das Gruppierungsattribut aufweisen. Regeln assoziieren Attributwerte, die in denselben Gruppen zusammen vorkommen. Die Anzahl der Gruppen sei als *G* bezeichnet[21].

s spezifiziert den Minimum-Support. Der Support einer Regel wird hier wie folgt definiert:

$s_r = G_r / G$, wobei G_r die Anzahl der Gruppen ist, die die Regel enthalten (also wird hier der relative Support benutzt),

c spezifiziert die Minimum-Confidence. Die Confidence wird hier wie folgt **definiert**:

$c_r = G_r / G_b$, wobei G_b die Anzahl der Gruppen ist, die mindestens den Body der Regel enthalten.

Ein Parameter des Metapatterns wird über die Punktnotation angesprochen:

z. B. mit *p.g* wird das Gruppierungsattribut *g* vom Pattern *p* gemeint.

In einem Metapattern kann entweder ein Attribut a von T oder ein Attribut b, dass man über die Dimension von a bekommt, ein Gruppierungsattribut (bzw. das „mined"-Attribut) sein: $p.g = a \rightarrow b$, bzw. $p.m = a \rightarrow b$.

Die Anwendung von Metapattern p an eine Instanz von Fakttabelle T produziert eine Menge von Assoziationsregeln (bezeichnet als R).

Angenommen, gegeben ist eine Tabelle (Tabelle 1) mit Transaktionen und ein Metapattern

p **: (***Transactions***,** *cust***,** *item***,** 0**,**18**,** 0**,**5**)**.

Dabei bedeutet:

„Transactions" die Faktentabelle, „cust" das Regel-(oder „mined"-) Attribut, „item" das Gruppierungsattribut, „0,18" den min-Support und „0,5" die min-Confidence.

Dieses Metapattern extrahiert die Regeln, die die Produkte assoziieren, die häufig bei einem Kunden gekauft wurden. Das mined-Attribut ist hier „*item*". Dabei sind die Transaktionen im

[21] Bei den Traditionellen Ansätzen ohne Gruppierung ist die Anzahl der Gruppen gleich der Anzahl der Transaktionen, da die Transaktionen nach ihren Nummern implizit gruppiert werden und jede Transaktion eine eindeutige Nummer hat. Folglich enthält jede Gruppe genau eine Transaktion.

Teil 1 der Tabelle nach Kunden gruppiert (Gruppierungsattribut „*cust*“). Relevant sind die Regeln, die bei mindestens 18 % der Kunden gelten (18% Support) und deren bedingte Wahrscheinlichkeit (also Confidence) mindestens 50 % beträgt.

cust	*date*	*item*	*store*
c_1	26/01/99	A	1
c_1	26/01/99	B	1
c_1	31/01/99	C	1
c_2	28/01/99	C	3
c_2	29/01/99	D	3
c_2	29/01/99	E	2
c_3	28/01/99	F	1
c_3	29/01/99	A	1
c_4	30/01/99	B	1
c_4	30/01/99	D	1
c_4	30/01/99	E	2
c_5	26/01/99	F	1
c_5	01/02/99	A	2
c_6	26/01/99	B	3
c_6	26/01/99	A	3
c_7	01/02/99	E	2
c_8	01/02/99	F	2
c_8	01/02/99	E	2
c_9	02/02/99	E	2
c_9	02 /02/99	A	2
c_9	03/02/99	B	1
c_9	03/02/99	C	1
c_{10}	04/02/99	B	2
c_{10}	04/02/99	A	2
c_{11}	05/02/99	B	3

cust	*date*	*item*	*store*	*cust* → *city*	*item* → *sub-cat*
c_1	26/01/99	A	1	$city_1$	K
c_1	26/01/99	B	1	$city_1$	X
c_1	31/01/99	C	1	$city_1$	K
c_2	28/01/99	C	3	$city_1$	K
c_2	29/01/99	D	3	$city_1$	X
c_2	29/01/99	E	2	$city_1$	Z
c_{11}	05/02/99	B	3	$city_1$	Z
c_3	28/01/99	F	1	$city_2$	K
c_3	29/01/99	A	1	$city_2$	X
c_{10}	04/02/99	B	2	$city_2$	X
c_{10}	04/02/99	A	2	$city_2$	Z
c_4	30/01/99	B	1	$city_3$	Z
c_4	30/01/99	D	1	$city_3$	K
c_4	30/01/99	E	2	$city_3$	X
c_5	26/01/99	F	1	$city_3$	K
c_5	01/02/99	A	2	$city_3$	Z
c_6	26/01/99	B	3	$city_3$	Z
c_6	26/01/99	A	3	$city_3$	Z
c_7	01/02/99	E	2	$city_4$	Z
c_8	01/02/99	F	2	$city_5$	K
c_8	01/02/99	E	2	$city_5$	X
c_9	02/02/99	E	2	$city_5$	K
c_9	02/02/99	A	2	$city_5$	X
c_9	03/02/99	B	1	$city_5$	K
c_9	03/02/99	C	1	$city_5$	X

Tabelle 1 Faktentabelle[22]

[22] Faktentabelle "Transactions", im Teil 1: gruppiert nach Kunden, im Teil 2: erweitert mit dem Dimensionsattributen *cust* → *city* und *item* → *sub-cat* und gruppiert nach Dimensionsattribut *cust* → *city.*

Dabei werden mit dem oberen Metapattern auf diesen Daten folgende Regeln produziert:

$A \Rightarrow B \; s_r = 0{,}364 \; c_r = 0{,}667$

$A \; B \Rightarrow C \; s_r = 0{,}182 \; c_r = 0{,}5$

$A \; C \Rightarrow B \; s_r = 0{,}182 \; c_r = 1$

$B \; C \Rightarrow A \; s_r = 0{,}182 \; c_r = 1$

$C \Rightarrow A \; s_r = 0{,}182 \; c_r = 0{,}667$

$C \Rightarrow B \; s_r = 0{,}182 \; c_r = 0{,}667$

$C \Rightarrow E \; s_r = 0.182 \; c_r = 0.667$

$D \Rightarrow E \; s_r = 0.182 \; c_r = 1$

$F \Rightarrow A \; s_r = 0.182 \; c_r = 0.667$

3.2.5.3. Simplified Metapatterns

Definition 21

Ein **vereinfachtes („simplified“) Metapattern** $\overline{p}:(T,g,m)$ ist ein Metapattern ohne Minimum-Support und Minimum-Confidence Parameter:

$\overline{p}:(T,g,m) = (T,g,m,0,0)$

3.2.5.4. Patterngeneralisierung

Es wird die Patterngeneralisierung eingeführt, die durch Generalisierung von entweder Gruppierungs- oder mined -Attributen realisiert wird. Es wird eine passende Metrik dargestellt, mit der das Interesse der Regeln und der Generalisierungsgrad bewertet werden können. Dann werden zwei Generalisierungsoperatoren definiert, die die vereinfachten Metapattern über die Dimensionen des Datenschemas generalisieren. Mit diesen Operatoren wird ein Verband gebildet.

Es sei p': $(Transaktions, cust \rightarrow city, item, 0{,}18, 0{,}5)$ ein Metapattern, das die Transaktionen über die Städte der Käufer gruppiert. (Siehe rechter Teil der Tabelle 1). Man sieht, dass die Gruppen, die jetzt von Pattern p' definiert werden, im Vergleich zu den Gruppen des vorherigen Patterns p größer werden und dass ihre Anzahl sinkt. Jede Gruppe enthält jetzt eine oder mehr Gruppen vom Pattern p. Als Effekt steigen die Regelgröße und der Support.

Als Konsequenz, wenn die Gruppen, die ein Metapattern definiert, zu groß sind, können wir alle möglichen Kombinationen der Werte eines Regelattributes in der entdeckten Regelmenge bekommen. Das gleiche gilt für die Generalisierung des mined-Attributes, weil die Anzahl der mined-Werte sinkt. D. h., eine zu starke Generalisierung kann dazu führen, dass man alle trivialen Regeln entdeckt.

Deshalb ist es wichtig, ein geeignetes Maß, oder eine „Metrik“, wie die Autoren sie nennen, zu haben, die eine Bewertung der Metapatterns und der produzierten Regeln ermöglicht. Sind die Metapatterns, für deren Bildung in Hierarchien die Gruppierungs- bzw. mined-Attribute benutzt werden, und die Menge der produzieren Regeln tatsächlich sinnvoll? Es wird versucht, mit Hilfe von dieser Metrik die Antwort auf diese Frage zu geben. Die Metrik soll nur auf den semantischen Eigenschaften von Metapatterns basieren (Gruppierungs- und Mined-Attributen). Eine Metrik, die auf Support und Confidence basieren würde, würde zu spezifisch sein und würde die Generalisierungs-Idee der Metapatterns nur schwierig verständlich machen. Natürlich, wenn die Regeln bereits extrahiert sind, kann der Benutzer nach minimum-Support und minimum-Confidence befragt werden. Aber das sind die erweiterten Einstellungsmöglichkeiten. Wichtig ist es, zunächst zu verstehen, welche Art von Regelmäßigkeiten man untersucht.

Deshalb werden weiterhin nur vereinfachte Metapatterns betrachtet.

Definition 22

Seien $\bar{p}$ *: (T,g,m)* ein vereinfachtes Metapattern und V_g und V_m die Domänen von *g* bzw. *m*

Wir bezeichnen:

mit z_i die Anzahl der unterschiedlichen Werte von *m*, die in der *i*-ten Gruppen g_i vorkommen;

mit $\bar{a}_{g,m}$ die mittlere Anzahl der unterschiedlichen Werte von dem Regelattribut in einer Gruppe

$\bar{a}_{g,m} = (\sum_{g_i} z_i)/|V_g|$,

$\bar{f}_{g,m} = \bar{a}_{g,m}/|V_m|) = (\sum_{g_i} z_i)/(|V_g| \times |V_m|)$

dann ist $\bar{f}_{g,m}$ die gesuchte Metrik. Diese Metrik zeigt den Anteil der unterschiedlichen Werte der Regelattributwerte in einer Gruppe. Je größer $\bar{f}_{g,m}$, desto größer die Anzahl der Attributwerte in einer Gruppe. Deshalb steigt $\bar{f}_{g,m}$ bei der Generalisierung über die Metapattern, wenn entweder die Anzahl der Gruppen, oder die Anzahl der Attributwerte sinkt, (also hat die Metrik *f* die Monotonieeigenschaft).

In dem Beispiel mit Pattern p': $(Transaktions, cust \rightarrow city, item, 0{,}18, 0{,}5)$ bekommt man den Wert von $\bar{f}_{g,m}$ aus der Tabelle (Tabelle1, Teil2) so:

Die Anzahl der Werte von mined-Attribut (*item*) $|V_m| = |\{A,B,C,D,E,F\}| = 6$;

Die Anzahl der Gruppen ($cust \rightarrow city$) $|V_g| = |\{citi_1, citi_2, citi_3, citi_4, citi_5\}| = 5$;

Die Summe der unterschiedlichen Werte der Attribute in jeder Gruppe $\sum_{g_i} z_i = 5+3+5+1+5 = 19$

Die gesuchte Metrik ist $\bar{f}_{g,m} = \frac{19}{30} \approx 0{,}63$. Bei dem Wert macht es noch vielleicht Sinn, dieses Pattern anzuwenden, meinen die Autoren. Je nach dem, welche Support- und Confidence-Werte man dann anschließend einsetzt.

Bei Anwendung des Patterns $(Transaktions, \{store \rightarrow city\}, \{item \rightarrow sub\text{-}cat\})$ steigt der Wert von der Metrik auf $\bar{f}_{g,m} = 0{,}8$. Das ist schon ein sehr hoher Wert und es hat keinen Sinn, so die Autoren, das Pattern anzuwenden, und zwar unabhängig von Support und Confidence. Die Autoren argumentieren, dass ein übermäßig hoher Wert von f auf eine sehr starke Generalisierung deutet, die dazu führt, dass viele triviale Regeln entdeckt werden, und zwar unabhängig von min-Support und min-Comfidence-Parametern. Beispielsweise, werden bei Anwendung des Metapatterns $(Transaktions, \{store \rightarrow city\}, \{item \rightarrow sub\text{-}cat\})$ alle möglichen Kombinationen der Produktkategorien entdeckt, weil es klar ist, dass jede Produktkategorie in jeder Stadt verkauft wird.

Die Autoren haben experimentell bestimmt, dass für die Obergrenze der Metrik ein Wert von $\bar{f}_{g,m} = 0{,}7$ am besten passt. [23]

3.2.5.5. Generalisierungsoperatoren und Verbände

Zwei Operatoren für die Generalisierung werden definiert: Einer, der die Gruppenattribute generalisiert, und ein anderer, der die mined(Regel-)-Attribute generalisiert.

Definitionen 23

Gegeben seien ein Datenbankschema S und zwei simplified-Metapatterns $\bar{p}_1$, $\bar{p}_2$. Der **Operator** $\uparrow g(\bar{p}_1) = \bar{p}_2$, wird **Generalisierungsoperator des Gruppierungsattributes** genannt und generalisiert $\bar{p}_1$ zu $\bar{p}_2$.

Analog, $\uparrow r(\bar{p}_1) = \bar{p}_2$ heißt **Generalisierungsoperator des Regelattributes** und generalisiert $\bar{p}_1$ zu $\bar{p}_2$.

Das Ziel dieser Operatoren ist, einen Formalismus zu schaffen, der die Generalisierungsbeziehung unter Metapatterns abbildet.

Es sei $\bar{p}_b$ ein simplified Metapattern, das nur aus den Attributen der Fakttabelle besteht. Dann wird es **Basic-Metapattern** genannt, weil es nicht mit Generalisierungsoperatoren abgeleitet werden kann.

$\bar{p}_b$ sei ein solches Pattern. Das Generalisierungsverhältnis *L* wird dann so definiert:

[23] Bemerkung: es ist fraglich, ob es tatsächlich in allen Situationen und für alle Daten ein passender Wert ist. Die Güte und die Genauigkeit dieser Metrik kann man in praktischen Experimenten noch mal überprüfen. Siehe dazu das spätere Kapitel 5.5

$L(\overline{p}_b) \subseteq P(\overline{p}_b) \times P(\overline{p}_b) \times \{\uparrow g, \uparrow r\}$, wo $P(\overline{p}_b)$ die Menge der Metapatterns ist, die die Generalisierung von $\overline{p}_b$ vereint mit $\overline{p}_b$ ist. Der Tupel $(\overline{p}_1, \overline{p}_2, \uparrow g) \in L(\overline{p}_b)$ (bzw. $(\overline{p}_1, \overline{p}_2, \uparrow r) \in L(\overline{p}_b)$) genau dann, wenn $\uparrow g(\overline{p}_1) = \overline{p}_2$ (bzw. $\uparrow r(\overline{p}_1) = \overline{p}_2$).

Dabei ist $L(\overline{p}_b)$ ein **Verband**.

Beispielsweise sieht ein mit dem Basic-Metapattern $\overline{p}_b = (T, cust, item)$ abgeleiteter Verband wie in Abbildung 14 aus. Man sieht, dass im Verband alle möglichen Kombinationen von beiden Attributhierarchien vorhanden sind. Die Operatoren eignen sich deshalb gut, um mehrere Hierarchien miteinander zu verbinden.

Kurz kann man das Verfahren von Psaila und Lanzi so **zusammenfassen**:

Man bildet alle möglichen Kombinationen aus den vorhandenen Hierarchien und bekommt dabei alle möglichen simplified-Metapatterns. Das erreicht man durch die Anwendung von Generalisierungsoperatoren.

Um festzustellen, welches oder welche von den gebildeten Metapatterns „gut“ sind, berechnet man die Metrik f und wendet sie so an: man bewegt sich von den weniger zu den mehr generalisierten Metapatterns in dem Metapattern-Verband, und zwar so lange, bis die Metrik f über dem vorher vorgegebenen Grenzwert liegt. Wegen der Monotonieeigenschaft von f ist es nicht notwendig, weiter zu gehen.

Erst dann sucht man die Regeln mit den „guten“ Metapatterns. Die Suche kann z. B. mit Apriori und mit Hilfe der Metapatterns gebildeten „neuen“ Transaktionen durchgeführt werden, wie im Kapitel 5.6 beschrieben ist.

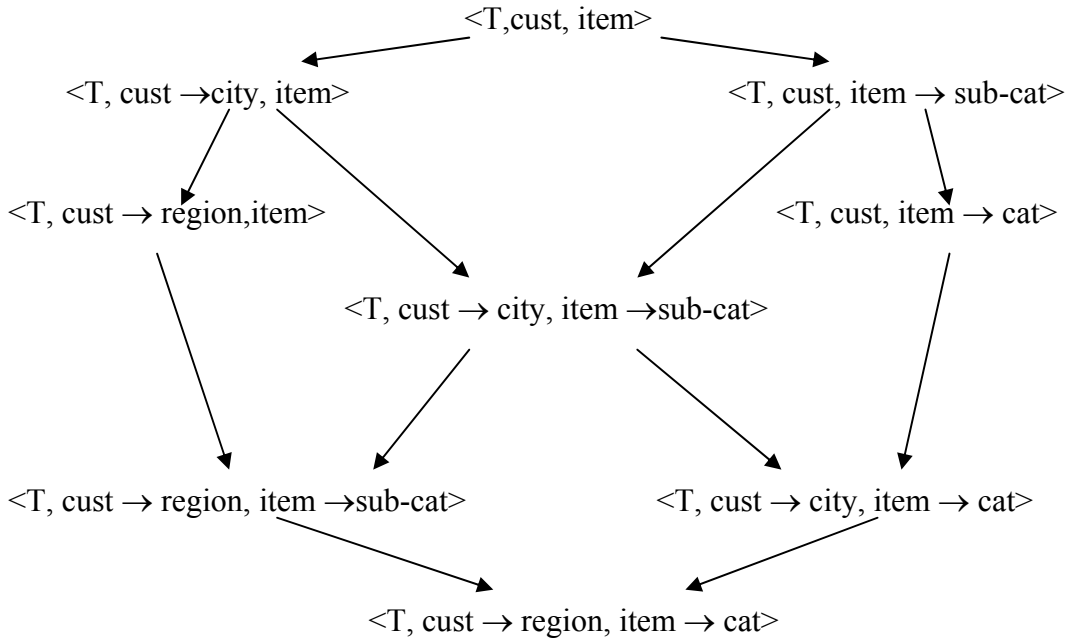


Abbildung 14 Verband aus Kunden- und Artikelattributen

3.2.5.6. Diskussion

Den Ansatz Psaila und Lanzi kann man als eine Verallgemeinerung der vorherigen Ansätze sehen. Die Autoren haben versucht, eine vom Mining-Algorithmus unabhängige Sichtweise auf die generalisierten Regeln zu schaffen. Der Idee, dass man bei der Regelentdeckung im Prinzip nicht an Parameter gebunden sein soll, sondern einfach nach bestimmten Mustern in den Daten suchen will und diese erst dann genau untersucht, ist entscheidend. Die Metapatterns, insbesondere die simplified Metapatterns stellen die formale Basis für diese Idee dar und sind eine Art von „mining queries".

Während die anderen Autoren immer nur die Hierarchie auf Basis der Artikelattribute benutzen bzw. bilden, wird bei Psaila und Lanzi die Kundeninformation für die Bildung einer zusätzlichen Hierarchie herangezogen. Ein anderer interessanter Unterschied besteht in der Sichtweise auf die Bedeutung der Transaktionen: Die Autoren gruppieren die Käufe z. B. nach Kunden, unabhängig vom Zeitpunkt des Zustandekommens der Käufe. D. h., wenn ein Kunde mehrere Transaktionen im gewöhnlichen Sinne getätigt hat, werden diese als eine generalisierte Transaktion interpretiert. (Siehe den linken Teil der Tabelle 1, die Transaktionen sind nach customer-id gruppiert.) Wenn man dann eine Regel „Artikel A $\Rightarrow$ Artikel B (s=x, c=y)" verwendet, dann heißt es lediglicht, dass Kunden, die irgendwann den Artikel A gekauft haben, irgendwann auch den Artikel B (mit Support=x und Confidence=y) gekauft haben. Der zeitliche Zusammenhang ist nicht mehr gegeben. Würde man aber nach Datum und Kunden gruppieren, so würde sich wieder die herkömmliche Form der Transaktionen ergeben. Somit sind die konventionellen Transaktionen ein Spezialfall dieser allgemeinen Gruppierungsmethode. Man sollte sich im Klaren sein, dass diese Art der Gruppierung die Transaktionen prinzipiell verändert, so dass die Support- und Confidence-Werte geändert werden. Folgende Situation, die in Tabelle 2 dargestellt ist, illustriert die Aussage:

Kunde	**Ursprungliche Transaktion**	**Nach Kunden gruppierte Transaktion**
Kunde1	**A, B, D, K, L, O**	**A, B, D, E, K, L, M, O**
Kunde1	**A, E, D, M**	
Kunde1	**B, O**	
Kunde2	**…**	**…**
Kunde2	**…**	

Tabelle 2 Gruppierung der Transaktionen

Wie man sieht, verschwindet nach der Gruppierung die Information, dass die Artikel A, B, D, und O von Kunde 1 mehrmals gekauft wurden, sprich höheren Support hatten. Auch die Bedeutung ist anders: der Kunde1 hat irgendwann die Produkte A, B, C, D, K, L M, O gekauft, und zwar wirklich irgendwann und nicht unbedingt zusammen, d. h. die Formulierung ist ziemlich allgemein.

Das Konzept der Gruppierungsattribute ermöglicht dem Benutzer, über jedes mögliche Attribut zu generalisieren. So ist dem Benutzer überlassen, wie er die Hierarchie bildet. Was bringt diese beliebige Gruppierung? Je nachdem, an welchen Regelmäßigkeiten oder Unregelmäßigkeiten in Transaktionsdaten der Benutzer interessiert ist, kann er gezielt danach suchen: Ist der Benutzer z. B. an Zusammenkäufen einer Produktgruppe von Kunden aus einer bestimmten Stadt interessiert, kann er die Daten nach diesen beiden Attributen gruppieren und die Regel, die zu dem so formulierten Metapattern passt, suchen. Die Anwendung der konventionellen Parametern im Anschluss kann die Ergebnisse noch weiter verbessern.

Bei dem Verfahren bleibt allerdings noch offen, was passiert, wenn mehrere Patterns durch die Bewertung mit *f* als „gut" bezeichnet werden. Mit welchem oder welchen Pattern soll man dann die Regelsuche durchführen? Mit einem, allen, oder mit einem der freien Wahl?

Ist es überhaupt sinnvoll, alle möglichen Hierarchiekombinationen zu berechnen, oder sollte man lieber wie Li und Sweeney nur die vorhandenen Daten benutzen, anstatt die allgemeine Struktur zu erzeugen, die sowieso nur teilweise benutzt wird? Denn jede Berechnung bedeutet einen Durchgang über die Datentabelle. Bei den für die Experimente vorliegenden Daten macht es kein Problem aus, kann aber im Allgemeinen zu teuer sein. Die späteren Experimente werden zeigen, ob der Ansatz von Psaila und Lanzi sinnvolle Ergebnisse auf den vorhandenen Daten liefert.

3.3. Interessensmaße und Filtern von Regeln

Das Problem der Entdeckung von interessanten Assoziationsregeln in drei Stufen aufgeteilt werden:

1. Finden aller häufigen Itemsets (entsprechend dem minimum-Support-Wert)
2. Finden aller Assoziationsregeln (entsprechend dem minimum-Confidence-Wert)
3. Filtern von uninteressanten Regeln (anhand des Interessensparameters) aus der Gesamtmenge der Assoziationsregeln.

Die vorherigen Kapitel haben die ersten beiden Stufen behandelt. Dieses Kapitel soll sich mit der dritten Stufe beschäftigen.

3.3.1. Interessensmaß für die Regeln von Agrawal und Srikant.

Wie kann man das Interesse einer Regel bewerten? Verschiedene Autoren haben unterschiedliche Antworten auf diese Frage vorgeschlagen. Weiter unten werden einige Kriterien für die Interessensbewertung der Regeln dargestellt.

An dieser Stelle wird aber der Ansatz von Agrawal und Srikant in diesem Aspekt weiter vorgestellt. Die Autoren finden nämlich, dass der Ansatz von [Piatetsky-Shapiro 91] nicht sehr geeignet ist, da sich dabei relativ wenig statistisch nicht signifikanten Regeln herausfiltern lassen. Und zwar, wird bei diesem Ansatz eine Regel $X \Rightarrow Y$ als nicht interessant betrachtet, wenn folgendes gilt: $Support(X \Rightarrow Y) \approx Support(X) \times Support(Y)$.

Die Autoren führen in [Agrawal und Srikant, 1995] ein Kriterium ein, um eine interessante Regel von einer uninteressanten zu unterscheiden, das zunächst spachlich so formuliert werden kann:

Wenn der Support oder die Confidence einer Regel sich von dem erwarteten Support oder der erwarteten Confidence einer Regel um mehr als einen vom Benutzer vorgegebenen Faktor unterscheiden (also kleiner oder größer sind), gilt diese Regel als interessant.

Für die Formalisierung werden noch einige Begriffe benötigt:

Ein Itemset $\hat{Z}$ ist ein Vorfahre vom Itemset *Z*, ($\hat{Z}, Z \subseteq I$), wenn $\hat{Z}$ von Z abgeleitet werden kann, indem die Elemente von Z mit ihren Vorfahren aus der Hierarchie ersetzt werden und die Gesamtanzahl der Elemente in Z und $\hat{Z}$ gleich ist. Die letztere Restriktion bedeutet, dass es nur dann Sinn macht, den erwarteten Support von $\hat{Z}$ aus *Z* zu berechnen, wenn diese die gleiche Anzahl von Elementen haben. Die Regeln $\hat{X} \Rightarrow Y, X \Rightarrow \hat{Y}$, oder $\hat{X} \Rightarrow \hat{Y}$ werden Vorfahren von der Regel $X \Rightarrow Y$ genannt. Aus einer gegebenen Menge von Regeln wird die Regel $\hat{X} \Rightarrow \hat{Y}$ direkter Vorfahre der Regel $X \Rightarrow Y$ genannt, wenn es keine Regel $X' \Rightarrow Y'$ gibt, so dass $X' \Rightarrow Y'$ ein Vorfahre von $X \Rightarrow Y$ und $\hat{X} \Rightarrow \hat{Y}$ ein Vorfahre von $X' \Rightarrow Y'$ wären. (Ähnliche Definitionen gelten auch für $\hat{X} \Rightarrow Y$ und $X \Rightarrow \hat{Y}$).

Es sei die Regel $X \Rightarrow Y$ gegeben und es sei $Z = X \bigcup Y$. Der Support von *Z* ist der gleiche wie von der Regel $X \Rightarrow Y$. Es sei $E_{\hat{Z}}[P(Z)]$ der erwartete Wert von $P(Z)$[24] bei gegebenem $P(\hat{Z})$, wo $\hat{Z}$ ein Vorfahre von Z ist. Es sei $Z = \{z_1,...,z_n\}$ und $\hat{Z} = \{\hat{z}_1,...,\hat{z}_j,\hat{z}_{j+1},...,\hat{z}_{n1}\}$, $1 \le j \le n$, wobei $\hat{z}_i$ ein Vorfahre von z_i ist. Dann gilt:

Definition 24

$E_{\hat{Z}}[P(Z)] = \frac{P(z_1)}{P(\hat{z}_1)} \times ... \times \frac{P(z_j)}{P(\hat{z}_j)} \times P(\hat{Z})$ ist der erwartete Wert von $P(Z)$ bei gegebenem Itemset $\hat{Z}$.

Ähnlich, sei $E_{\hat{X} \Rightarrow \hat{Y}}[P(Y \mid X)]$ die erwartete Confidence der Regel $X \Rightarrow Y$ bei gegebener Regel $\hat{X} \Rightarrow \hat{Y}$. Sei $Y = \{y_1,...,y_n\}$ und $\hat{Y} = \{\hat{y}_1,...,\hat{y}_j,\hat{y}_{j+1},...,\hat{y}_{n1}\}$, $1 \le j \le n$ und $\hat{y}_i$ ist ein Vorfahre von y_i Dann definieren wir:

Definition 25

$$E_{\hat{X} \Rightarrow \hat{Y}}[P(Y \mid X)] = \frac{P(y_1)}{P(\hat{y}_1)} \times ... \times \frac{P(y_j)}{P(\hat{y}_j)} \times P(\hat{Y})$$

Bemerkung: $E_{\hat{X} \Rightarrow Y}[P(Y \mid X)] = P[Y \mid \hat{X}]$

[24] $P(Z)$ bedeutet hier die Wahrscheinlichkeit von *Z*

Die Regel $X \Rightarrow Y$ ist "R Mal interessanter" als die Vorfahren-Regel $\hat{X} \Rightarrow \hat{Y}$, wenn der Support (oder die Confidence) um Rmal sich von dem erwarteten Support (oder die Confidence von der erwarteten Confidence) unterscheiden, der (die) auf Werten von der Regel $\hat{X} \Rightarrow \hat{Y}$ basieren.

Definition 26

Gegeben seien eine Menge der Regel S und ein Minimum-Interesse R. Die Regel $X \Rightarrow Y$ heißt **interessant** in S, wenn sie keine Vorfahren hat, oder sie ist Rmal interessanter als ihr nächster Vorfahre aus allen interessanten Vorfahren.

Wie die Autoren Han und Fu (vgl. [Han und Fu, 1999]) vorgeschlagen haben, können aus den gefundenen Regelnmengen die redundanten Regeln gelöscht werden. Den Begriff der **redundanten** Regel hatten die Autoren so formuliert:

„Eine AR ist redundant, wenn sie von einer AR der höheren Stufe berechnet werden kann …unter der Annahme der gleichen Daten-Verteilung…"
und wie folgt formal definiert:

Definition 27

eine Assoziationsregel $R, A_1 \wedge A_2 ... A_n \Rightarrow B_1 \wedge B_2 ... B_m$ ist redundant, wenn es eine Assoziationsregel $R', A'_1 \wedge A'_2 ... A'_n \Rightarrow B'_1 \wedge B'_2 ... B'_m$ gibt, wobei jedes Element in der Assoziationsregel R ein Nachkommen des Elementes des korrespondierenden Elements oder dasselbe Element in der AR R' ist, und $\varphi(R) \in [exp(\varphi(R)) - \alpha, exp(\varphi(R)) + \alpha]$,

wobei $\exp(\varphi(\mathrm{R})) = (\sigma(\mathrm{B_n}) / \sigma(B'_n)) \times (\sigma(\mathrm{B_n}) / \sigma(B'_n)) \times ... \times (\sigma(\mathrm{B_n}) / \sigma(B'_n)) \times \varphi(R')$ und α eine vom Benutzer definierte Abweichungs-Konstante und σ der Support von einem Itemset ist.

Für das Löschen solcher AR wird ein minimum-confidence Test durchgeführt, bei dem für jede starke AR *R* geprüft wird, ob AR *R'* von ihr ein Nachkomme ist. Wenn die Confidence von R $\varphi(R)$, innerhalb der Grenzen von der erwarteten Confidence mit Abweichung α liegt, wird die Regel verworfen.

Dieser Filter-Ansatz ähnelt sehr dem oben beschriebenen Interessensmaße von Agrawal und Srikant, da sowohl bei Agrawal und Srikant, als auch bei Han un Fu die Erwartungswerte, die auf Basis der Hierarchieinformationen und den Nachkommen-Regeln berechnet werden den tatsächlichen Werten gegenübergestellt werden und eine vom Benutzer frei definierbare Grenze für Abweichungen als Entscheidungskriterium benutzt wird.

3.3.2. Interessensmaße von Webb und Zhang.

Bei der Entdeckung der Assoziationsregeln ist die Voraussagbarkeit einer Regel nicht das einzige Kriterium für die Bewertung des Interesses der Regel. Oft sind die Stärke der Korrelation zwischen Hypothese und Konklusion und der erwartete Wert dieser

Korrelationsstärke gemeinsam eine Funktion für die Bewertung des Interesses einer Regel (vgl. [Piatetsky-Shapiro, 1991]).

Z. B., eine Regel „wer die Kosmetik kauft, kauft auch die Süßigkeiten“ (mit Confidence 95%) ist uninteressant, wenn 95 % der Kunden sowieso Süßigkeiten kaufen (vergl. [Webb und Zhang, 2003]) .

Diese Parameter der Regeln erlauben unterschiedliche Maße für die Bewertung der Differenz zwischen beobachtetem und erwartetem Grad der Korrelation zu Benutzen, um alle Regeln zu finden, die den benutzerdefinierten Nebenbedingungen (Constraints) entsprechen. Allerdings beruhen die meisten Maße auf der Benutzung der Minimum-Support-Bedingung. Diese werden für die Beschränkung des Suchraumes benutzt, um die Berechnung möglichst effizient zu machen. Dabei ist der Support oft nicht direkt mit dem Interesse der Regel zusammenhängend.

Ein berühmtes Beispiel ist das so genannte „Wodka und Kaviar“-Problem. Starke Korrelation zwischen Ketel Wodka und Beluga Kaviar kann ziemlich interessant sein, da diese Produkte einen großen Profit erzielen, auch wenn das Verkaufsvolumen dieser Produkte relativ klein ist und sie deswegen wahrscheinlich den Minimum-Support nicht erreichen. Auch wenn der minimum-Support nicht direkt mit dem Interesse der Regeln zusammenhängend ist, hat er einen großen Einfluss auf die Regelentdeckung: es besteht ein Risiko, dass viele wirklich interessante Regeln nicht gefunden werden.

Es muss nicht unbedingt eine natürliche untere Schranke für Support existieren. Assoziationsregeln mit dem Supportwert kleiner als nominiertem Minimum-Support werden nicht gefunden. Nicht häufige Itemsets können oft auch interessant sein, wie im obigen Beispiel mit „Wodka und Kaviar“-Problem, wobei hochwertige Artikel in vielen Fällen relativ unhäufig sein können. Nichtsdestotrotz sind diese von großem Interesse.

Folgende Gründe sind ein Argument gegen die Verwendung des Supports als Maß für die Interessensbewertung der Regeln:

1. Wenn sogar eine natürliche untere Schranke für den Support existiert, kann sie oft von den Analytikern nicht identifiziert werden.
2. Wenn sogar eine relevante minimale Häufigkeit spezifiziert werden kann, kann die Menge der häufigen Itemsets zu groß werden, um noch berechenbar zu sein.
3. Der Ansatz der häufigen Itemsets kann nicht die Nebenbedingungen für die Effizienz nutzen, die aus der Hypothese, aus der Konklusion, oder aus ihrer Union hervorgehen. D. h., es wird nur die Bedingung des Supports genutzt. Die Bedingung der Confidence hat dagegen fast keinen Einfluss auf die Berechnung, wobei gerade die Confidence die Beziehung zwischen dem Support der Hypothese und dem Support der Hypothese in Union mit der Konklusion widerspiegelt.

Ein möglicher Ansatz in dieser Hinsicht ist, den Minimum-Support abhängig von Items in den Itemsets variieren zu lassen. Das bringt größere Flexibilität, löst aber keine der oben beschrieben Probleme, obwohl das „Wodka-Kaviar-Problem“ dadurch gelöst wird (s. Kapitel 5.9.1). Die meisten Arbeiten auf dem Gebiet der AR-Discovery beschäftigen sich mit der Effizienzsteigerung des Entdeckungsprozesses. Dabei adressieren sie ebenso nicht die oben beschriebenen Probleme.

Es werden mehrere **Maße** für Interessensbewertung der Regeln in [Webb und Zhang, 2003] diskutiert:

1. $coverage(X \Rightarrow Y) = cover(X)$
2. $confidence(X \Rightarrow Y) = \frac{support(X \Rightarrow Y)}{coverage(X \Rightarrow Y)}$, also etwas anders als sonst definiert.
3. $leverage(X \bigcup Y) = support(X \Rightarrow Y) - cover(X) \times cover(Y)$ [25]
4. $lift(X \bigcup Y) = \frac{support(X \Rightarrow Y)}{cover(X) \times cover(Y)}$

In [Piatetsky-Shapiro 91] wurde argumentiert, dass viele Interessensmaße auf der Differenz zwischen der beobachteten gemeinsamen Häufigkeit der Hypothese und Konklusion ($support(X \Rightarrow Y)$) und der Häufigkeit, die erwartet werden würde, wenn X und Y unabhängig wären ($cover(X) \times cover(Y)$), basiert sind. Die einfachste Methode, diesen Unterschied zu bewerten, wäre das Maß Leverage anzuwenden. Dabei kann Leverage auch so formuliert werden:

$leverage(X \bigcup Y) = cover(X) \times (confidence(X \Rightarrow Y) - cover(Y))$. In dieser Form ausgedrückt, kann es auch als „wheigthed relative accuracy“ ([Todorovski et al., 2000]) bezeichnet werden. Das Maß Leverage ist interessant, weil es die Unabhängigkeit bzw. Abhängigkeit zwischen X und Y zeigt.

Dieses Maß Lift stellt ein Verhältnis zwischen der beobachteten Häufigkeit der Konklusion (Y) im Kontext der Hypothese (X) und der erwarteten Häufigkeit, wenn man die Unabhängigkeit von X und Y annimmt, dar.

Die Maße *Leverage* und *Lift* können später im praktischen Teil der Arbeit als Bewertungs- bzw. Filterkriterien für die Regeln eingesetzt werden.

[25] die Definitionen von cover, support siehe im früheren Kapitel 3.1.1

4. Planung der praktischen Schritte

Hier wird beschrieben, welche praktischen Aufgaben im Rahmen dieser Diplomarbeit bearbeitet werden mögen. Aus zeitlichen Gründen werden nur die wichtigten und interessantesten Schritte gemacht. Es wird jedoch versucht, die für die Integration in das vorhandene und im Kapitel 2.1 beschriebene Informationssystem relevanten praktischen Experimente durchzuführen und die Ergebnisse in das System zu integrieren. Die genauen Abläufe dieser Schritte werden in späteren Kapiteln diskutiert. An dieser Stelle sollen nur der Plan und die Vorgehensweise vorgestellt werden.

4.1. Anwendung von Apriori an vorhandene Daten zwecks Regelentdeckung

In diesem Schritt muss zunächst eine Auswahl der Implementierung von Apriori getroffen werden. Dabei werden mehrere freiverfügbaren Implementierungen in Betracht gezogen. Die Vor- und Nachteile der einzelnen werden in Bezug auf die vorliegenden Daten, Geschwindigkeit und die Integrationsumgebung diskutiert. Gegebenfalls wird eine Anpassung der ausgewählten Implementierung vorgenommen. Eine eigene Implementierung des Apriori-Algorithmus bleibt als letzte Auswegmöglichkeit immer noch, falls es nicht gelingt, eine der gefundenen zu verwenden.

Die Daten müssen an die ausgewählte Implementierung angepasst werden. Danach müssen die gefundenen Regeln interpretiert und in eine lesbare Form gebracht werden. D.h., die Datenvor- und Nachverarbeitungsphasen werden durchgeführt.

Durch die mehrfachen Experimente und Proben sollen die optimalen Parameter, Datenformate und Darstellungsmöglichkeiten für die Ergebnisse gefunden werden. Es werden sowohl einfache, als auch generalisierte Regeln gesucht. Bei den generalisierten kann es sich um *Cross-Level* und auch *Multiple-Level* Regeln handeln. Außerdem kann die Regelentdeckung zunächst auf der gesamten Menge der Transaktionen, und danach auch auf den deren Untermengen als gezielte Suche innerhalb von bestimmten Produkt- oder Warengruppen durchgeführt werden. Dadurch haben die Items bzw. Itemsets innerhalb der ausgewählten Gruppen einen höheren Support und bilden deshalb mehr Regeln. Falls man zuvor wenig Regeln hatte, wird es vielleicht hilfreich, wenn man mehr Regeln „provozieren“ kann. Man muss sich aber bewusst sein, das diese Regeln allerdings andere Bedeutung haben, als die Regeln, die über die gesamten Transaktionsdaten gelernt werden. Sie bilden Zusammenhänge der Elemente innerhalb der Gruppen ab. Diesen Zusammenhang wird weiter noch untersucht (s. die Neugruppierung im Kapitel 4.3).

4.2. Verbesserung der generalisierten Regeln

Nach dem die Regelsuche einige Regel liefern wird, sollen diese auf ihre Aussagekräftigkeit, und Rechtfertigkeit analysiert werden. Die gefundenen Regeln sind vielleicht nicht alle interessant oder es sind zu viele bzw. zu wenige Regeln. Deshalb sollen die Ergebnisse analysiert und die Regelmenge „verbessert“ werden. Das bedeutet, dass die redundanten und uninteressanten Regeln herausgefiltert werden müssen. Dies kann einerseits mit Filterung anhand der oben vorgestellten Parameter (*lift*, *leverage*, Maß von Agrawal und Srikant, etc.. s.

Kapitel 3.3) für die Interessenmaße, anderseits durch herausfiltern von *starken* und von *robusten* Regeln realisiert werden.

Außerdem sollen die Generalisierungseigenschaften, d. h. die Güte der Generalisierung und die Gültigkeit der produzierten Regeln untersucht werden. Es soll geprüft werden, ob die Regeln gerechtfertigt sind, das heißt, ob die Elemente aus den höheren Hierarchiestufen als „Vertreter“ der Elemente aus den niedrigeren Stufen benutzt werden dürfen. Dafür werden die Gruppenauslastungen bei den Transaktionen berechnet. Dabei wird untersucht, wie stark die eine oder die andere Gruppe, (am besten die Produktgruppe), in den Transaktionen ausgenutzt wird. Beispielsweise wird bei einigen Gruppen nur ein relativ kleiner Anteil der Artikel gekauft, wohingegen bei den anderen Gruppen sehr viele Artikel gekauft werden. Je nach dem, welche Grenze für die Gruppenausnutzung gewählt wird, werden die Gruppenelemente durch die Gruppe generalisiert oder nicht.

Außerdem soll die oben vorgestellte Metrik f für die Bewertung und Verbesserung der Generalisierungseigenschaft benutzt werden. Allerdings soll zunächst ermittelt werden, ob diese Metrik auf den vorhandenen Daten überhaupt anwendbar ist.

Diese Techniken werden in den Pre- und Postprocessing-Phasen angewandt, so dass der Entdeckungsalgorithmus an sich (Apriori) unverändert bleibt.

4.3. Hierarchien verbessern bzw. bilden

Wie es bereits im vorherigen Textabschnitt gesagt wurde, sollte vielleicht nicht immer die Gruppe als generalisierte Struktur in die Regel übernommen werden, weil es nicht gerechtfertigt ist, für nur wenige Artikel gleich die ganze Gruppe verantwortlich zu machen. Stattdessen sollte vielleicht eine andere Gruppenbildung vollzogen werden, bei der die Gruppenelemente in einem Zusammenhang stehen. Dafür wird Analyse der vorhandenen Hierarchien und Vergleich dieser mit den vorliegenden Transaktionen durchgeführt. Dies wird mit folgender Methode gemacht:

Man gruppiert paarweise die ähnlichen häufigen Zusammenkäufe (Itemsets) so, dass sie in allen Elementen bis auf jeweils eins gleich sind.

Beispiel: seien 2 „ähnliche“ häufige Itemsets gegeben:

S_1: = {A, B, C, D} und S_2:={A, B, C, E}. Diese Itemsets unterscheiden sich nur um ein Element. Vermutlich handelt es sich bei den Elementen D und E um ähnliche Artikel. Manche Kunden kaufen D, manche E, aber auf jeden Fall sind die oft zusammen mit A, B und C gekauft worden. Dann kann man die neuen Gruppen so bilden, dass sie entweder durch die Vereinigung der Artikeln aus S_1und S_2 ({A, B, C, D, E}), oder als Gruppe, die aus diesen beiden unterscheidenden Artikel ({D, E}) gebildet werden.

Wenn man aber genau überlegt, dann ist die Gruppe {D, E} als selbständige Gruppe (sagen wir *M*) eigentlich nicht sinnvoll: diese Artikel werden ja gerade nicht, (oder höchstens so selten, dass sie keinen häufigen Itemset ausmachen), zusammen gekauft. Die Gruppe {A, B, C, D, E} ist aber sehr wohl sinnvoll, da sie einen häufigen Zusammenkauf darstellt. Man muss natürlich an der Stelle den Unterschied zwischen der neu gebildeten Gruppe (sagen wir *N*) und den gewöhnlichen Itemsets (z. B. wie S_1 oder S_2) sehen:

$N := A \wedge B \wedge C \wedge (D \vee E)$, (also sowohl „„UND“-verknüpfte“ als auch „ODER“-verknüpfte Elemente), während $S_1 := A \wedge B \wedge C \wedge D$, $S_2 := A \wedge B \wedge C \wedge E$, (also nur „„UND“-verknüpfte“ Elemente.)

also ist $N := S_1 \vee S_2$.

Um diese neuen Gruppen als Hierarchiegruppen betrachten zu können, werden alle Elemente darin „ODER“-verknüpft. Denn eine Hierarchiegruppe ist eine „ODER“-Verknüpfung ihr zugehörender Artikel.

Diese neuen Gruppen kann man mit den vorhandenen Produktgruppen vergleichen und Gemeinsamkeiten und Unterschiede feststellen. Man könnte diese neuen Gruppen als eine Art der Gruppenbildung betrachten. Die so neu gebildete Gruppe kann in die generalisierten Regeln einfließen, anstatt die vorgefertigten Gruppen aus der vorhandenen Hierarchie zu benutzen. Dann ist die Genauigkeit und Gerechtfertigkeit der generalisierten Regeln gewährleistet.

In Bezug auf die oben angesprochene Gruppenauslastung würden solche neuen Gruppen voll ausgelastet sein, denn sie wären ja nur aus den gekauften Artikeln gebildet. Wie im vorherigen Abschnitt bereits erwähnt, ist eine möglichst volle Auslastung wünschenswert, da sie eine Benutzung der Gruppen anstatt der Gruppenelemente als Generalisierung ermöglicht, ohne die Semantik der Regel zu ändern. Ein Problem würde aber die (automatische) Namensgebung für manche dieser Gruppen darstellen. Schließlich sollen diese Gruppen in Regeln erscheinen, die von Menschen gelesen und verstanden werden müssen.

Dieses Verfahren könnte man auch für automatische Gruppenbildung nutzen, falls keine Hierarchie vorhanden ist. Die so entstandenen Gruppen wären wahrscheinlich für Menschen von ihrem Zusammenhang her nur schwer verständlich, trotzdem würden sie die zusammengehörige Artikel im Bezug auf ihre Käufe zu den Gruppen zusammenfassen.

Denkbar ist, dass solche automatische Gruppenbildung für die Fälle nützlich sein könnte, wo eine manuelle Zuordnung der Artikel zu Gruppen, also eine Hierarchiebildung, zu teuer und eine „nicht ganz genaue“, aber immerhin automatisch gebildete Hierarchie wünschenswert wäre.

4.4. Anreichern der Transaktionen mit Zusatzdaten

Durch die ersten Probexperimente wurde festgestellt, dass die Regeln, die gefunden werden, nicht immer interessant sind, auch wenn sie mit Interessensmaßen und Parametern bewertet und gefiltert werden. Es kann sein, dass die Regeln wenig Interessantes preisgeben, obwohl sie den Interessensmaßen entsprechen und mit besten Filtern gefiltert wurden.

Um eine breitere „Palette“ an interessanten Regeln zu bekommen, bzw. um mehr Regelmäßigkeiten oder Unregelmäßigkeiten im Kaufverhalten verschiedener Kunden zu entdecken, könnte man die Transaktionen mit zusätzlichen Daten erweitern. Eine Möglichkeit wäre, die Artikelattribute in die Regelendeckung mit einzubeziehen, eine andere, die Transaktionen mit Kundeninformationen anzureichern. Die Artikelattribute wären als Gruppierungsattribute (wie bei [Psaila und Lanzi, 2000] vorgeschlagen und oben beschrieben) gut geeignet. Man könnte bei der Erweiterung der Transaktionen mit Gruppen (nach dem Srikant- und Agrawal-Prinzip) die neuen Gruppen benutzen, die anhand der Artikelattribute

gebildet worden wären. Doch dadurch, dass die Artikel sehr viele Attribute haben, wäre die Auswahl der interessanten davon sicher schwierig.

Welche Artikel-Attribute könnten überhaupt in Frage kommen? (Zu den existierenden Attributen siehe das Kapitel 2.2.1.) Man könnte z. B. den Preis als Gruppierungsattribut verwenden, indem man die Artikel in mehrere Preisgruppen unterteilt. Es gibt aber unterschiedliche Preis-Attribute, wie den Einkaufspreis und den empfohlenen Verkaufspreis, außerdem sind die Preise für unterschiedliche Mengen angegeben, die über die Preiseinheiten erst erfragt werden müssten. Der Versuch, den Preis als Gruppierungsattribut zu verwenden, ist mit zusätzlichem Rechenaufwand verbunden und kann nicht für alle Artikel gleichermaßen angewandt werden.

Man könnte die Artikel nach dem Attribut „Mengeneinheit“ gruppieren und diese Gruppenzugehörigkeit benutzen: die Gruppen wären dann „Stückartikel“, „Kilogrammartikel“, „Tonnenartikel“, „Sackartikel“, etc. Die Verteilung der Artikel wäre aber sehr ungleichmäßig, weil die meisten Artikel die „Stückartikel“ wären. Ähnlich sieht es mit der Verpackungseinheit aus, wenn man diese als Gruppierungsattribut benutzen würde: die Verteilung wäre auch hier sehr ungleichmäßig.

Man könnte aber diese Artikelattribute trotz der erwähnten Nebeneffekte bei der Neugruppierung benutzen. Ein eventuell passendes Attribut wäre das Artikelbild. Z. B. kann man anhand des gleichen Artikelbildes die Artikel in die Gruppen aufteilen. Höchstwahrscheinlich sind diese Artikel aber von Anfang an in den gleichen Produktgruppen. Durch die Experimente sollte sich herausfinden lassen, welche der Attribute am besten für die Neugruppierung passen würden.

Aber jetzt noch einmal zurück zu den Zusatzelementen in den Transaktionen.

Wenn man aber wirklich die Transaktionen „anreichern“ möchte, sollten die Kundeninformationen herangezogen werden. Außerdem haben die Transaktionen die Eigenschaft, dass sie immer von nur einem Kunden gebildet worden sind. Und deswegen hat die Kundeninformation in einer Transaktion einen Zusammenhang mit allen in ihr enthaltenen Artikeln.

Die Kunden haben weniger Attribute, die in Frage kommen könnten, um die Transaktionen mit Zusatzinformationen zu versorgen. Man kann die Attribute, wie die Stadt, in der die Firma ansässig ist (die Kunden sind in diesem Fall gewerbliche Kunden), oder die Größe der Firma etc. als ein oder mehrere zusätzliche „Artikel“ in die Transaktion einfügen. Diese Information würde neue Regeln entstehen lassen.

Die Regeln, die aus den mit den Kundeninformationen erweiterten Transaktionen gewonnen werden, haben auch andere Bedeutung. Sie sagen nicht nur über die Zusammenkäufe der Artikel etwas aus, sondern auch etwas über die Käufer. Andererseits kann man diese Informationen als Gruppierungsattribute (wie die Artikelattribute oben, nach dem [Psaila und Lanzi, 2000]-Verfahren) verwenden. Eine solche Regel könnte z.B. „Kunden aus Dortmund, die Artikel aus den Produktgruppen A und B gekauft haben, haben auch Artikel aus Produktgruppe C gekauft…“ lauten.

Eventuell kann man die Transaktionen auch so gruppieren, dass man alle Transaktionen eines Kunden als eine Transaktion sieht, unabhängig von der tatsächlichen Zeit der Bestellungen.

Ein anderes für die Gruppierung passendes Kundenattribut wäre die Postleitzahl, nach der man die Kunden z. B. in west- und ostdeutsche Kunden aufteilen kann (je nach der ersten

Stelle der PLZ, die bei den ostdeutschen Kunden eine „0“ wäre). Wenn man dann die Regel entdeckt, kann der Manager sie zur besseren Angebots- und Preisgestaltung abhängig von den Kaufgegebenheiten benutzen. Dann entstehen völlig anders gruppierte Transaktionen. Mit den neuen Gruppen werden wieder andere, parallel existierende Hierarchien gebildet.

In beiden Fällen (1. Erweiterung der Transaktionen durch Kundenattribute und 2. Neugruppierung nach Kundenattributen) würden die entdeckten Regeln die Zusatzinformationen über den Kunden (Firmen) abbilden.

Eine weitere Möglichkeit wäre, die Transaktionen in Untermengen zu gruppieren und auf diesen getrennt die Regeln zu entdecken. Es würde den relativen Support und die Confidence der Regeln vergrößern, was die zuvor wegen des kleinen Support-Wertes unentdeckten Regeln finden ließe.

4.5. Integration der Ergebnisse in das vorhandene System

Abschließend sollen die erreichten Ergebnisse nutzbar gemacht werden: die Regelentdeckungsroutinen samt verschiedenen Filterungs- und Suchkriterien sollen in das System mit entsprechenden Darstellungsmöglichkeiten integriert werden. Die Such- und Darstellungsmöglichkeitsparadigmen sollen entsprechend den im System vorhandenen Services gestaltet werden. Um eine effiziente Suche mit vielen unterschiedlichen Parametern für die Regeln zu ermöglichen, sollen die Regeln automatisch mit vielen unterschiedlichen Parametern als Cronjob[26] regelmäßig generiert und in die Datenbank gespeichert werden. Außerdem kann das in [Klementitinen et. al, 1996] vorgestellte Konzept der Templates benutzt werden, bei dem die Suchergebnisse bestimmten Vorgaben entsprechen sollen. Als Ausblick kann man sich die Benutzung der entdeckten Regeln für eine Vorschlaggenerierung bei der Produktsuche sowie die Integration in einen im System vorhandenen E-Mail-Benachrichtigungs-Service vorstellen, um so das Kaufinteresse der Kunden und dadurch den Umsatz zu steigern.

[26] „Der Cronjob , oder Cron Daemon ist eine Jobsteuerung von Unix bzw. Unix-artigen Betriebssystemen wie Linux, die wiederkehrende Aufgaben oder Befehle zu einer bestimmten Zeit ausführen kann. Die auszuführenden Einträge werden in der sog. crontab gepflegt, wobei jeder Benutzer seine eigene crontab besitzt. Außerdem gibt es eine systemweite Tabelle“, Quelle: [Wikipedia]

5. Praktische Experimente

5.1. Implementierungsauswahl

5.1.1. Datenvorverarbeitung

Um die Experimente durchführen zu können, muss zunächst die passende Implementierung des Kernalgorithmus Apriori gewählt werden. Die Wahl der Implementierung hängt sehr von den vorliegenden Daten ab. Viele der frei verfügbaren Implementierungen sind nur für kleinere Datenmengen und begrenzte Datenformate ausgelegt. Außerdem ist ihre Laufzeit sehr unterschiedlich. Hier noch einmal eine kurze Beschreibung der Daten, die für diese Arbeit vorliegen, und ihrer Formate.

Die gesamte Artikelmenge beinhaltet ca. 25000 Artikel, die in mehr als 3000 Produktgruppen unterteilt sind, die wiederum ca. 40 Warengruppen bilden. Dabei sind die Artikelnummern 10-stellig, wobei 6 zusammenstehende Stellen der Artikelnummer bereits eindeutig einen Artikel identifizieren. Welche 6 Stellen es sind, ist vorher nicht bekannt. Die Produktgruppennummern können unterschiedlich lang sein, jedoch sind die Zahlen höchstens 4-stellig. Analog sind die Warengruppen nummeriert.

Der Datenbestand umfasst zurzeit über 60000 Transaktionen, von denen aber aus Datenschützgründen noch nicht alle zur Analyse verwendet werden dürfen. Die Transaktionen liegen in Form von Dateien vor. Die Dateien beinhalten unter anderem Informationen über die gekauften Artikel, die Kundennummer und das Bestelldatum. Diese Informationen sollen aus den Dateien herausgeparst werden. Danach können die anderen notwendigen Informationen über Produktgruppen und Warengruppen sowie zusätzliche Kundeninformationen aus der Datenbank selektiert werden.

Da die meisten Implementierungen von Apriori eine Datei mit Transaktionen als Eingabe benötigen, sollen alle Transaktionen in eine solche gemeinsame Transaktionen-Datei überführt werden. Ein im Rahmen dieser Arbeit extra dafür entwickeltes Programm löst diese Aufgabe. Es parst die gesamte Menge der Transaktionsdateien, findet die benötigten Informationen, selektiert die zusätzlichen Informationen aus der Datenbank und fügt alles zusammen zu einer bzw. mehreren Dateien mit allen Transaktionen[27]. Warum sollen mehrere Dateien geschrieben werden? Unterschiedliche Kombinationen der Hierarchiestufen werden gleichzeitig in unterschiedliche Transaktionsdateien geschrieben, so dass diese später nicht noch einmal berechnet werden müssen. Die höheren Hierarchiestufen werden dabei mit Präfixen versehen (s.u.). Zu diesem Zeitpunkt sollte die Apriori-Implementierung gewählt worden sein, um das Format seiner Eingabedatei(en) zu bestimmen, die von dem erwähnten Parseprogramm geschrieben werden.

[27] Die Laufzeit bei der Ausführung dieses Programms und bei ca. 28000 Transaktionsdateien lag je nach Systemauslastung bei ca. 7 bis 8 Minuten.

5.1.2. Verfügbare Implementierungen

Nach der umfangreicher Suche im Internet und vielen Tests, konnten mehrere Implementierungen ausselektiert werden. Hier sind die interessantesten tabellarisch dargestellt. Die kurzen Beschreibungen beruhen auf den weiter unten beschriebenen Experimenten. Hier soll nur kleine Übersicht gemacht werden (Tabelle 3).

Autor und Link	Programmier-Sprache	Vorteile	Nachteile[28]	Bemerkung
Ferenc Bodon, http://www.cs.bme.hu/~bodon/en/index.htm	**C++, sources und binary verfügbar**	**Eingabe: nur 1 Datei mit Transaktionen, minsup und (optional) minconf. Es wird nur 1 Datei mit sowohl häufigen Itemsets als auch Assoziationsregeln erzeugt, die später in einem Schritt bearbeitet werden kann. Unabhängig von Transaktionslänge und Gesamtanzahl der Items. Sehr schnell, wenn die Items entsprechend mit kleineren Zahlen kodiert sind. Bei größeren Zahlen immer noch schnell genug. Maximale Itemsetlänge konfigurierbar. Anzahl der Elemente im Head der Regel unbegrenzt. (s. u.**)**	**Schwer zu ändern, da im Gegensatz zu Programmierung in Java wenig Erfahrung in C++-Programmierung vorhanden, deshalb Vorzug einer Java-basierter Implemenierung. Die Eingabedaten dürfen nur „unsigned integer“ als Darstellung der Items in Transaktionen (s. u. Bemerkung *) beinhalten. Relativ hoher Speicher-Verbrauch. Keine Filter bzw. Interessensmaße. Letztere Versionen des Programms erzeugen keine Regeln. Kein GUI**	**Die Laufzeit abhängig von Zahlengrößen, mit denen die Items kodiert sind. Passend für die Entdeckung der Generalisierten und Robusten Regeln, da kein Beschränkung der Head-Länge. Schwierigere Datenvorverarbeitung, da Itemkodierungs-Möglichkeit begrenzt.**
Christian Borgelt http://fuzzy.cs.uni-magdeburg.de/~borgelt/apriori.html	**C++, sources und binary verfügbar**	**Sehr schnell, erwartet als Eingabe eine Datei mit Transaktionen. Items können als „signed long“ kodiert werden (s. u. * warum für hier wichtig), optionale Filterung der Regel mit mehreren Kriterien möglich. Anzahl der Items und Transaktionslänge unbegrenzt. GUI zu älteren Versionen vorhanden.**	**Schwierigkeit bei der Änderung mit gleicher Argumentation wie oben. Großer Nachteil für bevorstehende Experimente: nur 1 Element im Head der Regeln möglich (s. Bemerkung **). Relativ hoher Speicherverbrauch.**	**Viele kleinere Experimente durch GUI-Nutzung erleichtert. Jedoch unpassend bei der Entdeckung der generalisierten Regeln sowie robusten Regeln, die mehr als ein Element im Head beinhalten können.**

[28] Die Nachteile sind nicht als allgemeine Nachteile zu verstehen, sondern im Hinblick auf die Verwendung im Rahmen dieser Arbeit, die Möglichkeit der Änderung und der Anpassung des Quellcodes, die Programmiersprache, in der die jeweilige Implementierung vorlag, sowie Integrationsmöglichkeit in das vorliegende Informationssystem.

Bart Goethals http://www.cs.helsinki.fi/u/goethals/software	**C++, sources und binary verfügbar**	**Schnell. Benutzt effizientere DS (tries anstatt hashtree). Eingabe: Transaktionen und minsup, produziert nur häufige Itemsets, aus denen aber später die Regeln mit separatem zweitem Algorithmusteil erzeugt werden können.**	**Schwierigkeit bei der Änderung mit gleicher Argumentation wie oben. Keine negativen Itemkodierungen möglich. Keine Erzeugung der Häufigen Itemsets und Regeln in einem Schritt. Relativ hoher Speicherverbrauch. Kein Filter. Kein GUI**	**Ungefähr so schnell, wie die Implementierung von Ferenc Bodon. Gut geeignet zur Prüfung von deren Ergebnissen. Da aber zwei Schritte nötig, weniger geeignet im Allgemeinen.**
Frans Coenen http://www.csc.liv.ac.uk/~frans/KDD/Software/AprioriTFP/aprioriTFP.html	**Java**	**Java-Implementierung, relativ leichter zu verstehen und zu ändern. Modifizierter Apriori-Algorithmus namens AprioriTFP, ,unabhängig von Transaktionslänge. Sehr gute integrationsmögl in das Informationssystem, da Java-basiert**	**Sehr hoher Speicherverbrauch, da in Java geschrieben, bei kleinen Supportwerten (was bei den Daten notwendig ist) hohe Laufzeit und ungenügend Arbeitspeicher. Kein GUI**	**Nicht geeignet für praktische Experimente, da Speichermangel. Gut zum Verständnis geeignet.**

Tabelle 3 Implementierungsauswahl

Es gibt noch andere Java-basierte Implementierungen, die aber gar nicht in Frage kommen, da sie eine feste Transaktionslänge voraussetzen, in der binär das Vorkommen eines jeden Artikels abgebildet wird. Die unterschiedlich großen Transaktionen und die große Artikelmenge lässt dieses Format eher schwer verwenden. Außerdem gibt es größere Miningtools, die Apriori beinhalten. Diese benutzen aber auch andere Formate und können auch schlecht in das Bestellsystem später integriert werden

* Warum ist es bei den vorliegenden Daten wichtig, dass die Kodierung der Items mit beliebigen Zahlen erfolgen kann? Bei der Suche der generalisierten Regeln sollen die Produktgruppen und die Warengruppen sowie später Kundengruppen von dem oben beschriebenen Programm entsprechend kodiert und in die Transaktionen eingefügt werden. Für den Algorithmus würden sie ja immer noch Items darstellen, und er würde sie wie die „normalen“ Items, also wie Artikel behandeln. Dann würden Regeln entstehen, in denen Vertreter aus unterschiedlichen Hierarchielevels vorkommen. Um die Regeln zu interpretieren, müssen diese jedoch voneinander unterschieden werden können. Wenn z.B. die Produktgruppen genauso wie Artikel und Warengruppen kodiert sind, z. B. durch positive Integer, kann man sie später nicht mehr auseinander halten. Man könnte evt. noch die Artikel von den Produktgruppen und von den Warengruppen unterscheiden, weil die Artikelnummer in den Transaktionen 10- oder 6-stellig sein könnten (bei 10-stelligen Nummern versagen aber alle der aufgeführten Implementierungen auf dem zur Verfügung stehenden Rechner, die 6-stelligen Nummern sind dagegen verarbeitbar), und die Produkt- und Warengruppennummern sind höchstens 4-stellig oder kleiner. Man kann sicher sein, dass es auch in Zukunft nicht mehr als 10000 Produkt- und Warengruppen geben wird. Aber wie sollen die Produkt- und Warengruppen von einander unterschieden werden? Und wie sollen sie später von Kundengruppen unterschieden werden? Deshalb entstand die Idee, die Produktgruppen z. B. mit einem Präfix „-1“, die Warengruppen mit „-2“ und die Kundengruppen mit „-3“ zu erweitern. Damit könnte man sie später eindeutig als solche identifizieren: eine negative Zahl

wäre auf jeden Fall keine Artikelnummer, und anhand der ersten Ziffer könnte man die Gruppen sofort zuordnen. Leider funktioniert nur eine der oberen Implementierungen mit negativen Zahlen. Genau diese kann aber keine Regeln mit mehreren Elementen im Head produzieren. Deshalb mussten andere Präfixe für die höheren Hierarchien gefunden werden, die keine „Überschneidungen" mit Artikelnummern verursachen können und nicht zu lang sind, um die Zahlengrößen möglichst klein zu halten, damit die Laufzeit nicht zu groß wird. Als Präfixe wurden für Produktgruppen „7777" und für Warengruppen „8888" gefunden[29]. Diese Zahlensequenzen kommen in keiner Artikelnummer vor, sind positiv und die Zahlen bleiben noch relativ klein. Z. B. wäre eine Produktgruppe, die die Produktgruppen-ID „1024" hat, dann als „77771024" kodiert, und die Warengruppe mit der ID „34" wäre mit „888834" kodiert. Einen Artikel mit solchen Nummern gibt es definitiv nicht. Somit können alle Hierarchievertreter auseinander gehalten werden. Diese Präfixe wurden zunächst bei der Berechnung verwendet.

** Warum ist es bei den Experimenten wichtig, dass die produzierten Regeln mehrere Elemente im Head haben können? Zum Beispiel bei der Entdeckung von robusten Regeln, wo die im Body der Regel stehenden Elemente aus höheren Hierarchielevels im Head spezialisiert werden:

Beispiel: „Wenn die Kunden aus der Kundengruppe K1 die Artikel aus den Warengruppe *A* gekauft haben, dann waren es Kunden mit *PLZ 44147* und das Produkt war *a*".

Aus den oberen Implementierungen wurde für die weitere Arbeit die Implementierung von Ferenc Bodon gewählt. Abgesehen von den genannten subjektiven Nachteilen, ist sie universal einsetzbar, und hat eine gute Laufzeit (Beispiele für die Laufzeit bei der Entdeckung der generalisierten Regeln später). Bei Anwendung nur an die unterste Hierarchieebene (also nur die Artikel) beträgt die Laufzeit unter 1 Sekunde! Allerdings verlangsamte sich durch die großen Zahlen, die bei Verwendung der oben genannten Präfixe für die Kodierung der oberen Stufen entstanden sind, die Berechnung etwas, und die Laufzeit des Apriori lag bei 20 Sekunden bis 1 Minute, je nachdem, welche Hierarchiestufen in die Berechnung miteinbezogen waren. Das war zwar keine große Laufzeit, aber trotzdem störend. Deshalb habe wurde etwas die Implementierung der Regelinterpretation weiter verbessert und es konnte dann doch die Kodierung der Stufen in dem Parseprogramm mit Präfixen „1" bzw. „2" verwenden. Außerdem hatte wurden die Kundengruppen direkt in dem Parseprogramm gebildet und mit Präfix „3" versehen. Jetzt konnte man alles auseinander halten. Dabei wurde die Laufzeit des Apriori bei der Crosslevel-Regelberechnung wieder gesenkt und lag bei minsup 0,1% und minconf 40% und Berechnung auf allen 3 Stufen der Artikelhierarchie, ohne Regelinterpretation bei ca. 8-9 Sekunden. Das war auch für die Echtzeitverwendung[30] akzeptabel.

[29] Nur in der Anfangsphase, später wurden jedoch andere Präfixe festgelegt und verwendet. Später ist es jedoch gelungen, kleinere Präfixe zu verwenden: „1" für die Kodierung der Produkt-, „2" der Waren- und „3" der Kundengruppen. Das reduzierte die Laufzeit.

[30] Unter Echtzeitverwendung wird hier und weiter hin die Verwendung in einer reellen praktischen Managerarbeit verstanden, die bestimmten zeitlichen Einschränkungen unterliegt. Z.B. soll die maximale Antwortzeit nach einer Anfrage an das System nicht mehrere Minuten überschreiten. Dabei ist zu beachten, dass der Anwender kein unerfahrener Benutzer ist, sondern die eventuell etwas erhöhte Antwortzeit des Systems in solchen Anwendungsfällen vermutet und in Kauf nimmt.

5.2. Erste Experimente und Ergebnisse

Die ersten Experimente wurden auf den unterschiedlichen Hierarchieebenen und ihren Kombinationen durchgeführt. Die Experimente haben gezeigt, dass die unterste Ebene einen sehr kleinen Support aufweist. Bei einem minsup von 1% werden kaum häufige Itemsets und dementsprechend Regel gefunden. Bei einem minsup von 0,1% werden wesentlich mehr häufige Itemets gefunden. Trotzdem dominieren sehr stark bestimmte Artikel, wie z. B. diverse Bohrer, die unterschiedliche Maße haben, aber im Grunde sehr ähnlich sind. Ähnlich sieht es mit den Meißeln aus. Es gibt aber auch viele Artikeln, die nicht den Minsup-Wert erreichen, obwohl gleichzeitig viele andere sehr ähnliche, aber doch andere Artikel in den Transaktionen vorkommen. Würden diese alle gleich bezeichnet, würden sie doch den minsup erreichen. Das bestätigt fast die Notwendigkeit, solche ähnlichen Artikel nicht im Einzelnen zu betrachten und von einander genau zu unterscheiden, sondern sie evtl. mit einem „Vertreter“ zu ersetzen, der z. B. für alle Bohrer einer bestimmten Sorte und unabhängig von unterschiedlichen Durchmessern in den Regeln agieren würde.

Die Interpretierung der Regeln wurde im Groben so implementiert:

Eine der Dateien, die mit Apriori von Ferenc Bodon erzeugt wurde und die die häufigen Itemsets und die Regeln enthält, wird in Hauptspeicher zeilenweise eingelesen. Es werden die Supportwerte aller häufigen Itemsets gemerkt, die in der Datei bereits vorhanden sind. Bei den Regeln werden die Body- und die Head-Seiten gemerkt und die Support- und Confidence-Werte der Regeln gemerkt. Gleichzeitig wird beim Lesen geprüft, zu welcher Hierarchiestufe die Elemente gehören. Der Support, der Lift und die Leverage-Werte der Regel werden bestimmt und gemerkt, um anschließend damit filtern zu können. Die Confidence und der absolute Support der Regel werden von dem Apriori bereits geliefert und stehen in der Datei. Dann werden die Regeln darauf geprüft, ob sie nicht die Abbildung der Hierarchie darstellen. Zur Erinnerung: vor dem Apriori-Lauf wurden die Transaktionen mit den Elementen der höheren Hierarchiestufen erweitert. Jetzt kann es natürlich sein, (und ist auch tatsächlich in vielen Fällen so), dass es Regeln gibt, die nichts anderes besagen, als die Zugehörigkeit der Artikel zu Produktgruppen und/oder Warengruppen, oder Produktgruppen zu Warengruppen. Z. B. „Produkt Bohrer 5mm $\Rightarrow$ Produktgruppe Bohrer“ oder „Produktgruppe Bohrer $\Rightarrow$ Warengruppe Werkzeuge und Bauelemente“. Dabei enthält der Body nur solche Elemente, die die Elemente der Unterstufen von Elementen aus dem Head sind. Solche Regeln sind natürlich uninteressant und werden herausgefiltert. Die Regeln, die bleiben, werden in Listen eingefügt, und diese werden nach vorgegebenen Parametern sortiert. Die Elemente sind in dem Moment noch in ihrer ID-kodierten Form dargestellt und müssen mit ihren textuellen Namen ersetzt werden. Diese Informationen werden aus der Datenbank selektiert. Danach wird die Liste der Regeln mit Hilfe einer JSP-Seite, die in das System integriert ist, dargestellt (Abbildung 15).

Die Implementierung ist für die ersten Experimente und zum Testen der Filter gedacht und wird noch verbessert. Z. B. wenn die Regeln später nicht aus der Datei gelesen werden sollen, sondern in die Datenbank vorher importiert worden sind, ist der Arbeitsspeicherbedarf nicht so groß und die Laufzeit bei der Darstellung der Regeln kleiner. Zur Zeit dauert es relativ lange für einen Echtzeitbetrieb, wenn die Regeln interpretiert und angezeigt werden. Zwar dauert es bei den 1% minsup und 60% minconf-Regeln mit allen 3 Artikelhierarchie-Stufen (ca. 200 Regeln) nur ca. 0,5 Sekunden, was noch sehr gut ist, aber bei 0,1% minsup, wenn es über 36000 Regeln gibt, dauert es schon mehrere Minuten, bis die Seite mit den Ergebnissen

angezeigt wird. Außerdem läst sich die Suche nach bestimmten Regeln (mit Metapattern-Konzept, wie oben im Kapitel 3.2.5 beschrieben), leichter und bequemer realisieren.

5.3. Filtern der Regel mit Leverage und Lift

In den früheren Kapiteln wurde über unterschiedliche Filterungsmöglichkeiten der erzeugten Regeln berichtet. In den Experimenten sollten zunächst die Leverage und der Lift der Regeln berechnet und damit die Regeln gefiltert werden(die Definitionen dieser Parameter s. o. im Kapitel 3.3). Bei der Anwendung an der untersten Hierarchieebene (Artikel) wurden folgende Ergebnisse gefunden[31]:

Bei minsup 0,1% und minconf 30% wurden in 21777 Transaktionen (nur eine Teilmenge von allen Transaktionen, bedingt durch den Zeitpunkt der Experimente) 220 Regeln gefunden. Auf der Abbildung 15 sind die ersten 20 Regeln zu sehen.

Während die minimale Leverage 0,00096264 betrug, lag die maximale Leverage bei 0,00256149. Der minimale Lift lag bei 21,23 und der maximale bei 1.472,84. Die meisten der Regel haben einen sehr hohen Liftwert, was darauf zurückzuführen ist, dass der Support sehr klein ist. Einen genauen Wert für die Parameter, um die Regeln damit zu filtern, kann man eigentlich nicht nennen. Es werden auch keine Werte in der Literatur empfohlen. Deswegen wurde zunächst probiert, mit der minleverage, die knapp über dem minsup liegt, zu filtern, in diesem Fall mit 0,0012. Dabei wurde ungefähr die Hälfte der Regeln ausgefiltert. Es blieben nur 107 Regeln. Bei einer minleverage von 0,0015 wurden ca. 80% ausgefiltert und es sind nur noch 38 Regeln geblieben. Somit kann man sehen, dass bereits bei sehr klein definierter minleverage viele Regeln ausgefiltert werden können. Ob es wirklich die „uninteressanten" Regeln sind, die ausgefiltert werden, ist natürlich noch nicht klar. Beim Versuch, einen passenden Wert für den Lift zu finden, wurde einen Mittelwert zwischen den minimalen und maximalen Liftwerten verwendet. In diesem Fall war es ca.740. Dabei wurde ca. ein Drittel der Regeln ausgefiltert. Bei gleichzeitiger Anwendung von minleverage=0,0012 und minlift=740 blieben nur 33 Regeln übrig.

Nr.	Regel	Häufigkeit	Support	Confidence	Leverage	Lift
Nr. 1	[SPITZMEISSEL SDS-PLUS 250,-PROMAT- (produkt)] ➞ [FLACHMEISSEL SDS-PLUS 250,-PROMAT- (produkt)]	39	0,00179088	0,95122000	0,00178650	408,51515152
Nr. 2	[SPIRALBOHRER 338 HSS 3,5,-PROMAT- (produkt) , SPIRALBOHRER 338 HSS 5,0,-PROMAT- (produkt)] ➞ [SPIRALBOHRER 338 HSS 4,0,-PROMAT- (produkt)]	23	0,00105616	0,82142900	0,00105377	441,68518519
Nr. 3	[VERBANDSCHRANK 47X40X11 L (produkt)] ➞ [FUELLUNG 72 TLG.DIN13157 (produkt)]	28	0,00128576	0,75675700	0,00128262	409,78225806
Nr. 4	[MASCH.GEW.B.371 HSSE B M8,-PROMAT- (produkt) , MASCH.GEW.B.371 HSSE B M5,-PROMAT- (produkt)] ➞ [MASCH.GEW.B.371 HSSE B M6,-PROMAT- (produkt)]	30	0,00137760	0,75000000	0,00137365	348,99038462
Nr. 5	[KNARREN-RINGMAULSCHL.19MM,MIT RINGRATSCHE -PROMAT- (produkt)] ➞ [KNARREN-RINGMAULSCHL.17MM,MIT RINGRATSCHE -PROMAT- (produkt)]	24	0,00110208	0,75000000	0,00110092	946,82608696
Nr. 6	[SCHLUESSELANHAENGER GELB (produkt)] ➞ [SCHLUESSELANHAENGER BLAU (produkt)]	23	0,00105616	0,74193500	0,00105366	421,60858586
Nr. 7	[VERBINDUNGSPLAETTCHEN 10 (produkt)] ➞ [VERBINDUNGSPLAETTCHEN 20 (produkt)]	22	0,00101024	0,73333300	0,00096264	21,22514620
Nr. 8	[SCHLUESSELANHAENGER GRUEN (produkt)] ➞ [SCHLUESSELANHAENGER BLAU (produkt)]	24	0,00110208	0,72727300	0,00109958	439,93939394
Nr. 9	[SPIRALBOHRER HSS S10 16,0 (produkt)] ➞ [SPIRALBOHRER HSS S10 14,0 (produkt)]	27	0,00123984	0,69230800	0,00123761	556,79829545
Nr. 10	[SCHLUESSELANHAENGER ROT (produkt)] ➞ [SCHLUESSELANHAENGER BLAU (produkt)]	32	0,00146944	0,68085100	0,00146694	586,58585859
Nr. 11	[SCHRAUBENDR. PH 1/ 80MM,MEHRKOMP.HEFT -PROMAT- (produkt)] ➞ [SCHRAUBENDR. PH 2/100MM,MEHRKOMP.HEFT -PROMAT- (produkt)]	25	0,00114800	0,67567600	0,00114487	366,86320755
Nr. 12	[SCHLUESSELANHAENGER GRUEN (produkt)] ➞ [SCHLUESSELANHAENGER ROT (produkt)]	22	0,00101024	0,66666700	0,00100922	989,86363636
Nr. 13	[KNARREN-RINGMAULSCHL.10MM,MIT RINGRATSCHE -PROMAT- (produkt)] ➞ [KNARREN-RINGMAULSCHL.13MM,MIT RINGRATSCHE -PROMAT- (produkt)]	27	0,00123984	0,65853700	0,00123507	259,93766578
Nr. 14	[SPIRALBOHRER 338 HSS 5,0,-PROMAT- (produkt) , SPIRALBOHRER 338 HSS 3,0,-PROMAT- (produkt)] ➞ [SPIRALBOHRER 338 HSS 4,0,-PROMAT- (produkt)]	28	0,00128576	0,65116300	0,00128348	564,58888889
Nr. 15	[SPIRALBOHRER 338 HSS 4,0,-PROMAT- (produkt) , SPIRALBOHRER 338 HSS 5,0,-PROMAT- (produkt)] ➞ [SPIRALBOHRER 338 HSS 3,0,-PROMAT- (produkt)]	28	0,00128576	0,65116300	0,00127967	211,06126687
Nr. 16	[GEWINDEFEILEN ZOLL,-PROMAT- (produkt)] ➞ [GEWINDEFEILEN METRISCH,-PROMAT- (produkt)]	26	0,00119392	0,65000000	0,00118999	303,91948470
Nr. 17	[SPIRALBOHRER 338 HSS 3,5,-PROMAT- (produkt) , SPIRALBOHRER 338 HSS 3,0,-PROMAT- (produkt)] ➞ [SPIRALBOHRER 338 HSS 4,0,-PROMAT- (produkt)]	27	0,00123984	0,64285700	0,00123825	777,75000000
Nr. 18	[SPIRALBOHRER 338 HSS 3,5,-PROMAT- (produkt) , SPIRALBOHRER 338 HSS 4,0,-PROMAT- (produkt)] ➞ [SPIRALBOHRER 338 HSS 3,0,-PROMAT- (produkt)]	27	0,00123984	0,64285700	0,00123465	238,91873222
Nr. 19	[KNARREN-RINGMAULSCHL.17MM,MIT RINGRATSCHE -PROMAT- (produkt)] ➞ [KNARREN-RINGMAULSCHL.13MM,MIT RINGRATSCHE -PROMAT- (produkt)]	25	0,00114800	0,64102600	0,00114506	391,10991379
Nr. 20	[SPATMEISSEL SDS-PL.40X200,GERADE - PROMAT - (produkt)] ➞ [FLACHMEISSEL SDS-PLUS 250,-PROMAT- (produkt)]	32	0,00146944	0,62745100	0,00146538	362,00727273

Abbildung 15 Regeln, erste Hierarchiestufe (Artikel) [32]

[31] Eine Übersicht der Ergebnisse der ersten Experimente ist in Tabelle 4 dargestellt.

[32] Lift und Leverage berechnet, keine Filterung angewandt, sortiert nach Confidence.

Danach wurden die Filter auf die Regeln angewendet, die die Elemente aus der gesamten Hierarchie beinhalten. In der Implementierung der Regelinterpretation wurde die Möglichkeiten eingebaut, die Regeln nach Häufigkeit, Confidence, Support, Leverage und Lift zu sortieren, um einen besseren Überblick zu bekommen. Dabei wurden folgende unterschiedliche Variationen der Crosslevelregeln ausprobiert:

Erste, erste und zweite Ebene, erste, zweite und dritte Ebene, zweite und dritte Ebene und nur zweite. Es wurden jeweils unterschiedliche minsup- und minconf-Werte ausprobiert, wobei in jeder einzelnen Berechnung diese Werte für alle Ebenen gleich gesetzt waren. (Als Verbesserung des Ansatzes kann man sich die Anwendung der unterschiedlichen Werte für unterschiedliche Hierarchiestufen denken, nach dem im Kapitel 3.2.3 beschriebenem Ansatz von Han und Fu. Die Implementierung dieses Ansatzes ist im Kapitel 5.11.3.2 beschrieben. Hier sollte aber nur die Filterungsmöglichkeit mit Lift und Leverage untersucht werden. Allgemein kann man sagen, dass die minconf-Werte im Gegensatz zum minsup praktisch sehr wenig Einfluss auf die Anzahl der Regeln ausmachen. Bei den Crosslevel-Regeln, die mit allen 3 Hierarchieebenen berechnet werden, sind beim minsup=1 % keine Elemente aus der untersten Ebene (Artikel) zu finden, weil sie nicht genügend Support erreichen. Bei kleinerem minsup sind diese aber in den Regeln vertreten, wohingegen die Elemente aus den oberen Stufen (Produkt- und Warengruppe) in den genannten Regeln zu sehen sind, weil sie offensichtlich genug Support aufweisen. Es wurden unterschiedliche Einstellungen bei der Filterung ausprobiert und drei verschiedenen Werten sowohl bei min-Leverage als auch bei min-Lift jeweils getestet. Die Werte für minlift- und minleverage wurden im Hinblick auf die tatsächlich vorkommenden Leverage- und Lift-Werten gewählt, damit die Wirkung der Filterung sichtbar wurde. Die weiter unten dargestellte Tabelle 4 soll den Überblick dieser Experimente darstellen.

Die bloßen Zahlen sagen natürlich nicht viel aus. Deswegen werden die Regeln später noch genauer „unter die Lupe genommen“, um zu sehen, ob darunter auch wirklich interessante zu finden sind. Hier sind einige der gefundenen Regeln (ungefiltert), die aus allen drei Hierarchiestufen berechnet wurden (Abbildung 16).

Nr.	Regel	Häufigkeit	Support	Confidence	Leverage	Lift
Nr. 1	[Beschläge / Bauelemente (warengruppe) , Werkzeuge / Baugeräte (warengruppe)] ➞ [Arbeitsschutz / Technische Produkte (warengruppe)]	2.050	0,07127460	0,67656800	0,06356415	9,24389628
Nr. 2	[Werkstatt und Industriebedarf (warengruppe) , Werkzeuge / Baugeräte (warengruppe)] ➞ [Arbeitsschutz / Technische Produkte (warengruppe)]	2.021	0,07026632	0,71036900	0,06255931	9,11719552
Nr. 3	[Beschläge / Bauelemente (warengruppe) , Arbeitsschutz / Technische Produkte (warengruppe)] ➞ [Werkzeuge / Baugeräte (warengruppe)]	2.050	0,07127460	0,91436200	0,05598238	4,66084103
Nr. 4	[Werkstatt und Industriebedarf (warengruppe) , Arbeitsschutz / Technische Produkte (warengruppe)] ➞ [Werkzeuge / Baugeräte (warengruppe)]	2.021	0,07026632	0,90183000	0,04666011	2,97660329
Nr. 5	[Draht-, Eisenwaren und Schweißtechnik (warengruppe)] ➞ [Werkzeuge / Baugeräte (warengruppe)]	1.267	0,04405118	0,64841400	0,04326814	56,25695311
Nr. 6	[Beschläge / Bauelemente (warengruppe) , Werkstatt und Industriebedarf (warengruppe)] ➞ [Werkzeuge / Baugeräte (warengruppe)]	1.328	0,04617203	0,89851200	0,03902108	6,45676499
Nr. 7	[Beschläge / Bauelemente (warengruppe) , Werkstatt und Industriebedarf (warengruppe)] ➞ [Arbeitsschutz / Technische Produkte (warengruppe)]	1.117	0,03883596	0,75575100	0,03670029	18,18437522
Nr. 8	[Beschläge / Bauelemente (warengruppe) , Werkstatt und Industriebedarf (warengruppe)] ➞ [Arbeitsschutz / Technische Produkte (warengruppe) , Werkzeuge / Baugeräte (warengruppe)]	1.073	0,03730617	0,72598100	0,03517049	17,46807038

Nr. 1579	[Transportgeräte-Rollen (produktgruppe) , Befestigungstechnik (warengruppe)] ➞ [Arbeitsschutz / Technische Produkte (warengruppe)]	102	0,00354635	0,76119400	0,00333656	16,90468755
Nr. 1580	[Lufträder (produktgruppe) , Arbeitsschutz / Technische Produkte (warengruppe)] ➞ [Beschläge / Bauelemente (warengruppe)]	97	0,00337251	1,00000000	0,00333572	91,68602320
Nr. 1581	[Sechskant-Stiftschlüssel-Sätze lang mit Kugelkopf (produktgruppe) , Befestigungstechnik (warengruppe)] ➞ [KUGELK.STIFTSCHL.1,5-10MM,IM KUNSTSTOFFCLIP-PROMAT- (produkt)]	96	0,00333774	0,96969700	0,00333571	1.642,56513980
Nr. 1582	[KUGELK.STIFTSCHL.1,5-10MM,IM KUNSTSTOFFCLIP-PROMAT- (produkt) , Befestigungstechnik (warengruppe)] ➞ [Sechskant-Stiftschlüssel-Sätze lang mit Kugelkopf (produktgruppe)]	96	0,00333774	1,00000000	0,00333571	1.642,56513980
Nr. 1583	[KUGELK.STIFTSCHL.1,5-10MM,IM KUNSTSTOFFCLIP-PROMAT- (produkt) , Werkstatt und Industriebedarf (warengruppe)] ➞ [Beschläge / Bauelemente (warengruppe)]	99	0,00344204	0,60365900	0,00333545	32,29046744
Nr. 1584	[KUGELK.STIFTSCHL.1,5-10MM,IM KUNSTSTOFFCLIP-PROMAT- (produkt) , Sechskant-Stiftschlüssel-Sätze lang mit Kugelkopf (produktgruppe) , Werkstatt und Industriebedarf (warengruppe)] ➞ [Beschläge / Bauelemente (warengruppe)]	99	0,00344204	0,60365900	0,00333545	32,29046744
Nr. 1585	[Schlagzahlen-Sätze (produktgruppe)] ➞ [Arbeitsschutz / Technische Produkte (warengruppe)]	99	0,00344204	0,60736200	0,00333543	32,28570781
Nr. 1586	[Kabelmesser (produktgruppe)] ➞ [KABEL-KLAPPMESSER-PROMAT-,1-TEILIG,HOLZHEFT (produkt)]	96	0,00333774	0,68571400	0,00333534	1.391,70967742
Nr. 1587	[Holzschrauben (produktgruppe) , Draht-, Eisenwaren und Schweißtechnik (warengruppe) , Beschläge / Bauelemente (warengruppe)] ➞ [Befestigungstechnik (warengruppe) , Werkstatt und Industriebedarf (warengruppe) , Werkzeuge / Baugeräte (warengruppe)]	98	0,00340727	0,64052300	0,00333532	47,35678763

Abbildung 16 Beispiele der Regel mit Elementen aus allen Hierarchiestufen

Benutzte Stufen der Hierarchie bei Erzeugung	Min-sup, %	Min-conf, %	Anzahl der Regel	Minimale und maximale Leverage (tatsächlich)	Minleverage als Filter angewandt	Nach Filterung mit minleverage verbliebene Anzahl der Regel Abs. (ca. %)	Minimaler und Maximaler Lift (tatsächlich)	Minlift als Filter angewandt	Nach Filterung mit minlift verbliebene Anzahl der Regel Abs. (ca. %)
1, 2, 3	1	40	204	-0,0517826 0,1568644	0,01	154 (75%)	0,175 307,413	150	3 (1,5%)
					0,015	85 (41%)		100	6 (3%)
					0,02	47 (23%)		50	19 (10%)
1, 2, 3	1	50	173	-0,0517825 0,1568644	0,01	126 (72%)	0,175 307,413	150	1 (0,6%)
					0,015	65 (37%)		100	2 (1,2%)
					0,02	37 (22%)		50	13 (8%)
1, 2, 3	0,1	40	36914	-0,022229 0,1568644	0,01	185 (0,5%)	0,0448191 9.439,1343	100	9439 (26%)
					0,015	91 (2,5%)		1000	955 (2,6%)
					0,02	50 (1,4%)		4500	9 (0,024 %)
1,2	0,1	40	1340	0,00092916 0,01480993	0,005	13 (1%)	10,25383 11.602,852	100	1287 (96%)
					0,01	2 (0,15%)		1000	205 (15%)
					0,015	0 (0%)		3000	13 (1%)
2	0,1	40	197	0,00093619 0,01057624	0,0015	55 (28%)	4,2213 3.799,1677	100	164 (83%)
					0,002	35 (10%)		1000	28 (14%)
					0,005	3 (1,5%)		2000	7 (3,6%)
1	0,1	40	105	0,001612 0,001284	0,0013	45 (42%)	62,1453 1.377,860	100	103 (98%)
					0,0014	26 (25%)		500	44 (42%)
					0,0015	24 (23%)		1000	5 (5%)

Tabelle 4 Anwendung der Filterung[33]

Wie man auf der Abbildung 16 sieht, kommen bei den obersten Regeln (die Regeln sind in diesem Fall nach Leverage sortiert) ausschließlich Warengruppen vor (Ebene 3), etwas weiter unten gibt es aber auch Elemente aus den Ebenen 2 und 1. Manche Regeln sind Wiederholungen der anderen, mit der Ausnahme, dass sie ein Element mehr haben. Solche „uninteressanten“ Regeln könnte man ausfiltern. Man sieht aber auch Regeln, die fast der

[33] Anwendung der Filterung mit unterschiedlichen Minlift- und Minleverage- Parametern und bei Verwendung von unterschiedlichen Hierarchieebenen

Eigenschaft der robusten Regeln entsprechen: links höhere Hierarchiestufe (allgemeiner, z. B. Produktgruppe), rechts niedrigste (Artikel, also detaillierter), wobei bei den robusten Regeln links die höchste Hierarchiestufe stehen sollte (Warengruppe), was man so mit bloßem Auge nicht ohne weiteres aus der großen Regelmenge erkennen kann. Deswegen sollten solche Regeln, falls es sie tatsächlich gibt, später noch gezielt aus der Masse hervorgehoben werden (Experimente dazu s. im Kapitel 5.11.2).

Eine Betrachtung der Tabellenwerte (Tabelle 4) und die Durchführung der bis jetzt beschriebenen und anderen Experimente zeigen, dass:

1. Bei der Benutzung höherer Stufen mehr Regeln entdeckt werden.
2. Ein spezifischer Minsup-Wert für jede Taxonomiestufe notwendig ist. Z. B. wären für die unterste Stufe ein Wert zwischen 0,05 und 0,1%, für die mittlere Stufe ein Wert zwischen 0,1 und 1% und für die oberste Stufe ein Wert zwischen 0,5 und 2% passend.
3. Eine Filterung mit Minleverage bzw. Minlift oder ihrer Kombination eine eventuell unübersichtliche Anzahl der Regeln durchaus verkleinert.

Ein Zwischen-**Fazit**:

Nachdem die ersten Experimente und die Tests mit verschiedenen Kombinationen der Hierarchien sowie Filtern der Regeln durchgeführt waren, war klar, dass, wie auch vorher erwartet, die Gewinnung von Regeln mit Ausnutzung der Taxonomie einige Vorteile gegenüber den „einfachen“ Regelentdeckung ohne Hierarchien bietet:

Auf der einen Seite kann man durch Multilevel-Regeln die Zusammenhänge zwischen Elementen der gleichen Stufe etwas abstrakter anschauen, ohne ganz detailliert jeden einzelnen Artikel zu betrachten. Die Parametereinstellungen für die Regelentdeckung, insbesondere der Minsup-Wert, können höher sein als auf der untersten Ebene, wodurch mehr Regeln entstehen. Auf der anderen Seite bieten die Crosslevelregeln eine Möglichkeit, die Zusammenhänge innerhalb der Taxonomie zu untersuchen. Die aus den unterschiedlichen Stufen stammenden „Teilnehmer“ der Crosslevel-Regeln führen zu der Idee, eventuell die anderen Taxonomien in die Berechnungen zu involvieren. Diese Idee wird im Weiteren auch untersucht. Gleichzeitig gibt es aber auch Nachteile: durch höhere Supportwerte der Itemsets auf den oberen Stufen entstehen sehr viele Regeln. Dieses Problem kann man aber durch eine Filterung lösen.

5.4. Kundengruppen bilden

Wie oben beschrieben, sollten zusätzlich zu der Artikeltaxonomie die Kundeninformationen für die Regelentdeckung herangezogen werden (s. Kapitel 4.4).

Als erstes sollten die Kunden in Gruppen eingeteilt werden. Da die Kunden keine Endkunden, sondern Firmen sind, gibt es keine Möglichkeit bzw. macht es keinen Sinn, sie nach Namen, Alter oder anderen persönlichen Daten aufzuteilen. Vielmehr könnten dazu z. B. die geografischen Merkmale der Kunden passen. Es wurde die Postleitzahl als solches Merkmal gewählt. Für die Gruppenbildung wurden die ersten 2 Ziffern der PLZ benutzt. Allerdings funktionierte diese Aufteilung nur bei den deutschen Kunden. Die ausländischen Kunden

haben (in der vorliegenden Datenbank) ihr Landeskürzel als Präfix vor den Postleitzahlen. Deshalb wurden diese Kunden nicht nach der PLZ, sondern nach dem Landeskürzel gruppiert.

Bei der Gruppenbildung wurden zunächst alle Kunden, die in der Datenbank abgelegt sind, benutzt. Später, bei der *f*-Metrik-Berechnung, werden aber nur die Kunden berücksichtigt, die tatsächlich eingekauft haben.

Die Gruppierung wurde in die Implementierung des Datenvorverarbeitungsprogramms integriert. Es sind dabei 415 Gruppen entstanden. Diese Informationen wurden in die Transaktionsdateien mit verschiedenen Artikelhierarchiekombinationen eingefügt und dann wurde die Regelentdeckung und Interpretation durchgeführt. Dabei sind neue Regeln gefunden worden, die die Kundeninformation (PLZ-Gruppe) beinhalten. Abbildung 17 stellt ist ein Fragment der Regelmenge dar, die aus den 2 unteren Artikelhierarchiestufen und den Kunden-PLZ-Gruppen gewonnen wurde, die unter anderem die neuen Regeln mit den Kunden-PLZ-Gruppen enthalten. Die neuen Regeln sind rot markiert.

Nr. 694	[Steckschlüssel-Sätze 1/2" (produktgruppe) , PLZ-SK 03852] ➡ [STECKSCHL.SA.55/32 MM PR (produkt) , Steckschlüssel-Sätze 1/4"-1/2" (produktgruppe)]	37	0,00128642	0,51388900	0,00128221	305,45177956
Nr. 695	[TOPFBUERSTE M14 65/0,5 MM,-PROMAT- (produkt) , Spiralbohrer (produktgruppe)] ➡ [Topfbürsten (produktgruppe)]	37	0,00128642	1,00000000	0,00128137	254,83572797
Nr. 696	[FEUERLOESCHER KFZ 2 KG,F 2 G (produkt)] ➡ [PLZ-S 73121]	37	0,00128642	0,68518500	0,00128124	248,17957090
Nr. 697	[Storz-Armaturen (produktgruppe)] ➡ [PLZ-D 96***]	37	0,00128642	0,54411800	0,00128099	237,06705280
Nr. 698	[HANDWASCHP.SANDLOS 10 L (produkt) , Spiralbohrer (produktgruppe)] ➡ [Handwaschpaste (produktgruppe)]	37	0,00128642	1,00000000	0,00127466	109,38369822
Nr. 699	[SPIRALBOHRER 338 HSS 3,5,-PROMAT- (produkt) , Maschinengewindebohrer (produktgruppe)] ➡ [Spiralbohrer (produktgruppe)]	37	0,00128642	1,00000000	0,00127250	92,42608998
Nr. 700	[METALLSAEGEBOGEN -PROMAT- (produkt) , Schraubendreher-Sätze (produktgruppe)] ➡ [Metall-Sägebogen (produktgruppe)]	36	0,00125165	1,00000000	0,00125063	1.231,19143876
Nr. 701	[SPIRALBOHRER 338 HSS 5,0,-PROMAT- (produkt) , SPIRALBOHRER 338 HSS 3,0,-PROMAT- (produkt)] ➡ [SPIRALBOHRER 338 HSS 4,0,-PROMAT- (produkt)]	36	0,00125165	0,65454500	0,00125056	1.151,75973304
Nr. 702	[STECKSCHL.SA.1/4+1/2 ZOLL,4-34 MM, 65TEILIG,-PROXXON- (produkt) , Steckschlüssel-Sätze 1/2" (produktgruppe)] ➡ [Steckschlüssel-Sätze 1/4"-1/2" (produktgruppe)]	36	0,00125165	1,00000000	0,00125056	1.151,75973304
Nr. 703	[PLZ-I 16138, Atemschutz (produktgruppe)] ➡ [Sicherheitsschnürstiefel (produktgruppe)]	36	0,00125165	0,48000000	0,00125056	1.150,48000000
Nr. 704	[Schrauben metrisches Gewinde (produktgruppe) , PLZ-D 93***] ➡ [Scheiben (produktgruppe)]	36	0,00125165	0,40909100	0,00125053	1.115,76724138
Nr. 705	[PLZ-S 81136, Kontaktkleber (produktgruppe)] ➡ [PATTEX-KLEBER 4,5 KG,CLASSIC (produkt)]	36	0,00125165	0,92307700	0,00125049	1.081,95611285
Nr. 706	[PATTEX-KLEBER 4,5 KG,CLASSIC (produkt)] ➡ [PLZ-S 81136, Kontaktkleber (produktgruppe)]	36	0,00125165	0,63157900	0,00125049	1.081,95611285
Nr. 707	[PLZ-I 16138, Sicherheitshalbschuhe (produktgruppe)] ➡ [Sicherheitsschnürstiefel (produktgruppe)]	36	0,00125165	0,43373500	0,00125045	1.045,89090909
Nr. 708	[Steckschlüssel-Sätze 1/4" (produktgruppe) , Steckschlüssel-Sätze 1/4"-1/2" (produktgruppe) , PLZ-SK 03852] ➡ [STECKSCHL.SA.MEC1/4-13 PR (produkt)]	36	0,00125165	0,80000000	0,00125038	982,38330171
Nr. 709	[SPIRALBOHRER 338 HSS 5,0,-PROMAT- (produkt) , SPIRALBOHRER 338 HSS 3,0,-PROMAT- (produkt)] ➡ [SPIRALBOHRER 338 HSS 4,0,-PROMAT- (produkt) , Spiralbohrer (produktgruppe)]	36	0,00125165	0,65454500	0,00125038	982,38330171
Nr. 710	[PATTEX-KLEBER 4,5 KG,CLASSIC (produkt)] ➡ [PLZ-S 81136]	36	0,00125165	0,63157900	0,00125028	915,50132626
Nr. 711	[Storchschnabelzangen (produktgruppe) , Schraubendreher-Sätze (produktgruppe)] ➡ [Seitenschneider (produktgruppe)]	36	0,00125165	0,43902400	0,00125027	908,27368421

Abbildung 17 Regeln, die auch Informationen aus Kundenhierarchie beinhalten

Mit den Regeln, die mit Kundeninformationen berechnet wurden, kann man das Kaufverhalten verschiedener Kunden analysieren. So kann man z.B. die erste Regel in der Abbildung 17 so interpretieren, dass einige Artikel aus der Produktgruppe „Steckschlüssel-Sätze 1/2´´“ besonders oft gekauft werden und die Kunden, die diese Artikel kaufen der Kundengruppe PLZ-SK03852 (also die Kunden aus Slowakei) angehören. So bekommen die Regeln eine geographisch-ökonomische Bedeutung (s. Kapitel 5.11.3.4).

Im weiteren Verlauf der Arbeit wird die *f*-Metrik berechnet, die im Kapitel 3.2.5 beschrieben wurde. Diese Metrik soll der Bestimmung der optimalen Kombination der Artikel- und Kunden-Hierarchien dienen. Erst dann kann die Regelentdeckung auf dieser Kombination sinnvoll durchgeführt werden. Wie bereits oben gesagt, sind die Metrik und der von Psaila und Lanzi vorgeschlagene Schwellenwert für sie (0,7) nicht zwangsläufig korrekt oder für alle Daten verwendbar. Aus Anwendersicht ist die optimale Kombination die Kombination aus zweiter Stufe der Artikelhierarchie (Produktgruppe) und der zweiter Stufe der Kundenhierarchie (PLZ-Gruppe der Kunden). Es wird sich zeigen, ob es auch nach Berechnung und Anwendung der Metrik tatsächlich die optimale Kombination ist.

5.5. Berechnung der *f*-Metrik

Kurz zur Erinnerung: die *f*-Metrik soll (nach dem Ansatz von Psaila und Lanzi) die möglichen Generalisierungen der gegebenen Hierarchien auf ihre Güte überprüfen. Liegt der Wert der für die jeweilige Gruppierung berechneten Metrik unter dem vom Benutzer vorgegebenen Schwellenwert, ist das Metapattern „gut“ und man kann diese Gruppierung und die gewählte Hierarchiestufe für die weitere Berechnung der Regeln benutzen.

Für diesen Zweck wurden die Transaktionen in die Datenbank importiert, damit die Gruppierung anhand der SQL-Abfragen leichter gestalten werden kann. Dafür wurden drei Tabellen angelegt, die jeweils eine Relation zwischen einer Transaktion und einer der drei Artikelhierarchiestufen abbilden. Zusätzlich wurde eine Tabelle angelegt, die alle Transaktionen enthält und deren Spalten unter anderem Transaktionsnummer, Kundennummer, Kunden-PLZ-Gruppennummer und andere für die spätere Berechnungen relevanten Informationen sind. Außerdem wurden die Kundengruppierungsinformationen in eine separate Tabelle eingefügt. Somit wurde die Basis vorbereitet, auf der verschiedene Kombinationen aus verschiedenen Kunden- und Artikelhierarchiestufen gebildet werden konnten. Als Gruppierungsattribut wurde die Kundenhierarchie verwendet, und als Mined-Attribut die Artikelhierarchie. Mit diesen Informationen ließ sich ein Verband aus Kombinationen bilden, der auf der Abbildung 18 dargestellt ist.

Die Berechnung der *f*-Metrik wurde für alle diese Kombinationen (Metapatterns) durchgeführt. Das Ergebnis war die Feststellung, dass die *f*-Metrik bei allen Metapatterns unter der von Psaila und Lanzi vorgeschlagenen Grenze 0,7 liegt. D. h., alle Metapatterns sind „gut“ und können weiter für die Regelentdeckung verwendet werden. Wie bereits gesagt, wurde die Kundenhierarchie als Gruppierungsattribut verwendet. Das bedeutet, dass die Transaktionen nicht mehr in der ursprünglichen Form betrachtet, sondern immer nach Kundennummer bzw. Kunden-PLZ-Gruppennummer gruppiert wurden (vgl. Kapitel 3.2.5.6). Die Artikel bzw. Produktgruppen und Warengruppen aus allen reellen Transaktionen eines jeden Kunden bzw. einer jeden Kundengruppe wurden jeweils zu einer Transaktion disjunkt zusammengefasst: kamen z. B. bestimmte Artikel in vielen Transaktionen eines Kunden vor, so wurden sie in solch eine gruppierte Transaktion nur einmal übernommen.

Die Berechnung wurde auf der Datenmenge von ca. 28000 reellen Transaktionen durchgeführt. Es wurden nur die tatsächlich einkaufenden Kunden für die Kundengruppenbildung benutzt, im Gegensatz zu der Kundenbildung von oben, wo alle vorhandenen potentiellen und tatsächlich einkaufenden Kunden mitberücksichtigt wurden. Es hat sich herausgestellt, dass von rund 1760 potentiellen Kunden nur 213 tatsächlich aktiv waren (zumindest auf den betrachteten Transaktionen). Dadurch ist die Anzahl der Kundengruppen, die nach dem gleichen Verfahren, wie oben, also mit der Benutzung der PLZ gebildet wurden, kleiner geworden. Es sind nur 52 Kunden-PLZ-Gruppen gebildet worden.

In der unteren Tabelle (Tabelle 5) sind die berechneten *f*-Metrik-Werte dargestellt. Wie man sieht, sind die Werte der Metrik bei der Verwendung der untersten Artikelhierarchiestufen sehr niedrig und liegen sogar unter 0,1. Das bedeutet, dass die Gesamtzahl der zur Auswahl stehenden Items sehr groß im Verhältnis zu Gruppentransaktionslängen und Gruppenzahl ist. Dies deutet auf eine niedrige Generalisierung hin.

Bei der Verwendung der zweiten und dritten Artikelhierarchiestufen sind die *f*-Metrik-Werte dagegen höher. Der größte Wert der *f*-Metrik von ca. 0,45 wird bei den nach der Kunden-PLZ

gruppierten Transaktionen und den Warengruppen als Transaktionselementen erzielt (logischerweise wegen Monotonie von *f*-Metrik). Würde man in der Praxis aus solchen Daten Regeln gewinnen, würden diese (aus Anwendersicht) zu allgemein sein. Denn die Warengruppen sind sehr umfangreich. Für praktische Zwecke würde man in der Artikelhierarchie eher bei der zweiten Ebene stoppen, mit dem Wert von ca. 0,22.

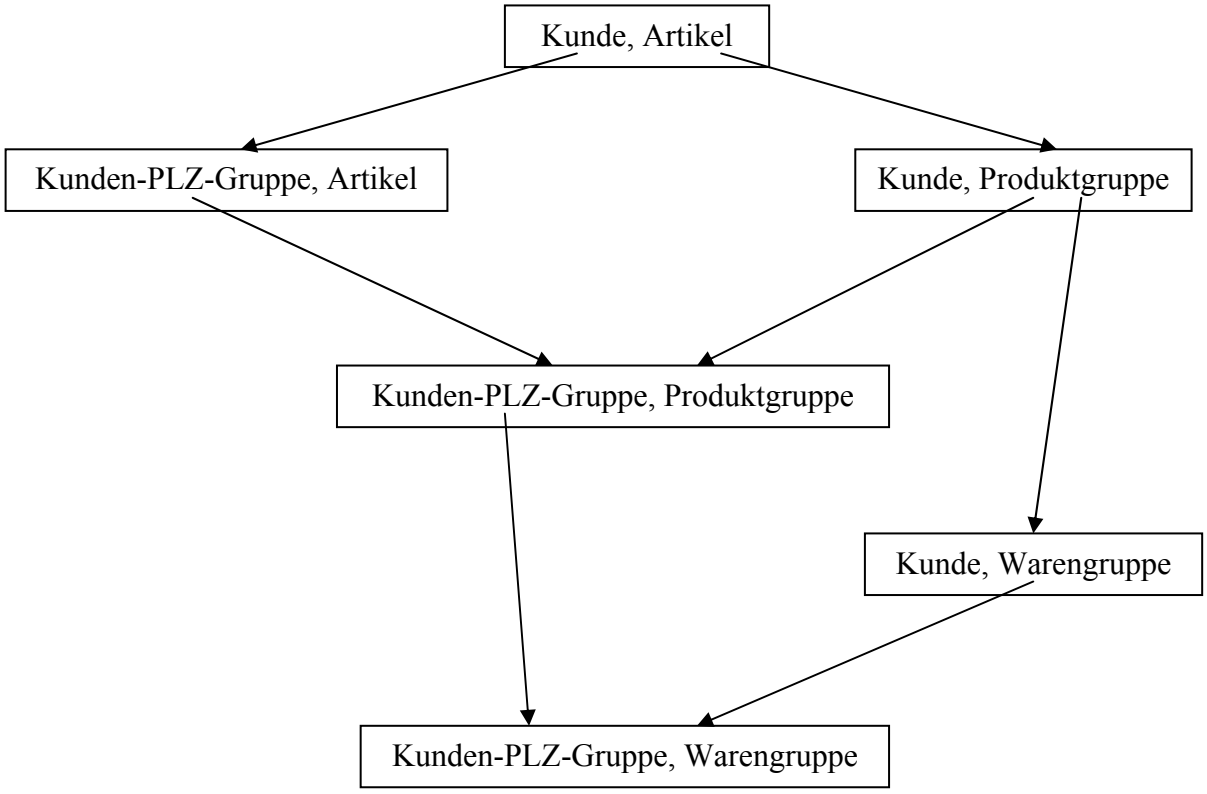


Abbildung 18 Verband aus Artikel- und Kundenhierarchie-Attributen

Gruppierung nach /Mined-Attribut der Artikelhierarchie-stufe	$\lvert V_g \rvert$ **Anzahl der Gruppen**	$\lvert V_m \rvert$ **Gesamtanzahl der Items in allen Gruppen, insgesamt disjunkt**	$\sum_{g_i} z_i$ **Gesamtanzahl der von allen Gruppen gekauften Items, in jeder Gruppe disjunkt**	$a_i = \sum_{g_i} z_i / \lvert V_m \rvert$ ϕ **Gruppenlänge (Transaktionslänge)**	$\bar{f}_{g,m} = (\sum_{g_i} z_i)/(\lvert V_g \rvert \times \lvert V_m \rvert)$ ***f*-Metrik**
Kunde /Artikel	**213**	**17419**	**94288**	**442,7**	**0,02541**
Kunde /Produktgruppe	**213**	**2036**	**34146**	**160,3**	**0,07874**
Kunde /Warengruppe	**213**	**25**	**1359**	**6,38**	**0,25521**

Kunden-PLZ-Gruppe /Artikel	**52**	**17419**	**80935**	**1556,4**	**0,08935**
Kunden-PLZ-Grupppe /Produktguppe	**52**	**2036**	**23098**	**444,2**	**0,21817**
Kunden-PLZ-Grupppe /Warengruppe	**52**	**25**	**584**	**11,2**	**0,44923**

Tabelle 5 *f*-Metrik

Eine Bemerkung über die Gruppierung sollte an diese Stelle noch gemacht werden:

In den früheren Experimenten (z.B. bei den ersten Experimenten in Kapiteln 5.2 und 5.3) waren die Transaktionen nicht nach dem Kundenattribut gruppiert. Trotzdem waren sie ja quasi automatisch nach der Transaktionsnummer gruppiert. Jede Gruppe enthielt dabei ein einziges Element, nämlich eine reelle Transaktion. Da alle Transaktionen unterschiedliche Nummern haben, ist die Gruppenanzahl genau so groß wie die Anzahl der Transaktionen.

Die *f*-Metrik ließ sich auch für diese implizite Gruppierung berechnen und betrug 0,0000331. Durch die sehr große Gruppenanzahl ist so ein kleiner Wert zustandegekommen.

Ein **Zwischenfazit**:

Im Allgemeinen lassen die berechneten Werte der *f*-Metrik sagen, dass ihre Verwendung nicht immer ein passendes Kriterium für die Bewertung einer Generalisierung ist. Die berechneten Werte, die größtenteils weit unter dem von Psaila und Lanzi vorgeschlagenen Grenzwert lagen, deuten darauf hin, dass eine starke Abweichung der vorliegenden Daten von den Daten, die die Autoren für ihre Forschungen benutzt haben, vorliegt. Der Nutzen des Ansatzes der Metapatterns besteht aber in der methodischen Vorgehensweise und nicht in einem bestimmten allgemein gültigen Grenzwert der *f*-Metrik. Der Ansatz selbst bietet eine interessante Möglichkeit, die Bestelldaten neu zu ordnen und auf eine andere Art in neue Transaktionen zu gruppieren. Diese methodische Vorgehensweise wird bei der Integration in das vorliegende Informationssystem weiter verwendet (s. Kapitel 5.6).

5.6. Berechnung der Regeln unter Verwendung der Gruppierung

Ein gesonderter und interessanter und problematischer Fall tritt ein, wenn man die Regeln mit der Benutzung von Gruppierungen nach Attributen der Kundenhierarchie erzeugt (s. dazu die Abbildung 19 im späteren Kapitel 5.11).

Es soll bei der Kundenhierarchie (*2[34]) die Gruppierung nach „Kunden“ oder nach „Kundengruppen“ eingestellt werden. Zunächst wurde der Fall der Gruppierung nach Kundengruppen analysiert. Wie werden die Transaktionen dabei erzeugt? Alle reellen Transaktionen aller Kunden aus der gleichen Gruppe „x“ werden als eine einzelne Transaktion aufgefasst, wobei die gleichen Elemente natürlich nicht mehrfach übernommen werden. D. h., alle Artikel bzw. Produkt- oder Warengruppen aus beispielsweise 500 reellen Transaktionen aller Kunden aus der gleichen Kundengruppe werden zu einer Transaktion. Die

[34] „*2“ Bezieht sich auf die Markierungen auf der Abbildung 19 im späteren Kapitel 5.11

mittlere Transaktionslänge wächst bei der gegebenen Auswahl an Artikeln (ca. 25000) enorm. Gab es ca. 28000 reelle Transaktionen, die im Durchschnitt eine Transaktionslänge von 10 Artikeln hatten, so gibt es dann nur 52 erzeugten Kundengruppen-Transaktionen (genau so viele wie Kundengruppen), die durchschnittlich 1500 Elemente enthalten, wobei es auch Transaktionen mit mehreren Tausenden von Elementen gibt. Da es insgesamt aber nur wenige Transaktionen sind, haben alle Itemsets einen sehr hohen Support und die Apriori-Laufzeit steigt sehr steil an. D. h., dass bei der Auswahl der Gruppierung nach Kundengruppen eine Berechnung praktisch (zumindest auf dem zur Verfügung stehenden Rechner) nicht möglich ist. Und, rein hypothetisch, falls sie möglich wäre, gäbe es sehr viele häufige Itemsets und dementsprechend sehr viele Regeln, die auch sehr lang wären und dadurch keine praktische Bedeutung hätten. Denn kein Mensch kann eine Regel interpretieren, die so viele Elemente beinhaltet.

Wählt man in der Artikelhierarchie eine höhere Stufe, damit die Transaktionen etwas kürzer werden, oder mit anderen Worten, navigiert man in dem oben abgebildeten Verband der Taxonomien (Abbildung 18) zu den Elementen mit größeren *f*-Metrik (s. obere Tabelle 5), so verbessert sich die Berechnungsmöglichkeit, und die durchschnittliche Transaktionslänge fällt. So sind die Transaktionen bei der Gruppierung nach Kundengruppen und Auswahl der Produktgruppe in der Artikeltaxonomie „nur noch ca. 442“ und bei der Auswahl der Warengruppe in der Artikeltaxonomie rund 11 Elemente lang. Bei der letzten Kombination betrug auch die *f*-Metrik ca. das Fünffache im Vergleich zu ihrem Wert aus dem ersten, problematischen Fall. Jetzt konnten die Berechnungen zwar durchgeführt werden, aber nur unter der Bedingung, dass der Minsup-Wert sehr hoch eingestellt wird (mind. 20 %), so dass das noch in einer akzeptablen Zeit passieren kann.

Ferner hatte wurden auch alle Kombinationen der Artikelhierarchiestufen und der Gruppierung nach Kunden (anstatt Gruppierung nach Kundengruppen wie vorher) ausprobiert. Bei den Einstellungen: Minsup=25%, Minconf=60%, Gruppierung Transaktionen nach Kunden, und Produktgruppe als Stufe der Artikelhierarchie sind ohne Filtern 1366 Regeln gefunden worden.

Fazit:

Zunächst schien die Berechnung der Regeln bei der Kombination, bei der die Gruppierung nach Kundengruppen geschehen soll überhaupt nicht Verwendbar zu sein, weil es zum einen sehr großen Rechenaufwand bedeutet hätte, und zum anderen sehr viele und nicht interpretierbare Regeln ergeben würde. Das widersprach einer früheren Behauptung (s. Kapitel 4.5), dass nach der *f*-Metrik-Berechnung die bessere Wahl der Gruppierungen die Kombination aus der Produktgruppe bei der Artikelhierarchie und den Kundengruppen bei der Kundenhierarchie wäre. Danach kam es aber trotzdem zu einem Ergebnis, dass bei den höheren Stufen der Artikelhierarchie oder / und bei der Gruppierung nach Kunden eine Durchführung der Berechnungen in einer durchaus akzeptablen Zeit möglich ist. Das Problem der sehr langen Transaktionen bei ihrer gleichzeitig kleinen Anzahl löste sich dabei. Auf diese Art konnte man durch diese Gruppierung der Elemente wirklich neue Transaktionen bilden und sie dann als Transaktionsmenge für die Regelentdeckung weiter anwenden. Hierbei spielte die *f*-Metrik eine nicht zu vernachlässigbare Rolle: ein höherer Wert der Metrik bedeutete bessere Ergebnisse. Da aber keine Werte von *f*-Metrik vorlagen, die über dem von

Psaila und Lanzi vorgeschlagenen Grenzwert von 0,7 lagen, bedeutete es, dass die Einstellungen mit möglichst hohem Wert der *f*-Metrik gewählt werden können.

Zum Abschluss des Kapitels folgt hier noch eine Tabelle (Tabelle 6) mit einer Übersicht über die durchschnittliche Länge der Transaktionen in Abhängigkeit von den verwendeten Gruppierungen in Kunden- und Artikelhierarchien.

	Artikel	**Produktgruppe**	**Warengruppe**
Transaktion	**6,33**	**4,45**	**1,56**
Kunde	**442,7**	**160,3**	**6,38**
Kundengruppe	**1556,4**	**444,2**	**11,2**

Tabelle 6 Durchschnittliche Transaktionslänge in Abh. von Gruppierung

5.7. Berechnung der Auslastung von Produktgruppen

Wie im Kapitel 4.2 geplant, sollte nun berechnet werden, welche Produktgruppen inwieweit genutzt werden und mit den gekauften Artikeln ausgelastet sind. Dies wird die Entscheidung bei der Generalisierung ermöglichen, ob die Produktgruppen als Stellvertreter für Artikel agieren können. Nach der Berechnung kann man sich einen Grenzwert überlegen, ab dem generalisiert werden darf. Z. B., wenn mindestens 40% der Artikel einer Produktgruppe gekauft werden, kann man die Transaktionen vor der Entdeckung der generalisierten Regeln mit dieser Gruppe erweitern, ansonsten nicht.

Diese Berechnung wurde durchgeführt und die jeweiligen Auslastungen der Produktgruppen wurden als Attribute in die Produktgruppentabelle in der Datenbank eingetragen, damit man auf sie später schnell zurückgreifen kann. Bei dieser Berechnung wurden wieder ca. 28000 Transaktionen benutzt. Es wurde festgestellt, dass aus insgesamt 3023 vorhandenen Produktgruppen nur ca. 2/3 überhaupt gekauft werden, nämlich nur 2036. Diese Gruppen wurden unterschiedlich stark, jedoch die meisten über 40% ausgelastet (ca. 90% aller überhaupt benutzten Gruppen).

Nachfolgende Tabelle (Tabelle 7) gibt einen Überblick über die Verteilung der Produktgruppenanzahl im Bezug auf ihre Auslastungen[35].

Wie man sieht, ist ca. die Hälfte der Produktgruppen voll ausgelastet, d. h., dass alle Artikel aus diesen Produktgruppen mindestens einmal gekauft wurden. Die Auslastungsgrenze von 40% scheint ein gutes Kriterium für spätere Experimente zu sein.

% Mindestauslastung (MA)	**10**	**20**	**30**	**40**	**50**	**60**	**70**	**80**	**90**	**100**
Produktgruppenanzahl, die MA erreichen	**2017**	**1977**	**1905**	**1833**	**1774**	**1574**	**1402**	**1275**	**1147**	**1030**
% Anteil von allen Benutzten Prod. Grup.	**99,1**	**97,1**	**93,6**	**90**	**87,1**	**76,3**	**68,9**	**62,6**	**56,3**	**50,1**

Tabelle 7 Auslastungen der Produktgruppen

[35] Bemerkung: Die gleiche Auslastungsberechnung könnte man auch mit den Warengruppen machen. Darauf wurde aber aus zeitlichen Gründen verzichtet.

5.8. Weitere Untersuchung einiger Regeln und Ausfilterung der redundanten

Eine bestimmte Art von Regeln sollte zusätzlich überprüft werden. Die spezielle Eigenschaft dieser Regeln besteht darin, dass sie im Head Elemente beinhalten, die sozusagen aus dem gleichen „Taxonomie-Zweig“ stammen. Nachfolgend wird erklärt, was gemeint wird. Angenommen, ein Bohrer B gehört zu der Produktgruppe Spiralbohrer SB und die Produktgruppe Spiralbohrer gehört zu der Warengruppe Werkzeuge W an. Es seien einige Regeln gefunden worden, die eine der folgenden Formen haben:

B, SB, .. $\Rightarrow$ …

B, W,.. $\Rightarrow$ …

SB, W,.. $\Rightarrow$ …

B, SB, W,… $\Rightarrow$ …

D. h., im Body der Regel stehen gleichzeitig Elemente, die sowohl einer direkten oder indirekten Unterstufe eines Elementes gehören, sowie das Element selbst.

Beispiel: Es sei eine Regel „Bohrer 8mm (Artikel), Spiralbohrer (Produktgruppe),..$\Rightarrow$…“ gegeben und der „Bohrer 8mm“ gehört gerade der Produktgruppe „Spiralbohrer“ an. Jetzt kann es vorkommen, dass die Regel nur dadurch entstanden ist, dass die ursprünglichen Transaktionen mit Elementen aus den höheren Stufen der Taxonomie erweitert wurden, um die generalisierten Regeln der Cross-Level Art zu bekommen. Genauer gesagt, die höheren Stufen sind in die Berechnung involviert worden, ohne einen anderen Sinn der Regel auszumachen. Ist das der Fall, so könnte man diese wieder aus den Transaktionen entfernen, ohne dabei Informationen zu verlieren.

Wann kann die Situation auftreten, dass ein Element aus der höheren Stufe neben dem Element aus der unteren Stufe im Head der gleichen Regel auftaucht und gar nichts zur Bedeutung der Regel beiträgt? Nur dann, wenn der Support des Elementes aus der höheren Stufe genau den gleichen Support aufweist wie das Element aus der unteren Stufe, d. h., wenn keine anderen Elemente aus der unteren Stufe in den Transaktionen vorkamen und den Support ihrer „Verallgemeinerungen“ vergrößert haben. Dadurch hätten die Elemente „B“ und „SB“ in dem Itemset „B, SB“ aus dem oberen Beispiel den gleichen Support untereinander (und auch den gleichen wie der Itemset „B, SB“). Der Bohrer „B“ wäre dann ein „Einzelakteur“ in seiner Produktgruppe und man könnte die Produktgruppe aus der Regel einfach entfernen.

Genau diese Eigenschaft sollte überprüft und ggf. die Regeln, bei denen sie erfüllt ist, herausgefunden werden. Sollte aber das Element aus der höheren Stufe einen höheren Support haben (einen kleineren kann es ja nicht haben), dann haben auch noch andere Elemente der Unterstufe seinen Support erhöht, die das Element als Oberstufe haben, wobei sie selbst aber nicht genügenden Support erreicht haben, um noch im häufigen Itemset bzw. in der Regel zu erscheinen. Alle erzeugten Cross-Level-Regeln, die dem Kriterium entsprechen, das oben als „gleicher Taxonomie-Zweig“ bezeichnet wurde, wurden diesem Test unterzogen. Erstaunlicherweise wurde festgestellt, dass es keine Regeln gibt, bei denen der Support eines Unterelements gleich dem Support eines Oberelements ist. D. h., es können keine Oberelemente aus irgendeiner Regel entfernt werden, ohne die Semantik der Regel zu verändern.

Hier muss noch gesagt werden, dass eine andere Art von „redundanten“ Regeln jedoch oft vorkommt. Allgemein formuliert, sind das solche Regeln, die nur die Taxonomie wiedergeben und im Body die Elemente aus der direkten oder indirekten Unterstufe der Elemente aus dem Head beinhalten (s. dazu Kapitel 3.2.2.2).

Genauer genommen sind es bei den gegebenen Daten die Regeln mit folgenden Eigenschaften:

1. Im Body nur Artikel, die zu der gleichen Produktgruppe gehören, die im Head steht
2. Im Body nur Produktgruppen, die zu der gleichen Warengruppe gehören, die im Head steht
3. Im Body nur Artikel, die zu der gleichen Produktgruppe gehören, im Head die Warengruppe, zu der die Produktgruppe gehört, zu der die Artikel aus dem Body gehören
4. Im Body nur Artikel, die zu der gleichen Produktgruppe gehören, im Head nur Produktgruppe und die Warengruppe, wie in Fällen 2 und 3

Solche Regeln filtere werden in jedem Fall ausgefiltert, weil sie nichts Neues an Informationen bedeuten und nur als Nebeneffekt der Transaktionen-Erweiterung entstanden sind. Da die Filterung nicht wie im von [Agrawal und Srikant, 1995] beschriebenen Verfahren in dem Algorithmus selbst erfolgt, (weil es eine Implementierung des Algorithmus für traditionelle Regeln benutz wird), wird dieses im Anschluss an den Algorithmuslauf gemacht, nachdem alle Regeln entdeckt wurden und bevor sie interpretiert werden[36].

5.9. Änderung vorhandener und Bildung neuer Hierarchien

Das im Kapitel 4.3 geplantes Verfahren wurde implementiert. Das Programm parst die Ausgabedateien von Apriori (F. Bodon) und sucht für alle k-Itemsets alle ähnlichen k-Itemsets mit gleichem k und Übereinstimung bis auf ein Item (für k>=2). Dann werden jeweils Vereinigungen dieser Itemsets gebildet. Die Länge dieser Vereinigungen ist k+1, da sie bei jeweils verglichenem Paar ein k-1-langes Itemset beinhaltet, der in beiden verglichenen Itemsets gleich ist, plus jeweils ein „exclusiv“-Element aus den beiden Itemsets. Es ist analog zur „candidate generation“-Methode von Apriori, nur mit dem Unterschied, dass der Support nicht berücksichtigt wird. Die Anzahl der neuen künstlich gebildeten k+1-Itemsets ist natürlich sehr von der Anzahl der k-Itemsets in den Eingaben und folglich von dem minsup abhängig, dass bei der Berechnung der häufigen Itemsets im Apriori verwendet wurde. Sie diente als Eingabedaten für diese Berechnung. Das Verfahren auf unterschiedliche Dateien angewandt, die mit unterschiedlichem minsup und unterschiedlichen Artikel-Hierarchieebenen erzeugt wurden. Dabei wurden keine „Cross-Level“-Itemsets genommen, weil es um die Bildung neuer Gruppen aus einer bestimmten Hierarchieebene geht. Die oberste Hierarchieebene wurde nicht verwendet.

[36] Diese Filterung ist in allen Experimenten berücksichtigt, d.h. alle redundanten Regeln werden immer herausgefiltert, damit die Korrektheit der Ergebnisse gegeben ist.

In der nachfolgenden Tabelle (Tabelle 8) ist ein Teil der Ergebnisse des Experimentes dargestellt. Bei einem minsup von 1% werden sowohl bei der Verwendung der untersten als auch der zweiten Hierarchiestufe keine neuen Itemsets gebildet.

Hierarchie- Level	Minsup %	Anzahl der neuen Itemsets
1	**1**	**0**
1	**0,05**	**2107***
1	**0,1**	**120***
2	**0,1**	**29259****
2	**0,5**	**86**
2	**1**	**0**

Tabelle 8 Neu gebildete Itemsets, Variante 1

* hierbei werden fast alle neuen Itemsets leider nur aus den Artikeln gebildet, die die „Ausreißer"-Artikel sind, nämlich verschiedenartige Bohrer, die ohnehin zu der gleichen Gruppe gehören. Wie die Daten zeigen, werden die Bohrer außerordentlich oft gekauft und haben deshalb einen sehr hohen Support im Vergleich zu anderen Artikeln. Deshalb bilden sie die meisten häufigen Itemsets (näheres zu dem Problem s. im Kapitel 5.9.1).

** Viele der neuen Itemsets sind untereinander auch ähnlich und beinhalten im Vergleich zu den reellen Produktgruppen wenige Elemente. Man kann sie noch verbessern, indem man die ähnlichen neuen Itemsets auf gleiche Weise untersucht und größere Itemsets bildet. Dies kann man solange wiederholen, bis die gewünschte Größe und die gewünschte Anzahl der neuen Itemsets erreicht sind, um dann die Itemsets als „neu gebildeten Gruppen" zu übernehmen. So wäre diese Methode für die groben und schnellen Neugruppierungen sogar in der Praxis passend. Genau diese Verbesserung der Methode wird im Kapitel 5.10 weiter diskutiert. Problematisch bliebe aber die Benennung dieser Gruppen zwecks ihrer weiteren Verwendung.

5.9.1. Problem der ungleichmäßigen Verteilung der Support-Werte

Das oben angesprochene Problem zog sich durch alle Experimente mit der untersten Artikelhierarchieebene hindurch. Wurde der minsup kleiner gesetzt, um noch anderen Artikeln zu ermöglichen, an den Berechungen teilzunehmen, so entstanden noch mehr von solchen häufigen Itemsets, die diese Ausreißer-Artikel beinhalteten, und es kam zu einem Arbeitsspeicherproblem, das sogar nach einer Arbeitsspeicherverdoppelung (von 512 MB auf 1GB) nicht behoben werden konnte.

Dieses Phänomen kann dadurch verhindert werden, dass man nicht den gewöhnliche Apriori-Algorithmus, sondern das MIS-Apriori (MIS: „Multiple Item Support") verwendet (näheres dazu im [Liu et. al, 1999]), um die häufigen Itemsets und die Regeln zu finden. Bei diesem Algorithmus kann der Benutzer angeben, welche Minsup-Werte für welche Items bzw. Itemsets gelten sollen, um diese in der Berechnung zu berücksichtigen, d. h., man kann

theoretisch für jedes Item einen minsup angeben. Außerdem kann man mit dem Algorithmus besser das „rare Item“-Problem angehen. Dies ist jedoch nicht das Thema dieser Arbeit und kann aus zeitlichen Gründen nicht ausführlich behandelt werden.

Um dem angesprochenen Problem entgegen zu wirken, wurde trotzdem mit einer Implementierung des MIS-Apriori-Algorithmus experimentiert. Es war eine Implementierung des gleichen Autors (Ferenc Bodon), von dem auch die Apriori-Implementierung stammt, die in anderen Experimenten benutzt wird. Leider waren die Experimente nahezu erfolglos, da die Implementierung nicht ganz fehlerfrei war.

Man kann versuchen, dieses Problem aber auf andere Weise zu umgehen, in dem man die „Ausreißer-Artikel“ (Bohrer etc.) aus den Transaktionen entfernt. Die Itemsets und die Regeln, die man dann findet, sind natürlich nicht mehr „komplett“ und semantisch nicht ganz den Daten entsprechend, aber so kann man den minsup viel kleiner angeben und trotzdem ohne gewaltige Speicherprobleme viele neue Itemsets und Regeln mit Apriori finden.

Zunächst wurden mit Apriori die Supportwerte der einzelnen Items bestimmt und die Items mit deren Supportwerten in die Datenbank importiert. Danach wurden die Transaktionen erneut geparst. Dabei wurde eine Grenze für den maximalen Support der Items vorgegeben und beim erneuerten Parsen der Transaktionsdateien die Supportwerte der Items aus den vorher in die Datenbank importierten Daten selektiert und mit dieser Grenze verglichen. War der aus der Datenbank selektierter Support eines Items größer als diese Grenze, wurde dieses Item nicht in die neue Gesamttransaktionsdatei, die vom Apriori verarbeitet werden soll, übernommen. Somit wurden die Items mit dem zu hohem Support ausgefiltert, um eine mehr oder weniger gleichmäßige Verteilung der Support-Werte der Items zu bekommen. Damit wurde die „Bohrerdominanz“ ausgeschaltet und es konnten bei einem sehr kleinen Minsup-Wert (0,03 %) für Apriori auch viele anderen häufige Itemsets entdeckt werden, (wobei diese natürlich „nicht so häufig“ wie Bohrer waren), ohne ein Speicherproblem wegen der Itemsets mit zu hohem Support zu bekommen. Dann konnten auch andere neue gebildet Gruppen werden. Dazu wurde das oben beschriebene Verfahren eingesetzt, allerdings in etwas modifizierter Form. Warum das Verfahren modifiziert wurde und wie diese modifizierte Vorgehensweise algorithmisch aussieht, wird nachfolgend beschrieben.

5.10. Vergleich der neuen Gruppen mit den vorhanden und Verbesserung der Methode für die Bildung der neuen Gruppen

Man kann die neuen Gruppen mit den vorhandenen Produktgruppen vergleichen, um die Güte dieser Gruppierung zu bewerten. Da in den neuen Gruppen aber fast ausschließlich Bohrer vorkamen, konnte man keinen Vergleich mit den vorhandenen reellen Produktgruppen machen. Das einzige positive Ergebnis dabei war, dass die Bohrer-Gruppe gut erkannt wurde. Interessant wäre aber, trotzdem neue Gruppen bilden, die nicht nur Bohrer. Deshalb wurden die Items mit zu hohem Support wie oben beschrieben „ausgeschaltet“.

5.10.1. Motivation

Nachdem viele neue Itemsets gefunden werden konnten, wurden diese „genauer“ angeschaut. Es fiel auf festgestellt, dass viele davon unter einander immer noch „ähnlich“ sind. Und zwar gab es zwei Arten von „Ähnlichkeiten“:

1. Die eine ist, wie bereits beschrieben: die „ähnlichen“ Itemsets sind gleich lang und haben alle Elemente bis auf eins gleich.
2. Die zweite Art von „Ähnlichkeit“ zeichnet sich dadurch aus, dass zwei Itemsets sich nur um ein Element in ihrer Länge unterscheiden, d. h. der eine Itemset ist eine Teilmenge der Elemente des anderen.

Um das zu verdeutlichen, sind hier einige Beispiele der neu gebildeten Itemsets dargestellt:

1. 810531 810551 810561
2. 810531 810532 810551
3. 810531 810541 810551
4. 810531 810541 810551 810561

Wie man sieht, könnte man die Itemsets 1 und 2, 1 und 3 sowie 2 und 3 paarweise miteinander wegen der Ähnlichkeitseigenschaft der ersten Art verbinden. Man würde folgende neue Itemsets bekommen:

a. 810531 810532 810551 810561 (1 und 2 vereinigt)
b. 810531 810541 810551 810561 (1 und 3 vereinigt)
c. 810531 810532 810541 810551 (2 und 3 vereinigt)

Die a, b und c könnten wiederum miteinander kombiniert werden, und es würden 5-elementige Itemsets entstehen. Außerdem ist ja der Itemset 4 fast genau der gleiche wie der Itemset 3, mit dem Unterschied, dass er ein Element mehr hat, sprich, man könnte den Itemset 4 übernehmen, und Itemset 3 auslassen.

Über die reellen Produktgruppen, die mit diesem Experiment automatisch und möglichst nah nachgebildet werden sollte, ist bekannt, dass sie natürlicherweise viel größer sind, also mehr Elemente beinhalten. Also möchte man versuchen, möglichst lange „neue“ Gruppen zu bekommen. Es entwickelte sich die Idee, immer weiter die neu gebildeten Itemsets zu vereinigen, solange es möglich ist. Wenn keine Vereinigung mehr möglich ist, bleiben nur die Itemsets mit der maximal möglichen Länge übrig. Diese Länge muss natürlich nicht unbedingt mit der Länge der reellen Produktgruppen übereinstimmen. Nach Durchführung des Experimentes kann man sogar sagen, dass sie viel kleiner ist. Nichtsdestotrotz sind die „verbesserten neuen“ Itemsets bereits viel länger als die ursprünglichen „neuen“ Itemsets.

5.10.2. Verbesserte Methode

Die Verbesserte Methode kann man prinzipiell in zwei Schritte aufteilen, die der Einfachheit halber „**Vorbereitungsschritt**“ und „**Abschlussschritt**“ genannt werden. Hier wird die Methode zunächst verbal beschrieben.

Der **Vorbereitungsschritt** läuft wie folgt ab:

Zunächst werden alle k-Itemsets aus der Apriori-Ausgabedatei gelesen.

Alle alle k-Itemsets mit gleichem k werden jeweils zu einer Collection[37] gruppiert. Jede solche Collection wird in einer globalen Hashtable gespeichert, und zwar so, dass „k“ als Schlüssel für die Speicherung der jeweiligen Collection mit k-langen Itemsets verwendet wird. Danach wird durch diese Hashtable iteriert. Es werden für alle k-Werte jeweils zwei Collections unter k und k+1 „unter die Lupe genommen“ (diese enthalten ja alle k- bzw. k+1-langen Itemsets). Dabei wird für alle k, wo $1 \leq k \leq n$ und n die maximale Länge der neu gebildeten Itemsets ist, wie folgt vorgegangen:

a) Die k-langen Itemsets werden zunächst miteinander verglichen und es werden Vereinigungen solcher gebildet, die sich genau um ein Element unterscheiden. Diese aus k-langen Itemsets berechneten Vereinigungen sind genau k+1 lang geworden. Bis hierher ist die Methode noch mit der oben beschriebenen ursprünglich geplanten und implementierten Methode identisch(s. Kapitel 4.3). Der Unterschied kommt ab hier.

b) Alle k-langen Itemsets, die an der Bildung von k+1-langen Vereinigungen teilgenommen haben, werden aus der entsprechenden Collection mit k-langen Itemsets gelöscht und die veränderte Collection wird wieder in die Hashtable unter dem Schlüssel „k“ zurück geschrieben.

c) Alle neuen k+1-langen Vereinigungen werden mit allen k+1-Itemsets aus der entsprechenden Collection verglichen und, falls sie darin noch nicht vorhanden waren, werden sie dorthin eingefügt. Die „überarbeiteten“ Collections mit k+1-langen Itemsets werden anschließend unter dem Schlüssel k+1 in die Hashtable gespeichert.

Hiermit ist der **Vorbereitungsschritt** zu Ende. Danach blieben in der Hashtable Collections mit nur solchen k-langen Itemsets (mit $2 \leq k \leq$ “größter Wert des Schlüssels in der Hashtable“), die nicht vereinigt werden konnten und bei der Bildung von k+1 langen Itemsets nicht verwendet werden konnten.

Jetzt folgt der **Abschlussschritt**.

Hierbei sollen die k-Itemsets aus der Collection, die unter dem größten k in der Hashtable gespeichert blieb, miteinander kombiniert werden. Und zwar rekursiv, solange es noch möglich ist, diese zu kombinieren. D. h., jedes Mal, wenn nach einem Vergleich der Itemsets aus der Collection neue Vereinigungen der „ähnlichen“ Itemsets möglich waren, entstehen neue k+1 lange Itemsets, die zu einer neuen Collection zusammengefasst und unter dem

[37] Implementierungstechnisch wurde als Collection-Datenstruktur für die Sammlung von k-langen Itemsets die Java-Datenstruktur „TreeSet“ aus dem „Collections-Framework“ gewählt, weil sie die Standardoperationen wie „add“, „remove“, „contains“ etc. in $O(\log n)$-Zeit ermöglicht und laut API die schnellste Implementierung der Collection-, (oder Set)-Datenstruktur sein soll und nach vielen Experimenten mit unterschiedlichen Datenstrukturen die besten Laufzeit-Ergebnisse ermöglichte. Im Prinzip handelt es sich um eine bestimmte Implementierungsart vom Binären Baum, oder genauer gesagt „Red-Black“-Baum, der gleichzeitig auch Funktionalitäten von TreeMap besitzt. Für nähere Infos s. Java-API .

Schlüssel k+1 in die Hashtable eingefügt werden. Danach sollen alle in der Hashtable enthaltenen Collections nur noch solche k-Itemsets beinhalten, die nicht mehr kombiniert werden können und somit die Ergebnismenge aller neuen *Itemsets mit maximaler Länge* darstellen.

Hier noch mal der beschriebene Algorithmus als Pseudo-Code in vereinfachter Form dargestellt.

1. //**Vorbereitungsschritt**
2. **read** alle häufigen Itemsets aus der Apriori-Ausgabe-Datei;
3. **for all *k***(*k*=1 bis *k*=maximale Itemset-Länge von gelesenen Itemsets)
4. {
5. **while** (es gibt Itemsets *IS* der Länge *k*)
6. füge *IS* der Collection *aKCol* hinzu;
7. speichere *aKCol* in der Hashtable *allCollections* unter *k*;
8. }
9. **for all k** (*k*=1 bis *k*= maximale Itemset-Länge von gelesenen Itemsets)
10. {
11. *aKCollection*= lese aus *allCollections* unter *k*;
12. *aKplusOneCol* = lese aus allCollections unter *k*+1;
13. *newKplusOneCol*=**buildNewKplusOneCol**(*aKCol*, *aKplusOneCol*);
14. //s. unten Zeile 23
15. speichere *newKplusOneCol* in *allCollections* unter *k+1*;
16. }
17. //**end** Vorbereitungsschritt.
18.
19. //**Abschlussschritt**
20. führe **rekursiveKPluseOneBuilding**() aus //s. unten, Zeile 42.
21. //**end** Abschlussschritt
22.
23. **function buildNewKplusOneCol**(*aKCol*, *aKplusOneCol*)
24. **for all *IS***(*IS* aus *aKCol*)
25. {
26. /*
27. bilde jede mögliche Vereinigung *newKplusOneIS* mit jedem „ähnlichen“ *IS* aus *aKCol*; „ähnlich“ bedeutet hier: gleiche Länge der beiden Itemsets des jeweilig verglichenen Itemset-Paars und alle Elemente in beiden Itemsets des jeweilig verglichenen Itemset-Paars bis auf eins sind gleich

```
28.  */
29.          füge newKplusOneIS der newKplusOneCol hinzu, falls noch nicht vorhanden;
30.          entferne alle IS aus aKCol, bei denen die Vereinigung(en) möglich waren;
31.          bilde jede mögliche Vereinigung newKplusOneIS
32.          mit jedem „ähnlichem“ Element aus aKplusOneCol;
33. /*
34. “ähnlich“ bedeutet hier: Längenunterschied der Itemsets im jeweilig
    verglichenen Paar=1 und alle Elemente bis auf das, was im k+1-langen-Itemset
    „zuviel ist“,  sind gleich
35. */
36.          füge newKplusOneIS der newKplusOneCol hinzu,
37.          falls noch nicht vorhanden;
38.          }
39. return newKplusOneCol;
40. }
41.
42. function rekursiveKPlusOneBuilding()
43. lese die aKCollection aus der allCollections unter dem größten k;
44. while (eine Vereinigung möglich)
45. {
46.          for all IS(IS aus aKCollection )
47.          {
48.          bilde jede mögliche Vereinigung newKplusOneIS mit
49.          allen IS aus aKCollection;
50.          }
51.          entferne alle IS aus aKCol, bei denen die Vereinigung(en) möglich waren
52.          füge newKplusOneIS der newKplusOneCol hinzu;
53.          speichere newKplusOneCol in allCollections unter k+1;
54.          führe rekursiveKPluseOneBuilding() aus;
55. }
56.
```

Bei Ausführung von diesem Experiment sind sehr große Performanzprobleme eingetreten, da eine große Menge an Itemsets als Eingabe verwendet wurde: ca. 7000 Itemsets. Da bei dieser Berechnung die Supportwerte nicht berücksichtigt werden und im Gegensatz zu Apriori keine neu gebildeten, aber „nicht häufigen“ Itemsets gelöscht werden, sprich kein „prune“-Schritt vorhanden ist und alle generierten Kandidaten auch übernommen werden, ist der Vergleichsaufwand viel größer gewesen. Es wurden diverse Strategien versucht, die aber aber keinen merkbaren Erfolg brachten. Die einzige mehr oder weniger spürbare Beschleunigung

(ca. 3-fach) konnte das Mapping von relativ großen, 6-stelligen IDs der Items auf die kleineren fortlaufenden Integer-Werte bringen. Trotzdem war die Laufzeit des Programms sehr groß. Der erste„Vorbereitungsschritt“, wie er oben genannt wurde, dauerte zwar relativ nicht lange (bei den gegebenen 7000 Itemsets ca. 14 min.), aber der „Abschlussschritt“ dauerte beträchtlich länger: über 4 Stunden.

Im Prinzip könnte man den ersten Schritt vereinfachen und als Eingabe nur die 2-langen Itemsets verwenden. Denn die 3-, 4-,….- langen Itemsets sind ja aus den 2-langen Itemsets mit Apriori berechnet worden. Allerdings kann man sich ein Usecase vorstellen, bei dem die Itemsets nicht unbedingt mit Apriori und nicht unbedingt alle aus der gleichen Transaktionsmenge gewonnen werden. Dann könnten die größeren Itemsets nicht zwangsläufig die Obermengen der kleineren Itemsets sein. Um diesen Aspekt zu berücksichtigen, wurden absichtlich alle vorhandenen Itemsets bei der Berechnung berücksichtigt.

5.10.3. Beobachtungen

Wie verhalten sich die neuen Itemsets im Hinblick auf die vorhandenen Produktgruppen?

Bei fast allen neuen Itemsets (und das waren nach dem Programmlauf über 100 Stück, und zwar solche, die nicht mehr untereinander kombiniert werden konnten) gehören alle Items im jeweiligen Itemset der gleichen Produktgruppe an! Bei den meisten kürzeren Itemsets gehören alle Elemente den gleichen Gruppen an, nur bei seltenen kleineren und bei längeren Itemsets sind allerdings mehrere reelle Produktgruppen involviert.

Z. B. sah es nach der Überprüfung eines neu gebildeten 23-elementigen Itemsets so aus (Tabelle 9):

Produkt-ID	Produktbezeichnung	Produktgruppe	Warengruppe
3000260000	Verbindungsplaettchen 0	Verbindungsplättchen	Beschläge/Bauelemente
3000260010	Verbindungsplaettchen 10	Verbindungsplättchen	Beschläge/Bauelemente
3000260312	Riffelduebel Buche 8X 40	Riffeldübel	Beschläge/Bauelemente
3000265383	Montageband55733-10, ExtraStark	Tesa Montagebänder	Beschläge/Bauelemente
3000265488	Tesa-Verlegeband 10M:50Mm	Verlegeband	Beschläge/Bauelemente
3000271214	Moebelfuss 4203-0800-2,11101	Möbelfüsse	Beschläge/Bauelemente
3000271814	Heizk-Klappenb.Kw57 Vern	Heizkörperklappenbeschläge	Beschläge/Bauelemente
3000274799	Schiene W 32 Weiss	Wandschienen	Beschläge/Bauelemente
3000274800	Schiene W 64 Weiss	Wandschienen	Beschläge/Bauelemente
3000274802	Schiene W 96 Weiss	Wandschienen	Beschläge/Bauelemente
3000274803	Schiene W 128 Weiss	Wandschienen	Beschläge/Bauelemente
3000274806	Schiene W 192 Weiss	Wandschienen	Beschläge/Bauelemente
3000274851	Tragarm T 17 Weiss	Tragarme	Beschläge/Bauelemente
3000274852	Tragarm T 22 Weiss	Tragarme	Beschläge/Bauelemente
3000274853	Tragarm T 27 Weiss	Tragarme	Beschläge/Bauelemente
3000274854	Tragarm T 32 Weiss	Tragarme	Beschläge/Bauelemente

3000274855	**Tragarm Tv 37 Weiss**	**Tragarme**	**Beschläge/Bauelemente**
3000274856	**Tragarm Tv 47 Weiss**	**Tragarme**	**Beschläge/Bauelemente**
3000274891	**Buchstuetze B 420 Weiss**	**Bücherträger**	**Beschläge/Bauelemente**
4000353438	**Pattex-Kleber 4,5 Kg,Classic**	**Kontaktkleber**	**Arbeitsschutz/Techn.Produkte**
4000355210	**Haushalt-Oel 100Ml Sb**	**Schmierstoffe**	**Arbeitsschutz/Techn.Produkte**
5000616682	**Karo-Scheibe Zn 20X6,4,Stahl**	**Scheiben**	**Befestigungstechik**

Tabelle 9 Beispiel eines neuen Itemsets aus 23 Elementen

Bei einem hier unten beispielhaft abgebildeten kürzeren 5-elementigen Itemset waren alle Elemente aus derselben Produktgruppe (Tabelle 10):

Produkt-ID	Produktbezeichnung	Produktgruppe	Warengruppe
4000827820	**Schraubendr. Torx 10/80Mm, Mehrkomp. Heft**	**Torx-Schraubendreher**	**Werkzeuge/Baugeräte**
4000827821	**Schraubendr.Torx 15/80Mm,Mehrkomp.Heft**	**Torx-Schraubendreher**	**Werkzeuge/Baugeräte**
4000827822	**Schraubendr. Tx 20/100Mm, Mehrkomp Heft**	**Torx-Schraubendreher**	**Werkzeuge/Baugeräte**
4000827823	**Schraubendr. Tx 25/100Mm, Mehrkomp. Heft**	**Torx-Schraubendreher**	**Werkzeuge/Baugeräte**
4000827825	**Schraubendr. Tx 30/115Mm, Mehrkomp. Heft**	**Torx-Schraubendreher**	**Werkzeuge/Baugeräte**

Tabelle 10 Beispiel eines neuen Itemsets aus 5 Elementen

Wenn man sich aber die Warengruppen bei den neuen Itemsets anschaut, sieht man sogar noch eine bessere Übereinstimmung mit reellen Warengruppen. Bei den kürzeren Itemsets, wie im Beispiel mit 5 Elementen, aber auch bei dem längeren Itemset ist bis auf 3 Elemente die Warengruppe überall gleich.

Das Ergebnis bestätigte die Erwartungen, (und übertraf sie sogar, wenn man die Übereinstimmung in Warengruppen der Artikel aus neu gebildeten Gruppen betrachtet): die neu gebildeten Gruppen waren tatsächlich sinnvoll. Der größere Unterschied zu den vorhandenen Gruppen besteht natürlich im Umfang der Gruppen. Die vorhandenen Produktgruppen sind viel größer als die neu gebildeten Itemsets. Die Längen neu gebildeten Gruppen, lagen zwischen 2 und 26. Insgesamt wurden es gebildet (Tabelle 11):

Größe d. Gruppe	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**
Anzahl d. Gruppen	**155**	**46**	**23**	**11**	**13**	**8**	**5**	**3**	**1**	**1**	**1**	**2**	-	-	-	-	**1**	-	-	-	-	**1**	-	-	**1**

Tabelle 11 Neu gebildete Itemsets. Berechnung mit dem modifizierten Verfahren

Nicht alle neu gebildeten Gruppen gehören tatsächlich unterschiedlichen Produktgruppen an, d.h. man sieht, dass manche Itemsets noch weiter verbunden werden sollten und größere Teile der reellen Gruppen bilden könnten, aber es gab anscheinend einfach keine passenden Kombinationen in den Daten, die eine Vereinigung dieser Itemsets ermöglicht hätte. Die meisten Gruppen sind jedoch wirklich Teile der selbständigen Produktgruppen.

Fazit: abgesehen von Performanzproblemen, kann man mit solchem Verfahren tatsächlich die Items aus den vorhandenen Itemsets gruppieren. Man bekommt zwar nicht große Gruppen, die aber trotzdem semantisch sinnvoll sind! D.h. Hämmer werden mit Hämmern, Nägel mit

Nägeln, Schraubendreher mit Schraubendreher gruppiert. Die Tatsache, dass dabei bei längeren neuen Gruppen auch mehrere reelle Gruppen vorkommen, ist darauf zurückzuführen, dass die Kunden natürlicherweise nicht Artikel aus einer einzigen Gruppe zusammen kaufen, sondern aus mehreren. Dabei sind die neuen Gruppen mit Hilfe der reellen Zusammenkäufen gebildet worden. Wenn man größere Gruppen bilden möchte, braucht man mehr Daten. Mehr Daten könnte man durch einen noch niedrigeren Minsup-Wert für Apriori bei der Itemset-Endeckung erreichen. Gleichzeitig aber würden mehr Daten mehr Performanzprobleme bringen, die evtl. mit einer Implementierung in einer anderen schnelleren Programmiersprache (z.B. in C) zu bewältigen wären.

Abschließend kann man sagen, dass die Hoffnung, mit dieser Methode die Artikel gruppieren zu können, sich erfüllt hat. Die Methode kann bei einer fehlenden Hierarchie als ein Hierarchiebildungswerkzeug angewendet werden.

Denkt man genau über die Ergebnisse von Apriori nach, so wird es ersichtlich, dass der Algorithmus Apriori selbst auch die Eigenschaft besitzt, Elemente auf eine bestimmte Art zu gruppieren, denn die häufigen Itemsets sind solche Gruppierungen von Elementen. Im Vergleich zum Apriori liefert die entwickelte Methode aber bessere Ergebnisse:

Die neuen Gruppen, die mit der Methode gebildet wurden, sind größer, als die von Apriori gebildeten häufigen Itemsets, und ihre Anzahl ist kleiner, als die Anzahl der häufigen Itemsets, die von Apriori entdeckt wurden. Während es 7000 Itemsets mit der maximalen Länge von 7 Elementen waren, die von Apriori entdeckt wurden, sind es nach Anwendung der verbesserten Gruppierungsmethode nur noch ca. 260 neue Gruppen, deren Länge dafür vergrößert wurde und maximal 26 Elemente erreicht.

5.11. Integration in vorhandenes Informationssystem und weitere Experimente

Nachdem die ersten Experimente abgeschlossen waren, sollte die Regelentdeckung mit allen möglichen Optionen in das vorhandene System integriert werden. So könnte man bequem weiter experimentieren und die Verbesserungs- und Optimierungsmöglichkeiten direkt aus der Benutzersicht leichter erkennen und durchführen. Außerdem könnten so die unterschiedlichen Optionen der Regelentdeckung verschiedener Art mit verschiedenen Filter-, Gruppierungs-, Sortier- Einstellungen sowie Involvierung der Kundenhierarchie untersucht werden, um so die bestmögliche Analyse der Bestelldaten zu erzielen.

Als Kernalgorithmus sollte weiterhin Apriori mit Implementierung von Ferenc Bodon dienen. Die Integrationsplattform sollte das am Anfang im Kapitel 2.2 beschriebene NIS sein.

5.11.1. Entwurf der Grafischen Benutzerschnittstelle

Die Grafische Oberfläche soll dem Benutzer den maximalen Komfort ermöglichen. Um allerdings dieses Ziel zu erreichen, müsste man eine Extraarbeit allein für dieses Thema schreiben. Deshalb wurde (zumindest im Anfangstadium) auf Raffinessen des GUI verzichtet und die Funktionalität in Vordergrund gestellt. Zumal sollte die Bedienung nicht durch einem unerfahrenen Benutzer, sondern z. B. durch einen Manager erfolgen, der sich bewusst ist, was er sucht, und sich vorstellen kann, was er finden wird. Angesichts der oberen Experimente und der bevorstehenden Untersuchungen sollten die nachfolgend beschriebenen und weiter unten abgebildeten Einstellungsmöglichkeiten bzw. Optionen in der GUI, und

dementsprechend in der Logikschicht vorhanden sein (s. Abbildung 19). Die Implementierung erfolgt mit JSP und Javabeans, d. h. der Browser dient als Frontend für den Benutzer.

Nachfolgend werden einige Einstellungen für die Suche beschrieben.

1. **Einstellung der Artikelhierarchiestufe:** Hierbei soll es eine Auswahl zwischen „Artikel", „Produktgruppe" oder „Warengruppe" geben, falls man Regel für jede Stufe für sich sucht. Alternativ kann man „Crosslevel" auswählen, falls Cross-Level-Regeln gesucht werden sollen(s. dazu Kapitel 3.2).
2. **Einstellung der Produktgruppenauslastung:** Bei der Auswahl „Produktgruppe" soll zusätzlich eine „erweiterte" Einstellung erscheinen, bei der, falls man möchte, eine minimale Ausnutzung der Produktgruppe vorliegen soll, um die Produktgruppe als solche in die Transaktion zu übernehmen (s. dazu Kapitel 4.2), andernfalls soll nicht die Gruppe, sondern die Artikel übernommen werden, die zu dieser Gruppe gehören und tatsächlich an der betroffenen Transaktion „teilgenommen" haben, wobei zwangsläufig einige Regeln zu Cross-Level-Regeln werden.
3. **Einstellung der Stufe der Kundenhierarchie:** Falls man die Kundenhierarchie in die Regeln involvieren möchte, soll es möglich sein, die Stufe der Kundenhierarchie einzustellen, die in die Transaktionen mit einfließt: dabei kann zwischen Kunden bzw. Kundengruppen (gebildet nach PLZ) gewählt werden (s. dazu Kapitel 4.4).
4. **Einstellung der Apriori-Parameter:** Selbstverständlich sollten die Minsup und Minconf einstellbar sein, um möglichst viele verschiedene Konstellationen für den Algorithmuslauf zu schaffen. Eventuell kann es später noch eine Möglichkeit der Einstellung von Maxsup für die einzelnen Artikel geben, um die oben beschriebene Dominanz der sehr häufigen Artikel zu vermeiden und dadurch die Laufzeit zu verbessern (s. dazu Kapitel 5.9.1).
5. **Einstellung der Gruppierung:** Diese soll nach dem Psaila / Lanzi –Verfahren die unterschiedliche Gruppierung der Daten in die für die Apriori-Berechnung verwendeten Transaktionen ermöglichen. Dabei können im einfachsten Fall die reellen Transaktionen als Transaktionen erfasst werden, also soll es die Auswahlmöglichkeit „Transaktionen" geben, oder es wird nach Kunden bzw. Kundengruppen gruppiert, und die gruppierten Daten bilden die weiter zu verwendenden Transaktionen, also Auswahlmöglichkeiten „Kunde" bzw. „Kundengruppe" (s. dazu Kapitel 3.2.5 und 5.5).
6. **Einstellung der Filterung:** Es soll einstellbar sein, ob eine Filterung der Regeln erfolgen soll. Falls „Filtern" ausgewählt wird, sollen Minlift und Minleverage unabhängig voneinander oder in Kombination einstellbar sein. (s. dazu Kapitel 5.3) Später kann eventuell noch das Maß von Agrawal und Srikant bei den Crosslevel Regeln hinzugefügt werden (z. Z. noch nicht implementiert).
7. **Einstellung der speziellen Suche nach den „robusten" Regeln:** Es soll möglich sein, explizit nach Robusten Regeln zu suchen (s. dazu Kapitel 3.2.4 und 5.11.3.1). Wenn die Checkbox selektiert ist, werden nur solche Regeln angezeigt (falls es sie gibt). Ansonsten werden sie einfach farblich hervorgehoben.
8. **Einstellung der Sortierung:** Es soll möglich sein, die gefundenen Regeln nach bestimmten Kriterien zu sortieren. Dabei können „Support", „Confidence", „Häufigkeit", „Lift" und „Leverage" als Sortieroptionen gewählt werden.
9. **Top-Down-Suche und Einstellung der speziellen Suche nach „starken" Regeln:** Eine Top-Down-Suche nach starken Regeln soll durchgeführt werden können (s. dazu Kapitel 3.2.3 und 5.11.3.2).

Auf der in der Abbildung 19 dargestellten Suchmaske sieht man die verschiedenen Einstellungsmöglichkeiten und Optionen für die Regelentdeckung. Die darunter abgebildeten Selectboxen (*1, *2, *3) sind die möglichen Einstellungen der jeweiligen Optionen. Außerdem erscheinen nicht alle möglichen Einstellungen immer, sondern sind je nach anderen ausgewählten Optionen ein- oder ausgeblendet. So erscheinen die Filtereinstellungen (*4) nur, wenn der Radiobutton „Filtern“ (*9) ausgewählt ist. Die Minimale Produktgruppen-Auslastung (5*) erscheint auch nur, wenn die Auswahl „Produktgruppe“ in der Artikelhierarchie ausgewählt ist. Alle Kombinationen dieser Einstellungen wurden in Rahmen der Arbeit implementiert und die Ergebnisse analysiert.

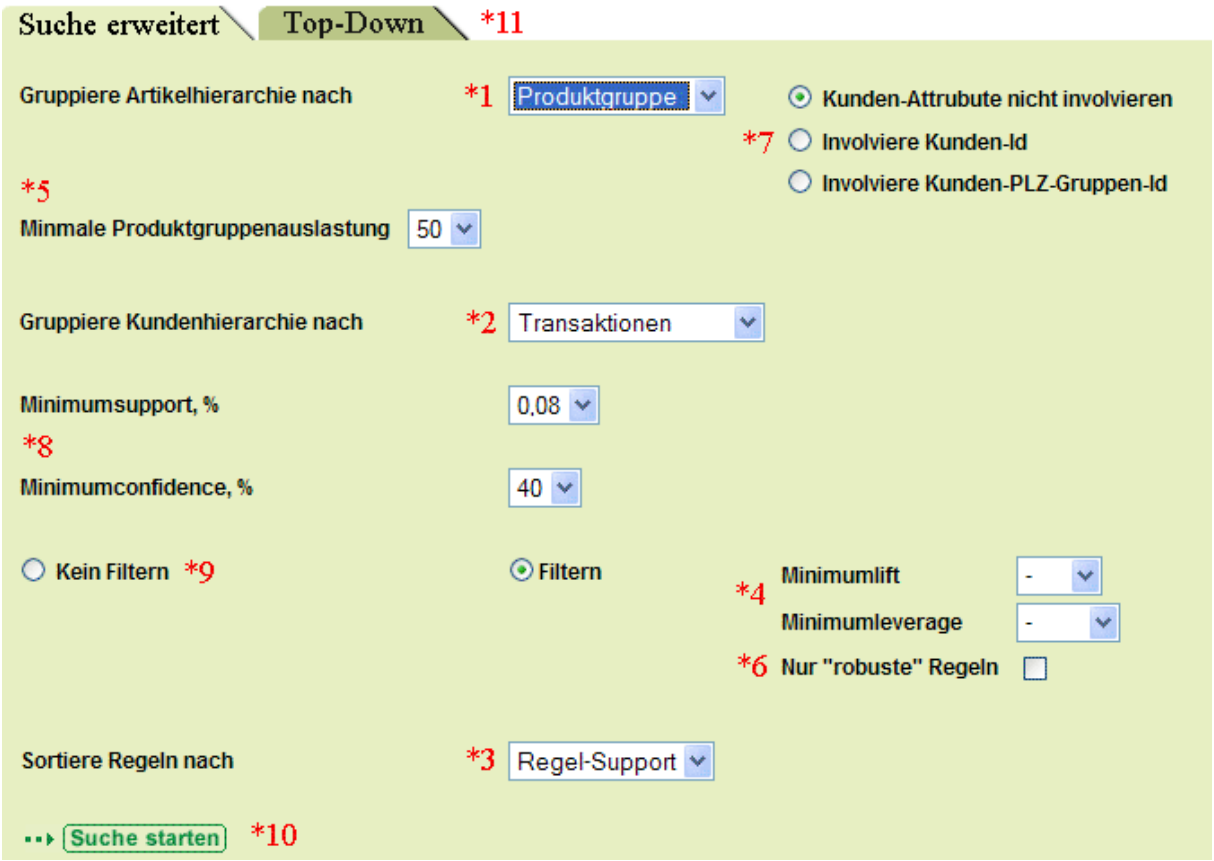
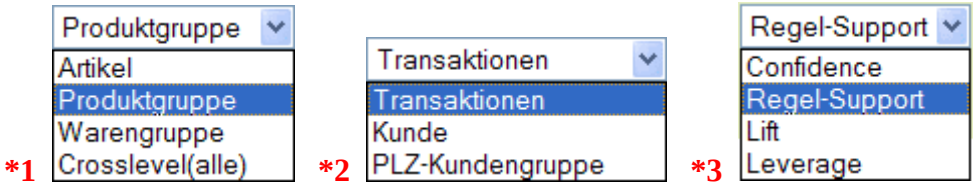


Abbildung 19 Suchmaske mit versch. Suchoptionen für die Regelentdeckung



5.11.2. Funktionsweise

Der einfachste Ablauf der Regelsuche, die die realisierte Implementierung bietet, wird nachfolgend beschrieben.

Der Benutzer wählt in der JSP, die die Suchmaske (Abbildung 19) darstellt, die gewünschten Suchoptionen aus. Angenommen, es wurde die einfachste Kombination ausgewählt: in der Artikelhierarchie (*1[38]) „Artikel“, und in der Gruppierung (*2) die „Transaktionen“. Es wurde „kein Filtern“ ausgewählt und die Kundenattribute werden auch nicht berücksichtigt. Es soll nach Confidence sortiert werden. Minsup wurden auf 0,1% und Minconf auf 40% eingestellt. („default“ Einstellungen).

Wenn auf „Suche starten“ (*10) geklickt wird, werden sämtliche Parameter von dem Browser an die Logik (implementiert mit Javabeans) weitergegeben. Diese werden von der Logik analysiert. Anhand der Parameter und in Abhängigkeit von ihnen werden SQL-Abfragen generiert und die benötigten Daten aus der Datenbank selektiert, die für die Transaktionsbildung benötigt werden. Diese Daten sind vorher aus den vielen Einzeltransaktionsdateien mit dem speziell dafür geschriebenen separaten Programm importiert worden. Sie beinhalten unter Anderem die Zugehörigkeit der Artikel zu den tatsächlichen Transaktionen, zu Produkt- und Warengruppen sowie Auslastungen der Produktgruppen, die auch vorher ausgerechnet wurden (bei den gewählten einfachsten Einstellungen sind die letzten 3 zunächst nicht von Bedeutung). Die Transaktionen werden aus diesen Daten gebildet und in eine Textdatei geschrieben, die für Apriori als Eingabedatei dienen soll. Eine Batch-Datei wird von der Javabean aufgerufen. Diese startet die externe C++ -Implementierung von Apriori und übergibt ihr die benötigten Argumente: die von der Suchmaske übergebenen minsup und minconf sowie die soeben erzeugte Transaktionsdatei. Der Apriori liest die Transaktionen aus der Datei und generiert die häufigen Itemsets sowie die Assoziationsregeln, die in eine Ausgabedatei geschrieben werden. Die Javabean-Logik wartet auf die Beendigung des Apriori, und liest anschließend seine Ausgabedatei. Die gelesenen Regeln werden analysiert, ggf. gefiltert (bei den beschriebenen Einstellungen zunächst nicht nötig) und für die Darstellung weitergegeben. Es öffnet sich die nächste JSP-Seite, auf der dem Benutzer diese Regeln in der von ihm eingestellten Sortierreihenfolge dargestellt werden. Der ganze Ablauf passiert in Echtzeit.

Das ist im Groben der Ablauf bei den einfachsten Einstellungen. Die Laufzeit ist dabei akzeptabel und liegt, (je nach eingestellten minsup und minconf) im Bereich von 1min[39], wobei die meiste Zeit für die Erzeugung der Transaktionen und die Regelinterpretation verbraucht wird.

Natürlich sind auch komplexere Sucheinstellungen möglich, mit deren Hilfe der Anwender noch gezielter seine Suche gestalten kann. Dabei bleibt der gesamte Suchablauf für den Benutzer transparent.

[38] „*1“ Bezieht sich auf die Markierungen auf der Abbildung 19.

[39] Mit einem kleineren Support gehen bei Apriori-Lauf die Arbeitspeicherbelegung und die Berechnungszeit über die maximal zumutbaren Werte. Deshalb wurde in der Suchmaske die Einstellung von Minsup bei der Ausgewählten Gruppierung nach Kunden automatisch nach unten begrenzt, so dass nur die Werte ab 20% und höher möglich sind. Bei der Gruppierung nach den Transaktionen in Kombinationen mit allen anderen möglichen Einstellungen sind kleinerer Minsup-Werte einstellbar.

Bei den komplexeren Suchoptionen werden auch andere Parameter berücksichtigt. So kann beispielsweise bei der Auswahl „Produktgruppe“ in der Artikelhierarchie die minimale Gruppenauslastung[40] vorgegeben werden. In diesem Fall überprüft die Logik die Auslastung von jeder Produktgruppe in jeder zu erzeugenden Transaktion und ersetzt ggf. die Gruppe durch die entsprechenden Produkte. Bei voreingestellten Filterparametern werden die Regeln im Anschluss an ihre Erzeugung noch gefiltert. Die Ausgefilterten Regeln kann man bei Bedarf in einem speziellen Logfile betrachten. Z.B. können aus allen erzeugten Regeln nur solche dargestellt werden, die die Eigenschaft „robust“ besitzen (wie in Kapitel 5.11.3.1 weiter beschrieben). Alle zusätzlichen Optionen wirken sich natürlich in der längeren Laufzeit aus. Die Darstellung der Regel erfolgt auch mittels JSP und hat das Tabellarische Layout, wie in die in den frühren Kapiteln dargestellten Beispiele. Die Liste der dargestellten Regeln lässt sich nach unterschiedlichen Kriterien sortieren, in dem man oben auf den Namen der Spalte klickt. Dabei wird die Liste nach dem Kriterium, das in der Spaltenüberschrift angeklickt wurde, absteigend sortiert. Möchte man eine aufsteigende Sortierung der Liste nach dem gleichen Kriterium bekommen, kann es durch ein erneuertes Anklicken der Spaltenüberschrift erfolgen.

Die schematische Darstellung er Funktionsweise in vereinfachter Form ist auf der Abbildung 20 zu sehen.

[40] Wie wird die Auslastung der Produktgruppen berechnet bzw. benutzt? Die Berechnung der Gruppenauslastung basiert auf dem Verfahren, das im Kapitel 4.6 Beschrieben ist. Diese Berechnung wird vor der Echtzeitnutzung des Systems durchgeführt und in die Auslastungswerte in die Datenbank importiert, so dass sie zum Zeitpunkt der Benutzung bereits in der Datenbank vorliegen.

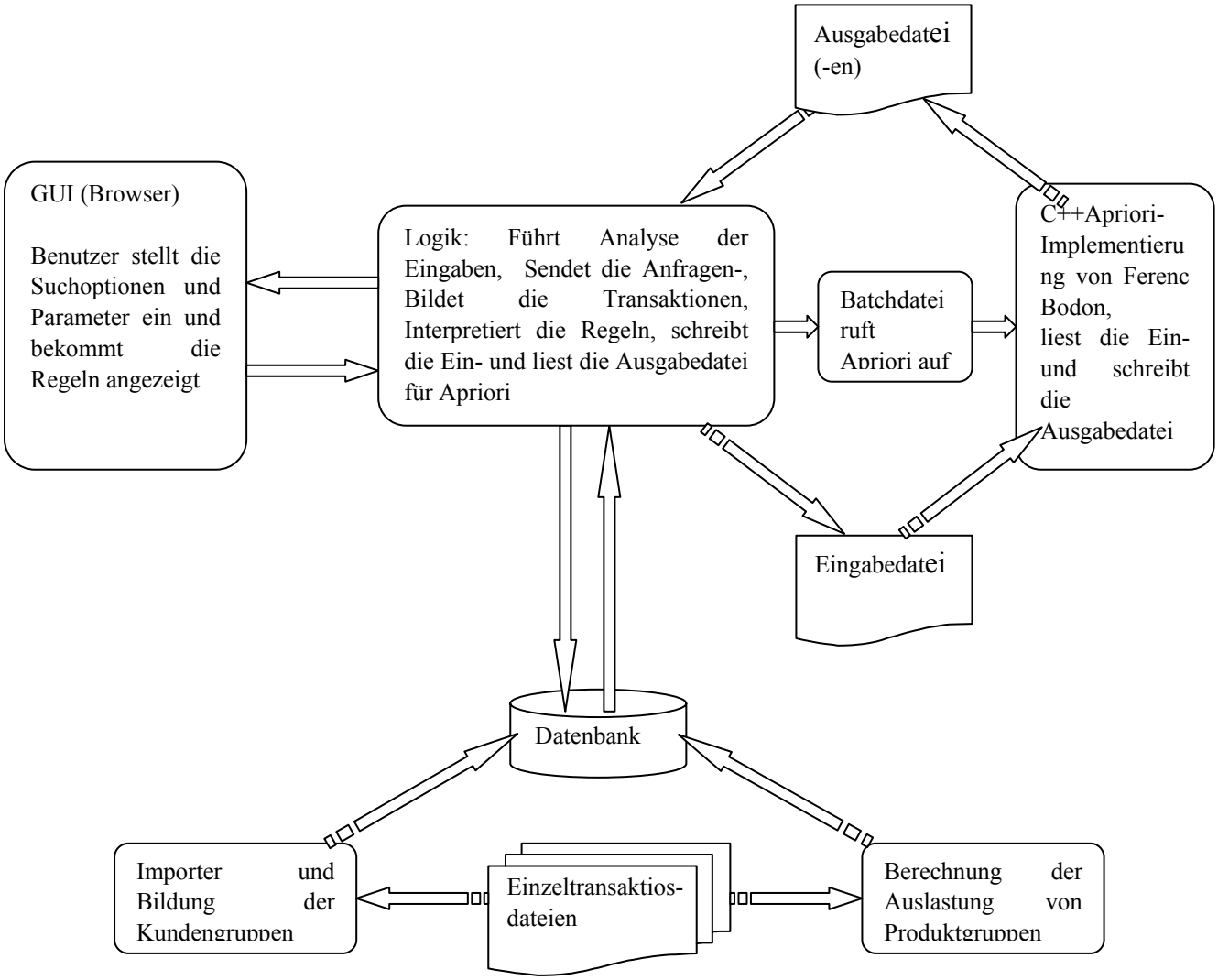


Abbildung 20 Schematische Darstellung der Funktionsweise

5.11.3. Erweiterte Analysemöglichkeiten

5.11.3.1. Entdeckung der robusten Regeln

Nun folgt eine Beschreibung der Experimente mit „robusten“ Regeln. (Die Eigenschaft wurde in Kapitel 3.2.4 beschrieben).

Auf den vorliegenden Daten sollten es solche Regeln sein, die auf der linken Seite eine Warengruppe, und auf der Rechten Seite ein Produkt beinhalten. Diese Eigenschaft wurde als Filter-Kriterium implementiert. Nachdem die Experimente mit gezielter Suche nach robusten Regeln durchgeführt wurden, stellte sich heraus, dass äußerst wenig robuste Regeln gefunden wurden: im Fall von 0,08% Minsup besaßen von über 35000 Regeln nur 4 (!) die Eigenschaft „robust“ (s. Abbildung 21).

Nr.	Regel	Häufigkeit	Support	Confidence	Leverage	Lift
Nr. 1	[Technische Armaturen (warengruppe)] ➡ [AUSBLASPISTOLE KUNSTSTOFF,M. VERLAENG.ROHR 115 MM,1/4" ANSCHLUSSGEWINDE (produkt)]	23	0,00084456	1,00000000	0,00084388	1.237,86363636
Nr. 2	[Büroausstattung (warengruppe)] ➡ [EDDING 750, WEISS,PAINTMARKER (produkt)]	40	0,00146881	0,46511600	0,00146803	1.904,40559441
Nr. 3	[Transportmittel (warengruppe)] ➡ [SACKKARRE 250KG LUFT BLAU (produkt)]	93	0,00341497	0,45365900	0,00341364	2.560,83822042
Nr. 4	[Verpackung (warengruppe)] ➡ [STRETCHFOLIE ROLLE A 300M,450 MM BREIT,20 MY STARK,3-SCHICHT-FOLIE,EXTRA ST. (produkt)]	64	0,00235009	0,41558400	0,00234904	2.228,78772379

Abbildung 21 Robuste Regeln

Betrachtet man die Regeln genau, so sieht man, dass die auf der rechten Seite stehenden Produkte in 3 von 4 Fällen genau zu den Warengruppen gehören, die auf der linken Seite stehen. Das bedeutet ja, dass aus diesen Warengruppen diese Produkte am meisten verkauft worden sind. Ist dieser Ausdruck wirklich Aussagekräftig? Erstens, ist eine Warengruppe zu allgemein, und zweitens, um dieses Ergebnis zu bekommen, bräuchte man eigentlich keine komplizierte Analyse mit den generalisierten Assoziationsregeln. Man könnte es mit einfachen SQL-Abfragen in wenigen Schritten aus der Datenbank bekommen. Deshalb soll die „Robustheit“ etwas umformuliert werden: interessanter würde die Eigenschaft sein, eine (oder evtl. mehrere) Produktgruppen auf der linken Seite der Regel zu haben, und ein Produkt auf der rechten Seite, was nicht zu der Produktgruppe gehört, die auf der linken Seite steht. Das würde bedeuten, dass wenn z.B. links die Produktgruppe „Holzschrauben“ steht, und rechts das Produkt „Spiralbohrer 8mm, Marke-XY“, dann heißt es, dass unabhängig davon, welche Holzschrauben gekauft werden (Hauptsache, sie werden gekauft), wird sehr oft diese bestimmte Bohrerart gekauft. Mit anderen Worten würde es einen Ausdruck der Form $A \vee B \vee C \ldots \Rightarrow X$ bedeuten, wobei A, B, C Produkte aus der Produktgruppe „Holzschrauben“ sind, die auf der linken Seite der Regel steht, und X das Produkt „Spiralbohrer 8mm, Marke-XY“, das auf der rechten Seite steht. Nach dem die Filter-Implementierung für die Suche der Robusten Regeln entsprechend der neuen Formulierung der „Robustheit“ geändert wurde, wurden bei den gleichen Parametereinstellungen wie oben wiederum nur 5 Regel (aus über 35000!) selektiert. Diese sind in der Abbildung 22 zu sehen. Alle Produkte, die jetzt auf den rechten Seiten stehen, gehören nicht zu den Produktgruppen, die auf den linken Seiten stehen. Allerdings, nach der Überprüfung, hat sich herausgestellt, dass diese in 4 von 5 Fällen einer Produktgruppe gehören, die sehr ähnlich der auf der linken Seite stehenden Produktgruppe ist. Z.B. in der Regel Nr.2 gehört das Produkt „Steckschl. SA…1/4’’ einer Produktgruppe „Steckschlüssel-Sätze 1/4 “, links steht aber eine Produktgruppe „Steckschlüssel-Sätze 1/4’’-1/2’’“, die sich wahrscheinlich nur durch schlechte manuelle Gruppierung von der erstgenanten unterscheidet. Man kann das als positives Ergebnis sehen: die Entdeckung dieser Regeln bedeutet, dass mit diesen Regeln solche Fälle entdeckt werden, bei denen die Gruppierung etwas verbessert werden könnte. Bei der Regel Nr.1 handelt es sich aber wirklich um ein interessante Regel: in 33 Fällen wurde „Füllung für Verbandschränke“ bestellt, wenn irgendwelche „Verbandschränke“ gekauft wurden (hierbei gehört der Inhalt der rechten und linken Seiten wirklich nicht durch unfeine Gruppierung in unterschiedlichen Hierarchiezweigen an).
Vielleicht würden bei einer anderen Datenlage mehr solcher Regeln entstehen. Ferner ist eine weitere Untersuchung der Regeln mit Involvierung der Kundenhierarchie geplant.

Nr.	Regel	Häufigkeit	Support	Confidence	Leverage	Lift
Nr. 1	Verbandschränke (produktgruppe) ➡ FUELLUNG 72 TLG.DIN13157 (produkt)	33	0,00121168	0,62264200	0,00121039	939,13793103
Nr. 2	Steckschlüssel-Sätze für Torx-Schrauben (produktgruppe) Steckschlüssel-Sätze 1/4"-1/2" (produktgruppe) ➡ STECKSCHL.SA.MEC1/4-13 PR (produkt)	24	0,00088122	0,50000000	0,00088051	1.235,61436673
Nr. 3	Durchtreiber-Satz (produktgruppe) ➡ SPLINTENTREIBERSATZ 6-TLG,-PROMAT- (produkt)	33	0,00121168	0,48529400	0,00121096	1.698,96975425
Nr. 4	Doppel-Ringschlüssel-Sätze (produktgruppe) Schraubendreher-Sätze (produktgruppe) ➡ DOPPEL-MAULSCHL.SATZ12TLG,-PROMAT- (produkt)	24	0,00088122	0,47058800	0,00088051	1.237,95454545
Nr. 5	Steckschlüssel-Sätze 1/2" (produktgruppe) Steckschlüssel-Sätze 1/4"-1/2" (produktgruppe) ➡ STECKSCHL.SA.MEC1/4-13 PR (produkt)	36	0,00132183	0,44444400	0,00132105	1.705,14782609

Abbildung 22 Robuste Regel, nach der "neuen Formulierung der Robustheit ".

5.11.3.2. Top-Down-Suche und Entdeckung der starken Regeln

In diesem Abschnitt sollen die Experimente zur Entdeckung der „starken“ Regeln[41] beschrieben werden. Die Implementierung dieser Experimente weicht leicht von dem in dem in Kapitel 3.2.3 beschriebenen Algorithmus ab, basiert aber größtenteils auf dem Ansatz von Han und Fu. Der Unterschied liegt darin, dass, während bei Han und Fu nur die „Candidate-Generation“-Methode des Apriori verwendet wird, wird hier der vollständige Apriori-Lauf verwendet. Die Erklärung dafür ist die Verwendung einer fertigen Implementierung von Apriori. Dabei beeinträchtigt es die Laufzeit nur unwesentlich und verursacht keine Performance-Probleme. Außerdem ist der Laufzeit von Apriori im Vergleich zur Weiterverarbeitungs- und Interpretationszeit der Regeln sehr gering, da die meiste Zeit bei den letzteren verbraucht wird.

Folgender Ablauf wurde implementiert:

es werden für jede Hierarchiestufe eigene Minsup- und Minconf- Werte vom Benutzer eingegeben. Die Suche wird sequenziell auf jeder Stufe der Hierarchie getrennt durchgeführt, wobei von der oberen Stufe angefangen wird. Die mit Elementen der obersten Stufe erzeugten Regeln werden vollständig übernommen. Die Suche wird auf der zweiten und der ersten Stufen fortgesetzt. Die Regeln aus der zweiter und der erster Stufen werden auf die Eigenschaft „stark“ geprüft: wurde für jedes Itemset einer aktuell betrachteten Regel ein häufiger Vorfahre auf der höher liegenden Stufe gefunden und ist der aktuell betrachtete Itemset auch häufig, so ist die aktuell betrachtete Regel stark, falls sie auch für ihre eigene Stufe häufig ist und die Minconfidence-Schwelle erreicht. Ansonsten ist die Regel nicht stark und wird verworfen, falls nur Starke Regeln gesucht werden. In der Abbildung 23 ist die für diese Suche vorgesehene Suchmaske dargestellt.

Abbildung 23 Top-Down-Suche nach starken Regeln

Der Benutzer kann die Anzeige der gefundenen Regeln noch zusätzlich beeinflussen:

[41] Die Beschreibung des Ansatzes und die Definition der „starken“ Regeln ist im Kapitel 3.2.3 beschrieben.

es können entweder alle nach diesem Ansatz gefundenen Regeln angezeigt werden, inklusive der Regeln aus der obersten Hierarchiestufe, oder nur die starken Regeln aus den beiden unteren Hierarchiestufen. Zusätzlich kann eine weitere Einschränkung bei der Suche aktiviert werden: es können gezielt nur solche Regeln gesucht werden, die auf ihrer beiden Seiten Elemente haben, deren Vorfahren unterschiedlich sind. D.h., bei dieser Einschränkung werden nicht Regeln mit Artikeln aus der gleichen Produktgruppe oder Produktgruppen aus der gleichen Warengruppe angezeigt (näheres dazu im nächsten Kapitel 5.11.3.3). Das ermöglicht eine gute Übersicht über Zusammenhänge zwischen unterschiedlichen Hierarchiezweigen. Die gefundenen starken Regeln werden auch farblich (grün) gekennzeichnet, wie in Abbildung 24 dargestellt ist.

Nr.	Regel	Häufigkeit	Support	Confidence	Leverage	Lift
Nr. 1	Spiralbohrer (produktgruppe) , Winkelverbinder (produktgruppe) ➡ Holzschrauben (produktgruppe)	24	0,00090703	0,53333300	0,00090138	160,64760941
Nr. 2	Pinsel (produktgruppe) , Sicherheitshalbschuhe (produktgruppe) ➡ Atemschutz (produktgruppe)	23	0,00086924	0,42592600	0,00086392	163,46494762
Nr. 3	STECKSCHL.SA.CV 1/2-29PR (produkt) , STECKSCHL.SA.55/32 MM PR (produkt) ➡ STECKSCHL.SA.MEC1/4-13 PR (produkt)	27	0,00099137	0,72973000	0,00098607	186,96796339
Nr. 4	PATTEX-KLEBER 4,5 KG,CLASSIC (produkt) ➡ VERBINDUNGSPLAETTCHEN 20 (produkt)	23	0,00084450	0,41071400	0,00084096	238,63047619
Nr. 5	STECKSCHL.SA.MEC1/4-13 PR (produkt) , STECKSCHL.SA.CV 1/2-29PR (produkt) ➡ STECKSCHL.SA.55/32 MM PR (produkt)	27	0,00099137	0,56250000	0,00098808	301,12407862
Nr. 6	STECKSCHL.SA.CV 1/2-29PR (produkt) ➡ STECKSCHL.SA.MEC1/4-13 PR (produkt)	48	0,00176244	0,53932600	0,00175714	332,38749047
Nr. 7	Spiralbohrer (produktgruppe) , Drahtstifte (produktgruppe) ➡ Holzschrauben (produktgruppe)	29	0,00109599	0,47541000	0,00109296	361,27118644
Nr. 8	VERBANDSCHRANK 47X40X11 L (produkt) ➡ FUELLUNG 72 TLG.DIN13157 (produkt)	29	0,00106481	0,70731700	0,00106218	405,03333333
Nr. 9	DOPPEL-MAULSCHL.SATZ12TLG,-PROMAT- (produkt) ➡ RINGSCHLUESSELSATZ 12TLG.,-PROMAT- (produkt)	44	0,00161557	0,42718400	0,00161196	447,47572816
Nr. 10	RINGSCHLUESSELSATZ 12TLG.,-PROMAT- (produkt) ➡ DOPPEL-MAULSCHL.SATZ12TLG,-PROMAT- (produkt)	44	0,00161557	0,51764700	0,00161196	447,47572816
Nr. 11	STECKSCHL.SA.MEC1/4-13 PR (produkt) , STECKSCHL.SA.55/32 MM PR (produkt) ➡ STECKSCHL.SA.CV 1/2-29PR (produkt)	27	0,00099137	0,55102000	0,00098932	484,41699605
Nr. 12	STECKSCHL.SA.CV 1/2-29PR (produkt) ➡ STECKSCHL.SA.55/32 MM PR (produkt)	37	0,00135855	0,41573000	0,00135650	663,83069829
Nr. 13	DURCHTREIBERSATZ 6-TLG.,-PROMAT- (produkt) ➡ SPLINTENTREIBERSATZ 6-TLG,-PROMAT- (produkt)	33	0,00121168	0,48529400	0,00121047	1.008,70370370

Abbildung 24 Starke Regeln

Optional kann die Top-Down-Suche auch ohne Überprüfung der Regeln auf Kriterium „stark“ erfolgen. Dabei werden alle gefundenen Regeln aus allen Stufen angezeigt.

Die Tabelle 12 zeigt die Ergebnisse der Top-Down-Suche mit und ohne Einschränkungen auf starke Regeln. Bei dem ersten Teil der in der Tabelle dargestellten Experimenten wurden bei der Suche nach starken Regeln nur solche starken Regeln übernommen, bei denen das Kriterium galt, dass die Vorfahren der Itemsets beider Seiten unterschiedlich sind (näheres dazu im nächsten Kapitel 5.11.3.3), bei dem zweiten Teil der Experimente wurde diese Einschränkung nicht aktiviert .

Alle Experimente (d.h. sowohl im Teil 1 als auch im Teil 2 der Tabelle) waren so aufgebaut, dass:

1. zunächst eine Top-Down-Suche durchgeführt wurde, die auf allen Hierarchiestufen gleiche Parameter angewendet hat (Zeilen 1, 2, 3 und 7, 8, 9 der Tabelle 12)
2. und danach eine Top-Down-Suche mit stufenspezifischen Parametern (Zeilen 4, 5, 6 und 10, 11, 12 der Tabelle 12).

Für jede dieser Parametereinstellungen wurden jeweils 3 Experimente gemacht:

1. nicht gezielt nach den starken Regeln gesucht und alle Ergebnisse anzeigen (Zeilen 1, 4, 7, 10 der Tabelle 12)
2. gezielt nach starken Regeln suchen, alle Regeln aus der obersten Stufe und nur starke Regeln aus beiden unteren Stufen anzeigen (Zeilen 2, 5, 8, 11 der Tabelle 12)
3. gezielt nach starken Regeln suchen und nur diese aus beiden unteren Stufen anzeigen (Zeilen 3, 6, 9, 12 der Tabelle 12).

	Art der Top-Down Suche	Minsup level3 %	Minsup level2 %	Minsup level1 %	Minconf level3 %	Minconf level2 %	Minconf level1 %	Anzahl der gefundenen Regeln	Anzahl der angezeigt. Regeln	Anzahl der starken	ausgefiltert. Regeln, %	Laufzeit, Sekunden
Nur Regeln mit unterschiedlichen Vorfahren für beide Seiten gesucht	Alle R. für Lev. 1,2,3	0,08	0,08	0,08	40	40	40	1205	523	Nicht gesucht	Nicht gefilt.	30
	Alle R. für Lev.3, nur starke R. für Lev.2,1	0,08	0,08	0,08	40	40	40	1205	523	13	57	37
	Keine R. Lev.3, nur starke R. für Lev.2,1	0,08	0,08	0,08	40	40	40	1205	13	13	99	37
	Alle R. für Lev. 1,2,3	1	0,08	0,05	30	30	30	3128	235	Nicht gesucht	Nicht gefilt.	195
	Alle R. für Lev.3, nur starke R. für Lev.2,1	1	0,08	0,05	30	30	30	3128	220	89	93	212
	Keine R. Lev.3, nur starke R. für Lev.2,1	1	0,08	0,05	30	30	30	3128	89	89	97	212
Geliche Vorfahren der beiden Seiten erlaubt	Alle R. für Lev. 1,2,3	0,08	0,08	0,08	40	40	40	1205	1205	Nicht gesucht	Nicht gefilt.	30
	Alle R. für Lev.3, nur Starke für Lev.2,1	0,08	0,08	0,08	40	40	40	1205	1203	693	0,17	37
	Keine R. Lev.3, nur starke R. für Lev.2,1	0,08	0,08	0,08	40	40	40	1205	693	693	48	37
	Alle R. für Lev. 1,2,3	1	0,08	0,05	30	30	30	3128	3128	Nicht gesucht	Nicht gefilt.	195
	Alle R. für Lev.3, nur starke R. für Lev.2,1	1	0,08	0,05	30	30	30	3128	3005	2974	4	210
	Keine R. Lev.3, nur starke R. für Lev.2,1	1	0,08	0,05	30	30	30	3128	2974	2974	5	210

Tabelle 12 Top-Down-Suche, Übersicht der Ergebnisse

Was kann man über die Ergebnisse dieses Ansatzes im Vergleich zu Experimentergebnissen mit dem Ansatz von Agrawal und Srikant sagen?

Zunächst, werden bei Top-Down-Suche keine Cross-Level-, sondern nur die Multiple-Level-Regeln gefunden (s Kapitel 3.2.3.3). Einerseits ist die Anzahl der gefundenen Regeln dabei viel kleiner, was dadurch zu erklären ist, dass keine Kombinationen der Itemsets aus verschiedenen Stufen erzeugt werden, andererseits sind die Regeln leichter zu verstehen bzw. zu interpretieren. Führt man die Top-Down-Suche ohne Einschränkung der Ergebnismenge auf starke Regeln, so degradiert die Suche zur sequenziellen Suche mit Ansatz von Apriori auf jeder Hierarchiestufe mit stufenspezifischen Minsup- und Minconf-Parameterwerten. Betrachtet man die Laufzeit, so kann man feststellen, dass bei gegebenen Daten die gezielte Suche nach starken Regeln nur unwesentlich die Laufzeit beeinflusst. Der Unterschied in der Laufzeit entsteht nicht bei der Regelerzeugung mit Apriori, sondern bei der anschließender Weiterverarbeitung und Interpretation der Regeln. Im Vergleich zur Laufzeit der Suche nach Cross-Level-Regeln mit gleichen Parameterwerten ist die Laufzeit der Suche mit Top-Down-Suche nach Multiple-Level-Regeln deutlich kleiner, da wie bereits erklärt, keine Itemsets aus Items verschiedener Hierarchiestufen erzeugt werden. Das macht die Top-Down-Suche vorteilhafter.

Während der Implementierung des Top-Down-Ansatzes ist eine neue Idee entstanden (und direkt implementiert worden), gezielt nach solchen Regeln zu suchen, deren Head- und Body-Seiten die Nachkommen unterschiedlicher Vorfahren sind. Die Ergebnisse dieser Suche sind, wie bereits erwähnt, im ersten Teil der Tabelle 12 dargestellt, um eine bessere Übersicht und Vergleichsmöglichkeit zu anderen Ergebnissen zu haben. Die Idee selbst ist aber nicht nur für den Top-Down-Ansatz spezifisch, sondern kann auch bei den Cross-Level-Regeln angewendet werden und wird jetzt diskutiert.

5.11.3.3. Entdeckung der Regeln, die unterschiedliche Hierarchiezweigen verbinden

Unabhängig von dem angewendeten Verfahren, sei es das von Agrawal und Srikant oder von Han und Fu, werden Regeln gefunden, die Elemente auf einer und derselben Hierarchiestufe verbinden. Bei dem Top-Down-Ansatz von Han und Fu werden zwar nur solche gesucht, aber bei der Crosslevel-Suche werden diese ja unter vielen anderen auch entdeckt, weil sie ja einen Speziellfall der Cross-Level-Regeln darstellen. Wenn man aber direkt nach solchen Regeln sucht, wird es nah liegend, eine Überprüfung der Vorfahren von Elementen der jeweils gefundenen Regel zu machen. Sind die Vorfahren der Elemente gleich, kann man sagen, dass die Regel einfach die meist vorkommenden Elemente eines einzigen Hierarchiezweiges abbildet, oder die Elemente, die die meiste Auslastung des Zweiges verursachen (s. Kapitel 5.7). Im Fall, dass die Vorfahren unterschiedlich sind, bedeutet eine solche Regel etwas anderes: sie verbindet unterschiedliche Hierarchiezweige und bildet deren Zusammenhang ab.

Dadurch gewinnt man z.B. im Falle der vorhandenen Daten eine bessere Übersicht über die Zusammenhänge zwischen verschiedenen Produktgruppen, die vielleicht für einen Manager sogar die interessantesten sind. Außerdem können bei solcher speziellen Suche die ungünstig, zu fein order gar falsch gewählten Gruppierungen entdeckt werden, weil z.B. die Produktgruppen der auf beiden Seiten stehenden Artikel direkt mit einander verglichen und deren Bildung beurteilt werden können. Sieht man diesen Ansatz als eine Filterungsmöglichkeit an, so wird es aus den in der Tabelle 12 dargstellten Ergebnissen ersichtlich, dass es tatsächlich viele Regeln sich damit ausfiltern lassen. Z.B. wurden bei den Experimenten mit dieser Einschränkung 93-97% der Regeln ausgefiltert (Zeilen 5, 6 der Tabelle 12), wohingegen ohne diese Einschränkung mit gleichen Parameterwerten nur 4-5% der Regeln wegfielen (Zeilen 11, 12 der Tabelle 12).

Es ist nicht notwendig, dieses Kriterium der Regelanalyse auch bei dem Crosslevel-Ansatz zu integrieren bzw. zu untersuchen, weil die vom Crosslevel-Ansatz entdeckten Regeln, die Elemente aus den gleichen Hierarchiestufen verbinden genau dieselben Regeln sind, die von Top-Down-Ansatz gefunden werden. Es könnten aber noch zwei spezielle Fälle beim Crosslevel-Ansatz eintreten, wenn man dort das Kriterium der unterschiedlichen Vorfahren der Seiten anwenden würde:

Zum einen könnte(n) im Body Element(e) aus einer(mehreren) höheren Stufe stehen, das (die) nicht ein direkter oder indirekter Vorfahre des (der) Elemente aus dem Head ist(sind). Das würde eine robuste Regel ausmachen, deren Entdeckung bereits untersucht wurde (s. Kapitel 5.11.3.1)

Zum anderen kann der genau umgekehrter Fall auftreten: im Body stehen Elemente aus einer (mehreren) niedrigerer(n) Stufe(n) als Element(e) im Head und kein Element aus dem Head ist ein Vorfahre der Elemente im Body.

Dieser Fall kann nur bei dem Crosslevel-Ansatz entdeckt werden und nicht von der Top-Down-Suche, wird aber nicht speziell weiter behandelt, da er keine besondere Bedeutung hat und ein Gegenteil einer robusten Regel bedeutet, außerdem tritt er bei den vorhandenen Daten nicht auf. Das lässt sich durch mangelnde Confidence solcher Regelnerklären.

5.11.3.4. Entdeckung der Regeln mit geografischer Bedeutung

Die Experimente, bei denen die Kundenhierarchie, insbesondere Kundengruppen involviert sind, zeigen, dass man durchaus anhand der Regeln eine grobe Vorstellung bilden kann, was in welchen Regionen gekauft wird, da die Kundengruppen ja nach PLZ gebildet wurden[42]. So bekommen die erzeugten Regeln eine geografisch-ökonomische Bedeutung.

Auf der Abbildung 25 sind einige Beispiele solcher Regeln.

Betrachtet z.B. ein Manager die Regel [Tragarme (produktgruppe), Kontaktkleber (produktgruppe)]⇒[PLZ-D 52***], sieht er sofort, dass im Gebiet der PLZ 52000 die Kunden überwiegend Artikel aus Produktgruppen „Tragarme“ und „Kontaktkleber“ kaufen. Wie oft und wie genau, kann er dann schon aus den numerischen Werten ablesen, falls es ihn interessiert. Untersucht er beispielsweise die Regel [Druckluftschläuche (produktgruppe)]⇒ [PLZ-S 59432] , so wird es für ihn deutlich, dass die Artikel aus der Produktgruppe „Druckluftschläuche“ überwiegend von Kunden aus Schweden gekauft werden.

Nr. 1	[Tragarme (produktgruppe) , Kontaktkleber (produktgruppe)] ➡ [PLZ-D 52***]	29	0,00106492	0,96666700	0,00104078	44,10163623
Nr. 2	[Steckschlüssel-Sätze 1/2" (produktgruppe) , Steckschlüssel-Sätze 1/4" (produktgruppe) , PLZ-S 55112] ➡ [Steckschlüssel-Sätze 1/4"-1/2" (produktgruppe)]	32	0,00117509	0,88888900	0,00117313	600,98206897
Nr. 14	[Steckschlüssel-Sätze 1/4" (produktgruppe) , Steckschlüssel-Sätze 1/4"-1/2" (produktgruppe) , PLZ-S 55112] ➡ [Steckschlüssel-Sätze 1/2" (produktgruppe)]	32	0,00117509	0,72727300	0,00117396	1.037,40952381
Nr. 15	[Maschinengewindebohrer (produktgruppe) , PLZ-S 57723] ➡ [Spiralbohrer (produktgruppe)]	34	0,00124853	0,72340400	0,00124405	278,88192771
Nr. 16	[Seitenschneider (produktgruppe) , PLZ-S 55112] ➡ [Kombizangen (produktgruppe)]	31	0,00113837	0,72093000	0,00113467	307,65014577
Nr. 186	[Meissel (produktgruppe) , PLZ-D 51***] ➡ [Rohrzangen (produktgruppe)]	28	0,00102820	0,45161300	0,00102277	189,39294585
Nr. 187	[PLZ-D 93***, Holzschrauben (produktgruppe)] ➡ [Schrauben metrisches Gewinde (produktgruppe)]	51	0,00187280	0,45132700	0,00187162	1.596,35862069
Nr. 188	[PLZ-D 39***, Sicherheitshalbschuhe (produktgruppe)] ➡ [Atemschutz (produktgruppe)]	32	0,00117509	0,45070400	0,00116952	210,94747035
Nr. 189	[Druckluftschläuche (produktgruppe)] ➡ [PLZ-S 59432]	32	0,00117509	0,45070400	0,00110505	16,77751251
Nr. 190	[PLZ-S 16858, Atemschutz (produktgruppe)] ➡ [Schutzhandschuhe (produktgruppe)]	31	0,00113837	0,44927500	0,00113688	766,05444646
Nr. 191	[Zylindersägen (produktgruppe) , PLZ-D 51***] ➡ [Stichsägeblätter (produktgruppe)]	30	0,00110165	0,44776100	0,00109575	186,94736842

Abbildung 25 Beispiele der Regeln mit der geografisch-ökonomischen Bedeutung

Die Untersuchung der geographischen Zusammenhänge kann man durch eine Erweiterung der Eigenschaft „robust“ noch weiter verfeinern, in dem man z.B. solche Regeln sucht, die die Form haben:

„Kunden aus der PLZ-Gruppe-XY, Warengruppe-W⇒Kunde Mustermann, Artikel A“

Solche Regel würde eine Spezialisierung vom Body im Kopf der Regel bedeuten, und gleichzeitig die Informationen sowohl aus der Artikel- als auch aus der Kundentaxonomie beinhalten würde.

[42] Die Aufteilung nach der PLZ gilt nur bei den deutschen Kunden, bei den ausländischen Kunden wurden die Gruppen nach dem Land gebildet. Diese Tatsache ändert aber nicht das Prinzip der regionalen Aufteilung der Kunden in Gruppen.

5.11.3.5. Weiterführende Funktionalitäten

Für die bessere Analyse der Ergebnisse soll es möglich sein, bestimmte Details der Regelelemente anzusehen. In der Implementierung wurde vorgesehen, dass jedes Element in jeder Regel angeklickt werden kann, damit z.B. der Manager die Informationen bekommt, die sich dahinter verbergen. So kann z.B. eine PLZ-Gruppe in einer Regel angeklickt werden. Man bekommt dabei die Informationen über die Gruppe in einem neuen Fenster angezeigt, die aus Gruppen-Nummer, Land, Anzahl der in der Gruppe enthaltenen Kunden (Mitglieder) sowie deren Kundennummer, PLZ und Stadt bestehen (s. Abbildung 26). Anhand dieser Informationen kann der Manager bestimmte Rückschlüsse ziehen. Ferner ist es möglich, beim Anklicken einer Mitgliedsnummer die Informationen über den Kunden zu bekommen (z. Z. o. Abbildung). Wird ein Element in einer Regel angeklickt, das ein Artikel ist, so bekommt man die Detailinformationen zu diesem Artikel angezeigt (s. Abbildung 5 im Kapitel 2.2 über die Systembeschreibung). Ist das angeklickte Element eine Produktgruppe, so werden sämtliche Artikel aus dieser Gruppe in einer Liste angezeigt (s. Abbildung 4 im Kapitel 2.2). Trifft man eine Warengruppe, bekommt man die Liste aller Produktgruppen, die zu dieser Warengruppe gehören zu sehen (s. Abbildung 3 im Kapitel 2.2).

Mit Hilfe dieser Querverweise hat man sofort den Überblick, was womit in jeweiliger Regel verbunden ist und bekommt die genaueren Informationen zu jedem Element mit einem Klick angezeigt.

Details der PLZ-Gruppe Nr. 103		
Land		de
Anzahl der Mitglieder in der Gruppe		4
Anzahl der Transaktionen in der Gruppe		7132
Mitgliedsnummer	Stadt	PLZ
9880	Friedberg	61150
7818	Friedrichsdorf	61381
7819	Friedrichsdorf	61381
7009	Oberursel	61440

Abbildung 26 Beispiel der Details einer PLZ-Gruppe

Was kann in der nächsten Zukunft noch implementiert und in das System integriert werden? Weiterhin sollen zusätzliche Suchoptionen in der Suchmaske eingebaut werden, eine präzisere Eingrenzung des Suchraumes ermöglichen sollen. Z.B. soll ein Maximum-Support-Wert für die Elemente eingegeben werden können, um das im Kapitel 5.9.1 angesprochene Problem zu adressieren. Eine weitere Option soll dem Benutzer eine Eingabe der bestimmten Produkt- oder Warengruppen erlauben, innerhalb deren gesucht werden soll. Es soll ebenso möglich sein, bestimmte Gruppen auszuschließen. Außerdem soll es möglich sein, den Zeitraum der Transaktionen einzugrenzen. Diese Optionen werden in der nächsten Zeit implementiert. Alternativ zu der Regelanzeige soll die Liste der häufigen Itemsets angezeigt werden können. Die erzeugten Regeln bzw. Itemsets könnten in der Liste selektiert und für weitere Verarbeitung gespeichert werden. Z.B. könnten anhand dieser Informationen die Vorschläge für die Kunden gemacht werden, die beispielsweise als News Letter verschickt oder bei der Produktsuche angezeigt werden könnten.

6. Zusammenfassung und Ausblick

Zusammenfassend wird hier ein Überblick über den Inhalt und die Ergebnisse der Arbeit gegeben.

In dieser Arbeit wurde versucht, die methodischen Werkzeuge zu untersuchen, die zur Analyse der Bestelldaten eines elektronischen Bestellsystems im Hinblick auf Taxonomien verwendet werden können. Dafür wurde zunächst eine theoretische Basis vorbereitet, in der einige Begriffe eingeführt und erklärt wurden. Da es bei der Analyse schwerpunktmäßig um die Entdeckung von häufigen Itemsets und Assoziationsregeln ging, die unter Berücksichtigung vorhandener Taxonomien erzeugt und untersucht werden sollten, wurden verschiedene Arten der generalisierten Regeln eingeführt und verglichen. Unter anderem wurde der Algorithmus „Apriori“ vorgestellt, der für die Entdeckung der Assoziationsregeln und insbesondere der generalisierten Regeln in einer leicht modifizierten Form eingesetzt wurde. Dieser Algorithmus eignete sich am besten für die Verarbeitung der vorhandenen Daten und ist zudem der meistangewendete Algorithmus in der Analyse der Bestelldaten. Aufbauend auf dieser Basis konnten die unterschiedlichen Ansätze zur Entdeckung von generalisierten Regeln und deren Evaluierung, die in der Literatur gefunden wurden, beschrieben und diskutiert werden. Die Aspekte der möglichen Generalisierungen im Hinblick auf mehrere parallel existierende Taxonomien wurden erläutert.

Die meisten der besprochenen Ansätze wurden in praktischen Experimenten untersucht und ausgewertet, darunter:

- die Ansätze zur Entdeckung der generalisierten Cross- und Multiple-Level-Assoziationsregeln
- die Untersuchung der Regelentdeckung mit Verwendung mehrerer Hierarchien
- die Filterung der entdeckten Regeln mit unterschiedlichen Parametern
- die gezielte Suche nach Regeln mit speziellen Eigenschaften wie „robust“ oder „stark“
- die gezielte Suche nach Regeln, die unterschiedliche Hierarchiezweige verbinden
- die Bewertung einer Generalisierung mit einer speziellen Metrik und die Anwendung von Metapatterns

Ferner wurde ein erfolgreicher Versuch gemacht, ein Verfahren zu entwickeln, das mit Hilfe von gefundenen häufigen Itemsets eine automatische Überprüfung von vorhandenen oder eine Bildung von neuen Hierarchien ermöglicht. Im Vergleich zu den konventionellen häufigen Itemsets, die vom „Apriori“ entdeckt werden und als Eingabedaten für das Verfahren dienten, waren die Ergebnisse dieses neuen Verfahrens für die Neugruppierungszwecke besser geeignet, da die mit diesem Verfahren gebildeten Gruppen eine größere Anzahl von Elementen aufwiesen, während die Anzahl dieser Gruppen selbst verkleinert werden konnte. Außerdem war der semantische Zusammenhang der Elemente in diesen Gruppen sichtbar.

Die Experimente wurden zunächst außerhalb des Bestellsystems durchgeführt. Danach wurden die implementierten Ansätze zur Suche, Filterung, Gruppierung und Evaluierung der generalisierten Regeln, die für praktische Zwecke gut geeignet waren, prototypisch in das Bestellsystem integriert.

Eingehend auf die in der Einleitung gestellten Fragen lassen sich folgende Ergebnisse festhalten:

1. Es werden in der Literatur zwei wesentliche Arten der generalisierten Assoziationsregeln diskutiert: zum einen sind es die Multiple-Level-Regeln, zum anderen die Cross-Level-Regeln. Diese unterscheiden sich auch in der Art ihrer Entdeckung. Es wurden zwei grundlegende Ansätze auf dem Gebiet der taxonomiebasierten Assoziationsregeln diskutiert, die in dieser Arbeit ausführlich untersucht wurden: bei den Cross-Level-Regeln werden nach dem Verfahren von Agrawal und Srikant alle Hierarchieebenen direkt in einem Suchvorgang in die Berechnung mit einbezogen, bei den Multiple-Level-Regeln wird ein Top-Down-Ansatz von Han und Fu verfolgt, bei dem Suche sequenziell, aber auf jedem Hierarchielevel getrennt, abläuft. Dabei ist die Anzahl der auf den gleichen Daten und mit den gleichen Parametern entdeckten Regeln bei dem Top-Down-Ansatz immer kleiner als bei dem Cross-Level-Ansatz. Weiterhin können die Regeln auf bestimmte Eigenschaften untersucht werden. Z. B. können unter den Multiple-Level-Regeln die „starken“ oder unter den Cross-Level-Regeln die„robusten“ Regeln gesucht werden.

2. Dadurch, dass bei der Top-Down-Suche nach starken Regeln auf einer bestimmten Hierarchiestufe die Regeln nur dann stark sind, wenn auf den höheren Stufen ihre Vorfahren auch stark sind, werden bei diesem Verfahren im Vergleich zur Cross-Level-Suche insgesamt wesentlich weniger Regeln entdeckt. Bei vorliegenden Daten unterschied sich die Anzahl der nach Han und Fu - Verfahren entdeckten Regeln um einen Faktor 20 von der Anzahl der Regeln, die mit Agrawal und Srikant- Verfahren gefunden wurden.

3. Die Eigenschaften „stark“ und „robust“ lassen sich ebenso wie die Regeln, die unterschiedliche Hierarchiezweige verbinden, nur mit Hilfe von Taxonomien einführen.

4. Es werden unterschiedliche Algorithmen verwendet, die aber meistens auf dem Apriori aufbauen. Die Bewertung der Interesse bzw. Filterung der Regel kann mit verschiedenen Kriterien erfolgen. Z.B. können zu diesen Zwecken Leverage und Lift verwendet werden. Eine Filterung mit Lift erschien sinnvoller: dadurch, dass der Lift im Gegensatz zur Leverage mit einem Quotient der Erwartungswerte zu den reellen Werten und nicht mit einer Differenz, wie die Leverage, gebildet wird, wird die Korrelationen zwischen Body und Head stärker hervorgehoben.

5. Die Taxonomien bringen den Vorteil, dass man auf einem abstrakteren Niveau die Zusammenhänge betrachten kann. Weiterhin bieten die Taxonomien die Möglichkeit, Regeln mit Verwendung von höheren Min-Support-Werten zu entdecken, die auf der untersten Ebene nicht möglich wären. Wurden es beispielsweise mit minimum-Support von 0,1% und minimum-Confidence von 40% auf den untersuchten 27000 Transaktionen nur 20 konventionelle Regeln entdeckt, so waren es 35000 bei dem Ansatz von Cross-Level-Regeln. Ein Nachteil dabei ist, dass eventuell zu viele oder uninteressante Regeln gefunden werden. Diesem Nachteil kann man mit Einführung von stufenspezifischen Parameterwerten entgegen wirken und anstatt Cross-Level-Suche die Top-Down-Suche anwenden.

6. Bei Involvierung einer zusätzlichen Taxonomie zu der Artikeltaxonomie, z. B. der Kundentaxonomie, bekommen die Regeln eine andere Bedeutung und werden mit zusätzlichen Informationen erweitert. Bei Verwendung der Gruppierung nach verschiedenen Kombinationen der Stufen aus zwei Taxonomien ändert sich die Semantik der Regeln und sie werden unter einem anderen Blickwinkel betrachtet.

7. Auf der Basis von den häufigen Itemsets, die mit Apriori gebildet werden, oder aber auf eine andere Weise erstellten Gruppen von Elementen, die unter einander eine gewisse Ähnlichkeit aufweisen, lässt sich eine Hierarchie neu bilden. Das kann als eine kostengünstigere Alternative zur manuellen Gruppierung der Elemente in der Praxis angewandt werden.

8. Eine Integration in ein bestehendes elektronisches Bestellsystem ist gut gelungen. Mit dem Einsatz von generalisierten und konventionellen Assoziationsregeln zu Management- und Marketingzwecken können die Angebotsgestaltung und die Sortimentanalyse effizienter gemacht werden. Auf der anderen Seite konnten bereits durch die Integration in das System die Methoden und die Vorgehensweisen der Analyse weiterentwickelt und verfeinert werden.

Was kann noch in der Zukunft gemacht werden? Welche Erweiterungen der Arbeit sind denkbar?

Beispielsweise kann in der Regelbewertung noch das Maß von Agrawal und Srikant implementiert und in das System integriert werden.

Das Problem der ungleichmäßigen Supportverteilung kann eventuell durch eine lauffähige Implementierung von Multiple-Itemset-Support-Apriori, was kurz angesprochen wurde, adressiert werden.

Bei der Bildung der Kundentaxonomie kann man ein anderes Gruppierungsattribut verwenden, wie z.B. „Jahresumsatz eines Kunden“. Dies wird den Regeln eine andere Semantik verleihen. Ebenso könnte die Artikeltaxonomie mit anderen Attributen gebildet werden, um die Wirkung der Taxonomie zu verstärken. Die Ergebnisse sollten danach mit den Ergebnissen der jetzigen Gruppierungen verglichen werden.

Nach erfolgreicher Integration von Analysemethoden in das Bestellsystem kann ein Mechanismus entwickelt werden, der auf der Basis von Analyseergebnissen Vorschläge generieren sollte. Die automatisch generierten Vorschläge könnten dem Anwender bei der Produktsuche eingeblendet oder an ihn als ein Newsletter verschickt werden. Nach einer längeren Einsatzzeit sollte eine Evaluierung der erzielten Effizienzsteigerung des Systems folgen.

Abbildungsverzeichnis

Tabellenverzeichnis

Literaturverzeichnis

[Agrawal et al., 1993] Agrawal, R., Imielinski, T., und Swami (1993), A. Mining Association Rules between Sets of Items in Large Databases. *In Proceedings of the ACM SIGMOD Conference on Management of Data,* Washington, D. C., USA.
URL: http://citeseer.ist.psu.edu/agrawal93mining.html, Verfügbar am 01.05.2005

[Agrawal und Srikant, 1994] Rakesh Agrawal and Ramakrishnan Srikant (1994), Fast Algorithms for Mining Association Rules. *In Proceedings of the VLDB Conference,* Santiago, Chile.
URL: http://citeseer.ist.psu.edu/agrawal94fast.html, Verfügbar am 01.05.2005

[Agrawal und Srikant, 1995] Rakesh Agrawal and Ramakrishnan Srikant (1995), Mining Generalized Association Rules. *In Proc. of the 21st Int'l Conference on Very Large Databases*, Zurich, Schweiz.
URL: http://citeseer.ist.psu.edu/srikant95mining.html, Verfügbar am 01.05.2005

[Amir, 2000] Amir Michail (2000), Data Mining Library Reuse Patterns using Generalized Association Rules. *In Proceedings of the 22nd international conference on Software engineering*, Limerick, Irland.
URL: http://www.cse.unsw.edu.au/~amichail/codeweb/icse2000.pdf, Verfügbar am 01.05.2005

[Bayardo, Agrawal] Robert J. Bayardo Jr. and Rakesh Agrawal (1999), Mining the most interesting rules. *In Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery in Databases & Data Mining (KDD99)*.
URL: http://www.almaden.ibm.com/software/quest/Publications/papers/kdd99.pdf, Verfügbar am 01.05.2005

[Bodon, 2003] Ferenc Bodon (2003), A fast APRIORI implementation, *In Workshop on Frequent Itemset Mining Implementations (FIMI'03),* Melbourne, Florida, USA.
URL: http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS//Vol-90/bodon.pdf, Verfügbar am 01.05.2005

[Borgelt, 2003] Christian Borgelt (2003), Efficient implementations of apriori and eclat. *In Bart Goethals and Mohammed J. Zaki, editors, Proceedings of the IEEE ICDM Workshop on Frequent Itemset Mining Implementations,* Melbourne, Florida, USA.
URL: http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS//Vol-90/borgelt.pdf, Verfügbar am 01.05.2005

[Borgelt, Kruse, 2002] Christian Borgelt and Rudolf Kruse (2002), Induction of Association Rules: Apriori Implementation, University of Magdeburg, *In 15th Conference on Computational Statistics (Compstat 2002),* Heidelberg, Germany, Physica Verlag.

URL: http://fuzzy.cs.uni-magdeburg.de/~borgelt/papers/cstat_02.pdf, Verfügbar am 01.05.2005

[Brin, Motwani, Silverstein, 1997] Sergey Brin, Rajeev Motwani, and Craig Silverstein (May 1997), Beyond market baskets: Generalizing association rules to correlations. *In SIGMOD 1997, Proceedings ACM SIGMOD International Conference on Management of Data,* Tucson, Arizona, USA.
URL: http://citeseer.ist.psu.edu/brin97beyond.html, Verfügbar am 01.05.2005

[Fung et.al. 2003] Benjamin C.M. Fung, Ke Wangy, Martin Esterz (2003), Hierarchical Document Clustering Using Frequent Itemsets. *In Proceedings of the SIAM International Conference on Data Mining*, San Francisco, California, USA.
URL: http://www.cs.sfu.ca/~ester/papers/FWE03Camera.pdf, Verfügbar am 01.05.2005

[Goethals, 2003] Bart Goethals (2003), Survey on Frequent Pattern Mining, HIIT Basic Research Unit, Department of Computer Science, University of Helsinki, Finland.
URL: http://www.adrem.ua.ac.be/~goethals/publications/survey.pdf, Verfügbar am 01.05.2005

[Görz, Rollinger, Schneeberger, 2000] G. Görz, C. Rollinger, J. Schneeberger, (Herausgeber) (2000), Einführung in die Künstliche Intelligenz.

[Han und Fu, 1999] Jiawei Han und Yongjian Fu (1999), Mining Multiple-Level Association Rules in Large Databases. *In IEEE's Transactions of Knowledge and Data Engineering, Vol.11, N.5*
URL: ftp://ftp.fas.sfu.ca/pub/cs/han/kdd/tkde99.pdf, Verfügbar am 01.05.2005

[Hu, 2003] Ya-Han Hu (2003), An efficient algorithm for discovering and maintenance of frequent patterns with multiple minimum supports, *Master Thesis*, Department of Information Management, National Central University, Taiwan.
URL: http://www.im.ncu.edu.tw/resource/papers/master/92m90423001.pdf, Verfügbar am 04.11.2004 (z. Z. nicht verfügbar)

[Klementitinen et. al., 1996] Mika Klementtinen, Heikki Mannila and Hannu Toivonen (1996), Interaktive Exploration of Discovered Knowledge: A Methodologie for Interaktion, and Usability Studies, Helsinki.

[Li und Sweeney, 2004] Yiheng Li, Latannya Sweeney (2004), Learnig Samantically Robust Rules from Data, School of Computer Science, Tech Report, CMU ISRI 05-101. Pittsburgh, Pennsylvania, USA.
URL: http://reports-archive.adm.cs.cmu.edu/anon/cald/CMU-CALD-04-100.pdf, Verfügbar 01.05.2005

[Liu et. al, 1999] Bing Liu, Wynne Hsu and Yiming Ma (1999), Mining Association Rules with Multiple Minimum Supports, *In ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD-99)*, San Diego, CA, USA.
URL: http://www.cs.uvm.edu/~xwu/kdd/Association-4.pdf, Verfügbar am 01.05.2005

[Lu et al., 2000] Hongjun Lu, Yuet Yeung Ng Zengping Tian (2000), T-Tree or B-Tree: Main Memory Database Index Structure Revisited.
URL: http://citeseer.ist.psu.edu/447405.html, Verfügbar am 01.05.2005

[Microsoft Encarta, 2005] Microsoft ® Encarta ® Enzyklopädie 2005 © Software, 1993-2004 Microsoft Corporation.

[Morik, 1999] Katharina Morik (1999), Maschinelles Lernen, Skript zur gleichnamigen Vorlesung, 3 Auflage, Universität Dortmund.

[Morik et al., 1993] Morik, K., Wrobel, S., Kietz, J.-U., und Emde, W. (1993), Knowledge Acquisition and Machine Learning - Theory, Methods, and Applications, Academic Press, London.

[Piatetsky-Shapiro 91] G.Piatetsky-Shapiro (l991), Discovery, analysis, and presentation of strong rules, *In G. Piatetsky-Shapiro and W.J. Frawley, Knowlage Discovery in Databases*, AAAI/MIT Press, Menlo Park, CA, USA.

[Psaila und Lanzi, 2000] G. Psaila, P. L. Lanzi (2000), Hierarchy-based Mining of Association Rules in Data Warehouses, *In Proceedings of the 2000 ACM symposium on Applied computing*, Como, Italien
URL: http://portal.acm.org/citation.cfm?id=335773, Verfügbar am 01.05.2005

[Tao, Murtagh, Farid, 2003] Feng Tao, Fionn Murtagh, and Mohsen Farid (2003), Weighted association rule mining using weighted support and significance framework. *In Proceedings of The Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (ACM SIGKDD 2003)*, Washington, DC, ACM Press.
URL: http://eprints.ecs.soton.ac.uk/7986/01/331.tao.pdf, Verfügbar am 01.05.2005

[Webb und Zhang, 2003] Geofrey I. Webb, Songmao Zhang (2003), Beyond Association Rules: Generalized Rule Discovery.
URL: http://www.csse.monash.edu.au/~webb/Files/WebbZhang03.pdf, Verfügbar am 01.05.2005

[Wikipedia] Online Enzyklopädie "Wikipedia",
URL: http://de.wikipedia.org, Verfügbar am 01.05.2005