Wissensentdeckung in Datenbanken mit Assoziationsregeln

Verfahren zur effizienten Regelgenerierung
und deren Integration in den Wissensentdeckungsprozeß

Dissertation

der Fakultät für Informations- und Kognitionswissenschaften
der Eberhard-Karls-Universität Tübingen
zur Erlangung des Grades eines
Doktors der Naturwissenschaften
(Dr. rer. nat.)

vorgelegt von

Dipl.-Inform. Jochen Hipp

aus Bad Soden

Tübingen
2003

Tag der mündlichen Qualifikation:	17.12.2003
Dekan:	Prof. Dr. Martin Hautzinger
1. Berichterstatter:	Prof. Dr. Ulrich Güntzer
2. Berichterstatter:	Prof. Dr. Gholamreza Nakhaeizadeh (DaimlerChrysler AG, Research & Technology, Ulm)

Zum Geleit

Wir sehen uns heute einer fast allgegenwärtigen Digitalisierung unserer Welt gegenüber. Der gleichzeitig zu beobachtende Trend der letzten Jahre, neben operativer Datenhaltung auch gezielt Informationssysteme als sogenannte Data Warehouses zu Analysezwecken zentralisiert aufzubauen, rückt die rechnergestützte Analyse großer Datenbestände verstärkt in das Interesse von Wirtschaft und Wissenschaft. Unter der Bezeichnung Wissensentdeckung in Datenbanken (Knowledge Discovery in Databases oder Data Mining) entwickelte sich vor diesem Hintergrund ein neues, interdisziplinäres Forschungsgebiet, das etablierte Disziplinen wie Mustererkennung, Maschinelles Lernen, Datenbanken, Statistik, Künstliche Intelligenz, Expertensysteme und Datenvisualisierung einbezieht.

Im Mittelpunkt der vorliegenden Arbeit steht mit der Generierung von Assoziationsregeln eine der grundlegenden Analysemethoden dieses neuen Forschungsgebiets. Assoziationsregeln modellieren Abhängigkeiten zwischen Ereignissen in transaktionsbasierten Datenbeständen. Die Einsatzmöglichkeiten sind vielfältig und reichen von der klassischen Warenkorbanalyse im Einzelhandel, über die Analyse der Nutzungsprofile von Internetseiten im World Wide Web, bis zu komplexen Anwendungen beispielsweise im Bereich der Garantie und Kulanz in der Automobilindustrie.

Die vorliegende Arbeit leistet einen wertvollen Beitrag zur Forschung durch die sorgfältige Systematisierung und Evaluierung der heute bekannten Verfahren, die Entwicklung neuer Algorithmen, welche durch wesentliche Effizienzsteigerungen der Analyse mittels Assoziationsregeln neue Anwendungen erschließen, und die Einbindung der Verfahren in einen Wissensentdeckungsprozeß. Wir wünschen der Arbeit eine gute Aufnahme und eine weite Verbreitung in Wissenschaft und Praxis.

Tübingen und Ulm,
im Dezember 2003

Ulrich Güntzer und
Gholamreza Nakhaeizadeh

Vorwort

Die vorliegende Arbeit entstand während meiner Tätigkeit als wissenschaftlicher Mitarbeiter am Wilhelm-Schickard-Institut für Informatik der Eberhard-Karls-Universität in Tübingen und am Forschungszentrum der DaimlerChrysler AG in Ulm.

Mein besonderer Dank gilt meinem Doktorvater Prof. Dr. Ulrich Güntzer für das stete Interesse an meiner Arbeit und die zahlreichen wertvollen Gespräche und Hinweise. Auch über die Dissertation hinaus hatte ich die Gelegenheit, sehr viel von Professor Güntzer zu lernen. Ich möchte mich außerdem bei Prof. Dr. Gholamreza Nakhaeizadeh bedanken, der mich ebenfalls über meine Arbeit hinaus mit vielen wichtigen Ratschlägen maßgeblich begleitete und sich bereit erklärt hat, das zweite Gutachten zu übernehmen.

Ebenso gilt mein Dank meinen Kollegen aus der Abteilung RIC/AM am Forschungszentrum in Ulm für die sehr gute Zusammenarbeit und Unterstützung, allen voran Daniel Sonntag, Christian Köpf, Udo Grimmer, Dr. Matthias Grabert, Dr. Kai Bartlmae und Dr. Rüdiger Wirth. Gleiches gilt für meine Kollegen am Wilhelm-Schickard-Institut, insbesondere für Andreas Ludwig, Dr. Jens Gramm und Bernd Heumesser, sowie für Christoph Mangold, der mich im Rahmen seiner Diplomarbeit unterstützte.

Erwähnen möchte ich auch Dr. Andreas Myka und Dr. Siegfried Bell, die mich durch die anspruchsvolle und motivierende Betreuung meiner Diplomarbeit darin bestärkten, mit der vorliegenden Arbeit zu beginnen.

Meinen Eltern danke ich für die unkomplizierte Ermöglichung meines Studiums und insbesondere meinem Vater dafür, daß er mein Interesse für die Informatik so früh weckte und meiner Mutter, daß dieses Interesse mich nicht zu einseitig bestimmte.

Ganz besonders möchte ich mich bei meiner Frau Christa Ginader bedanken. Ohne ihre inhaltlichen Hinweise, ihre sorgfältigen Korrekturen und ihre Geduld wäre die vorliegende Arbeit so nicht zustande gekommen.

Tübingen, im Dezember 2003 Jochen Hipp

Zusammenfassung

Die Datenanalyse mittels Assoziationsregeln ist eines der am häufigsten eingesetzten Data Mining-Verfahren und geht auf Arbeiten der Forschergruppe um Rakesh Agrawal am Forschungszentrum der IBM in Almaden, Kalifornien, USA, zurück. Dort wurden Anfang der neunziger Jahre Assoziationsregeln als Methode der Abhängigkeitsanalyse eingeführt und erste Algorithmen zur Assoziationsregelgenerierung entwickelt.

In der vorliegenden Arbeit werden die etablierten Verfahren zur Generierung von Assoziationsregeln analysiert und systematisiert, wodurch ein besseres Verständnis der in der Literatur bisher nicht im Zusammenhang dargestellten Verfahren möglich wird. In Verbindung mit einer umfassenden Evaluierung der Laufzeiten und des Speicherbedarfs führt dies zu einer Neubewertung der Ansätze.

Darauf aufbauend werden neue Verfahren zur Generierung von Assoziationsregeln abgeleitet. Diese beruhen auf einer optimierten Beschneidung des Suchraums, auf einem hybriden Vorgehen und auf der Einbeziehung einer eventuell vorhandenen Taxonomie. Im Rahmen einer Evaluierung erreichen die neu entwickelten Algorithmen in vielen Experimenten wesentlich kürzere Laufzeiten und einen geringeren Speicherbedarf als die bisherigen Algorithmen. Die vorgeschlagenen Verfahren sind insgesamt deutlich effizienter als die bisher bekannten Ansätze, insbesondere falls eine Taxonomie zu den Analysedaten zur Verfügung steht.

In Verbindung mit der Effizienz der Verfahren steht die Integration der Regelgenerierung in den Wissensentdeckungsprozeß. Ein iterativer und interaktiver Prozeß setzt kurze Antwortzeiten voraus, die von den Verfahren auf großen Datenmengen oft nicht erreicht werden können. Für diese von algorithmischen Aspekten in den Hintergrund gedrängte Problematik wird im Rahmen der vorliegenden Arbeit ein Regelcache als Lösung vorgeschlagen. Der Regelcache ist so aufgebaut, daß dieser auch für viele Anfragen gültig bleibt, die Selektionen der zugrundeliegenden Datensätze beinhalten, und dadurch für solche Anfragen nicht neu initialisiert werden muß.

Abstract

Data analysis using association rules belongs to the fundamental data mining approaches and was introduced as a method aiming at dependency analysis by Rakesh Agrawal at the IBM Research Center in Almaden, California, USA.

In this thesis, the established algorithms for association rule mining are analyzed and systemized. The chief goal is to learn more about the algorithms that thus far have not been described coherently. Together with the results of an exhaustive evaluation of runtime and memory usage, this leads to a changed appreciation of the different approaches.

On the basis of the results obtained, new algorithms for the generation of association rules are developed. These algorithms rely on an optimized pruning of the search space, a hybrid approach, and the incorporation of a potentially available taxonomy. In a multitude of experiments carried out during a comprehensive evaluation, the new algorithms achieved not only much shorter runtimes but also a greatly reduced memory usage as compared to established approaches. All in all, the algorithms introduced are much more efficient than conventional approaches, in particular when a taxonomy on the data is available.

Aligned with the efficiency of the algorithms is the aspect of integrating the rule generation into the process of knowledge discovery. An iterative and interactive process assumes short response times that cannot be reached by the algorithms on very huge datasets. For this often neglected problem, an extended rule cache is proposed. This rule cache stays valid even for many mining queries that include selections of the underlying data. Hence, for such queries, the cache does not need to be reinitialized.

Inhaltsverzeichnis

Kapitel 1

Einleitung

Die heutigen Möglichkeiten zur digitalen Erfassung und kostengünstigen Speicherung von Daten führen vielfach zu sehr umfangreichen Datenbeständen. Unternehmensdatenbanken erreichen nicht selten Größenordnungen von mehreren Terabyte, und in den Wissenschaften sind es unter anderem die Fortschritte im Bereich der Gentechnik, der Molekularwissenschaften, der Astronomie oder der Medizin, die immer größere Datenbestände entstehen lassen (vgl. u. a. [Fayyad et al., 1996c], [Hand et al., 2001], [Piatetsky-Shapiro, 2002b]). Auch die Vorgänge unseres täglichen Lebens erzeugen zunehmend digitalisierte Daten. Beispiele sind scannererfaßte Kassenbons im Einzelhandel, elektronische Kreditkartenabrechnungen, die Nutzung des Internets, die digitale Vermittlung von Telefongesprächen oder elektronische Maut- und Verkehrsleitsysteme.

Das Wachstum heutiger Datenbanken findet in zwei Dimensionen statt. Zum einen steigt die Anzahl der gespeicherten Datensätze, das heißt der Objekte aus der realen Welt, die erfaßt und abgelegt werden. Zum anderen wächst die Zahl der zu jedem Objekt gespeicherten Attribute. Die resultierenden Datensammlungen sind allerdings selten von direktem Nutzen. Zumeist läßt erst die Analyse der Daten, das heißt die Gewinnung von Wissen aus der Vielzahl erfaßter Objekte und Attributwerte, einen zusätzlichen Nutzen entstehen. Wissen wird in diesem Zusammenhang pragmatisch als Menge abgesicherter Erkenntnisse verstanden, die hinsichtlich einer bestimmten Domäne interessant und brauchbar sind (in Anlehnung u. a. an [Matheus et al., 1993, S. 905], [Wittmann, 1999, S. 5]).

Viele der Methoden aus dem Bereich der klassischen Statistik sind für die in heutigen Anwendungen immer häufiger erreichten Datenvolumina nicht ausgelegt (vgl. u. a. [Fayyad, 1998, S. 26f.], [Nakhaeizadeh et al., 1998, S. 2f.], [Wittmann, 1999, S. 10ff.]). Zugleich ermöglicht die in den letzten Jahren stark gestiegene Leistungsfähigkeit verfügbarer Rechnersysteme den Einsatz von Algorithmen, deren Entwicklung bisher aus Effizienzgründen nicht in Betracht gezogen wurde. Vor

diesem Hintergrund wächst seitens der Anwender und der Forschung das Interesse an neuen Ansätzen, welche die statistischen Methoden in bestimmten Bereichen ergänzen.

Data Mining als interdisziplinäres Forschungsgebiet hat die Entwicklung von Methoden zur effizienten Analyse großer Datenbanken zum Ziel und steht an der Schnittstelle etablierter Disziplinen wie Mustererkennung, Maschinelles Lernen, Datenbanken, Statistik, Künstliche Intelligenz, Expertensysteme und Datenvisualisierung. (Für einen allgemeinen Überblick siehe beispielsweise [Fayyad et al., 1996b], [Mannila, 1996], [Chen et al., 1996], [Nakhaeizadeh et al., 1998]). In [Hand et al., 2001, S. 1] wird Data Mining wie folgt definiert:

> *Data Mining* ist die Analyse meist größerer Datenmengen von beobachteten Attributwerten mit dem Ziel, bisher unbekannte Beziehungen zu finden und die Daten auf neue Art und Weise zusammenzufassen, so daß diese Zusammenfassung verständlich und zweckdienlich ist.[1]

Der englische Begriff „Data Mining“ läßt sich frei mit „Datenbergbau“ übersetzen. Demnach ist mit Data Mining die Vorstellung verbunden, während einer Analyse viel „taubes Gestein“ abtragen zu müssen, um an das in den Daten enthaltene wertvolle Wissen zu gelangen und dieses geeignet aufbereitet präsentieren zu können.

Die zusätzliche Schwierigkeit, in aus Sicht der Analyse überwiegend wertlosen Daten nach „verborgenen Schätzen“ zu suchen, ist unter anderem darin begründet, daß viele Datenbanken als Nebenprodukt elektronisch abgewickelter Prozesse entstehen. Die Aspekte der Analyse solcher operativer Datenbanken werden meist weder beim Datenbankdesign noch bei der späteren Datenerfassung ausreichend berücksichtigt. Im Einzelhandel erfolgt beispielsweise die Erfassung von Kundentransaktionen als Kassenbon primär zur Abrechnung, und die Analyse dieser Daten ist nur ein nachgelagerter Prozeß. Insbesondere der Zuschnitt auf Daten, die nicht unmittelbar zu Analysezwecken erhoben werden, unterscheidet Data Mining-Methoden von bisherigen Analyseansätzen, beispielsweise aus dem Maschinellen Lernen (vgl. u. a. [Holsheimer und Siebes, 1994, S. 16]). Eine Abgrenzung zum Gebiet der Statistik findet sich in [Smyth, 2000] oder [Klösgen und Zytkow, 2002, S. 637ff.].

Data Mining betrachtet vor allem die algorithmischen Aspekte der Datenanalyse. In praktischen Anwendungen zeigt sich, daß eine solche Sicht jedoch zu eng ist und eine erfolgreiche Analyse großer Datenbanken mehr als die Anwendung vielversprechender Algorithmen voraussetzt (vgl. u. a. [Brachman und Anand, 1996],

[1] Die gefundenen Beziehungen und Zusammenfassungen werden in der Literatur auch als Muster oder Modelle bezeichnet.

[Fayyad et al., 1996c]). Daher wird Data Mining als Teilschritt eines umfassenden Analyseprozesses betrachtet. Dieser sogenannte *Prozeß der Wissensentdeckung in Datenbanken (process of knowledge discovery in databases)* beinhaltet mehrere dem Data Mining sowohl vor- als auch nachgelagerte Phasen. Die Phasen reichen von der Analyse der Anwendungsdomäne, dem Verständnis, der Beschaffung und der Aufbereitung der Daten über das Data Mining bis hin zur Bewertung und Umsetzung der Ergebnisse. In praktischen Anwendungen sind es die dem Data Mining vorgelagerten Phasen, welche den größten Aufwand in einem Wissensentdeckungsprozeß verursachen (vgl. [Nakhaeizadeh et al., 1998, S. 20]).

Der Prozeß der Wissensentdeckung in Datenbanken wird allgemein als nicht trivial, iterativ und hochgradig interaktiv verstanden (vgl. z. B. [Fayyad et al., 1996a], [Brachman und Anand, 1996], [Wirth und Hipp, 2000]). Die Phasen der Wissensentdeckung werden nicht in einer festgelegten Reihenfolge durchlaufen, sondern der Analyst steuert den Prozeß anhand der Zwischenergebnisse. Typischerweise ist es erforderlich, Phasen mehrfach und in unterschiedlichen Kombinationen zu wiederholen, bis der Prozeß mit einem endgültigen Ergebnis beendet werden kann.

In einem solchen „human centered process“ trägt der Analyst mit seiner Kreativität und Kompetenz entscheidend zum Erfolg bei. Den Analysten geeignet zu unterstützen und damit eine der grundlegenden Voraussetzungen für die erfolgreiche Durchführung von Projekten im Rahmen der Wissensentdeckung zu schaffen, ist daher eine der wichtigsten Aufgaben der Forschung im Bereich der Wissensentdeckung in Datenbanken (vgl. auch [Brachman und Anand, 1996], [Nakhaeizadeh et al., 1998], [Wirth und Hipp, 2000]).

Assoziationsregeln (association rules) sind ein typisches Data Mining-Verfahren und gehen auf Arbeiten der Forschergruppe um Rakesh Agrawal am Forschungszentrum der IBM in Almaden, USA, zurück. Dort wurden Anfang der neunziger Jahre Assoziationsregeln als Methode der Abhängigkeitsanalyse eingeführt und erste Algorithmen zur Assoziationsregelgenerierung entwickelt (vgl. [Agrawal et al., 1993], [Agrawal und Srikant, 1994a]).

Die Datenanalyse mit Assoziationsregeln basiert auf Transaktionsdaten, das heißt auf Mengen von zu Transaktionen zusammengefaßten Ereignissen. Anschauliches Beispiel sind die Kundentransaktionen eines Einzelhandelsunternehmens. Die Artikelgruppen aus dem Warensortiment werden eindeutig mit jeweils einem Ereignis identifiziert. Als Transaktionen werden dann die Ereignismengen verstanden, die den von einem Kunden gleichzeitig gekauften Artikeln entsprechen.

Ausgehend von solchen Transaktionen modellieren Assoziationsregeln Abhängigkeiten zwischen Ereigniskombinationen bezüglich Gleichzeitigkeit. Beispielsweise kann eine Assoziationsregel aussagen, daß eine Transaktion, welche die Artikel a und b enthält, mit einer bestimmten Wahrscheinlichkeit, der sogenannten Konfidenz, ebenfalls den Artikel c enthält. Die Konfidenz einer Assoziationsregel ist in diesem Sinne eine bedingte Wahrscheinlichkeit.

Zu einer Menge von n Ereignissen existieren $3^n - 2^{n+1} + 1$ Assoziationsregeln (vgl. [Hipp et al., 2002a]). Bereits für kleine n in der Größenordnung von wenigen hundert Ereignissen würden für einen Analysten unüberschaubar viele Regeln erzeugt werden, die außerdem zum größten Teil, unter anderem aufgrund der geringen erreichten Konfidenzwerte, wenig aussagekräftig wären. Ziel der Verfahren zur Assoziationsregelgenerierung ist daher, vor dem Hintergrund typischer n in Größenordnungen von einigen zehntausend und mehr Ereignissen, effizient nur die Regeln zu erzeugen, welche vorgegebenen Schwellenwerten für bestimmte Gütemaße genügen.

Die Datenanalyse mittels Assoziationsregeln ist eines der am häufigsten eingesetzten Data Mining-Verfahren (vgl. etwa [Piatetsky-Shapiro, 2002b]). Die einfache Kommunizierbarkeit und unmittelbare Verständlichkeit haben zu der weiten Verbreitung von Assoziationsregeln in unterschiedlichen Anwendungsgebieten auch außerhalb der klassischen Warenkorbanalyse beigetragen. Für die Forschung im Bereich der Wissensentdeckung ist die Generierung von Assoziationsregeln ebenfalls von zentralem Interesse, wie die Vielzahl von Publikationen und Vorträgen auf wissenschaftlichen Tagungen belegt.

Gegenstand dieser Arbeit

Im Kontext der Wissensentdeckung in Datenbanken mit Assoziationsregeln sind insbesondere die Aspekte der effizienten Regelgenerierung und der Integration der Generierungsverfahren in einen Wissensentdeckungsprozeß als kritisch anzusehen.

In dieser Arbeit werden die etablierten Verfahren zur Generierung von Assoziationsregeln analysiert und systematisiert, wodurch ein besseres Verständnis der in der Literatur bisher nicht im Zusammenhang dargestellten Verfahren möglich wird. In Verbindung mit einer umfassenden Evaluierung der Laufzeiten und des Speicherbedarfs führt dies zu einer Neubewertung der Ansätze.

Darauf aufbauend werden neue Verfahren zur Generierung von Assoziationsregeln abgeleitet. Diese beruhen auf einer optimierten Beschneidung des Suchraums, auf einem hybriden Vorgehen und auf der Einbeziehung einer eventuell vorhandenen Taxonomie. Die entwickelten Algorithmen erreichen im Rahmen einer Evaluierung in vielen Experimenten wesentlich kürzere Laufzeiten und einen geringeren Speicherbedarf als die bisherigen Algorithmen. Die vorgeschlagenen Verfahren sind insgesamt deutlich effizienter als die bisher bekannten Ansätze, insbesondere falls eine Taxonomie zu den Analysedaten zur Verfügung steht.

In Verbindung mit der Effizienz der Verfahren steht die Integration der Regelgenerierung in den Wissensentdeckungsprozeß. Ein iterativer und interaktiver Prozeß setzt kurze Antwortzeiten voraus, die von den Verfahren für große Datenmengen oft nicht erreicht werden können. Für diese von algorithmischen Aspekten in den

Hintergrund gedrängte Problematik, wird im Rahmen der vorliegenden Arbeit ein Regelcache als Lösung vorgeschlagen. Ist der Regelcache initialisiert, können die während des Wissensentdeckungsprozesses typischen Analysefragen meist innerhalb weniger Sekunden beantwortet werden. Der Regelcache ist so aufgebaut, daß dieser auch für viele Anfragen gültig bleibt, die Selektionen der zugrundeliegenden Datensätze beinhalten, und dadurch für solche Anfragen nicht neu initialisiert werden muß. Außerdem wird ein Ansatz zur Integration der Verfahren mit relationalen Datenbanksystemen vorgestellt, der insbesondere im Hinblick auf den Transfer der Forschungsresultate in praktische Anwendungen von Bedeutung ist.

Aufbau dieser Arbeit

Im Anschluß an das einleitende Kapitel werden in Kapitel 2 die Grundlagen der Wissensentdeckung in Datenbanken mit Assoziationsregeln eingeführt. Anhand eines Prozeßmodells wird der Wissensentdeckungsprozeß dargestellt und auf die einzelnen Phasen der Wissensentdeckung näher eingegangen. Des weiteren wird ein Überblick über die wichtigsten Methoden der Wissensentdeckung in Datenbanken gegeben. In diesem Zusammenhang werden Assoziationsregeln als ein Verfahren der Abhängigkeitsanalyse eingeführt, formal definiert und an einem Beispiel veranschaulicht. Neben der Darstellung verschiedener Regelgütemaße und der Einführung von taxonomen Assoziationsregeln, das heißt von Regeln, welche die Berücksichtigung von Taxonomien erlauben, werden verschiedene weitere Regeltypen vorgestellt. Das Kapitel schließt mit der Beschreibung diverser Anwendungsszenarien für Assoziationsregeln, wobei die Ausreißererkennung mit Assoziationsregeln erstmals vorgestellt wird.

In Kapitel 3 werden die etablierten Verfahren zur Assoziationsregelgenerierung analysiert und evaluiert. Die Aufgabe der Assoziationsregelgenerierung wird dazu als sogenanntes Assoziationsproblem formalisiert. Es folgt die Darstellung der den Algorithmen zugrundeliegenden Ansätze für das Durchlaufen des Suchraums und der Häufigkeitsbestimmung von Ereigniskombinationen. Auf Basis dieser Grundlagen werden die Algorithmen selbst eingeführt und systematisiert. Das Kapitel endet mit einer Evaluierung der beschriebenen Verfahren anhand von Datensätzen, welche zum einen gezielt mit bestimmten Eigenschaften synthetisch erzeugt werden und zum anderen aus realen Analyseszenarien stammen.[2] Die Breite der einbezogenen Verfahren und die Erkenntnisse hinsichtlich des bisher in der Literatur kaum untersuchten Hauptspeicherbedarfs führen zu einer Neubewertung der etablierten Verfahren.

In Kapitel 4 werden verschiedene Optimierungen der bekannten Algorithmen und

[2]Die verwendeten Daten aus realen Anwendungen gehen auf verschiedene Projekte zurück, die am Forschungszentrum der DaimlerChrysler AG in Ulm von der Abteilung RIC/AM zusammen mit Anwendern aus den unterschiedlichsten Konzernbereichen durchgeführt wurden.

neue Ansätze zur Generierung von Assoziationsregeln vorgestellt. Grundlage dieser Weiterentwicklungen sind die Erkenntnisse aus der Analyse der etablierten Verfahren in Kapitel 3. Zunächst werden neue Ansätze zur Generierung einfacher Assoziationsregeln eingeführt. Zum einen basieren diese auf einer optimierten Suchstrategie, welche es bei einer Tiefensuche ermöglicht, den Suchraum anhand nichthäufiger Teilmengen von Ereigniskombinationen zu beschneiden. In verschiedenen Experimenten entfällt dadurch die Bestimmung der Häufigkeiten von bis zu 30% der Kandidaten. Zum anderen wird ein hybrider Ansatz vorgestellt, der durch Kombination verschiedener Verfahren weit kürzere Laufzeiten und einen geringeren Speicherbedarf als die zugrundeliegenden Algorithmen erreichen kann. Die beschriebenen Ansätze werden im Vergleich mit den etablierten Verfahren evaluiert. Darauf folgt die Einführung neuer Ansätze zur Generierung von Assoziationsregeln unter Einbeziehung von Taxonomien. Zunächst werden die etablierten Algorithmen erweitert, so daß diese Taxonomien in die Regelgenerierung einbeziehen können. Es schließt sich die Vorstellung neuer Verfahren an, welche die Taxonomie effizient zur Beschneidung des Suchraums heranziehen. Eine Evaluierung zeigt, daß diese Verfahren wesentlich kürzere Antwortzeiten erreichen als die erweiterten etablierten Algorithmen.

In Kapitel 5 werden zentrale Aspekte der Integration von Verfahren zur Assoziationsregelgenerierung in einen Wissensentdeckungsprozeß identifiziert und zu diesen Aspekten eigene Ansätze vorgestellt. Für den Datenzugriff wird die Anbindung der Verfahren an relationale Datenbanksysteme vorgeschlagen und anhand einer realen Anwendung werden verschiedene Anforderungen für den Datenzugriff abgeleitet. Der resultierende Ansatz wird zusammen mit kommerziellen Produkten evaluiert, wobei die hier vorgestellte Lösung komplexere Analyseanfragen bearbeiten kann und kürzere Laufzeiten erreicht. Des weiteren wird ein Regelcache eingeführt, der einen interaktiven Wissensentdeckungsprozeß trotz der unzureichenden Laufzeiten der Verfahren zur Assoziationsregelgenerierung erlaubt. Mit diesem Ansatz ist es möglich, aus dem initialisierten Cache einen Großteil der für einen Wissensentdeckungsprozeß typischen Anfragen ohne Rückgriff auf die Daten zu beantworten. Die Antwortzeiten sind unabhängig von der Anzahl der zugrundeliegenden Transaktionen und liegen in den durchgeführten Experimenten meist im Bereich weniger Sekunden. Es folgt die Einführung einer Anfragesprache für den Cache, welche über andere Data Mining-Sprachen hinausgeht, indem die zugrundeliegende Definition der Assoziationsregeln erweitert wird. Das Kapitel schließt mit einer Erweiterung des Regelcaches, so daß aus diesem auch Anfragen beantwortet werden können, die auf der Selektion von Datensätzen beruhen. Ohne auf die zugrundeliegenden Daten selbst zurückzugreifen, werden mit diesem Ansatz Regeln aus dem Cache abgeleitet, welche sonst durch einen erneuten Generierungslauf auf einer Teilmenge der Datensätze erzeugt werden müßten. Während eines Wissensentdeckungsprozesses

sind solche Fokussierungen auf Teilmengen der zugrundeliegenden Daten häufig. Die Möglichkeit, auf Selektionen beruhende Anfragen aus dem Cache beantworten zu können, ist daher grundlegend für den Einsatz des eingeführten Ansatzes zur interaktiven Wissensentdeckung.

Die Arbeit schließt in Kapitel 6 mit einer Zusammenfassung der Ergebnisse und einem Ausblick. Zur Vervollständigung sind im Anhang die Ergebnisse weiterer Experimente angeführt.

Kapitel 2

Wissensentdeckung mit Assoziationsregeln

In diesem Kapitel werden die Grundlagen der Wissensentdeckung in Datenbanken mit Assoziationsregeln eingeführt. Dazu folgt in Abschnitt 2.1 die Darstellung des Wissensentdeckungsprozesses anhand eines Prozeßmodells. Des weiteren wird ein Überblick über die wichtigsten Ziele der Wissensentdeckung in Datenbanken gegeben. In Abschnitt 2.2 werden Assoziationsregeln als ein Verfahren der Abhängigkeitsanalyse eingeführt, formal definiert sowie anhand eines Beispiels veranschaulicht. Neben der Darstellung verschiedener Regelgütemaße und der Berücksichtigung von Taxonomien enthält dieser Abschnitt die Beschreibung weiterer Regeltypen. Das Kapitel schließt in Abschnitt 2.3 mit der Einführung diverser Anwendungsszenarien, wobei die Ausreißererkennung mit Assoziationsregeln erstmals vorgestellt wird.

2.1 Wissensentdeckung in Datenbanken

Als Reaktion auf die Herausforderungen durch immer größer werdende Datenmengen wächst das Interesse an neuen Ansätzen zur Datenanalyse, die heute unter den Begriffen „Data Mining“ und „Wissensentdeckung in Datenbanken“ subsumiert werden. Während Data Mining vor allem die algorithmischen Aspekte der Datenanalyse in den Mittelpunkt stellt, umfaßt die Wissensentdeckung in Datenbanken auch die der eigentlichen Analyse vor- und nachgelagerten Aufgaben. Insbesondere wird die Wissensentdeckung in Datenbanken als ein Prozeß gesehen, der das Data Mining als einen von mehreren Teilschritten beinhaltet.

2.1.1 Wissensentdeckung als Prozeß

Die Wissensentdeckung in Datenbanken wird allgemein als nicht trivialer, iterativer und hochgradig interaktiver Prozeß verstanden (vgl. z. B. [Wrobel et al., 1996], [Fayyad et al., 1996c], [Brachman und Anand, 1996], [Johnson und Dasu, 1998], [Wirth und Hipp, 2000], [Hipp et al., 2002a]). Wissensentdeckung ist kein statisch zu durchlaufender Prozeß, sondern der Analyst interagiert mit den Daten sowie den Zwischenergebnissen und entscheidet aufgrund seiner Bewertungen während des gesamten Prozesses wiederholt darüber, wie fortzufahren ist. Typischerweise werden die einzelnen Phasen mehrfach und in unterschiedlichen Kombinationen durchlaufen, bis ein endgültiges Ergebnis feststeht, das heißt Wissen entdeckt oder die Wissensentdeckung als aussichtslos abgebrochen wird.

In Forschung und Praxis wurden verschiedene Prozeßmodelle für die Wissensentdeckung in Datenbanken ausgearbeitet (vgl. etwa [Fayyad et al., 1996a], [Fayyad et al., 1996c], [Brachman und Anand, 1996], [Adriaans und Zantinge, 1996], [Williams und Huang, 1996], [Berry und Linoff, 1997], [John, 1997], [Barth, 1998], [Krahl et al., 1998]). Im folgenden wird der Prozeß der Wissensentdeckung in Datenbanken, stellvertretend für die verschiedenen Prozeßmodelle, anhand von *CRISP-DM – **Cr**oss **I**ndustry **S**tandard **P**rocess Model for **D**ata **M**ining –* aus [Chapman et al., 2000], [Wirth und Hipp, 2000] veranschaulicht (vgl. auch [Hipp und Lindner, 1999], [Hipp et al., 2002a]). CRISP-DM hebt sich durch den Detailgrad und die erwiesene Praxistauglichkeit von den meisten anderen Prozeßmodellen ab (vgl. z. B. [Welte, 1999]). Nach einer Umfrage in [Piatetsky-Shapiro, 2002a] ist CRISP-DM zudem das zur Zeit am weitesten verbreitete Prozeßmodell.

CRISP-DM teilt als Projektleitfaden den Prozeß der Wissensentdeckung in Datenbanken in die folgenden sechs Phasen auf (vgl. auch [Bartlmae, 2002, S. 15ff.]):

1. Domänen- beziehungsweise Anwendungsanalyse

   In dieser einleitenden Phase gilt es, die Grundlage für das Projekt zu schaffen. Zunächst wird dazu die Domäne beziehungsweise das Anwendungsgebiet analysiert. Darauf basierend erfolgt die Formulierung der Projektziele und der jeweiligen Anforderungen aus Domänensicht. Dies umfaßt insbesondere die Ableitung von Erfolgskriterien, die Identifikation notwendiger Datenquellen und die Einplanung zeitlicher, technischer und personeller Ressourcen. Aufgabe ist es, die Projektziele adäquat in ein Wissensentdeckungsproblem abzubilden und einen ersten Projektablaufplan zu fixieren.

2. Datenbeschaffung und Datenverständnis

   Auf Basis der bisherigen Erkenntnisse ist das erste Ziel dieser Phase, die identifizierten Daten zu beschaffen und mit den dann verfügbaren Daten

vertraut zu werden. Dazu dienen beispielsweise einfache Abfragen, einfache deskriptive statistische Analysen oder Visualisierungen der Daten. Das zweite Ziel dieser Phase ist, sämtliche relevanten Attribute sowie die zugehörigen Attributwerte zu verstehen. Außerdem soll bereits ein erster Eindruck der Potentiale gewonnen werden, welche die Daten hinsichtlich der Analyseziele bieten.

3. Datenaufbereitung

   Ziel dieser Phase ist Erzeugung der Eingabe für die Data Mining-Algorithmen aus den Rohdaten. Die Datenaufbereitung umfaßt sowohl syntaktische Aspekte, das heißt für die jeweiligen Implementierungen der Algorithmen notwendige Formattransformationen, als auch semantische Aspekte, wie die Selektion geeigneter Tabellen, Attribute und Datensätze. Des weiteren werden gegebenenfalls aus den vorhandenen Daten höherwertige oder andersartige Attribute abgeleitet, die ursprünglich lediglich implizit vorhanden sind. Beispielsweise kann der Wochentag aus einer Datumsangabe erzeugt werden.

4. Data Mining oder Modellierung

   In dieser Phase findet die eigentliche Analyse statt. Basierend auf den Erkenntnissen und Ergebnissen der vorangehenden Schritte werden Analysealgorithmen sowie entsprechende Parametereinstellungen ausgewählt und auf die selektierten und aufbereiteten Daten angewandt.

5. Bewertung

   Die Bewertung der Data Mining-Ergebnisse umfaßt drei Hauptaspekte: Zunächst muß überprüft werden, ob die Analysealgorithmen die aufbereiten Daten auf der syntaktischen Ebene richtig einlesen und interpretieren konnten. Ebenso ist es notwendig, die Plausibilität und Verläßlichkeit der von den Datenanalysealgorithmen erzeugten Ergebnisse kritisch zu hinterfragen. Während einige Algorithmen entsprechende Signifikanz- oder Qualitätskennzahlen erzeugen, wird der Analyst von vielen Verfahren bei der Bewertung der Ergebnisse nicht unterstützt. Abschließend sind in dieser Phase die erzielten Data Mining-Ergebnisse mit den in der ersten Phase aus Domänen- beziehungsweise Anwendungssicht formulierten Zielen abzugleichen.

6. Umsetzung

   Nachdem die Analyseergebnisse erzeugt und bewertet sind, werden diese in die Anwendungsdomäne transferiert. Im einfachsten Fall wird ein Bericht oder eine Präsentation erstellt, worin die erzielten Ergebnisse so aufbereitet sind, daß diese auch für die Personen aus der Anwendungsdomäne, etwa die Auftraggeber, verständlich werden, ohne Kenntnisse der zugrundeliegenden

Data Mining-Verfahren vorauszusetzen. Der Transfer der Ergebnisse kann jedoch beispielsweise auch darin bestehen, daß ein kompletter Wissensentdeckungsprozeß in die operativen Geschäftsprozesse eines Unternehmens integriert wird.

Jede der beschriebenen sechs Phasen zerfällt in eine Reihe von Teilaufgaben. Die Phasen und Teilaufgaben werden typischerweise mehrfach und in wechselnder Reihenfolge durchlaufen (vgl. u. a. [Nakhaeizadeh et al., 1998], [Wirth und Hipp, 2000]). In Abbildung 2.1 sind die wichtigsten Übergänge zwischen den Phasen wiedergegeben. Der Analyst steuert den Prozeß, indem er fortlaufend die erreich-

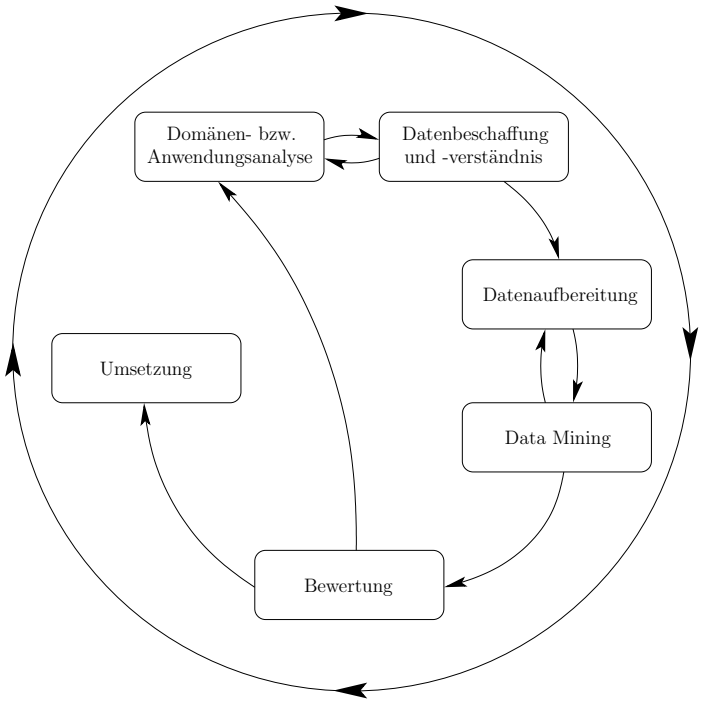


Abbildung 2.1: Die Phasen von CRISP-DM mit den wichtigsten Übergängen (in Anlehnung an [Chapman et al., 2000], [Wirth und Hipp, 2000])

ten Zwischenergebnisse mit seinen Erwartungen abgleicht und entsprechend die nächsten Schritte plant. Dabei ist es durchaus üblich, zu bereits durchlaufenen Phasen zurückzukehren.

Iterationen können zum Beispiel bereits in den ersten Phasen notwendig werden, wenn bei der Datenbeschaffung in Phase 2 offensichtlich wird, daß die in der Anwendungsanalyse formulierten Ziele mit den vorhandenen Daten nicht erreicht werden können und in einem Rückschritt zu Phase 1 die Ziele neu formuliert werden

müssen. Eine Diskrepanz zwischen dem Potential der vorhandenen Daten und den verfolgten Zielen kann jedoch auch erst bei der eigentlich abschließenden Bewertung in Phase 5 deutlich werden. Ebenfalls wird zu Phase 1 zurückgekehrt oder der gesamte Analyseprozeß ohne befriedigendes Ergebnis abgebrochen. Möglich ist auch, daß die Bewertung lediglich Probleme der Datenaufbereitung zeigt und mit Phase 3 fortgefahren werden kann. Die Mängel der Datenaufbereitung können ebenso schon in Phase 4, das heißt bei Anwendung der Analysealgorithmen, sichtbar werden etc. Diese Beispiele geben einen ersten Eindruck von der Komplexität und dem iterativen Charakter des Wissensentdeckungsprozesses, der sich in praktischen Anwendungen als schwer zu planen erweist (vgl. auch [Hipp und Lindner, 1999], [Wirth und Hipp, 2000]). Dies gilt allgemein für den Prozeß der Wissensentdeckung in Datenbanken und nicht nur für CRISP-DM als eine mögliche Konkretisierung.

Offensichtlich ist der Erfolg eines derart auf den Analysten angewiesenen Prozesses entscheidend von dessen Erfahrungshorizont, Motivation und Kreativität abhängig. Der Unterstützung des Analysten kommt daher eine zentrale Bedeutung für den Prozeß der Wissensentdeckung in Datenbanken zu (vgl. [Brachman und Anand, 1996], [Hipp et al., 2002a]).

2.1.2 Methoden der Wissensentdeckung

Um die Analyseziele verfolgen zu können, die sich aus den verschiedensten Anwendungen ergeben, stehen im Rahmen der Wissensentdeckung in Datenbanken diverse Analyse- beziehungsweise Data Mining-Methoden zur Verfügung. Im folgenden werden die nach [Fayyad et al., 1996b] wichtigsten Methoden vorgestellt und an Beispielen erläutert. Die Beschreibung erfolgt aus der Sicht der Wissensentdeckung in Datenbanken.

Klassifikation

Die *Klassifikation (classification)* setzt voraus, daß jedes Objekt sinnvoll klassifiziert, das heißt jeder Datensatz geeignet einer Klasse aus einer vorgegebenen Menge von Klassen zugeordnet werden kann. Das Ziel der Klassifikation ist es, ein Modell zu erzeugen, das neue, bisher ungesehene Datensätze verläßlich klassifizieren kann. Das jeweilige Modell wird auf Grundlage von Datensätzen erzeugt, deren Klassenzugehörigkeit bekannt ist (vgl. u. a. [Quinlan, 1993], [Hand, 1997], [Berry und Linoff, 1997, S. 116ff.]).

Ein Beispiel ist die Einteilung von Kunden einer Bank in die Klassen „kreditwürdig“ und „nicht kreditwürdig“ aufgrund historischer Daten über Kreditausfälle und Kontobewegungen sowie eventuell weiteren Informationen.

Regression

Die *Regression (regression)* stellt im Gegensatz zur Klassifikation einen funktionalen Zusammenhang zwischen den Merkmalen eines Datensatzes und einer metrisch skalierten Zielvariablen her (vgl. etwa [Wittmann, 1999, S. 36f.], [Fayyad et al., 1996b], [Klösgen und Zytkow, 2002, S. 298ff.]).

Ein Beispiel ist die Prognose der Umsätze eines Unternehmens für kommende Perioden aus Vergangenheitswerten.

Segmentierung

Das Ziel der *Segmentierung (clustering)* ist, die zugrundeliegenden Daten geeignet in verschiedene disjunkte Teilmengen, sogenannte Segmente, aufzuteilen. Es sind keine Kategorien vorgegeben, sondern die Aufgabe besteht darin, Segmente derart zu bilden, daß die Objekte bezüglich eines vorgegebenen Maßes innerhalb eines jeden Segments möglichst homogen sind und gleichzeitig die Segmente untereinander eine größtmögliche Heterogenität aufweisen (vgl. u. a. [Berry und Linoff, 1997, S. 55], [Krahl et al., 1998, S. 78f.], [Klösgen und Zytkow, 2002, S. 386ff.]).

Ein Beispiel ist die Segmentierung von Kunden im Einzelhandel anhand von Daten, welche deren Einkaufshistorie dokumentieren.

Datenbeschreibung und -zusammenfassung

Die *Datenbeschreibung und -zusammenfassung (data description and summarization)* kann ein eigenständiges Ziel der Wissensentdeckung sein. Eine sorgfältige Datenbeschreibung trägt wesentlich zum Verständnis der Prozesse bei, von welchen die Daten erzeugt wurden, und generiert damit Wissen im Sinne der Wissensentdeckung in Datenbanken. Auch als Grundlage für weitergehende Untersuchungen ist die Datenbeschreibung und -zusammenfassung eine zentrale Methode der Wissensentdeckung in Datenbanken. Zur Beschreibung und Zusammenfassung der Daten wird unter anderem auf einfache deskriptive Statistiken, wie Mittelwerte oder Varianzen, zurückgegriffen (vgl. auch [Nakhaeizadeh et al., 1998, S. 9f.], [Fayyad et al., 1996b], [Berry und Linoff, 1997, S. 55f.]).

Ein Beispiel ist die Darstellung der Gesamtumsätze von Kunden nach Wochentagen oder der Umsatzveränderungen zur Vorperiode für die wichtigsten Kunden.

Abhängigkeitsanalyse

Die *Abhängigkeitsanalyse (dependency analysis)* generiert Modelle, die signifikante Abhängigkeiten zwischen Merkmalswerten beschreiben. Solche Abhängigkeiten

können sich auf Attributwerte innerhalb eines Datensatzes beziehen oder auf die zeitliche Abfolge von Attributwerten (vgl. etwa [Nakhaeizadeh et al., 1998, S.10], [Bartlmae, 2002, S. 22]).

Im Einzelhandel kann zum Beispiel der Kauf von Artikel a den gleichzeitigen Kauf von Artikel b durch denselben Kunden bedingen, oder, wenn ein zeitlicher Bezug gegeben ist, den zukünftigen Kauf von Artikel b durch diesen Kunden. Neben solchen durch einfache Regeln darstellbaren Abhängigkeiten, können auch komplexere Zusammenhänge modelliert werden (vgl. z. B. [Kischka, 1998, S. 152ff.], [Klösgen und Zytkow, 2002, S. 64ff. u. S. 396ff.]).

Abweichungserkennung

Die *Abweichungserkennung (deviation detection)* identifiziert Objekte, deren Merkmalswerte nicht einer erwarteten Ausprägung entsprechen (vgl. [Nakhaeizadeh et al., 1998, S. 10]). Solche Ausreißer können Hinweise auf die Existenz von Datenqualitätsproblemen geben. Eine Möglichkeit ist, die auffälligen Attribute oder Datensätze aus den Daten zu filtern, damit diese weitere Analysen nicht verzerren (vgl. auch [Hipp et al., 2001a], [Bartlmae, 2002, S. 22]).

Ein Beispiel für einen Ausreißer ist ein Umsatz nahe null für einen bestimmten Artikel aus einem Einzelhandelswarensortiment, obwohl dieser in den Vorperioden vergleichsweise gut verkauft wurde.

2.2 Assoziationsregeln

In diesem Abschnitt werden Assoziationsregeln zunächst formal definiert und anhand eines Beispiels erläutert. Daraufhin werden verschiedene Erweiterungen und Regelgütemaße eingeführt. Das Kapitel schließt mit der Beschreibung von Anwendungsszenarien für die Assoziationsregelgenerierung im Rahmen der Wissensentdeckung in Datenbanken.

2.2.1 Definition und Beispiel

Sei $\mathcal{E}$ eine Menge von *Ereignissen (items)*. Eine Menge $X \subseteq \mathcal{E}$ mit $|X| = k$ heiße *k-Ereigniskombination (k-itemset)* oder kurz *Ereigniskombination (itemset)*. Sei die Transaktionsdatenbank $\mathcal{D}$ eine Multimenge von Ereigniskombinationen. Zu jeder Ereigniskombination X ist die *Häufigkeit (frequency)* definiert als der Anteil von Transaktionen in $\mathcal{D}$, welche die Ereigniskombination X als Teilmenge enthalten.

Die Häufigkeit wird auch als *Support (support)* bezeichnet und wie folgt berechnet:

$$\mathsf{supp}_{\mathcal{D}}(X) = \frac{|\{T \in \mathcal{D} \mid X \subseteq T\}|}{|\mathcal{D}|}.$$

Seien X und Y nichtleere Ereigniskombinationen mit $X \cap Y = \emptyset$. Eine *Assoziationsregel (association rule)* ist dann eine Implikation

$$X \to Y$$

mit *Prämisse (antecedent)* X und *Konklusion (consequent)* Y. Zu jeder Assoziationsregel sind außerdem die Gütemaße *Support (support)* und *Konfidenz (confidence)* definiert. Der Support einer Regel ist der Anteil von Transaktionen in $\mathcal{D}$, die sowohl die Ereignisse aus der Prämisse als auch aus der Konklusion enthalten:

$$\mathsf{supp}_{\mathcal{D}}(X \to Y) = \mathsf{supp}_{\mathcal{D}}(X \cup Y) = \frac{|\{T \in \mathcal{D} \mid X \cup Y \subseteq T\}|}{|\mathcal{D}|}.$$

Der Support einer Regel gibt an, wie oft diese in den Transaktionsdaten erfüllt ist, und kann damit als ein Maß für die Signifikanz einer Regel betrachtet werden.

Als Basis für die Berechnung der Konfidenz dienen jeweils die Transaktionen aus $\mathcal{D}$, welche sämtliche Ereignisse aus der Prämisse enthalten, und damit die Prämisse der jeweiligen Regel erfüllen. Für diese Teilmenge von $\mathcal{D}$ gibt die Konfidenz den Anteil von Transaktionen an, welche die Konklusion ebenfalls erfüllen:

$$\mathsf{conf}_{\mathcal{D}}(X \to Y) = \frac{|\{T \in \mathcal{D} \mid X \cup Y \subseteq T\}|}{|\{T \in \mathcal{D} \mid X \subseteq T\}|}.$$

Die Konfidenz $\mathsf{conf}_{\mathcal{D}}$ einer Regel $X \to Y$ entspricht damit der bedingten Wahrscheinlichkeit $P(Y|X)$, daß eine zufällig aus $\mathcal{D}$ gezogene Transaktion, die X enthält, ebenfalls Y enthält. Für Assoziationsregeln, deren Prämisse in keiner der Transaktionen vorkommt, ist die Konfidenz nicht definiert. Diese Regeln sind auf die gegebene Transaktionsdatenbank $\mathcal{D}$ nicht anwendbar.

In Tabelle 2.1 ist ein Beispiel für eine Transaktionsdatenbank dargestellt. Die Transaktionen der Datenbank $\mathcal{D} = \{T_1, \ldots, T_8\}$ sind jeweils Teilmengen der Menge von Ereignissen $\mathcal{E} = \{a, b, c, d, e\}$. Beispielsweise gilt für den Support der Assoziationsregel $\{a, b\} \to \{c\}$:

$$\mathsf{supp}_{\mathcal{D}}(\{a, b\} \to \{c\}) = \frac{|\{T_1, T_2, T_3\}|}{|\mathcal{D}|} = 37,5\%.$$

Damit erfüllen 37,5% der Transaktionen aus $\mathcal{D}$ die Regel $\{a, b\} \to \{c\}$. Für die Konfidenz dieser Regel gilt:

$$\mathsf{conf}_{\mathcal{D}}(\{a, b\} \to \{c\}) = \frac{|\{T \in \mathcal{D} \mid \{a, b, c\} \subseteq T\}|}{|\{T \in \mathcal{D} \mid \{a, b\} \subseteq T\}|} = \frac{|\{T_1, T_2, T_3\}|}{|\{T_1, T_2, T_3, T_4\}|} = 75\%.$$

Transaktion	Ereigniskombination
T_1	$\{a, b, c, d\}$
T_2	$\{a, b, c, e\}$
T_3	$\{a, b, c\}$
T_4	$\{a, b, e\}$
T_5	$\{a, c, d, e\}$
T_6	$\{a, c, e\}$
T_7	$\{b, d, e\}$
T_8	$\{b, d, e\}$

Tabelle 2.1: Beispiel für eine Transaktionsdatenbank

Enthält also eine zufällig aus $\mathcal{D}$ gezogene Transaktion die beiden Ereignisse a und b, dann enthält diese Transaktion mit einer Wahrscheinlichkeit von 75% ebenfalls das Ereignis c.

Insgesamt existieren für die exemplarisch angenommene Menge $\mathcal{E}$ von Ereignissen, die mit $|\mathcal{E}| = 5$ sehr wenige Ereignisse enthält, bereits fast 200 Assoziationsregeln (vgl. dazu Abschnitt 3.1.1).[1] Eine Auswahl verschiedener Assoziationsregeln zu der Ereignismenge $\mathcal{E}$ ist zusammen mit den anhand der Transaktionsdatenbank $\mathcal{D}$ aus Tabelle 2.1 bestimmten Support- und Konfidenzwerten in Tabelle 2.2 zu finden.

Assoziationsregel	Support	Konfidenz
$\{a\} \rightarrow \{b\}$	$4/8 = 50\%$	$4/6 \approx 66{,}7\%$
$\{b\} \rightarrow \{a\}$	$4/8 = 50\%$	$4/6 \approx 66{,}7\%$
$\{a, b\} \rightarrow \{c\}$	$3/8 = 37{,}5\%$	$3/4 = 75\%$
$\{a, c\} \rightarrow \{b\}$	$3/8 = 37{,}5\%$	$3/5 = 60\%$
$\{b, c\} \rightarrow \{a\}$	$3/8 = 37{,}5\%$	$3/3 = 100\%$
$\{a\} \rightarrow \{b, c\}$	$3/8 = 37{,}5\%$	$3/6 = 50\%$
$\vdots$	$\vdots$	$\vdots$

Tabelle 2.2: Assoziationsregeln zu den Beispieltransaktionen

Die eingeführten Assoziationsregeln werden auch als einfache Assoziationsregeln bezeichnet. Üblicherweise werden die Maße Support und Konfidenz durch weitere Gütemaße ergänzt, wie im folgenden Abschnitt beschrieben.

[1] Wie oben erwähnt, müssen jedoch nicht für alle Regeln die bedingten Wahrscheinlichkeiten, das heißt die Konfidenzwerte, auf Grundlage der Transaktionsdatenbank $\mathcal{D}$ bestimmbar sein.

2.2.2 Ergänzende Gütemaße

In praktischen Anwendungen zeigt sich, daß Regeln mit hoher Konfidenz fälschlicherweise kausale Abhängigkeiten suggerieren können:
Bezeichne $P(X)$ die Wahrscheinlichkeit, aus $\mathcal{D}$ zufällig eine Transaktion zu ziehen, die X enthält, und bezeichne $P(X \cup Y)$ die Wahrscheinlichkeit, eine Transaktion zu ziehen, die sowohl X als auch Y enthält. Seien X und Y stochastisch unabhängig (vgl. z. B. [Bauer, 1991, S. 42]), das heißt es gilt:

$$P(X)P(Y) = P(X \cup Y).$$

Für die Konfidenz der Regel $X \rightarrow Y$ folgt dann:

$$\mathsf{conf}_{\mathcal{D}}(X \rightarrow Y) = P(Y|X) = \frac{P(X \cup Y)}{P(X)} = P(Y).$$

Ist Y Teilmenge einer großen Anzahl der Transaktionen in $\mathcal{D}$, dann wird der Regel $X \rightarrow Y$ ein entsprechend hoher Konfidenzwert zugewiesen. Separat betrachtet kann diese hohe Konfidenz eine Abhängigkeit zwischen den Ereigniskombinationen X und Y suggerieren, obwohl diese stochastisch unabhängig sind.
Unter anderem aufgrund dieser Problematik wurden weitere Gütemaße entwickelt, von denen die bekanntesten im folgenden vorgestellt werden:

Lift

Das Maß *Lift (lift)* drückt die Abweichung der Konfidenz einer Regel von der a priori-Wahrscheinlichkeit für die Konklusion dieser Regel aus (vgl. [Brin et al., 1997b; Tan und Kumar, 2000]):[2]

$$\mathsf{lift}_{\mathcal{D}}(X \rightarrow Y) = \frac{\mathsf{conf}_{\mathcal{D}}(X \rightarrow Y)}{P(Y)} = \frac{\mathsf{conf}_{\mathcal{D}}(X \rightarrow Y)}{\mathsf{supp}_{\mathcal{D}}(Y)} = \frac{\mathsf{supp}_{\mathcal{D}}(X \cup Y)}{\mathsf{supp}_{\mathcal{D}}(X) \cdot \mathsf{supp}_{\mathcal{D}}(Y)}.$$

Bei stochastischer Unabhängigkeit von X und Y gilt somit $\mathsf{lift}_{\mathcal{D}}(X \rightarrow Y) = 1$. $\mathsf{lift}_{\mathcal{D}}(X \rightarrow Y) > 1$ bedeutet, daß, wenn eine Transaktion X enthält, sich dadurch die Wahrscheinlichkeit erhöht, daß diese Transaktion ebenfalls Y enthält. Der Wert $\mathsf{lift}_{\mathcal{D}}(X \rightarrow Y)$ gibt den Faktor dieser Erhöhung an. Umgekehrt sinkt diese Wahrscheinlichkeit für Werte von $\mathsf{lift}_{\mathcal{D}}(X \rightarrow Y) < 1$. Das Maß $\mathsf{lift}_{\mathcal{D}}$ ist symmetrisch.

Konviktion

Sei $P(\neg Y)$ die Wahrscheinlichkeit, zufällig aus $\mathcal{D}$ eine Transaktion zu ziehen, die Y nicht enthält, und sei $P(X, \neg Y)$ die Wahrscheinlichkeit, zufällig eine Transaktion

[2] In der englischsprachigen Literatur wird das Maß Lift auch mit „interest“ bezeichnet.

aus $\mathcal{D}$ zu ziehen, die zwar X, nicht aber Y enthält. Das Maß *Konviktion (conviction)* aus [Brin et al., 1997b] basiert auf der logischen Äquivalenz der Aussagen $X \Rightarrow Y$ und $\neg(X \wedge \neg Y)$ und ist wie folgt definiert:

$$\mathsf{conv}_{\mathcal{D}}(X \rightarrow Y) = \frac{P(X)P(\neg Y)}{P(X, \neg Y)}.$$

$\mathsf{conv}_{\mathcal{D}}(X \rightarrow Y)$ drückt aus, wie stark X und $\neg Y$ von stochastischer Unabhängigkeit abweichen und nimmt Werte aus dem Intervall $[0, \infty)$ an. Hohe Werte für die Konviktion lassen darauf schließen, daß die Regelprämisse tatsächlich die Regelkonklusion impliziert.

Rule Interest

Das Maß *Rule Interest (rule interest)* mißt die Differenz zwischen der Häufigkeit des beobachteten gemeinsamen Auftretens von Prämisse und Konklusion und der Häufigkeit bei angenommener stochastischer Unabhängigkeit (vgl. u. a. [Piatetsky-Shapiro, 1991; Tan und Kumar, 2000]):

$$\mathsf{RI}_{\mathcal{D}}(X \rightarrow Y) = P(X \cup Y) - P(X) \cdot P(Y) = \mathsf{supp}_{\mathcal{D}}(X \cup Y) - \mathsf{supp}_{\mathcal{D}}(X) \cdot \mathsf{supp}_{\mathcal{D}}(Y).$$

Dieses Maß nimmt Werte zwischen -0,25 und +0,25 an, insbesondere den Wert 0 im Fall stochastischer Unabhängigkeit von X und Y.

Neben den vorgestellten Gütemaßen wurden weitere Ansätze und Maße entwickelt, um interessante Regeln zu identifizieren. Für einen allgemeinen Überblick sei unter anderem auf [Ramaswamy et al., 1998], [Sahar, 1999], [Shah et al., 1999], [Tan und Kumar, 2000], [Hilderman und Hamilton, 2002] verwiesen.

2.2.3 Berücksichtigung von Taxonomien

In vielen Anwendungsszenarien existieren *Taxonomien (taxonomies)*, die Ereignisse hierarchisch zu Ereignisgruppen zusammenfassen, welche rekursiv ebenfalls zu Gruppen zusammengefaßt sein können. Ein Ereignis beziehungsweise eine Ereignisgruppe kann mehreren Gruppen zugeordnet werden, wobei jedoch keine Zyklen entstehen dürfen. Werden die Ereignisse sowie die Ereignisgruppen als Knoten und die Beziehungen zwischen diesen als gerichtete Kanten aufgefaßt, kann eine Taxonomie als gerichteter azyklischer Graph *(directed acyclic graph, DAG)* dargestellt werden.

Eine Taxonomie $\mathcal{T}$ zu den Ereignissen $\mathcal{E} = \{a, b, c, d, e\}$ aus den Tabellen 2.1 und 2.2 ist in Abbildung 2.2 dargestellt. Die ursprüngliche Ereignismenge $\mathcal{E}$ wird um die Ereignisse $\{f, g, h, i, j\}$ erweitert, die jeweils für Ereignisgruppen stehen.

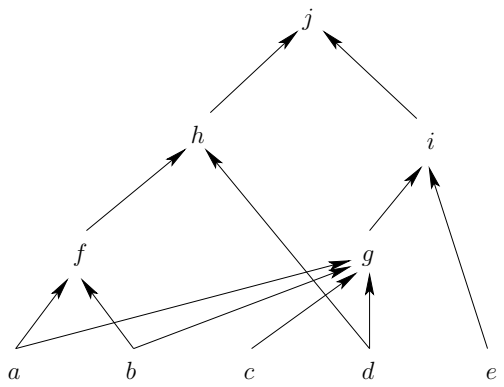


Abbildung 2.2: Beispiel für eine Taxonomie

Falls keine Taxonomie existiert, kann in vielen Anwendungsszenarien ein Taxonomie speziell für die Wissensentdeckung erzeugt werden.

Sogenannte *taxonome Assoziationsregeln (generalized association rules)* berücksichtigen Taxonomien und können Ereignisse beziehungsweise Ereignisgruppen aus verschiedenen Abstraktionsebenen der Taxonomie enthalten.

Sei $\mathcal{T} = \mathcal{T}(V, E)$ ein DAG mit der Knotenmenge V und der Kantenmenge E. Sei $\mathcal{T}$ eine Taxonomie auf der Ereignismenge $\mathcal{E}$, gelte folglich $V = \mathcal{E}$ und $E \subseteq \mathcal{E} \times \mathcal{E}$. Ein Ereignis $x \in \mathcal{E}$ heißt *Nachkomme (descendant)* eines Ereignisses $\hat{x} \in \mathcal{E}$ genau dann, wenn in der transitiven Hülle von $\mathcal{T}$ eine Kante von x nach $\hat{x}$ existiert. In diesem Fall heißt außerdem $\hat{x}$ *Vorfahr (ancestor)* von x. Sei $\mathsf{desc}(\hat{x})$ die Menge aller Nachkommen eines Ereignisses $\hat{x}$ und $\mathsf{anc}(x)$ die Menge aller Vorfahren eines Ereignisses x. Für die Berechnung der Konfidenz sei nicht mehr ausschließlich entscheidend, ob ein Ereignis $\hat{x}$ selbst in einer Transaktion T enthalten ist. Vielmehr genüge es, wenn T einen Nachkommen $x \in \mathsf{desc}(\hat{x})$ von $\hat{x}$ enthält.[3] Die Definitionen für den Support und die Konfidenz der Regeln werden entsprechend erweitert:

$$\mathsf{supp}_{\mathcal{D}}(X) = \frac{|\{T \in \mathcal{D} \mid \forall \hat{x} \in X : (\{\hat{x}\} \cup \mathsf{desc}(\hat{x})) \cap T \neq \emptyset\}|}{|\mathcal{D}|}$$

und

$$\mathsf{conf}_{\mathcal{D}}(X \to Y) = \frac{|\{T \in \mathcal{D} \mid \forall \hat{x} \in X \cup Y : (\{\hat{x}\} \cup \mathsf{desc}(\hat{x})) \cap T \neq \emptyset\}|}{|\{T \in \mathcal{D} \mid \forall \hat{x} \in X : (\{\hat{x}\} \cup \mathsf{desc}(\hat{x})) \cap T \neq \emptyset\}|}.$$

Um triviale Regeln auszuschließen, wird für taxonome Assoziationsregeln ebenfalls die Bedingung $X \cap Y = \emptyset$ erweitert. Kein Ereignis aus der Konklusion Y darf

[3] Transaktionen können folglich beliebige Ereignisse aus $\mathcal{E}$ enthalten, das heißt auch solche, die für Ereignisgruppen stehen.

Vorfahr eines Ereignisses aus der Prämisse X sein:

$$\bigcup_{x \in X} \mathsf{anc}(x) \cap Y = \emptyset.$$

Außerdem dürfen weder Prämisse noch Konklusion ein Ereignis und einen Nachkommen dieses Ereignisses gleichzeitig enthalten, das heißt es muß gelten:

$$X \cap \mathsf{desc}(X) = \emptyset \text{ und } Y \cap \mathsf{desc}(Y) = \emptyset.$$

Durch die Einführung einer Taxonomie können Regeln erzeugt werden, die Ereignisse aus unterschiedlichen Abstraktionsniveaus enthalten. Die Taxonomie aus Abbildung 2.2 faßt die Ereignisse a und b zu g zusammen. Beispielsweise beträgt die Konfidenz der Regel $\{g\} \rightarrow \{e\}$ für die Transaktionen aus Tabelle 2.1

$$\mathsf{conf}_{\mathcal{D}}(\{g\} \rightarrow \{e\}) = \frac{|\{T_2, T_4, T_5, T_6, T_7, T_8\}|}{|\{T_1, T_2, T_3, T_4, T_5, T_6, T_7, T_8\}|} = \frac{6}{8} = 75\%.$$

Diese Regel $\{g\} \rightarrow \{e\}$ kann als Zusammenfassung der Regeln $\{a\} \rightarrow \{e\}$, $\{b\} \rightarrow \{e\}$ und $\{a, b\} \rightarrow \{e\}$ auf abstrakterem Niveau verstanden werden.

Taxonome Assoziationsegeln, wie oben definiert, gehen auf [Srikant und Agrawal, 1995; Srikant, 1996] zurück. In [Han und Fu, 1995; Fu, 1996] wurden zur gleichen Zeit sogenannte *ebenenübergreifende Assoziationsregeln (level crossing association rules)* eingeführt. Die Idee, unter Berücksichtigung hierarchischer Beziehungen Regeln zu generieren, liegt beiden Ansätzen zugrunde, wobei die Definition aus [Srikant und Agrawal, 1995] allgemeiner gehalten ist und Ausgangspunkt dieser Arbeit sein soll.

2.2.4 Weitere Regeltypen

Motiviert durch verschiedene Anwendungsszenarien wurden neben den beschriebenen einfachen und taxonomen Assoziationsregeln eine Reihe weiterer Assoziationsregeltypen und verwandter Analysemethoden entwickelt.

Die wichtigste dieser Erweiterungen sind *quantitative Assoziationsregeln (quantitative association rules)* aus [Srikant und Agrawal, 1996a] (vgl. auch [Fukuda et al., 1996], [Miller und Yang, 1997], [Zhang et al., 1997], [Büchter und Wirth, 1998], [Weber, 1998], [Hong et al., 1999]). Quantitative Assoziationsregeln setzen als zusätzliche Information die Anzahl der Vorkommen eines Ereignisses in jeder Transaktion voraus. Mittels geeigneter Intervallbildung können Regeln dann die Anzahl der Vorkommen modellieren. Eine solche Regel könnte beispielsweise wie folgt aussehen: Enthält eine Transaktion m- bis n-mal das Ereignis a und gleichzeitig das Ereignis b mindestens einmal, dann enthält diese Transaktion mit einer

bestimmten Konfidenz das Ereignis c genau k-mal:

$$\{m \leq \mathsf{anzahl}(a) \leq n,\ \ 1 \leq \mathsf{anzahl}(b)\} \ \ \rightarrow \ \ \{\mathsf{anzahl}(c) = k\}.$$

Intervalle können beliebig in Prämisse und Konklusion auftreten.

Eine andere Erweiterung sind *negative Regeln (negative rules)* aus [Savasere et al., 1998]. Diese Regeln können die Abwesenheit von Ereignissen in Transaktionen berücksichtigen. Außerdem sind verschiedene Ansätze entwickelt worden, Assoziationsregeln durch Fuzzy Logic zu erweitern (vgl. etwa [Chan und Au, 1997; Kuok et al., 1998; Yang und Singhal, 1999; Gyenesei, 2001]).

Von größerer Bekanntheit sind ebenso die Ansätze zur Erzeugung *räumlicher Assoziationsregeln (spatial association rules)* für die Analyse geographischer Datenbanken (vgl. u. a. [Koperski und Han, 1995; Clementini et al., 2000; Li et al., 2003a]).

Weitere, sehr spezielle Regeltypen sind beispielsweise in [Aggarwal et al., 1998; Lu et al., 1998; Tung et al., 1999] finden.

Im Zusammenhang mit der Generierung von Assoziationsregeln sind ebenfalls *sequentielle Muster (sequential patterns)* zu erwähnen (vgl. u. a. [Agrawal und Srikant, 1995], [Mannila et al., 1995], [Hätönen et al., 1996], [Srikant und Agrawal, 1996b], [Srikant, 1996], [Mannila et al., 1997]). Sequentielle Muster sind keine Erweiterung der Assoziationsregeln, jedoch sind beide Wissensentdeckungsmethoden bezüglich der Generierung und der verwendeten Begrifflichkeiten eng miteinander verwandt. Grundlage der Generierung sequentieller Muster sind um zeitliche Informationen angereicherte Transaktionsdaten. Anstelle der Modellierung von Abhängigkeiten bezüglich Gleichzeitigkeit beschreiben sequentielle Muster häufige zeitliche Abfolgen von Ereignissen und Ereigniskombinationen.

2.3 Anwendungsszenarien

Assoziationsregeln sind als bedingte Wahrscheinlichkeiten auf intuitive Art und Weise verständlich. Statistische Grundkenntnisse reichen aus, um Analyseergebnisse nachvollziehen zu können. Insbesondere die einfache Kommunizierbarkeit und unmittelbare Verständlichkeit haben mit zur weiten Verbreitung von Assoziationsregeln als Analyseverfahren beigetragen. Heute gehört die Generierung von Assoziationsregeln zu den Grundlagen der Wissensentdeckung in Datenbanken und ist laut einer Umfrage in [Piatetsky-Shapiro, 2002b] unter den acht am häufigsten eingesetzten Verfahren.

Ursprünglich motiviert war die Datenanalyse mit Assoziationsregeln durch die neuen Möglichkeiten der Warenkorbanalyse im Einzelhandel, die sich mit der Einführung von Scannerkassensystemen eröffneten (vgl. [Agrawal et al., 1993]). Assoziationsregeln werden jedoch nicht nur im traditionellen Einzelhandel eingesetzt.

Ein bekanntes Beispiel aus dem Bereich des E-Commerce ist der Internethändler Amazon (vgl. z. B. [Saunders, 1999]) der seinen Absatz durch gezielte Kaufvorschläge fördert. Während ein Kunde die Seiten des Internetauftritts von Amazon besucht und Artikel auswählt, bekommt er Hinweise, welche Artikel andere Kunden zusätzlich zu diesen gekauft oder näher betrachtet haben. Eine Möglichkeit, solche Abhängigkeiten aufzudecken, ist die Generierung von Assoziationsregeln.

Assoziationsregeln sind nicht auf das klassische Gebiet der Warenkorbanalyse beschränkt, sondern es lassen sich Anwendungsszenarien in den unterschiedlichsten Bereichen finden. Im folgenden werden exemplarisch einige Ansätze kurz vorgestellt. Auf einen Teil der hier beschriebenen Szenarien wird in dieser Arbeit noch mehrfach Bezug genommen werden.

Abhängigkeitsanalyse

Im Rahmen der Abhängigkeitsanalyse läßt sich das Konzept des Warenkorbs beziehungsweise der Ereigniskombinationen auf eine Vielzahl weiterer Anwendungsgebiete unmittelbar übertragen.

Ein Beispiel ist die Untersuchung von Abhängigkeiten zwischen Ausstattungsmerkmalen von Personenkraftwagen aus [Hipp et al., 2002a]. Die Ereignisse stehen in diesem Szenario für die Ausstattungsmerkmale und die Transaktionen entsprechen den einzelnen Fahrzeugen. Die erzeugten Regeln geben Beziehungen zwischen Ausstattungsmerkmalen wieder. Jede Regel entspricht einer Aussage der Form, daß Fahrzeuge, die mit den Ausstattungen $\{x_1, x_2, \ldots, x_n\}$ produziert werden, gleichzeitig auch $\{y_1, y_2, \ldots, y_m\}$ als Ausstattungsmerkmale aufweisen. Solche Informationen können für das Marketing im Bereich Sonderausstattungen von Interesse sein.

Werden zu diesen Daten Informationen über Werkstattaufenthalte und Reparaturen hinzugefügt, dann können Aussagen über Zusammenhänge von Ausstattungsmerkmalen und späteren Reparaturen abgeleitet werden (vgl. [Hipp und Lindner, 1999]). Auch Daten aus dem laufenden Fahrzeugbetrieb können auf ähnliche Weise mit Assoziationsregeln ausgewertet werden (vgl. [Wirth et al., 2001]).

Ein weiteres Beispiel ist die Analyse von Log-Dateien, wie sie Server im WWW – World Wide Web – erzeugen. In diesen Dateien werden Zugriffe auf die angebotenen WWW-Seiten im laufenden Betrieb protokolliert. Unter anderem können aus den Log-Dateien sogenannte Benutzersitzungen abgeleitet werden.[4] Eine solche Benutzersitzung beinhaltet sämtliche WWW-Seiten, die ein Benutzer während einer Sitzung besucht. Wird jede Seite als ein Ereignis und jede Sitzung als eine Trans-

[4] Da die Informationen in den Log-Dateien zumeist unvollständig sind, ist die Ableitung von Benutzersitzungen eine komplexe Aufgabe (vgl. u. a. [Berendt et al., 2001], [Berendt et al., 2002], [Spiliopoulou et al., 2003]).

aktion verstanden, dann können aus den aufbereiteten Log-Dateien direkt Assoziationsregeln erzeugt werden. Diese Regeln beschreiben Zusammenhänge zwischen den Seiten hinsichtlich der erfolgten Besuche. Die Wissensentdeckung im Kontext des WWW wird auch als Web Mining bezeichnet (vgl. etwa [Mobasher et al., 1996], [Moore et al., 1997], [Garofalakis et al., 1999], [Spiliopoulou, 1999], [Cooley, 2000], [Spiliopoulou et al., 2003]).

Klassifikation

Der Klassifikation mittels Assoziationsregeln liegt die Idee zugrunde, aus bereits klassifiziert vorliegenden Datensätzen Assoziationsregeln zu erzeugen, deren Konklusion aus genau einem Ereignis besteht, welches einer der Klassen entspricht. Die Klassenzugehörigkeit eines jeden Datensatzes muß entsprechend in den Ausgangsdatensätzen als Ereignis kodiert enthalten sein. Datensätze können dann klassifiziert werden, indem auf diese die Regelmenge angewandt und dadurch eine Klasse vorausgesagt wird. Die Schwierigkeit besteht zum einen darin, anhand der Regelmenge die Zugehörigkeit der Datensätze zu einer der Klassen festzulegen. Für viele der Datensätze werden die Aussagen der Regeln widersprüchlich sein, so daß geeignet abgewogen werden muß. Zum anderen ist es notwendig, bereits bei der Erzeugung der Regelmenge durch Filterung eine Menge von Regeln zu erzeugen, die gute Ergebnisse bezüglich der Klassifikation erreicht. Als Filterkriterien können die Ereignisse in Prämisse und Konklusion der Regeln dienen oder Schwellenwerte für die Gütemaße vorgegeben werden. Zur Klassifikation mittels Assoziationsregeln sei auf [Liu et al., 1998; Megiddo und Srikant, 1998; Bayardo und Agrawal, 1999; Meretakis und Wüthrich, 1999; Freitas, 2000; Li et al., 2001] verwiesen.

Abweichungsanalyse

Die Abweichungsanalyse mittels Assoziationsregeln basiert darauf, daß eine Menge von Assoziationsregeln einen Teil der Struktur der dieser Menge zugrundeliegenden Datensätze, die als Transaktionen verstanden werden, widerspiegelt. Die Regelmenge wird in diesem Sinne als eine Verallgemeinerung der Daten verstanden. Einzelne Datensätze können auf ihre Konsistenz mit dieser Regelmenge überprüft werden, und in Abhängigkeit von der Anzahl widersprechender, das heißt von einem Datensatz verletzter Regeln, werden Datensätze gegebenenfalls als Ausreißer klassifiziert. Die Datensätze können sowohl aus der zugrundeliegenden Datenbasis stammen, als auch bei der Regelerzeugung noch unbekannt gewesen sein. Ähnlich wie bei der Klassifikation liegt die Schwierigkeit darin, durch Filterung geeignete Regelmengen zu erzeugen und außerdem ein Maß für die Inkonsistenz eines Datensatzes mit einer Regelmenge unter Verwendung der verschiedenen Regelgütemaße abzuleiten.

Assoziationsregeln basieren auf symbolischen Werten, während der Schwerpunkt existierender Ansätze zur Ausreißererkennung im Bereich metrisch skalierter Werte liegt (vgl. etwa [Arning et al., 1996]). Zur Ausreißererkennung mittels Assoziationsregeln sei auf [Hipp et al., 2001a] verwiesen.

Weitere Anwendungsszenarien

Neben den oben beschriebenen Anwendungsszenarien werden Assoziationsregeln in weiteren Bereichen eingesetzt. Insbesondere auf dem Gebiet der Segmentierung existieren Erfahrungen mit der Anwendung von Assoziationsregeln (vgl. u. a. [Lent et al., 1997], [Han et al., 1997a], [Ramkumar und Swami, 1998]).[5] Zu nennen sind ebenfalls die Bereiche der Prognose (vgl. u. a. [Lu et al., 1998]) oder der automatischen Bildanalyse (vgl. z. B. [Ordonez und Omiecinski, 1999], [Rushing et al., 2002], [Zaïane, 2003]).

[5] Diese Verfahren basieren auf häufigen Ereigniskombinationen, einem Zwischenergebnis der Assoziationsregelgenerierung, siehe Abschnitt 3.1.

Kapitel 3

Analyse der etablierten Verfahren

In diesem Kapitel werden die heute etablierten Verfahren zur Erzeugung von Assoziationsregeln analysiert. Dazu wird zunächst in Abschnitt 3.1 die Aufgabe der Erzeugung von Assoziationsregeln formalisiert und eine Einführung in die Grundlagen der Assoziationsregelgenerierung gegeben. Es werden verschiedene Strategien zum Durchlaufen des Suchraums und zur Bestimmung von Häufigkeiten zusammen mit entsprechenden Datenstrukturen eingeführt, wobei die Darstellung der Grundlagen unabhängig von konkreten Algorithmen erfolgt. In Abschnitt 3.2 werden basierend auf den beschriebenen Grundlagen die etablierten Verfahren zur Generierung einfacher und taxonomer Assoziationsregeln vorgestellt. Durch die gewählte Aufbereitung sind die Ansätze anhand der zugrundeliegenden Vorgehensweisen vergleichbar, und es gelingt, eine Systematisierung der Verfahren abzuleiten. In Abschnitt 3.3 schließt das Kapitel mit einer Evaluierung der beschriebenen Verfahren im Rahmen verschiedener Experimente. Die Grundlagen dieser Experimente werden dargestellt und die von den Verfahren erzielten Ergebnisse interpretiert. Insbesondere führt der bisher in der Literatur kaum untersuchte Hauptspeicherbedarf zu einer Neubewertung der etablierten Verfahren.

3.1 Grundlagen

In diesem Abschnitt werden unabhängig von den Algorithmen die Grundlagen der Assoziationsregelgenerierung eingeführt. Zunächst wird eine formale Beschreibung des sogenannten Assoziationsproblems gegeben. Es folgt die Darstellung von Strategien zum Durchlaufen des Suchraums sowie zur Bestimmung der Häufigkeiten von Ereigniskombinationen. Der Abschnitt endet mit der Beschreibung verschiedener Datenstrukturen, welche die effiziente Implementierung der eingeführten Ansätze ermöglichen.

3.1.1 Formale Problembeschreibung

Sei $\mathcal{E}$ mit $|\mathcal{E}| = n$ eine wie in Abschnitt 2.2.1 definierte Menge von Ereignissen. Sei $\mathcal{R} = \mathcal{R}(\mathcal{E})$ die Menge aller einfachen Assoziationsregeln zu $\mathcal{E}$. Für die Mächtigkeit der Menge $\mathcal{R}$ gilt dann (vgl. [Hipp et al., 2002a]):

$$|\mathcal{R}| = 3^n - 2^{n+1} + 1.$$

Beweis: Für $x, y \in \mathbb{R}$ und $n \in \mathbb{N}$ gilt der binomische Lehrsatz:

$$(*) \qquad (x+y)^n = \sum_{k=0}^{n} \binom{n}{k} x^{(n-k)} y^k$$

Mit $x = 1$ und $y = 1$ folgt:

$$(**) \qquad 2^n = \sum_{k=0}^{n} \binom{n}{k}$$

Sowohl Prämisse als auch Konklusion sind nichtleere Teilmengen von $\mathcal{E}$. Außerdem ist der Schnitt dieser beiden Mengen nach der Definition aus Abschnitt 2.2.1 stets leer. Prämisse beziehungsweise Konklusion enthalten daher mindestens ein Ereignis und maximal $n-1$ Ereignisse. Sei k die Anzahl von Ereignissen in der Prämisse einer Regel $r \in \mathcal{R}$. Die Konklusion der Regel r enthält dann zwischen einem und maximal $n-k$ Ereignissen, und $|\mathcal{R}|$ kann wie folgt berechnet werden:

$$\begin{aligned}
|\mathcal{R}| &= \sum_{k=1}^{n-1} \left(\binom{n}{k} \cdot \sum_{l=1}^{n-k} \binom{n-k}{l} \right) \\
&= \sum_{k=1}^{n-1} \left(\binom{n}{k} \cdot \left(\sum_{l=0}^{n-k} \binom{n-k}{l} - \binom{n-k}{0} \right) \right) \\
&\overset{(**)}{=} \sum_{k=1}^{n-1} \left(\binom{n}{k} \cdot (2^{(n-k)} - 1) \right) = \sum_{k=1}^{n-1} \left(\binom{n}{k} 2^{(n-k)} \right) - \sum_{k=1}^{n-1} \binom{n}{k} \\
&= \sum_{k=0}^{n} \left(\binom{n}{k} 2^{(n-k)} 1^k \right) - 2^n - 1 - \sum_{k=0}^{n} \binom{n}{k} + 2 \overset{(*)}{=} 3^n - 2^{n+1} + 1
\end{aligned}$$

□

In praktischen Anwendungen (vgl. Abschnitt 2.3) nimmt $n = |\mathcal{E}|$ typischerweise Werte zwischen einigen Hundert bis zu mehreren Hunderttausend an. Entsprechend viele Regeln existieren zu $\mathcal{E}$, und folglich kann das Ziel der Assoziationsregelgenerierung lediglich sein, eine Teilmenge $\mathcal{R}'$ von $\mathcal{R}$ zusammen mit den Werten für die Gütemaße der Regeln aus $\mathcal{R}'$ zu erzeugen.

In der vorliegenden Arbeit erfolgt die Beschränkung der erzeugten Regelmengen durch Vorgabe von Mindestwerten für den Support und die Konfidenz der einzelnen Regeln. Dieser als *Support-Konfidenz-Ansatz (support-confidence framework)* bekannt gewordene und bereits in [Agrawal et al., 1993] eingeführte Ansatz liegt nahezu sämtlichen heutigen Verfahren zugrunde.[1] Formal stellt sich die Assoziationsregelgenerierung also wie folgt dar:

Sei $\mathcal{E}$ eine Menge von Ereignissen und die Transaktionsdatenbank $\mathcal{D}$ eine Multimenge von Teilmengen von $\mathcal{E}$. Ziel der Assoziationsregelgenerierung sei es, die Menge aller Assoziationsregeln zu $\mathcal{E}$, welche die vorgegebenen minimalen Schwellenwerte $\mathsf{minsupp} \in (0, 1]$ und $\mathsf{minconf} \in [0, 1]$ für die Transaktionsdatenbank $\mathcal{D}$ erfüllen, vollständig zu erzeugen, das heißt folgende Regelmenge zusammen mit den Ausprägungen der Gütemaße für $\mathcal{D}$ zu generieren:

$$\mathcal{R}' = \{r \in \mathcal{R}(\mathcal{E}) \mid \mathsf{supp}_{\mathcal{D}}(r) \geq \mathsf{minsupp} \wedge \mathsf{conf}_{\mathcal{D}}(r) \geq \mathsf{minconf}\}.$$

Die Einschränkung $\mathsf{minsupp} \neq 0$ präzisiert die ursprüngliche Definition aus [Agrawal et al., 1993], [Agrawal und Srikant, 1994a]. Für jede Regel $r = X \rightarrow Y$, $r \in \mathcal{R}'$, wird durch diese Vorgabe sichergestellt, daß der Nenner $\mathsf{supp}_{\mathcal{D}}(X)$ für die Berechnung der Konfidenz von r ungleich null und damit $\mathsf{conf}_{\mathcal{D}}(r)$ definiert ist.[2]

Dem Support-Konfidenz-Ansatz liegt die Annahme zugrunde, daß eine Kombination von Schwellenwerten vorgegeben werden kann, so daß diese die Regelmenge auf eine Obermenge der potentiell interessanten Regeln beschränkt. Neben Support und Konfidenz können weitere Gütemaße wie Lift und Konviktion aus Abschnitt 2.2.2 zu den Regeln berechnet und ebenfalls zur Beschränkung der Ergebnismenge genutzt werden.

Die heute im Rahmen des Support-Konfidenz-Ansatzes etablierten Verfahren unterteilen die Regelgenerierung in zwei Schritte, die Generierung sogenannter häufiger Ereigniskombinationen und die Ableitung der Regeln aus den häufigen Ereigniskombinationen (vgl. z. B. [Agrawal und Srikant, 1994a], [Savasere et al., 1995a], [Zaki et al., 1997c], [Han et al., 2000], [Hipp et al., 2000a]). Eine Ereigniskombination $X \subseteq \mathcal{E}$ heiße *häufige Ereigniskombination (frequent itemset)* genau dann, wenn X in mindestens $\mathsf{minsupp} \cdot |\mathcal{D}|$ der Transaktionen aus $\mathcal{D}$ als Teilmenge vorkommt:

$$X \text{ häufig} \Leftrightarrow \mathsf{supp}_{\mathcal{D}}(X) \geq \mathsf{minsupp}.$$

[1] Eine Ausnahme ist etwa der in [Webb, 2000] vorgestellte Algorithmus MagnumOpus, der aus Effizienzgründen jedoch nur für spezielle Analyseszenarien von praktischer Bedeutung ist (vgl. auch [Zheng et al., 2001], [Motwani et al., 2000]).

[2] Die explizite Beschränkung des minimalen Supports minsupp auf Werte größer null hat vor allem formale Bedeutung. Durch positive Schwellenwerte für den Support wird das Problem undefinierter Konfidenzwerte gelöst, ohne daß gleichzeitig die Definition des Assoziationsproblems wesentlich komplizierter wird. In praktischen Anwendungen wird ohnehin meist $\mathsf{minsupp} > 0$ vorgegeben.

Jede nichtleere Teilmenge einer häufigen Ereigniskombination ist ebenfalls häufig. Diese Eigenschaft beruht auf der *Abgeschlossenheitseigenschaft des Supports (closure property of support)* und besagt, daß für Ereigniskombinationen X und X' mit $\emptyset \neq X' \subseteq X \subseteq \mathcal{E}$ gilt $\mathsf{supp}_{\mathcal{D}}(X') \geq \mathsf{supp}_{\mathcal{D}}(X)$. Der Beweis folgt unmittelbar aus der Definition für den Support von Ereigniskombinationen in Abschnitt 2.2.1 (vgl. auch [Agrawal und Srikant, 1994a; Mannila et al., 1994a]).

Sei $\mathcal{F} = \{H \subseteq \mathcal{E} \mid H \text{ häufig}\}$ die Menge aller häufigen Ereigniskombinationen, und sei $\mathcal{R}'$, wie oben definiert, die Menge aller Regeln zu $\mathcal{E}$, welche die Schwellenwerte minsupp und minconf erfüllen. Für jede Regel $r \in \mathcal{R}'$ mit $r = X \rightarrow Y$ gilt dann definitionsgemäß $X \cup Y \in \mathcal{F}$. Durch die Menge $\mathcal{F}$ ist eine Obermenge von $\mathcal{R}'$ festgelegt, die Menge aller Regeln, die zwar minsupp, nicht aber in jedem Fall minconf erfüllen.

Im ersten Schritt der Regelgenerierung wird die Menge $\mathcal{F}$ zusammen mit den Supportwerten der häufigen Ereigniskombinationen bestimmt. Im zweiten Schritt werden aus $\mathcal{F}$ die Regeln abgeleitet, welche die Schwellenwerte für die Konfidenz und eventuelle weitere Gütemaße erfüllen. Dazu genügt es, alle nichtleeren echten Teilmengen $X \subsetneq H$ jeder häufigen Ereigniskombination $H \in \mathcal{F}$ zu betrachten. Die resultierenden Regeln $X \rightarrow H \setminus X$ werden bezüglich minconf und eventueller Schwellenwerte für weitere Gütemaße überprüft. Aufgrund der Abgeschlossenheitseigenschaft des Supports sind alle dazu notwendigen Häufigkeiten bekannt.

In [Agrawal und Srikant, 1994a] wird dieses Verfahren wie folgt optimiert: Erreicht die Regel $X \rightarrow H \setminus X$ den Schwellenwert minconf nicht, dann gilt das auch für die Regeln $X' \rightarrow H \setminus X'$ mit $X' \subsetneq X$, so daß die Regeln für alle Teilmengen X' von X zu einem H nicht weiter betrachtet werden müssen.

Beweis: Aufgrund der Abgeschlossenheitseigenschaft des Supports gilt für alle Ereigniskombinationen $X', \emptyset \neq X' \subsetneq X$: $\mathsf{supp}_{\mathcal{D}}(X') \geq \mathsf{supp}_{\mathcal{D}}(X)$. Die Behauptung folgt unmittelbar aus

$$\mathsf{conf}_{\mathcal{D}}(X' \rightarrow H \setminus X') = \frac{\mathsf{supp}_{\mathcal{D}}(H)}{\mathsf{supp}_{\mathcal{D}}(X')} \leq \frac{\mathsf{supp}_{\mathcal{D}}(H)}{\mathsf{supp}_{\mathcal{D}}(X)} = \mathsf{conf}_{\mathcal{D}}(X \rightarrow H \setminus X).$$

□

Die Laufzeiten der Verfahren werden typischerweise vom ersten Schritt, der Generierung der häufigen Ereigniskombinationen, bei weitem dominiert (vgl. [Agrawal und Srikant, 1994a]). Optimierungen des zweiten Schritts, der Ableitung der Regeln aus den häufigen Ereigniskombinationen, sind kaum von praktischer Bedeutung. Im Rahmen des Support-Konfidenz-Ansatzes ist es daher üblich, das Assoziationsproblem auf die Generierung aller häufigen Ereigniskombinationen zu reduzieren. So wird in der vorliegenden Arbeit ebenfalls das Assoziationsproblem als das Problem der Bestimmung aller häufigen Ereigniskombinationen und deren Häufigkeiten zu einer Ereignismenge $\mathcal{E}$, einer Datenbank $\mathcal{D}$ von Transaktionen

zu $\mathcal{E}$ und einem vorgegebenen Schwellenwert minsupp für die Mindesthäufigkeit verstanden.

3.1.2 Durchlaufen des Suchraums

Wird das Assoziationsregelproblem wie beschrieben auf das Problem der Generierung aller häufigen Ereigniskombinationen und deren Häufigkeiten reduziert, dann besteht der Suchraum aus allen potentiell häufigen Ereigniskombinationen. Da grundsätzlich jede Ereigniskombination häufig sein kann, umfaßt der Suchraum die Potenzmenge der Menge aller Ereignisse, $\mathcal{P}(\mathcal{E})$, wobei die leere Menge für die Ableitung der Regeln aus den häufigen Ereigniskombinationen keine Bedeutung hat und daher nicht berücksichtigt werden muß. Alle Ereigniskombinationen aus $\mathcal{P}(\mathcal{E}) \setminus \{\emptyset\}$ auf Häufigkeit zu überprüfen ist für große $n = |\mathcal{E}|$ wegen $|\mathcal{P}(\mathcal{E})| = 2^n$ offensichtlich nicht praktisch durchführbar.

Für die Beispieldatenbank aus Tabelle 2.1 mit der Ereignismenge $\mathcal{E} = \{a, b, c, d, e\}$ ist der vollständige Suchraum in Abbildung 3.1 wiedergegeben. Zur Veranschauli-

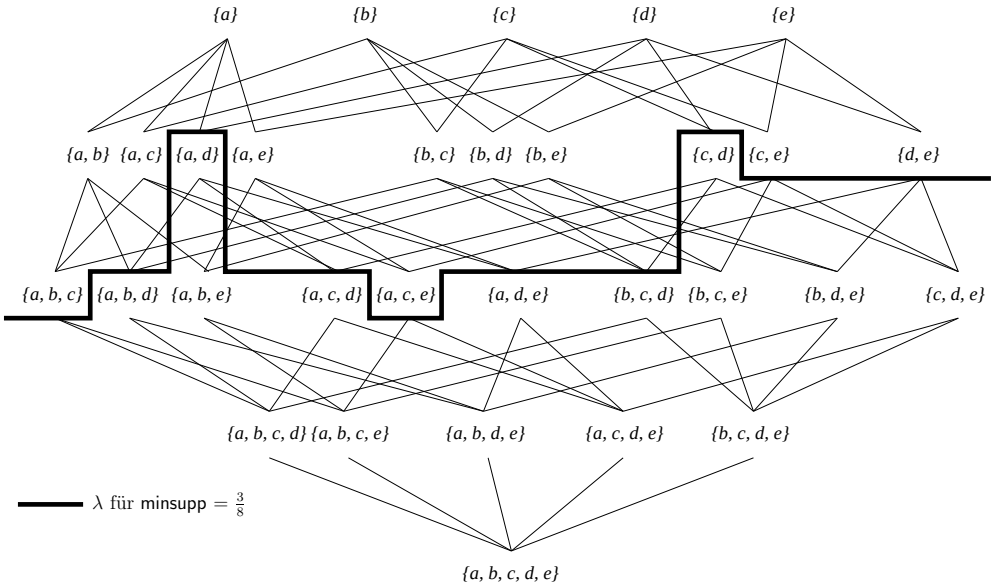


Abbildung 3.1: Suchraum zu $\mathcal{E} = \{a, b, c, d, e\}$ mit Häufigkeitsgrenze λ für minsupp $= \frac{3}{8}$

chung wurde hier minsupp $= \frac{3}{8}$ exemplarisch als Schwellenwert für die Häufigkeit gesetzt. Damit ist beispielsweise die Ereigniskombination $\{b, c\}$ häufig, da diese Teilmenge der drei Transaktionen T_1, T_2 und T_3 ist, das heißt exakt den minimalen Support von $\frac{3}{8}$ erreicht. Umgekehrt erreicht die Ereigniskombination $\{b, c, d\}$ als Teilmenge von lediglich einer Transaktion, T_1, diesen Schwellenwert nicht und

ist somit in diesem Fall nichthäufig. Die Häufigkeiten der einzelnen Ereigniskombinationen sind in Tabelle 3.1 wiedergegeben. Die Häufigkeitsgrenze λ teilt den

Ereignis-kombination	Häufigkeit in Datenbank	Ereignis-kombination	Häufigkeit in Datenbank
$\{a\}$	$6/8 = 75\%$	$\{a,b,d\}$	$1/8 = 12{,}5\%$
$\{b\}$	$6/8 = 75\%$	$\{a,b,e\}$	$2/8 = 25\%$
$\{c\}$	$5/8 = 62{,}5\%$	$\{a,c,d\}$	$2/8 = 25\%$
$\{d\}$	$4/8 = 50\%$	$\{a,c,e\}$	$3/8 = 37{,}5\%$
$\{e\}$	$6/8 = 75\%$	$\{a,d,e\}$	$1/8 = 12{,}5\%$
$\{a,b\}$	$4/8 = 50\%$	$\{b,c,d\}$	$1/8 = 12{,}5\%$
$\{a,c\}$	$5/8 = 62{,}5\%$	$\{b,c,e\}$	$1/8 = 12{,}5\%$
$\{a,d\}$	$2/8 = 25\%$	$\{b,d,e\}$	$2/8 = 25\%$
$\{a,e\}$	$4/8 = 50\%$	$\{c,d,e\}$	$1/8 = 12{,}5\%$
$\{b,c\}$	$3/8 = 37{,}5\%$	$\{a,b,c,d\}$	$1/8 = 12{,}5\%$
$\{b,d\}$	$3/8 = 37{,}5\%$	$\{a,b,c,e\}$	$1/8 = 12{,}5\%$
$\{b,e\}$	$4/8 = 50\%$	$\{a,b,d,e\}$	0
$\{c,d\}$	$2/8 = 25\%$	$\{a,c,d,e\}$	$1/8 = 12{,}5\%$
$\{c,e\}$	$3/8 = 37{,}5\%$	$\{b,c,d,e\}$	0
$\{d,e\}$	$3/8 = 37{,}5\%$	$\{a,b,c,d,e\}$	0
$\{a,b,c\}$	$3/8 = 37{,}5\%$		

Tabelle 3.1: Häufigkeiten der Ereigniskombinationen zur Beispieltransaktionsdatenbank

Suchraum $\mathcal{P}(\{a,b,c,d,e\}) \setminus \{\emptyset\}$ aus Abbildung 3.1 in häufige Ereigniskombinationen oberhalb und nichthäufige Ereigniskombinationen unterhalb von λ.

Der Verlauf der Häufigkeitsgrenze λ ist abhängig von der Datenbank $\mathcal{D}$ und dem Schwellenwert $\mathsf{minsupp}$. Die Veranschaulichung anhand einer Häufigkeitsgrenze ist wegen der Abgeschlossenheitseigenschaft des Supports für jede Datenbank $\mathcal{D}$ möglich.

Als *Grenzmenge (border set)* $\mathcal{B}$ wird die Menge von Ereigniskombinationen aus dem Suchraum bezeichnet, für deren sämtliche Elemente X gilt, daß alle echten Teilmengen von X häufig und alle echten Obermengen von X nichthäufig sind (vgl. [Toivonen, 1996b, S. 40f.]):

$$\mathcal{B} = \{X \in \mathcal{P}(\mathcal{E}) \setminus \{\emptyset\} \mid \forall H \subsetneq X : \mathsf{supp}_{\mathcal{D}}(H) \geq \mathsf{minsupp} \ \wedge \ \forall K \supsetneq X : \mathsf{supp}_{\mathcal{D}}(K) < \mathsf{minsupp}\}$$

Die Grenzmenge $\mathcal{B}$ wird von der Häufigkeitsgrenze λ ebenfalls in häufige und nichthäufige Ereigniskombinationen aufgeteilt. Die Teilmenge der häufigen Ereigniskombinationen aus $\mathcal{B}$ wird als der *positive Rand (positive border)*, die Menge

der nichthäufigen Ereigniskombinationen als der *negative Rand (negative border)* bezeichnet.

Heutige Verfahren nutzen λ, um den Suchraum effizient zu beschneiden. Ist λ zu einer Datenbank $\mathcal{D}$ und einem vorgegebenen Schwellenwert für die Häufigkeit bekannt, dann ist zur vollständigen Erzeugung der häufigen Ereigniskombinationen sowie deren Häufigkeiten lediglich die Betrachtung der Ereigniskombinationen oberhalb – im Sinne der Darstellung in Abbildung 3.1 – von λ erforderlich.

Sei $\mathsf{map} : \mathcal{E} \to \{1, \ldots, |\mathcal{E}|\}$ eine bijektive Abbildung, die jedes Ereignis aus $\mathcal{E}$ auf eine natürliche Zahl abbildet. Über die Relation $<$ auf $\mathbb{N}$ sind die Ereignisse aus $\mathcal{E}$ dann vollständig geordnet. Für das Beispiel in diesem Abschnitt bilde map jedes Ereignis $\{a, b, c, d, e\}$ auf dessen Position im Alphabet ab, also a auf 1, b auf 2 etc. Außerdem sei für jedes $X \subseteq \mathcal{E}$ folgende Abbildung definiert (vgl. [Hipp et al., 1998]):

$$X.\mathsf{item} : \{1, \ldots, |X|\} \to \mathcal{E} : n \mapsto X.\mathsf{item}_n,$$

wobei $X.\mathsf{item}_n$ für das n-te Ereignis der aufsteigend sortierten Ereignisse $x \in X$ steht. Das n-Präfix $P_n(X)$ einer Ereigniskombination $X \subseteq \mathcal{E}, n \in \{0, 1, \ldots, |X| - 1\}$, sei dann definiert durch

$$P_n(X) = \{X.\mathsf{item}_m \mid 1 \leq m \leq n\},$$

wobei insbesondere $P_0(X) = \emptyset$ gilt.

Damit läßt sich eine Äquivalenzrelation auf dem Suchraum $\mathcal{P}(\mathcal{E}) \setminus \{\emptyset\}$ ableiten: Zwei n-Ereigniskombinationen, $n > 0$, gehören zu derselben Äquivalenzklasse, wenn sie ein gemeinsames $(n-1)$-Präfix haben, also bis auf das n-te Ereignis übereinstimmen. Beispielsweise gehören $\{a, b, c\}$ und $\{a, b, d\}$ zu der durch das Präfix $\{a, b\}$ eindeutig bestimmten Äquivalenzklasse $E(\{a, b\})$. Formal sind auf dem Suchraum $\mathcal{P}(\mathcal{E}) \setminus \emptyset$ die Äquivalenzklassen $E(P), P$ Präfix, $E(P) \neq \emptyset$[3], wie folgt definiert:

$$E(P) = \{X \subseteq \mathcal{E} \mid |X| = |P| + 1 \text{ und } P \text{ ist ein Präfix von } X\}.$$

Daß es sich um Äquivalenzklassen handelt, läßt sich wie folgt beweisen: Sei

$$R = \{(X', X'') \in \mathcal{P}(\mathcal{E}) \setminus \{\emptyset\} \times \mathcal{P}(\mathcal{E}) \setminus \{\emptyset\} \mid \exists P \subseteq \mathcal{E} : X', X'' \in E(P)\}$$

eine Relation auf den Ereigniskombinationen aus $\mathcal{P}(\mathcal{E}) \setminus \{\emptyset\}$. R ist via Gleichheit definiert, damit reflexiv, symmetrisch und transitiv, und folglich Äquivalenzrelation mit den Äquivalenzklassen $E(P)$. □

[3]Für jede Menge von Ereignissen $\mathcal{E}$ gilt immer $E(P) = \emptyset$, wenn P das im Sinne der Relation $<$ größte Ereignis enthält. Im Beispiel gilt unter anderem $E(\{e\}) = \emptyset$ und $E(\{a, e\}) = \emptyset$. Im folgenden sollen lediglich die nichtleeren $E(P)$ betrachtet werden.

Gilt wie bisher $\mathcal{E} = \{a, b, c, d, e\}$, dann folgt $E(\emptyset) = \{\{a\}, \{b\}, \{c\}, \{d\}, \{e\}\}$, $E(\{a\}) = \{\{a, b\}, \{a, c\}, \{a, d\}, \{a, e\}\}$, $E(\{b\}) = \{\{b, c\}, \{b, d\}, \{b, e\}\}$ etc. Für die Beispieldatenbank aus Tabelle 2.1 und den zugehörigen Suchraum aus Abbildung 3.1 sind die Äquivalenzklassen in Abbildung 3.2 vollständig dargestellt. Wieder ist $\mathsf{minsupp} = \frac{3}{8}$ exemplarisch als Schwellenwert für die Mindesthäufigkeit gesetzt.

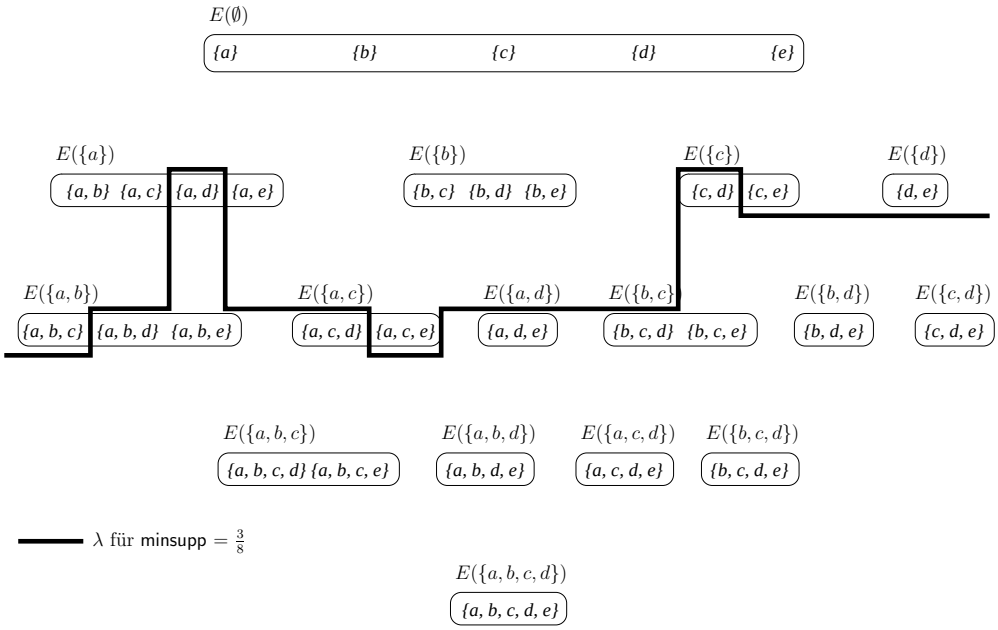


Abbildung 3.2: Äquivalenzklassen zu $\mathcal{E} = \{a, b, c, d, e\}$

Auf der Menge aller Äquivalenzklassen $E(P)$, wird im folgenden ein Baum definiert: Eine Klasse $E(P'')$ sei genau dann Sohn einer Klasse $E(P')$, wenn $P'' \in E(P')$. So ist beispielsweise $E(\{a, c\})$ Sohn von $E(\{a\})$, wegen $\{a, c\} \in E(\{a\})$. Da aufgrund der Äquivalenzklasseneigenschaften jede Ereigniskombination aus $\mathcal{P}(\mathcal{E}) \setminus \{\emptyset\}$ in genau einer Äquivalenzklasse $E(P)$ und $\emptyset$ in keiner der Äquivalenzklassen enthalten ist[4] (vgl. dazu auch die Abbildungen 3.1 und 3.2) folgt:

(a) $E(\emptyset)$ ist die einzige Klasse ohne Vater.

(b) Jede Klasse $E(P), P$ Präfix, $P \neq \emptyset$, hat genau einen Vater.

Der resultierende Graph ist folglich ein Baum auf sämtlichen Äquivalenzklassen $E(P), P$ Präfix, mit Wurzel $E(\emptyset)$, wie in Abbildung 3.3 dargestellt.

[4]Eine Äquivalenzklasse $E(P)$ kann definitionsgemäß nie die leere Menge enthalten, da stets $|\emptyset| = 0 \neq |P| + 1$ gilt.

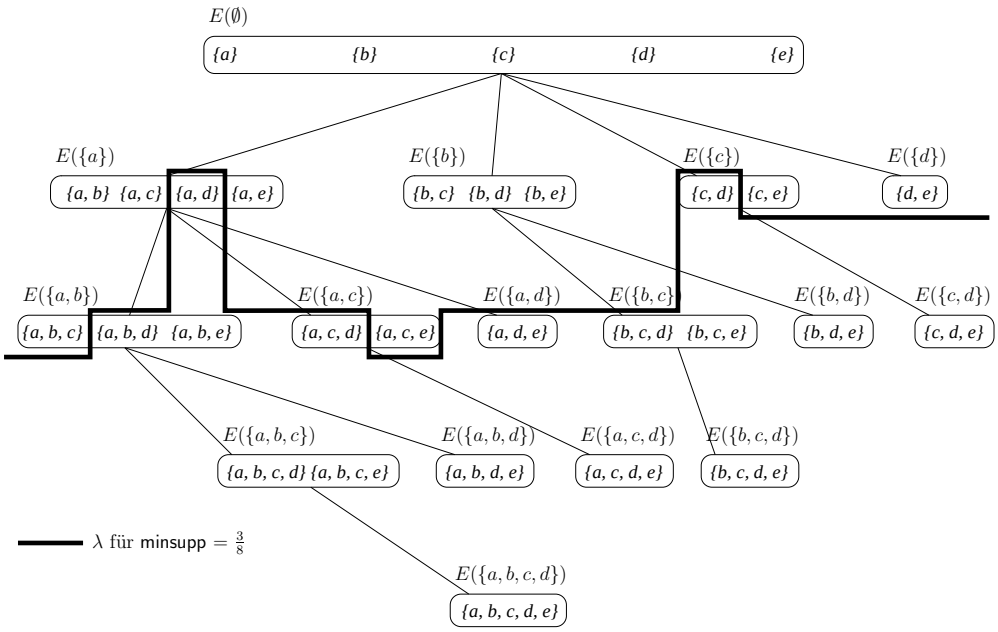


Abbildung 3.3: Baum auf den Äquivalenzklassen zu $\mathcal{E} = \{a, b, c, d, e\}$

□

Ist $E(P'')$ Sohn von $E(P')$, dann gilt außerdem:

$$\forall X \in E(P'') : \exists V, W \in E(P') \text{ mit } V \cup W = X.$$

Das heißt, sämtliche Elemente aus der Sohnklasse $E(P'')$ können durch Vereinigung zweier Ereigniskombinationen aus der Vaterklasse $E(P')$ erzeugt werden.
Beweis:

$$\begin{aligned} & E(P'') \text{ Sohn von } E(P') \\ \Rightarrow\ & \forall X \in E(P'') : P_{|X|-2}(X) = P' \\ \Rightarrow\ & \forall X \in E(P'') \exists v, w \in \mathcal{E} : \max(P') < v < w \wedge X = P' \cup \{v\} \cup \{w\} \\ \Rightarrow\ & \text{Behauptung} \end{aligned}$$

□

Unter anderem folgt aus obiger Behauptung in Verbindung mit der Abgeschlossenheitseigenschaft des Supports, daß, sobald eine Äquivalenzklasse weniger als zwei häufige Ereigniskombinationen enthält, sämtliche Äquivalenzklassen unterhalb dieser Äquivalenzklasse – im Sinne der Darstellung in den Abbildungen 3.1 bis 3.3 – keine häufigen Ereigniskombinationen mehr enthalten können und damit die Häufigkeitsgrenze λ erreicht ist.

Für die Suche nach der Häufigkeitsgrenze λ entlang der Pfade des Baums auf den Äquivalenzklassen eignet sich sowohl eine *Breitensuche (breadth-first search)* als auch eine *Tiefensuche (depth-first search)*. Ausgangspunkt ist immer die Wurzel $E(\emptyset)$. Von dort aus werden entweder in jedem Schritt die Äquivalenzklassen einer Tiefe gleichzeitig betrachtet (Breitensuche) oder die Pfade von der Wurzel zu den Blättern werden einzeln rekursiv verfolgt (Tiefensuche). Ist der exakte Verlauf der Häufigkeitsgrenze λ zusammen mit den Häufigkeiten sämtlicher Ereigniskombinationen oberhalb von λ bestimmt, dann sind alle häufigen Ereigniskombinationen sowie deren Häufigkeiten bezüglich der Transaktionen aus der Datenbank $\mathcal{D}$ und dem vorgegebenen Schwellenwert $\mathsf{minsupp}$ bekannt.

Zur Illustration sei folgendes Beispiel gegeben, wobei die Häufigkeiten wieder basierend auf den Transaktionen aus Tabelle 2.1 bestimmt werden. Die Häufigkeitsgrenze sei ebenfalls durch $\mathsf{minsupp} = \frac{3}{8}$ vorgegeben. Im Baum aus Abbildung 3.3 beginnt die Suche nach λ in der Äquivalenzklasse $E(\emptyset)$, der Wurzel des Baums. Zunächst werden die Häufigkeiten der Ereigniskombinationen aus $E(\emptyset)$, das heißt der einzelnen Ereignisse aus $\mathcal{E}$, bestimmt. Es gilt $\mathsf{supp}_{\mathcal{D}}(\{a\}) = \frac{6}{8}$, $\mathsf{supp}_{\mathcal{D}}(\{b\}) = \frac{6}{8}$, $\mathsf{supp}_{\mathcal{D}}(\{c\}) = \frac{5}{8}$ etc. Im Beispiel sind alle Ereignisse häufig. Das weitere Vorgehen richtet sich nach der gewählten Suchstrategie.

Bei einer Breitensuche werden im zweiten Schritt die Klassen $E(\{a\})$, $E(\{b\})$, $E(\{c\})$ und $E(\{d\})$ gleichzeitig betrachtet. $E(\{a\})$ sowie $E(\{b\})$ enthalten jeweils drei häufige Ereigniskombinationen und sowohl $E(\{c\})$ als auch $E(\{d\})$ jeweils genau eine häufige Ereigniskombination. Folglich werden im dritten Schritt lediglich die Sohnklassen von $E(\{a\})$ und $E(\{b\})$ betrachtet, wobei $E(\{a, d\})$ direkt verworfen werden kann, da $\{a, d\}$ nichthäufig ist. Es genügt, die Häufigkeiten der Ereigniskombinationen in $E(\{a, b\})$, $E(\{a, c\})$ sowie $E(\{b, c\})$ und $E(\{b, d\})$ zu bestimmen. Ein vierter Schritt erübrigt sich, da keine der im dritten Schritt betrachteten Klassen mehr als eine häufige Ereigniskombination enthält. Es kann auf den genauen Verlauf von λ geschlossen werden, denn die verbleibenden Klassen $E(\{a, b, c\})$, $E(\{a, b, d\})$, $E(\{a, c, d\})$, $E(\{b, c, d\})$ und $E(\{a, b, c, d\})$ liegen mit Sicherheit unterhalb von λ.

Bei einer Tiefensuche werden dieselben Klassen durchlaufen, lediglich die Reihenfolge ist eine andere. Zu welcher Klasse jeweils abgestiegen wird, ist im allgemeinen nicht festgelegt. Nach der Wurzel $E(\emptyset)$ könnte beispielsweise $E(\{a\})$ besucht werden. Diese Klasse enthält mehrere häufige Ereigniskombinationen und als nächstes wird beispielsweise zu $E(\{a, b\})$ abgestiegen. Diese Klasse enthält nur eine häufige Ereigniskombination, so daß deren Sohnklassen nicht weiter betrachtet werden müssen, das heißt die Klassen $E(\{a, b, c\})$ und $E(\{a, b, d\}$ mit Sicherheit keine häufigen Ereigniskombinationen enthalten. Der aktuelle Zweig der Rekursion bricht in $E(\{a, b\})$ ab und setzt in $E(\{a\})$ wieder auf, von wo beispielsweise zu $E(\{a, c\})$ abgestiegen wird. Dieses Vorgehen wird fortgesetzt, bis sämtliche Äquivalenzklassen besucht sind, die eventuell häufige Ereigniskombinationen enthalten.

Die vorgestellten Suchstrategien können dahingehend optimiert werden, daß nur die Ereigniskombinationen betrachtet werden, welche tatsächlich in den Transaktionen vorkommen. Beim Abstieg im Suchraum werden bei einer sogenannten *datenorientierten Suche (data oriented search)* die Ereigniskombinationen von vornherein ausgeschlossen, welche nicht Teilmenge mindestens einer der zugrundeliegenden Transaktionen sind.

Das Potential der in diesem Abschnitt beschriebenen Vorgehensweise wird offensichtlich, wenn, statt des bisher aus Übersichtlichkeitsgründen mit $|\mathcal{E}| = 5$ sehr klein gewählten Beispiels, realistischere Ereignismengen $\mathcal{E}$ betrachtet werden. Die dem beschriebenen Ansatz zugrundeliegende Annahme ist, daß in praktischen Anwendungen zwar die maximal mögliche Mächtigkeit häufiger Ereigniskombinationen mit $|\mathcal{E}|$ sehr groß ist, aber trotzdem die häufigen Ereigniskombinationen im Mittel vergleichsweise wenige Ereignisse enthalten. So kann $\mathcal{E}$ beispielsweise über 100.000 Ereignisse enthalten, während zugleich die häufigen Ereigniskombinationen im Schnitt lediglich aus weniger als 10 Ereignissen bestehen (vgl. dazu auch Abschnitt 3.3.1). Entsprechend liegt die Häufigkeitsgrenze λ weit im oberen Bereich des Suchraums, also in der Nähe der Wurzel, so daß in praktischen Anwendungen mit den beschriebenen Suchstrategien typischerweise nur ein sehr kleiner Teil des Suchraums betrachtet werden muß.

Es bleibt anzumerken, daß weitere Maße (vgl. Abschnitt 2.2.2) nicht ohne weiteres für das Beschneiden des Suchraums herangezogen werden können. Der Support weist als einziges der etablierten Maße die Abgeschlossenheitseigenschaft auf, welche Voraussetzung für die Existenz einer eindeutigen Häufigkeitsgrenze λ ist (vgl. dazu [Brin et al., 1997a]).

3.1.3 Bestimmung von Häufigkeiten

Während des Durchlaufens des Suchraums werden die Häufigkeiten verschiedener potentiell häufiger Ereigniskombinationen bestimmt. Eine Ereigniskombination K heiße *Kandidat (candidate, candidate itemset)*, sobald vorgesehen ist, die Häufigkeit dieser Ereigniskombination zu bestimmen. Für das Bestimmen der Häufigkeiten bieten sich zwei grundsätzliche Vorgehensweisen in verschiedenen Varianten an:

Häufigkeitsbestimmung durch direktes Zählen von Vorkommen

Zum einen kann die Häufigkeit eines Kandidaten anhand der vorliegenden Datenbank $\mathcal{D}$ durch *direktes Zählen von Vorkommen* bestimmt werden. Dazu wird der Kandidat in den Transaktionen aus $\mathcal{D}$ gesucht und jedes Vorkommen des Kandidaten als Teilmenge einer Transaktion gezählt. Aus Effizienzgründen werden Kandi-

daten im allgemeinen zu Kandidatenmengen zusammengefaßt, deren Häufigkeiten gemeinsam gezählt werden (vgl. Abschnitt 3.1.4).

Indirekte Häufigkeitsbestimmung durch Schneiden von Transaktionsmengen

Alternativ kann die Häufigkeit eines Kandidaten auch *indirekt durch Schneiden von Transaktionsmengen* bestimmt werden. Dazu wird die Datenbank $\mathcal{D}$ nicht als Menge von Transaktionen[5] betrachtet, die Ereignisse enthalten, sondern als Menge von Ereignissen, die in Transaktionen enthalten sind.

In Abbildung 3.4 ist die Beispieldatenbank aus Tabelle 2.1 in Form einer Matrix dargestellt. Es gilt $d_{ij} = 1$, wenn die Transaktion $i \in \mathcal{D}$ das Ereignis $j \in \mathcal{E}$

$$\begin{pmatrix} d_{T_1,a} & d_{T_1,b} & \cdots & d_{T_1,e} \\ d_{T_2,a} & d_{T_2,b} & \cdots & d_{T_2,e} \\ \vdots & \vdots & \ddots & \vdots \\ d_{T_8,a} & d_{T_8,b} & \cdots & d_{T_8,e} \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 \end{pmatrix}$$

Abbildung 3.4: Matrixdarstellung einer Transaktionsdatenbank

enthält, sonst $d_{ij} = 0$. Aus dieser Matrix läßt sich nicht nur die transaktionsbasierte Darstellung horizontal ableiten, sondern vertikal auch die ereignisbasierte Darstellung. Letztere ist für die Beispieldatenbank aus Tabelle 2.1 und die Matrix aus Abbildung 3.4 in Tabelle 3.2 wiedergegeben (vgl. auch [Mannila et al., 1994a], [Hipp et al., 2000a], [Bykowski und Rigotti, 2001]). Zu jedem Ereignis wird die *Transaktionsmenge (transaction set)*, das heißt die Menge aller Transaktionen, die dieses Ereignis enthalten, angegeben. Da die Datenbank $\mathcal{D}$ als Multimenge gleiche Transaktionen mehrfach enthalten kann, müssen die Transaktionsmengen formal gesehen ebenfalls als Multimengen aufgefaßt werden. Alternativ werden im folgenden die einzelnen Transaktionen aus der Datenbank jeweils durch eindeutige Bezeichner $T_1, \ldots, T_{|\mathcal{D}|}$ identifiziert. Die Datenbank $\mathcal{D}$ wird hier also vereinfacht als Menge von Bezeichnern verstanden. Auf diese Weise kann eine Transaktionsmenge

[5] Formal handelt es sich wieder um eine Multimenge von Transaktionen, da Transaktionen, die in den Ereignissen übereinstimmen, in der Datenbank $\mathcal{D}$ vorkommen dürfen.

Ereignis	Transaktionsmenge
a	$\{T_1, T_2, T_3, T_4, T_5, T_6\}$
b	$\{T_1, T_2, T_3, T_4, T_7, T_8\}$
c	$\{T_1, T_2, T_3, T_5, T_6\}$
d	$\{T_1, T_5, T_7, T_8\}$
e	$\{T_2, T_4, T_5, T_6, T_7, T_8\}$

Tabelle 3.2: Ereignisbasierte Darstellung einer Transaktionsdatenbank

mehrere in den Ereignissen übereinstimmende Transaktionen enthalten, ohne formal als Multimenge definiert zu sein. Beispielsweise enthält die Transaktionsmenge zu dem Ereignis b die Transaktionen T_7 und T_8, die sich lediglich im Bezeichner unterscheiden (vgl. Abbildung 3.4 und Tabelle 3.2).

Eine Transaktionsmenge existiert ebenfalls zu jeder Ereigniskombination. Sei $\mathsf{tids}_{\mathcal{D}}$ eine Abbildung mit

$$\mathsf{tids}_{\mathcal{D}} : \mathcal{P}(\mathcal{E}) \to \mathcal{P}(\mathcal{D}) : X \mapsto X.\mathsf{tids}_{\mathcal{D}},$$

die jede Ereigniskombination X auf ihre Transaktionsmenge $X.\mathsf{tids}_{\mathcal{D}}$ bezüglich der Datenbank $\mathcal{D}$ abbildet, also auf die Menge aller Bezeichner der Transaktionen, welche die Ereigniskombination X enthalten. Insbesondere gilt $\emptyset.\mathsf{tids}_{\mathcal{D}} = \mathcal{D}$.

Ist die Transaktionsmenge zu einer Ereigniskombination X bekannt, ergibt sich die Häufigkeit von X aus

$$\mathsf{supp}_{\mathcal{D}}(X) = \frac{|X.\mathsf{tids}_{\mathcal{D}}|}{|\mathcal{D}|}.$$

Die Transaktionsmengen für die einzelnen Ereignisse, das heißt für die Ereigniskombinationen X mit $|X| = 1$, werden üblicherweise direkt aus der Datenbank $\mathcal{D}$ abgeleitet.

Allgemein gilt definitionsgemäß für eine Ereigniskombination K und n Ereigniskombinationen $X_i, i = 1, \ldots, n$, $n \in \mathbb{N}$:

$$K = \bigcup_{i=1}^{n} X_i \quad \Rightarrow \quad K.\mathsf{tids}_{\mathcal{D}} = \bigcap_{i=1}^{n} X_i.\mathsf{tids}_{\mathcal{D}}.$$

Insbesondere gilt für K, X, Y mit $K = X \cup Y$:

$$K.\mathsf{tids}_{\mathcal{D}} = X.\mathsf{tids}_{\mathcal{D}} \cap Y.\mathsf{tids}_{\mathcal{D}}.$$

Wie im vorangehenden Abschnitt gezeigt, kann jede Ereigniskombination aus einer Sohnklasse $E(P'')$ durch Vereinigung zweier Ereigniskombinationen aus der

zugehörigen Vaterklasse $E(P')$ dargestellt werden. Die Transaktionsmengen zu Kandidaten K mit $|K| \geq 2$ lassen sich folglich während des Abstiegs im Suchraum jeweils aus den Transaktionsmengen zu zwei Ereigniskombinationen aus der jeweiligen Vaterklasse ableiten. Die entsprechenden Transaktionsmengen sind zu den Ereigniskombinationen aus den Vaterklassen bereits bekannt, wenn deren Häufigkeiten ebenfalls durch Schnittmengenbildung bestimmt beziehungsweise die Transaktionsmengen für die Ereigniskombinationen der Mächtigkeit 1 direkt aus der Datenbank erzeugt wurden.

Sollen beispielsweise im Suchraum aus Abbildung 3.3 die Häufigkeiten der Ereigniskombinationen aus der Klasse $E(\{a, b\})$ bestimmt werden, dann kann die Transaktionsmenge $\{a, b, c\}.\mathsf{tids}_{\mathcal{D}}$ des Kandidaten $\{a, b, c\}$ durch den Schnitt zweier Transaktionsmengen aus der Vaterklasse $E(\{a\})$ erzeugt werden, vorausgesetzt, diese Transaktionsmengen sind bereits bekannt:

$$\{a, b, c\}.\mathsf{tids}_{\mathcal{D}} = \{a, b\}.\mathsf{tids}_{\mathcal{D}} \cap \{a, c\}.\mathsf{tids}_{\mathcal{D}}.$$

Die Häufigkeit des Kandidaten ergibt sich aus

$$\mathsf{supp}_{\mathcal{D}}(\{a, b, c\}) = \frac{|\{T_1, T_2, T_3, T_4\} \cap \{T_1, T_2, T_3, T_5, T_6\}|}{|\mathcal{D}|} = \frac{3}{8}.$$

Indirekte Häufigkeitsbestimmung durch Ableitung aus Differenzmengen

Eine Variante der indirekten Häufigkeitsbestimmung durch Schnittmengen ist das Ableiten der Häufigkeiten aus sogenannten *Differenzmengen (diffsets)* (vgl. [Zaki und Gouda, 2001]). Die Differenzmenge der Ereigniskombination X bezüglich des Präfixes P von X ist definiert als die Menge aller Transaktionen, die das Präfix P von X, nicht aber X selbst enthalten:[6]

$$D_P(X) = P.\mathsf{tids}_{\mathcal{D}} \setminus X.\mathsf{tids}_{\mathcal{D}}.$$

Die Häufigkeit einer Ereigniskombination X läßt sich aus der Häufigkeit von P und der Differenzmenge $D_P(X)$ ableiten:

$$\mathsf{supp}_{\mathcal{D}}(X) = \mathsf{supp}_{\mathcal{D}}(P) - \frac{|D_P(X)|}{|\mathcal{D}|}.$$

Sei P eine Ereigniskombination und seien x, y Ereignisse mit $x > p, \forall p \in P$ und $y > x$. Nach der obigen Gleichung gilt:

$$\mathsf{supp}_{\mathcal{D}}(P \cup \{x, y\}) = \mathsf{supp}_{\mathcal{D}}(P \cup \{x\}) - \frac{|D_{P \cup \{x\}}(P \cup \{x, y\})|}{|\mathcal{D}|}.$$

[6] Die Notation in [Zaki und Gouda, 2001] erschließt sich teilweise erst aus dem Zusammenhang und wird im Rahmen der vorliegenden Arbeit als nicht zufriedenstellend angesehen. An dieser Stelle wird daher ergänzend eine eigene Notation eingeführt.

Sind die Differenzmengen von $P \cup \{x\}$ und $P \cup \{y\}$ bezüglich des gemeinsamen Präfixes P bekannt, so kann die Differenzmenge von $P \cup \{x, y\}$ bezüglich $P \cup \{x\}$ wie folgt bestimmen werden:

$$D_{P\cup\{x\}}(P \cup \{x, y\}) = D_P(P \cup \{y\}) \setminus D_P(P \cup \{x\}).$$

Beweis[7]: Die Transaktionsmenge $P.\mathsf{tids}_\mathcal{D}$ läßt sich anhand von $(P \cup \{x\}).\mathsf{tids}_\mathcal{D}$, $(P \cup \{y\}).\mathsf{tids}_\mathcal{D}$ und $(P \cup \{x, y\}).\mathsf{tids}_\mathcal{D}$ in vier disjunkte Teilmengen zerlegen (vgl. auch Abbildung 3.5):

$$\begin{aligned}
A &= \{t \in P.\mathsf{tids}_\mathcal{D} \mid t \notin P \cup \{x\}.\mathsf{tids}_\mathcal{D} \wedge t \notin P \cup \{y\}.\mathsf{tids}_\mathcal{D}\},\\
B &= \{t \in P.\mathsf{tids}_\mathcal{D} \mid t \in P \cup \{x\}.\mathsf{tids}_\mathcal{D} \wedge t \notin P \cup \{y\}.\mathsf{tids}_\mathcal{D}\},\\
C &= \{t \in P.\mathsf{tids}_\mathcal{D} \mid t \notin P \cup \{x\}.\mathsf{tids}_\mathcal{D} \wedge t \in P \cup \{y\}.\mathsf{tids}_\mathcal{D}\},\\
D &= \{t \in P.\mathsf{tids}_\mathcal{D} \mid t \in P \cup \{x\}.\mathsf{tids}_\mathcal{D} \wedge t \in P \cup \{y\}.\mathsf{tids}_\mathcal{D}\}.
\end{aligned}$$

Es folgt:

$$\begin{aligned}
D_{P\cup\{x\}}(P \cup \{x, y\}) &= (P \cup \{x\}).\mathsf{tids}_\mathcal{D} \setminus (P \cup \{x, y\}).\mathsf{tids}_\mathcal{D}\\
&= B\\
&= (A \cup B) \setminus (A \cup C)\\
&= (P.\mathsf{tids}_\mathcal{D} \setminus (P \cup \{y\}).\mathsf{tids}_\mathcal{D}) \setminus (P.\mathsf{tids}_\mathcal{D} \setminus (P \cup \{x\}).\mathsf{tids}_\mathcal{D})\\
&= D_P(P \cup \{y\}) \setminus D_P(P \cup \{x\})
\end{aligned}$$

□

Mit den beiden obigen Gleichungen

$$\mathsf{supp}_\mathcal{D}(X) = \mathsf{supp}_\mathcal{D}(P) - \frac{|D_P(X)|}{|\mathcal{D}|}$$

und

$$D_{P\cup\{x\}}(P \cup \{x, y\}) = D_P(P \cup \{y\}) \setminus D_P(P \cup \{x\})$$

ist es somit möglich, aus dem Support von $P \cup \{x\}$ und den Differenzmengen zu $P \cup \{x\}$ und $P \cup \{y\}$ die Häufigkeit von $P \cup \{x, y\}$ direkt abzuleiten. Es genügt demnach sowohl bei der Breiten- als auch bei der Tiefensuche, die Häufigkeiten der ausgezählten Ereigniskombinationen mitzuführen, um die Häufigkeiten der betrachteten Kandidaten anhand der entsprechenden Differenzmengen anstatt basierend auf Transaktionsmengen bestimmen zu können.

Wenn beispielsweise im Suchraum aus Abbildung 3.3 die Häufigkeiten und Differenzmengen der Ereigniskombinationen aus der Klasse $E(\{a, b\})$ bestimmt werden

[7] Für eine alternative Beweisführung siehe [Zaki und Gouda, 2001].

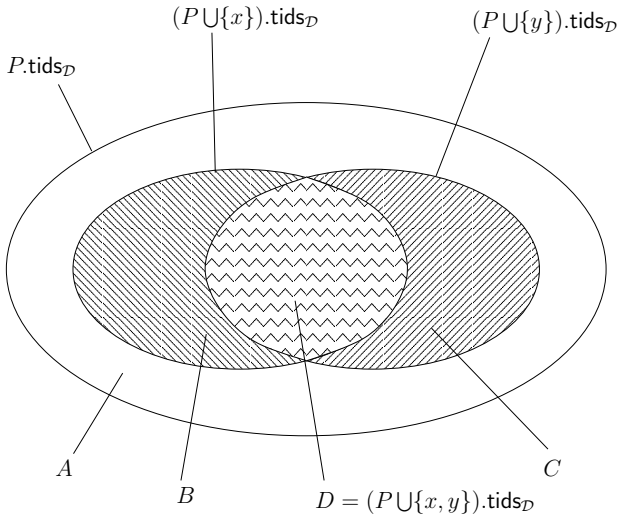


Abbildung 3.5: Die anhand der Transaktionsmengen $(P \cup \{x\}).\mathsf{tids}_{\mathcal{D}}$, $(P \cup \{y\}).\mathsf{tids}_{\mathcal{D}}$ und $(P \cup \{x, y\}).\mathsf{tids}_{\mathcal{D}}$ in vier disjunkte Teilmengen A, B, C, D zerlegte Transaktionsmenge $P.\mathsf{tids}_{\mathcal{D}}$

sollen, wird folgendermaßen verfahren: Die Häufigkeiten der Ereigniskombinationen aus $E(\{a\})$ sowie deren Differenzmengen bezüglich des Präfixes $\{a\}$ sind bereits berechnet. Werden die Transaktionen aus Tabelle 2.1 zugrundegelegt, gilt $\mathsf{supp}_{\mathcal{D}}(\{a, b\}) = \frac{4}{8}$ und $\mathsf{supp}_{\mathcal{D}}(\{a, c\}) = \frac{5}{8}$ sowie

$$\begin{aligned} D_{\{a\}}(\{a, b\}) &= \{a\}.\mathsf{tids}_{\mathcal{D}} \setminus \{a, b\}.\mathsf{tids}_{\mathcal{D}} \\ &= \{T_1, T_2, T_3, T_4, T_5, T_6\} \setminus \{T_1, T_2, T_3, T_4\} \\ &= \{T_5, T_6\} \end{aligned}$$

und

$$\begin{aligned} D_{\{a\}}(\{a, c\}) &= \{a\}.\mathsf{tids}_{\mathcal{D}} \setminus \{a, c\}.\mathsf{tids}_{\mathcal{D}} \\ &= \{T_1, T_2, T_3, T_4, T_5, T_6\} \setminus \{T_1, T_2, T_3, T_5, T_6\} \\ &= \{T_4\}. \end{aligned}$$

Für die Differenzmenge des Kandidaten $\{a, b, c\}$ bezüglich des Präfixes $\{a, b\}$ folgt:

$$\begin{aligned} D_{\{a,b\}}(\{a, b, c\}) &= D_{\{a\}}(\{a, c\}) \setminus D_{\{a\}}(\{a, b\}) \\ &= \{T_4\}. \end{aligned}$$

Für die Häufigkeit des Kandidaten $\{a, b, c\}$ gilt damit:

$$\mathsf{supp}_{\mathcal{D}}(\{a, b, c\}) = \mathsf{supp}_{\mathcal{D}}(\{a, b\}) - \frac{|D_{\{a,b\}}(\{a, b, c\})|}{|\mathcal{D}|} = \frac{4 - 1}{8} = \frac{3}{8}.$$

Entsprechend können die Differenzmengen und die Supportwerte der weiteren Kandidaten bestimmt werden.

Die Auswirkungen der verschiedenen Ansätze zur Häufigkeitsbestimmung auf die Laufzeit werden in Abschnitt 3.3 untersucht und in Abschnitt 4.1 erneut aufgegriffen.

3.1.4 Datenstrukturen

Für die effiziente Implementierung der beschriebenen Such- und Zählstrategien haben sich verschiedene Datenstrukturen etabliert. Die wichtigsten dieser Datenstrukturen werden im folgenden gemeinsam mit den auf ihnen typischen Operationen beschrieben.

Ereigniskombinationen

Ereigniskombinationen sind Mengen von Ereignissen, wobei die Ereignisse selbst durch natürliche Zahlen dargestellt werden, vergleiche dazu auch die Abbildung map aus Abschnitt 3.1.2. Ereigniskombinationen werden gewöhnlich als Arrays von Integerwerten implementiert. Eine häufige Operation auf Ereigniskombinationen ist die Suche nach Teilmengen, beispielsweise ob Ereigniskombination X Ereigniskombination Y als Teilmenge enthält. Um solche Suchoperationen effizient zu unterstützen und um eine eindeutige Repräsentation der Mengen im Sinne von Abschnitt 3.1.2 zu erreichen, werden die für die Ereignisse stehenden Integerwerte in den Arrays aufsteigend sortiert abgelegt.

Transaktionsmengen

Bei der Implementierung von Transaktionsmengen ist grundsätzlich zu unterscheiden, ob die Transaktionen als atomar betrachtet werden oder ob auf die einzelnen Ereignisse in den Transaktionen zugegriffen wird.

Im Fall atomarer Betrachtung der Transaktionen, bietet sich die folgende Herangehensweise an: Zuerst wird jeder Transaktion durch Numerierung eineindeutig ein Integerwerte zugeordnet.[8] Auf diese Weise können Multimengen von Transaktionen, das heißt Transaktionsmengen, die bezüglich der enthaltenen Ereignisse identische Transaktionen enthalten, als einfache Mengen von Integerwerten implementiert werden. Die dominierende Operation auf solchen Transaktionsmengen ist das Bilden von Schnittmengen. Entsprechend werden in der Regel Transaktionsmengen ebenfalls wie Ereigniskombinationen als aufsteigend sortierte Arrays

[8] Praktisch ergibt sich die Numerierung zumeist aus der Reihenfolge der Transaktionen in der Implementierung der Datenbank $\mathcal{D}$.

implementiert (vgl. u. a. [Savasere et al., 1995a], [Savasere et al., 1995b], [Mueller, 1995], [Zaki et al., 1997c], [Zaki et al., 1997a], [Hipp et al., 2000a]). Eine Speicherung der Transaktionsmengen als Bitstrings ist wenig vielversprechend, da die Bitstrings zumeist sehr dünn besetzt wären. Eine geeignete Komprimierung müßte die Bildung von Schnittmengen effizient unterstützen.[9]

Sogenannte *Schnellschnitte (fast intersections)* (vgl. [Zaki et al., 1997a], [Zaki et al., 1997c]) optimieren die Schnittmengenbildung dahingehend, daß nur Schnitte abgeschlossen werden, die zu einer Transaktionsmenge führen, deren Mächtigkeit den minimalen Schwellenwert für den Support erreicht. Ein Schnitt wird abgebrochen, sobald erkennbar ist, daß die resultierende Transaktionsmenge nicht zu einer häufigen Ereigniskombination gehört und diese Transaktionsmenge daher nicht für die nachfolgenden Schritte der Häufigkeitsbestimmung benötigt wird.

Im zweiten Fall, wenn auf die einzelnen Ereignisse in den Transaktionen zugegriffen werden soll, beispielsweise bei der Implementierung einer Transaktionsdatenbank $\mathcal{D}$, wird die Transaktionsmenge gewöhnlich als Array von Ereigniskombinationen umgesetzt. Die typische Operation auf einer solchen Transaktionsmenge ist das sequentielle Durchlaufen, wobei die Transaktionen nacheinander jeweils komplett abgearbeitet werden. Wird eine Transaktionsmenge in einem zusammenhängenden Speicherblock auf einem externen Speichermedium abgelegt, so muß dieser Block für effiziente Zugriffe nicht vollständig im Hauptspeicher liegen, sondern kann während des Durchlaufens der Transaktionen in Teilblöcken sukzessive nachgeladen werden. Es wird eine Art Hauptspeicherfenster über die Daten geschoben, die sich auf dem externen Speichermedium befinden. Durch diese Implementierung ist es möglich, Transaktionsmengen effizient zu bearbeiten, die so groß sind, daß sie nicht vollständig in dem vorhandenen Hauptspeicher abgelegt werden können.

Alternativ können Transaktionsdatenbanken implementiert werden, indem die einzelnen Transaktionen in aufbereiteter und für das Zählen bereits entsprechend aggregierter und optimierter Form gespeichert werden. Diesen Weg geht die Implementierung durch sogenannte *FP-Bäume (fp-tree, frequent pattern-tree)* (vgl. [Han et al., 2000], [Han und Pei, 2000]). Ein FP-Baum faßt die Datenbanktransaktionen aufgrund gemeinsamer Präfixe zusammen. Dabei gehen Informationen verloren, das heißt, es ist nicht mehr möglich, die Ausgangstransaktionen allein anhand eines FP-Baums wiederherzustellen. Trotzdem können die Häufigkeiten sämtlicher Ereigniskombinationen in den Ausgangsdaten mit Hilfe des abgeleiteten FP-Baums bestimmt werden. FP-Bäume unterstützten die Häufigkeitsbestimmung effizient durch Verkettungen der Knoten.

Der Aufbau und die Erzeugung eines FP-Baums aus einer Transaktionsdatenbank soll im folgenden mittels der Beispieldaten aus Tabelle 2.1 veranschaulicht wer-

[9] Letztendlich kann die aufsteigend sortierte Speicherung der Transaktionsnummern in einem Array als eine kompakte Repräsentation eines Bitstrings verstanden werden.

den. Wie in Abschnitt 3.1.2 werden die Transaktionen als geordnet angenommen. Zunächst wird ein einzelner Wurzelknoten als leerer FP-Baum erstellt. Darauf werden nacheinander die einzelnen Transaktionen in den FP-Baum wie folgt eingetragen: Beginnend mit dem ersten Ereignis der betrachteten Transaktion wird für jedes Ereignis im Baum abgestiegen. Existiert bereits ein entsprechender Knoten im Baum, so wird der Zähler dieses Knotens erhöht, ansonsten wird ein neuer Knoten erzeugt und dessen Zähler mit 1 initialisiert.

Für $T_1 = \{a, b, c, d\}$, die erste Transaktion der Beispieldatenbank, wird aus dem leeren FP-Baum der Baum aus Abbildung 3.6(a) erzeugt. Zu jedem Ereignis wird ein neuer, mit 1 initialisierter Knoten erstellt. Die zweite einzutragende Transaktion $\{a, b, c, e\}$ impliziert hingegen nur einen neuen Knoten. Der Anfangspfad über a, b und c ist bereits bekannt und für diese Knoten wird lediglich deren Zähler entsprechend erhöht. Der für das letzte Ereignis e anzulegende Knoten wird an Knoten c angehängt (siehe Abbildung 3.6(b)). Entsprechend werden die Transaktionen $\{a, b, c\}$ und $\{a, b, e\}$ als dritte beziehungsweise vierte Transaktion eingetragen (vgl. die Abbildungen 3.6(c) und 3.6(d)). Sind sämtliche Transaktionen zu dem FP-Baum hinzugefügt (vgl. Abbildung 3.7) und die zu denselben Ereignissen gehörenden Knoten miteinander verkettet, resultiert der in Abbildung 3.8 dargestellte Baum für die Beispieldatenbank.

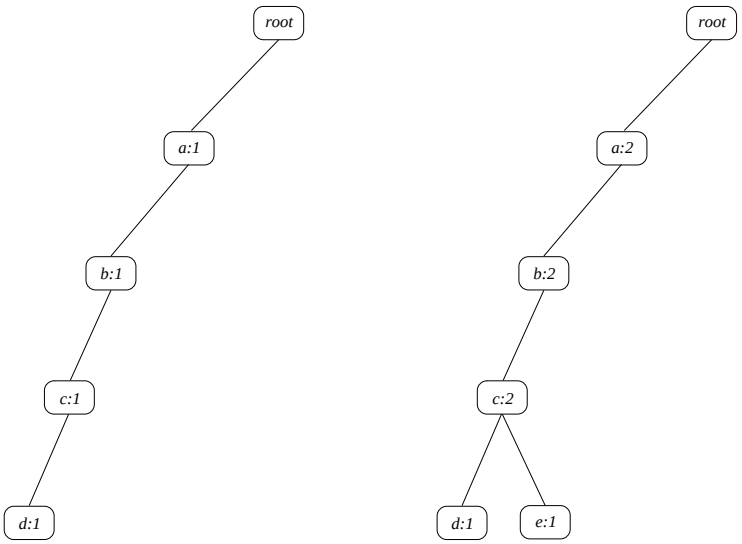
Wie folgt kann ein solcher FP-Baum durch Aufbereitung der hinzuzufügenden Transaktionen kompakter aufbereitet werden:

- Ereignisse, die nichthäufig sind, können nicht Element häufiger Ereigniskombinationen sein. Solche Ereignisse können daher vor dem Hinzufügen zu einem FP-Baum aus den Transaktionen gestrichen werden.

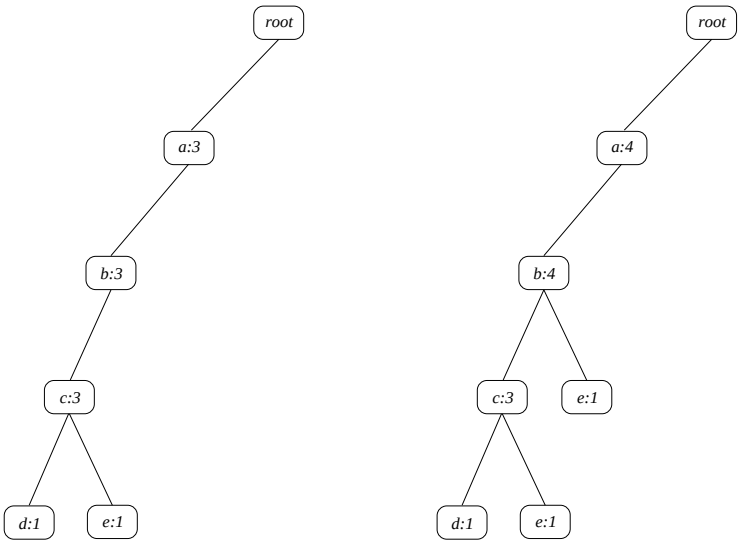
- FP-Bäume zielen darauf, Transaktionen anhand gemeinsamer Präfixe zusammenzufassen. Um die Anzahl gemeinsamer Präfixe zu erhöhen, können vor dem Hinzufügen die Ereignisse innerhalb der Transaktionen absteigend nach ihrer Häufigkeit in der Datenbank sortiert werden. Befinden sich die häufigen Ereignisse „vorne“ in den Transaktionen, so nimmt die Wahrscheinlichkeit für gemeinsam genutzte Pfade zu. Im Beispiel profitiert offensichtlich die Kompaktheit des resultierenden FP-Baums davon, daß a mit einem Vorkommen von 6 als häufigstes Ereignis an erster Stelle in jeder der Transaktionen steht.

Ein FP-Baum als kompakte Darstellung einer Transaktionsdatenbank ermöglicht das effiziente und trotz der Aggregation exakte Zählen von Häufigkeiten. Dazu werden rekursiv sogenannte *bedingte FP-Bäume (conditional fp-trees)* erzeugt.[10] Beginnend mit dem nach der zugrundeliegenden Ordnung größten Ereignis, wird

[10]Der soeben beschriebene FP-Baum kann als der durch die leere Menge bedingte FP-Baum verstanden werden.

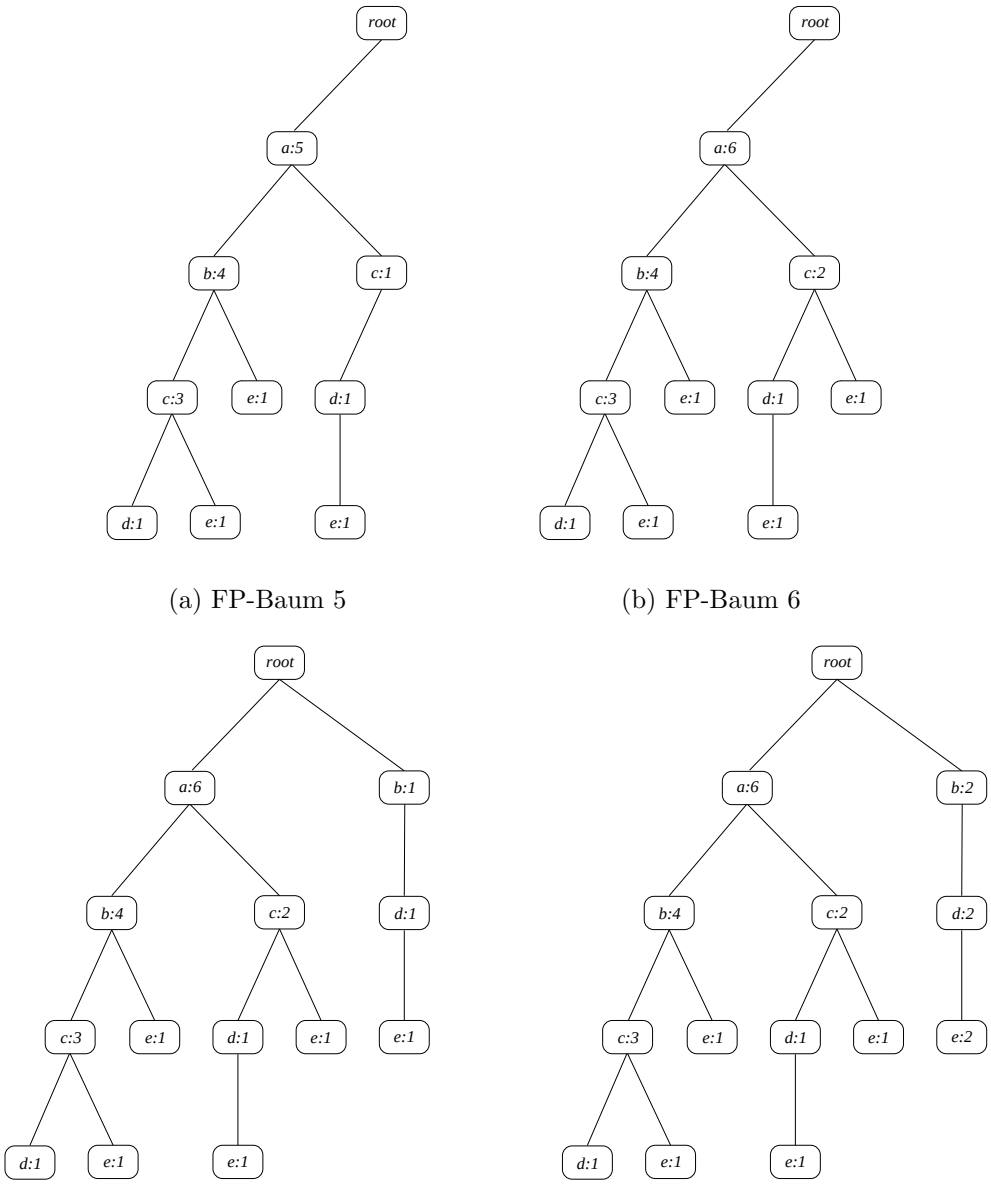


(a) FP-Baum 1 (b) FP-Baum 2



(c) FP-Baum 3 (d) FP-Baum 4

Abbildung 3.6: Schrittweiser Aufbau eines FP-Baums zur Beispieldatenbank (I)



(a) FP-Baum 5

(b) FP-Baum 6

(c) FP-Baum 7

(d) FP-Baum 8

Abbildung 3.7: Schrittweiser Aufbau eines FP-Baums zur Beispieldatenbank (II)

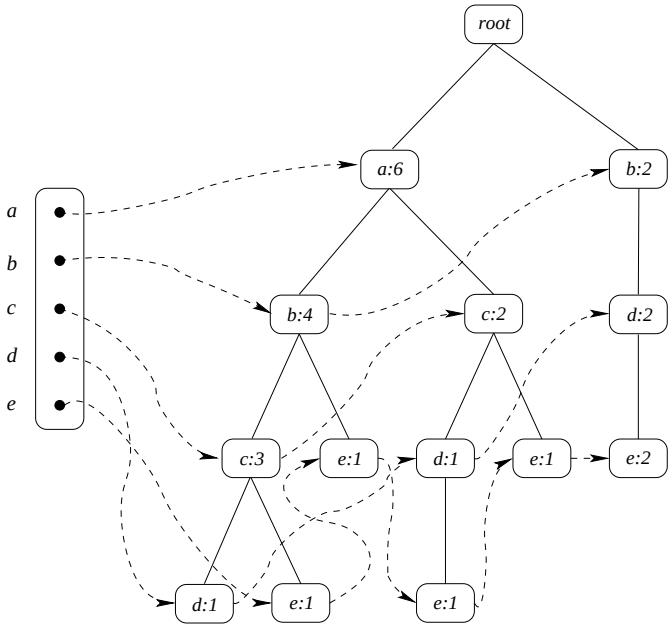


Abbildung 3.8: Vollständiger FP-Baum zur Beispieldatenbank mit Verkettung

in absteigender Reihenfolge für jedes Ereignis die Menge der maximalen[11], dieses Ereignis als im Sinne der Relation $<$ größtes Ereignis enthaltenden und in den Transaktionen vorkommenden Ereigniskombinationen generiert. Im nächsten Schritt wird für diese Ereigniskombinationen der entsprechende FP-Baum aufgebaut. Im Beispiel ist e das größte Ereignis aus $\mathcal{E}$. Liegt für eine Transaktionsdatenbank ein FP-Baum wie in Abbildung 3.8 vor, können die Ereignis e als größtes Ereignis enthaltenden und in den Transaktionen vorkommenden maximalen Ereigniskombinationen direkt abgelesen werden, indem der Verkettung für e gefolgt wird. Der Weg von der Wurzel bis zu jedem der erreichten Knoten entspricht einer Ereigniskombination. Im einzelnen sind es die Ereigniskombinationen $\{a, b, c, e\}$, $\{a, b, e\}$, $\{a, c, d, e\}$, $\{a, c, e\}$, die einmal in den Transaktion vorkommen, sowie die Ereigniskombination $\{b, d, e\}$, die zweimal vorkommt. Die Häufigkeiten lassen sich jeweils als Wert des Zählers in den Knoten bei e ablesen. Die Häufigkeit des Ereignisses e selbst ergibt sich aus der Summe über die Zähler in den verketteten Knoten. Das Ereignis e wird aus jeder der Ereigniskombinationen gestrichen und aus den resultieren Ereigniskombinationen entsprechend dem oben beschriebenen Vorgehen der sogenannte durch $\{e\}$ bedingte FP-Baum erzeugt. Ergebnis ist der in Abbildung 3.9 dargestellte FP-Baum. Das größte Ereignis in diesem Baum ist d

[11] Maximal ist in dem Sinne zu verstehen, daß keine Obermenge zu dieser Ereigniskombination existiert.

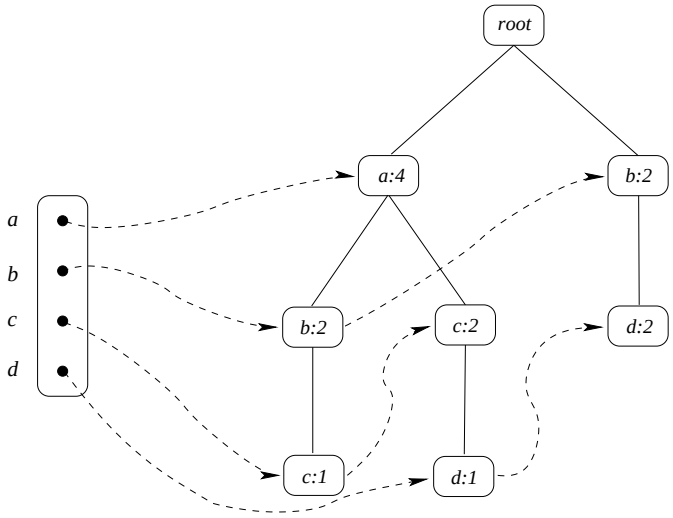


Abbildung 3.9: Der durch die Ereigniskombination $\{e\}$ bedingte FP-Baum

und das beschriebene Verfahren wird rekursiv für diesen FP-Baum wiederholt: Die verkettete Liste zu d führt zu den maximalen Ereigniskombinationen, die in den Transaktionen vorkommen und mit $\{d, e\}$ enden. Dies sind die Ereigniskombinationen $\{a, c, d, e\}$ und $\{b, d, e\}$, die, ablesbar am Zähler bei d, einmal beziehungsweise zweimal in den ursprünglichen Transaktionsdaten vorkommen. Der resultierende, durch $\{d, e\}$ bedingte FP-Baum ist in Abbildung 3.10 dargestellt. Der nächste re-

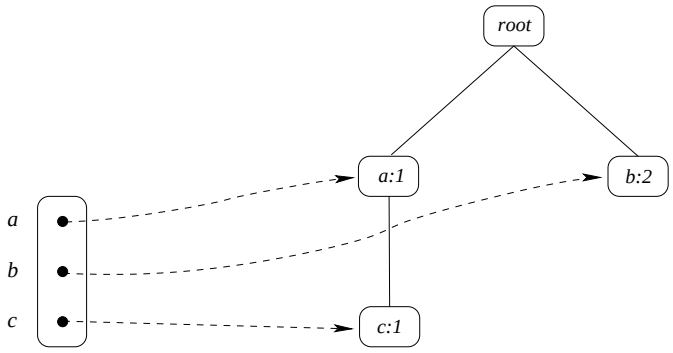


Abbildung 3.10: Der durch die Ereigniskombination $\{d, e\}$ bedingte FP-Baum

kursiv erzeugte FP-Baum ist der durch die Ereigniskombination $\{c, d, e\}$ bedingte FP-Baum. Dieser enthält lediglich einen Pfad, die Kante von der Wurzel zu Knoten a (vgl. Abbildung 3.11), und damit endet der Rekursionszweig. Als Ergebnis können die Häufigkeiten aller mit $\{c, d, e\}$ endenden und in den Transaktionen

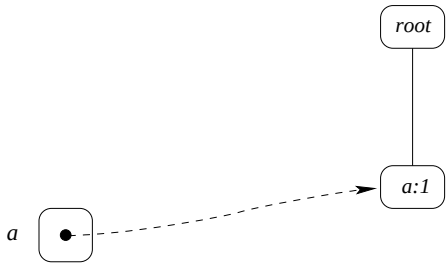


Abbildung 3.11: Der durch die Ereigniskombination $\{c, d, e\}$ bedingte FP-Baum

vorkommenden Ereigniskombinationen abgelesen werden, indem die Potenzmenge ohne die leere Menge der auf diesem Pfad liegenden Ereignisse, in diesem Fall $\mathcal{P}(\{a\}) \setminus \emptyset = \{\{a\}\}$, mit $\{c, d, e\}$ kombiniert wird. Als absolute Häufigkeit für die Ereigniskombination $\{a, c, d, e\}$ ergibt sich somit 1.

Die Rekursion setzt bei dem durch $\{d, e\}$ bedingten FP-Baum wieder auf. Auf das bearbeitete Ereignis c folgt das Ereignis b, das zu einem leeren bedingten FP-Baum führt. Die Rekursion bricht daher ab, und die Häufigkeit 2 der Ereigniskombination $\{b, d, e\}$ kann aus dem Zähler im Knoten von b abgelesen werden. Entsprechend ergibt sich eine Häufigkeit von 1 für $\{a, d, e\}$, und der durch $\{d, e\}$ bedingte FP-Baum ist vollständig bearbeitet. Die Rekursion setzt mit dem Ereignis c in dem durch $\{e\}$ bedingten FP-Baum wieder auf und erzeugt den durch $\{c, e\}$ bedingten FP-Baum. Das Verfahren wird fortgesetzt, bis sämtliche Häufigkeiten bestimmt sind.

Die Berücksichtigung eines Schwellenwertes minsupp für die minimale Häufigkeit beschleunigt das beschriebene Verfahren. Ereignisse, welche die vorgegebene Mindesthäufigkeit nicht erreichen, werden vor Erzeugung der bedingten FP-Bäume ausgefiltert. Dabei wird ihre Häufigkeit nicht mehr auf der Gesamtmenge der Transaktionen bestimmt, sondern auf der Menge der in den bedingten FP-Baum einzutragenden Ereigniskombinationen. Die Häufigkeit der Ereignisse c und d beträgt 3, während a und b eine Häufigkeit von 4 aufweisen, jeweils für die in den durch $\{e\}$ bedingten FP-Baum einzutragenden Ereigniskombinationen. Wird im Beispiel als absoluter minimaler Schwellenwert für die Häufigkeit 4 vorgegeben, dann werden in dem durch $\{e\}$ bedingten FP-Baum aus Abbildung 3.9 bereits die Ereignisse c und d nicht mehr eingetragen, und für die durch $\{c, e\}$ und $\{d, e\}$ bedingten FP-Bäume erfolgt keine Rekursion mehr.

Wird eine Transaktionsdatenbank als FP-Baum aufbereitet und werden dann mit dem oben beschriebenen Verfahren Häufigkeiten bestimmt, so ist eine Vorhersage der auf den Baum erfolgenden Zugriffe nicht trivial und der Zugriff über ein Hauptspeicherfenster ist nicht möglich. Damit ist die Verwendung von FP-Bäum-

en auf solche Transaktionsdatenbanken beschränkt, die aufbereitet vollständig im Hauptspeicher gehalten werden können.[12]

Kandidatenmengen

Als Datenstruktur zur Speicherung von Kandidatenmengen wird in [Agrawal und Srikant, 1994a] ein sogenannter *Hashbaum (hashtree)* vorgeschlagen. Ein Hashbaum speichert Kandidaten gleicher Mächtigkeit, mit dem Ziel, das direkte Zählen von Vorkommen (vgl. dazu Abschnitt 3.1.3) effizient zu unterstützen. Dazu wird jeder Kandidat, der als Ereigniskombination wie beschrieben aufsteigend sortiert in Form eines Arrays von Integerwerten gespeichert wird, zusammen mit einem Zähler für die Häufigkeit im Hashbaum abgelegt. Der Aufbau des Hashbaums ermöglicht es, effizient sämtliche Teilmengen einer Transaktion T in der Menge von Kandidaten zu finden und die zugehörigen Zähler zu erhöhen.

Das direkte Zählen von Vorkommen ist gewöhnlich so implementiert, daß sämtliche Transaktionen aus der Datenbank $\mathcal{D}$ nacheinander bearbeitet und die Zähler der Kandidaten aus der Kandidatenmenge erhöht werden, die in der jeweils bearbeiteten Transaktion enthalten sind. Die dominierende Operation auf einem Hashbaum nimmt als Argument folglich eine Transaktion T, und für jede Teilmenge von T, die im Hashbaum als Kandidat enthalten ist, wird der entsprechende Zähler im Baum um 1 erhöht.

Die Kandidaten sind gemeinsam mit ihren Zählern in Listen abgelegt, welche in den Blättern des Hashbaums gespeichert werden. Die inneren Knoten des Hashbaums enthalten Hashtabellen, wobei jeder Eintrag einer Hashtabelle entweder als Kante auf einen tieferliegenden Knoten zeigt oder undefiniert ist.

Ein leerer Hashbaum besteht nur aus der Wurzel, die eine leere Liste von Kandidaten enthält. Wird ein Kandidat K zu einem Hashbaum hinzugefügt, dann beginnt in der Wurzel des Baums die Suche nach einem geeigneten Blatt, um den Kandidat zu speichern. In jedem inneren Knoten der Tiefe d wird jeweils auf das d-te Ereignis des Kandidaten K, das heißt $K.\mathsf{item}_d$ (vgl. dazu Abschnitt 3.1.2) die Hashfunktion angewendet und so der nächste Knoten mit der Tiefe $d+1$ bestimmt. Es wird im Baum abgestiegen, bis entweder ein Blatt erreicht oder ein Hasheintrag undefiniert ist. Ist ein Blatt erreicht, dann wird der Kandidat K zu der Liste von Kandidaten dieses Blatts hinzugefügt. Übersteigt die Anzahl der in einem Blatt gespeicherten Ereigniskombinationen einen vorgegebenen Schwellenwert, dann wird dieses Blatt in einen inneren Knoten umgewandelt und die gespeicherten Ereigniskombinationen werden auf die neu entstehenden Blätter verteilt. Die neuen Blätter werden

[12] Speziell für die Auslagerung auf Sekundärspeicher entwickelte Varianten des FP-Baums werden in [Han et al., 2000] eingeführt. Diese sind jedoch weit weniger effizient als ursprüngliche FP-Baum aus [Han et al., 2000], welcher der vorliegenden Arbeit zugrunde liegt.

entsprechend mit dem umgewandelten Knoten über dessen Hashtabelle verknüpft. Wird hingegen kein Blatt erreicht, sondern ist auf dem Weg von der Wurzel zu den Blättern ein Hasheintrag undefiniert, dann wird an dieser Stelle ein leeres Blatt eingefügt und der Kandidat K in diesem gespeichert.

In Abbildung 3.12 ist ein Beispiel für einen Hashbaum dargestellt, der Kandidaten der Mächtigkeit 3 speichert. Die Ereignisse aus der zugrundeliegenden Ereignis-

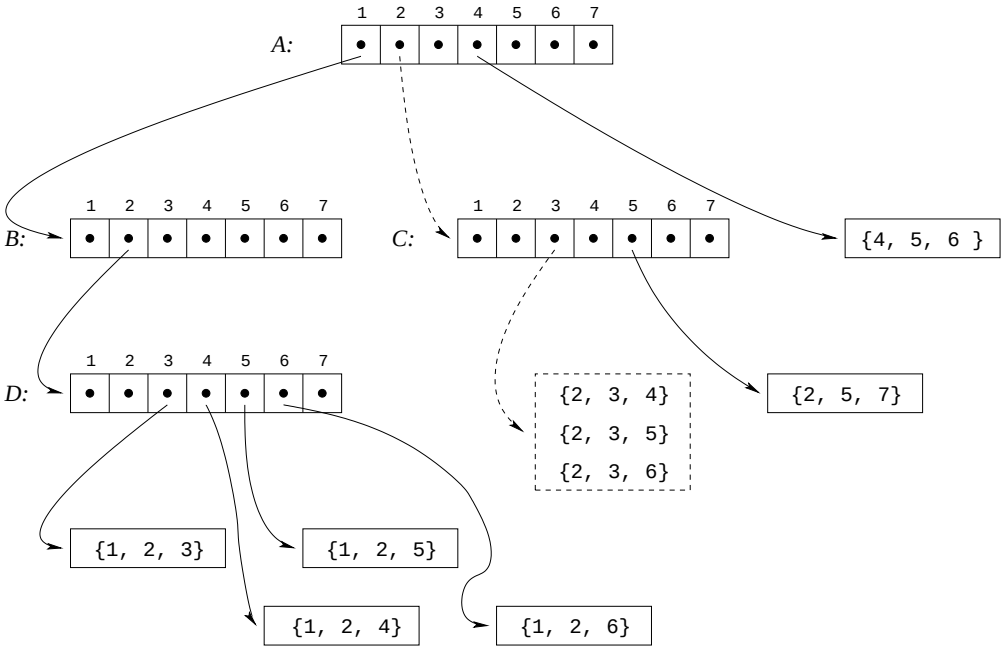


Abbildung 3.12: Beispiel für einen Ereigniskombinationen der Mächtigkeit 3 speichernden Hashbaum

menge sind auf die natürlichen Zahlen $1, \ldots, 7$ abgebildet, und die Hashtabellen in den inneren Knoten bestehen jeweils aus 7 Einträgen. Der n-te Eintrag in jeder der Hashtabellen verweist für das als n kodierte Ereignis auf den nächsten Knoten. Soll zu dem Hashbaum aus Abbildung 3.12 beispielsweise die Ereigniskombination $\{2, 3, 7\}$ hinzugefügt werden, dann werden beginnend in der Wurzel nacheinander für die Ereignisse $\{2, 3, 7\}.\mathsf{item}_1 = 2$ und $\{2, 3, 7\}.\mathsf{item}_2 = 3$ die Hashwerte berechnet und es wird entsprechend im Baum abgestiegen. Dadurch wird das gestrichelt markierte Blatt über die Knoten A und C erreicht. Dann wird der Kandidat zu dem gefundenen Blatt hinzugefügt, oder wenn, wie hier angenommen, der Schwellenwert für die maximale Anzahl von Ereigniskombinationen pro Blatt überschritten ist, das Blatt in einen inneren Knoten umgewandelt. In letzterem Fall werden die in dem umgewandelten Blatt gespeicherten Ereigniskombinationen zusammen mit dem hinzukommenden Kandidaten auf die neu entstehenden Blätter verteilt (siehe Abbildung 3.13).

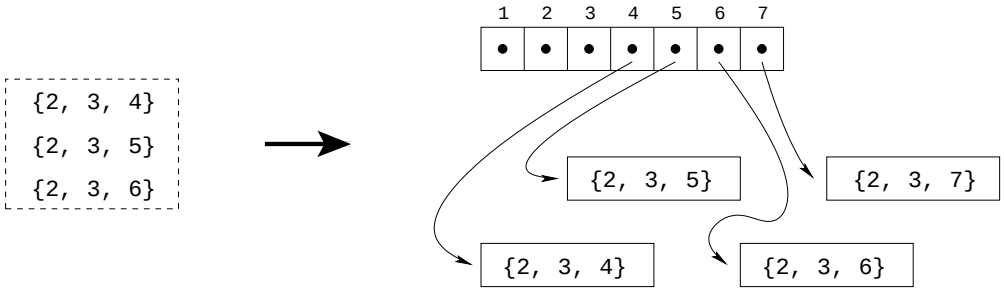


Abbildung 3.13: Umwandlung eines Blatts in einen inneren Knoten und Verteilung der gespeicherten Ereigniskombinationen sowie des neu hinzukommenden Kandidaten

Zweck eines solchen Hashbaums ist es, wie bereits erwähnt, effizient sämtliche Teilmengen einer Transaktion T in der Menge von Kandidaten zu finden und die zugehörigen Zähler zu erhöhen. Diese Operation wird als Tiefensuche auf dem Hashbaum mittels der Funktion increment_counts() realisiert (vgl. Abbildung 3.14).

Bei der Suche der Teilmengen von T im Baum wird wie folgt vorgegangen: Die Funktion increment_counts(T, n, p) wird zuerst für den Wurzelknoten $n = A$ und die undefinierte Position $p = 0$ aufgerufen.[13] In Zeile 3 wird überprüft, ob der aktuelle Knoten n ein Blatt ist. Trifft das zu, werden in den Zeilen 4 bis 12 die Supportwerte aller in diesem Blatt enthaltenen Ereigniskombinationen, die Teilmenge der Transaktion T sind, um 1 erhöht.

Andernfalls wird rekursiv im Hashbaum abgestiegen. Die Schleife in den Zeilen 15 bis 22 durchläuft dazu nacheinander die auf die Position p in der Transaktion T folgenden Ereignisse. Gilt für eines dieser Ereignisse $\mathsf{hashtable}[T.\mathsf{item}_q] \neq \mathsf{false}$, das heißt führt das Ereignis an Position q in T zu einem weiteren Knoten $m = \mathsf{hashtable}[T.\mathsf{item}_q]$, dann erfolgt ein rekursiver Aufruf von increment_counts() für diesen Knoten m und die neue Position q. Kehrt die Rekursion von diesem Aufruf zurück, wird mit den auf $T.\mathsf{item}_q$ in T folgenden Ereignissen auf die gleiche Weise fortgefahren bis als letztes das Ereignis $T.\mathsf{item}_{|T|}$ erreicht ist und der aktuelle Aufruf abbricht.

Dazu folgendes Beispiel: Sei $T = \{1, 2, 4, 7\}$ eine Transaktion, zu deren Teilmengen in der Kandidatenmenge, welche im Hashbaum aus Abbildung 3.12 abgelegt ist, die Zähler gesucht und erhöht werden sollen. Die Rekursion beginnt in der Wurzel A des Hashbaums. Der Knoten A ist ein innerer Knoten, und es wird in Zeile 15 begonnen, über die auf Position $p = 0$ folgenden Ereignisse im Baum abzusteigen. Für Position $q = 1$ führt $\mathsf{hashtable}[T.\mathsf{item}_1 = 1]$ zu Knoten B. In den Zeilen 18 und 20 wird die Funktion increment_counts() für Knoten B und die Position $q = 1$

[13] Die Abbildung $T.\mathsf{item} : \{1, \ldots, |T|\} \to \mathcal{E}$ ist erst ab Position $p = 1$ definiert.

```
(1)  increment_counts(Transaktion T, Knoten n, Position p)
(2)      {
(3)      if (is_leaf(n))
(4)          {
(5)          forall (X ∈ n)
(6)              {
(7)              if (X ⊆ T)
(8)                  {
(9)                  ++(X.count);
(10)                 }
(11)             }
(12)         }
(13)     else
(14)         {
(15)         for (q = p + 1; q ≤ |T|; ++q)
(16)             {
(17)             m = n.hashtable[T.item_q];
(18)             if(m)
(19)                 {
(20)                 increment_counts(T, m, q);
(21)                 }
(22)             }
(23)         }
(24)     }
```

Abbildung 3.14: Pseudocode für die Funktion increment_counts()

rekursiv aufgerufen. Da B ebenfalls ein innerer Knoten ist, wird in den Zeilen 15 bis 23 weiter im Baum abgestiegen. Das nächste betrachtete Ereignis aus T ist das Ereignis $T.\mathsf{item}_2 = 2$. Der entsprechende Eintrag in der Hashtabelle führt von Knoten B zu Knoten D. Knoten D ist erneut ein innerer Knoten, und der rekursive Abstieg im Baum wird mit dem Ereignis $T.\mathsf{item}_3 = 4$ fortgesetzt. Das Hashing in Knoten D über Ereignis 4 führt zum Aufruf von increment_counts() für ein Blatt. In den Zeilen 5 bis 11 werden dann die in diesem Blatt abgelegten Ereigniskombinationen in T gesucht. Die Liste des Blatts enthält nur eine Ereigniskombination, $\{1, 2, 4\}$, die in T gesucht und gefunden wird. Der mit dem Kandidat $\{1, 2, 4\}$ zusammen im Hashbaum gespeicherte Zähler wird daher um 1 erhöht.

Der erste Zweig der Rekursion bricht ab, und eine Rekursionstiefe höher wird in Knoten D wieder aufgesetzt. Auf das Ereignis 4 an Position $q = 3$, das letzte für Knoten D betrachtete Ereignis, folgt an Position $q = 4$ das Ereignis $T.\mathsf{item}_4 = 7$. Für Ereignis 7 ist der Eintrag in der Hashtabelle leer, und es kann nicht weiter im Baum abgestiegen werden. Da außerdem auf Position $q = 4$ in T keine weiteren Ereignisse mehr folgen, bricht die Rekursion in Knoten D ab und kehrt zu Knoten B zurück. Dort wurde als letztes das Ereignis $T.\mathsf{item}_2 = 2$ aus T bearbeitet. Auf Position $q = 2$ folgen die Ereignisse $T.\mathsf{item}_3 = 4$ und $T.\mathsf{item}_4 = 7$, die beide in B auf leere Einträge in der Hashtabelle verweisen. Die Rekursion bricht ab, und es wird in den Wurzelknoten A zurückgekehrt. Das letzte in A bearbeitete Ereignis war $T.\mathsf{item}_1 = 1$. Entsprechend wird über das Ereignis $T.\mathsf{item}_2 = 2$ an Position $q = 2$ abgestiegen und der Knoten C erreicht. Die auf Position $q = 2$ folgenden Ereignisse 4 und 7 führen von Knoten C aus nicht weiter, so daß die Rekursion abbricht. Über das Hashing von 4 in Wurzelknoten A kann zwar ein weiteres Blatt erreicht werden, aber die einzige dort gespeicherte Ereigniskombination $\{4, 5, 6\}$ kommt in T nicht als Teilmenge vor. Damit sind mit $\{1, 2, 4\}$ alle Teilmengen von T, welche die im Baum gespeicherte Kandidatenmenge enthält, gefunden und die entsprechenden Zähler erhöht worden.

Dem Hashbaum liegt die Idee zugrunde, nicht sämtliche Kandidaten in jeder der Transaktionen zu suchen, sondern diese Suche jeweils nur für eine Obermenge der tatsächlich in der gerade zu bearbeitenden Transaktionen durchzuführen. Für jede Ereigniskombination K mit $K \subseteq T$ gilt $K.\mathsf{item}_1 \in T$. Dadurch, daß in der Wurzel nacheinander über jedes Ereignis aus T abgestiegen wird, ist folglich sichergestellt, daß genau die Kandidaten K mit $K.\mathsf{item}_1 \notin T$, die mit einem Ereignis „beginnen“, das nicht in T enthalten ist, ignoriert werden. Auf die gleiche Weise läßt sich für die tieferliegenden Knoten argumentieren, so daß bei Ankunft in einem Blatt die dort vorgefundenen Kandidaten ein gemeinsames, in K und in T enthaltenes Präfix P aufweisen.

Da ein gemeinsames Präfix nicht hinreichend dafür ist, daß die im Blatt vorgefundenen Kandidaten in der aktuellen Transaktion enthalten sind, müssen diese

einzeln überprüft werden. Dazu wird die gerade bearbeitete Transaktion T in einen Bitvektor umgeformt, und sämtliche Vergleiche auf $K \subseteq T$ erfolgen effizient anhand dieses temporären Bitvektors.

Zur Implementierung der Hashbäume ist anzumerken, daß die ersten Einträge jeder Hashtabelle in Abhängigkeit von deren Position im Hashbaum ungenutzt bleiben. Folglich kann durch Anpassung der Hashtabellen in jedem Knoten der Speicherbedarf für den Hashbaum insgesamt signifikant reduziert werden. Außerdem müssen in den Hashtabellen lediglich Einträge für häufige Ereignisse vorgesehen werden (vgl. auch [Agrawal und Srikant, 1994a]).

Eine Variante des Hashbaums ist der sogenannte *Präfixbaum (prefix tree)* (vgl. u. a. [Mueller, 1995], [Brin et al., 1997b]). Im Gegensatz zu einem Hashbaum speichert ein Präfixbaum die Ereigniskombinationen nicht als Listen in den Blättern, sondern implizit verteilt auf den Knoten. Dazu werden die Kanten des Baums mit den Ereignissen aus $\mathcal{E}$ benannt. Jedem Kandidaten K ist genau ein Knoten im Präfixbaum zugeordnet, wobei sich der jeweilige Kandidat aus den Ereignissen auf dem Pfad von der Wurzel bis zu diesem Knoten ergibt. Der Knoten für Kandidat K wird ausgehend von der Wurzel durch sukzessives Absteigen über die Ereignisse $K.\mathsf{item}_1$ bis $K.\mathsf{item}_{|K|}$ erreicht. Damit enthält ein Präfixbaum nicht wie ein Hashbaum nur Ereigniskombinationen einer Mächtigkeit, sondern Ereigniskombinationen verschiedener Mächtigkeiten. Jeder Knoten speichert für den entsprechenden Kandidaten dessen Zähler und eventuelle weitere Informationen. Im Gegensatz zu einem Hashbaum bedingt der Aufbau eines Präfixbaums, daß, wenn ein Kandidat K gespeichert ist, auch jedes Präfix $P_n(K)$ des Kandidaten K im Baum gespeichert sein muß.

3.2 Etablierte Verfahren

Im folgenden werden anhand der im vorangehenden Abschnitt eingeführten und formalisierten Grundlagen die heute im Rahmen des Support-Konfidenz-Ansatzes etablierten Verfahren zur Generierung von Assoziationsregeln beziehungsweise häufiger Ereigniskombinationen vorgestellt.[14] Dabei wird gezeigt, wie sich die einzelnen

[14]Neben dem klassischen Assoziationsproblem im Sinne des Support-Konfidenz-Ansatzes, das dieser Arbeit zugrunde liegt, werden in der Forschung verschiedene Varianten der Assoziationsregelgenerierung diskutiert. So wurden beispielsweise Algorithmen entwickelt, die lediglich Teilmengen der häufigen Ereigniskombinationen beziehungsweise der Assoziationsregeln erzeugen. Einige dieser Ansätze basieren auf abstrakten Konzepten, welche die direkte Erzeugung von Assoziationsregeln nicht vorsehen (vgl. u. a. [Mannila und Toivonen, 1996], [Zaki et al., 1997c], [Zaki et al., 1997a], [Bayardo, 1998], [Lin, 1998], [Lin und Kedem, 1998], [Pasquier et al., 1999], [Pasquier, 2000], [Pei et al., 2000], [Bykowski und Rigotti, 2001]). Des weiteren können auf diese Weise Einschränkungen, welche durch den Analysten vorgegeben werden, mit in die Regelge-

Algorithmen aus den identifizierten Bausteinen zusammensetzen. Im Vordergrund stehen nicht die Implementierungsdetails, sondern es werden die Gemeinsamkeiten und Unterschiede der Verfahren dargestellt. Abschließend wird aus den Ergebnissen dieses Abschnitts eine Systematisierung der etablierten Assoziationsregelverfahren abgeleitet.

Eine vollständige Erfassung und Aufbereitung aller veröffentlichten Verfahren wird aufgrund der Dynamik und der umfangreichen Forschungstätigkeiten auf dem Gebiet der Assoziationsregelgenerierung im Rahmen dieser Arbeit nicht angestrebt. Trotzdem ist die folgende Evaluierung und Systematisierung als erschöpfend anzusehen, da die aufgrund der Rezeption in der Literatur heute als vielversprechend und bedeutsam einzuschätzenden Verfahren berücksichtigt werden.

3.2.1 Generierung einfacher Assoziationsregeln

AIS [Agrawal et al., 1993], [Agrawal und Srikant, 1994a]

AIS, nach den Autoren Agrawal, Imielinski und Srikant benannt, ist der erste veröffentlichte Algorithmus zur Assoziationsregelgenerierung und wurde in [Agrawal et al., 1993] zusammen mit den Assoziationsregeln eingeführt.

AIS realisiert die Generierung der häufigen Ereigniskombinationen als Breitensuche mit direktem Zählen von Vorkommen. Die Kandidatenmengen K_k werden während des Auszählens dynamisch anhand der Transaktionen erzeugt und enthalten jeweils sämtliche Kandidaten der Mächtigkeit k. Für jede Transaktion T werden nacheinander die Teilmengen von T zur Kandidatenmenge K_k hinzugenommen, die sich als Vereinigung einer häufigen $(k-1)$-Ereigniskombination mit

nerierung einfließen (vgl. z. B. [Bayardo, 1997], [Srikant et al., 1997], [Ng et al., 1998], [Ng et al., 1999], [Lakshmanan et al., 1998], [Bayardo et al., 1999], [Webb, 2000], [Li et al., 2001]). Diese Erweiterung wird in Abschnitt 5.2 aufgegriffen. Einige Verfahren erlauben die Vorgabe mehrerer minimaler Häufigkeiten (vgl. u. a. [Liu et al., 1999], [Fujiwara et al., 2000], [Wang et al., 2000]) oder ermöglichen dem Analysten, während der Regelgenerierung aufgrund von Zwischenergebnissen in die laufende Regelgenerierung einzugreifen (vgl. u. a. [Aggarwal und Yu, 1997], [Aggarwal und Yu, 1998], [Hidber, 1998], [Hidber, 1999]). Verschiedentlich werden auch Stichproben herangezogen, um eine Näherung der häufigen Ereigniskombinationen zu erhalten (vgl. etwa [Kivinen und Mannila, 1993], [Kivinen und Mannila, 1994], [Toivonen, 1996b], [Domingo et al., 1998], [Lee et al., 1998]). Weiterhin werden Algorithmen mit dem Ziel entwickelt, in der Datenbanksprache SQL formuliert und direkt auf einem relationalen Datenbanksystem ausgeführt zu werden (vgl. z. B. [Houtsma und Swami, 1993], [Houtsma und Swami, 1995], [Holsheimer et al., 1995], [Sarawagi et al., 1998a]). Diese Ansätze erreichen allerdings bei weitem nicht die Laufzeiten der Implementierungen in den üblichen prozeduralen Sprachen. An dieser Stelle sei außerdem auf die vielfältigen Ansätze zur Parallelisierung der Assoziationsregelgenerierung verwiesen (vgl. u. a. [Park et al., 1995a], [Park et al., 1995b], [Zaki et al., 1997b], [Han et al., 1997b], [Shintani und Kitsuregawa, 1998], [Parthasarathy et al., 1998], [Sousa et al., 1998], [Skillicorn, 1998], [Zaki, 1999]).

einem häufigen Ereignis darstellen lassen.[15] AIS basiert somit auf einer datenorientierten Suchstrategie.

Durch dieses Vorgehen wird der Suchraum von zwei Seiten beschränkt. Zum einen werden lediglich für die Ereigniskombinationen Zähler angelegt, die auch tatsächlich in mindestens einer der Transaktionen vorkommen. Zum anderen findet die Abgeschlossenheitseigenschaft des Supports (vgl. Abschnitt 3.1.1) ansatzweise Verwendung, indem jeder Kandidat als Vereinigung einer häufigen Ereigniskombination und eines häufigen Ereignisses darstellbar ist.

In [Agrawal et al., 1993] werden keine speziellen Datenstrukturen zur Speicherung der häufigen Ereigniskombinationen und Kandidatenmengen angegeben. Unter anderem für das direkte Auszählen der Kandidaten wird jedoch nach [Agrawal und Srikant, 1994a] ein Hashbaum (vgl. Abschnitt 3.1.4) herangezogen.

OCD [Mannila et al., 1994a], [Mannila et al., 1994b]

Die Bezeichnung für den Algorithmus *OCD* leitet sich von „**o**ffline **c**andidate **d**etermination“ ab. Im Gegensatz zu dem Vorgehen, das AIS zugrunde liegt, werden die Kandidaten nicht während des Auszählens anhand der Transaktionen „online“ generiert, sondern „offline“ allein aufgrund der bereits in früheren Durchläufen als häufig erkannten Ereigniskombinationen. Grundlage ist erstmals die vollständige Anwendung der Abgeschlossenheitseigenschaft des Supports (vgl. dazu Abschnitt 3.1.1). Aus der Menge der häufigen $(k-1)$-Ereigniskombinationen wird die Menge der k-Kandidaten als Menge der k-Ereigniskombinationen, deren sämtliche Teilmengen der Mächtigkeit $(k-1)$ häufig sind, abgeleitet.

Das Vorgehen von OCD läßt sich als Breitensuche mit direktem Zählen von Vorkommen zusammenfassen. Allerdings werden noch keine speziellen Datenstrukturen (vgl. Abschnitt 3.1.4) eingesetzt.

Apriori [Agrawal und Srikant, 1994a], [Agrawal und Srikant, 1994b], [Agrawal et al., 1996]

Der Algorithmus *Apriori* basiert wie OCD auf einer Breitensuche in Kombination mit dem direkten Zählen von Vorkommen und einer separaten Kandidatengenerierung auf Grundlage der Abgeschlossenheitseigenschaft des Supports. Wie bei OCD wird aus der Menge der häufigen $(k-1)$-Ereigniskombinationen die Menge der k-Kandidaten als Menge der k-Ereigniskombinationen, deren sämtliche $(k-1)$-Teilmengen häufig sind, abgeleitet. Hinzu kommt die Verwendung eines Hashbaums (vgl. Abschnitt 3.1.4) um die Häufigkeiten der Kandidaten effizient zu bestimmen.

[15] Die häufigen $(k-1)$-Ereigniskombinationen sind zu diesem Zeitpunkt aus der vorherigen Iteration bereits bekannt.

AprioriTID [Agrawal und Srikant, 1994a], [Agrawal und Srikant, 1994b], [Agrawal et al., 1996]

Der Algorithmus *AprioriTID* ist eine Variante von Apriori, die auf höherwertigen Transaktionen basiert, welche aus der ursprünglichen Transaktionsdatenbank abgeleitet werden. Jede der aufbereiteten Transaktionen T'_k enthält nach dem Auszählen der Kandidaten auf der k-ten Ebene des Suchraums (vgl. Abschnitt 3.1.2) genau die k-Kandidaten, die Teilmenge der korrespondierenden Transaktion T aus der ursprünglichen Transaktionsdatenbank sind. Diese k-Kandidaten sind ausreichend, um die Häufigkeiten der $(k+1)$-Kandidaten zu bestimmen und außerdem jede Transaktion T'_k zu T'_{k+1} fortzuschreiben, ohne daß auf die ursprüngliche Datenbank zurückgegriffen werden muß.

Das Zählen der Kandidaten wie auch das Fortschreiben der aufbereiteten Transaktionen wird über spezielle sequentielle Datenstrukturen implementiert.

DHP [Park et al., 1995a], [Park et al., 1997]

DHP ist eine weitere Variante von Apriori. Der Algorithmus DHP, „**d**ynamic **h**ashing and **p**runing“, optimiert das Zählen der 2-Kandidaten, indem während der Supportbestimmung der Einzelereignisse bereits aggregierte Informationen über die Häufigkeiten der 2-Ereigniskombinationen gesammelt werden. Über eine Hashfunktion werden n Zähler den m möglichen 2-Ereigniskombinationen zugewiesen, wobei $n \ll m$ gilt. Aufgrund dieser ersten Abschätzung, die einen Teil der nichthäufigen 2-Ereigniskombinationen erkennt, kann die 2-Kandidatenmenge vor dem eigentlichen Auszählen zusätzlich beschnitten werden.

Als weitere Optimierung werden anhand der aggregierten Häufigkeiten der 2-Ereigniskombinationen vor dem Auszählen der 2-Kandidaten solche Ereignisse aus den Transaktionen der zugrundeliegenden Datenbank entfernt, für die bereits erkannt wurde, daß sie nichthäufig und nicht Element einer häufigen 2-Ereigniskombination sind. Diese aufbereitete Datenbank ist Grundlage aller nachfolgenden Datenbankdurchläufe, für die der unveränderte Apriori-Algorithmus eingesetzt wird.

Partition [Savasere et al., 1995a], [Savasere et al., 1995b]

Partition basiert auf der indirekten Häufigkeitsbestimmung durch Schneiden von Transaktionsmengen.[16] Die Transaktionsmengen werden für Partition als aufsteigend sortierte Listen von Integerwerten implementiert (vgl. Abschnitt 3.1.4), und der Suchraum wird mittels Breitensuche durchlaufen.

[16] Das Schneiden von Transaktionsmengen wird auch in [Holsheimer et al., 1995] beschrieben. Holsheimer et al. gehen jedoch nicht auf die in [Savasere et al., 1995a,b] angesprochene Hauptspeicherproblematik ein.

Für die Breitensuche mit indirekter Häufigkeitsbestimmung werden aus Effizienzgründen jeweils die Transaktionsmengen zu sämtlichen Ereigniskombinationen einer Ebene k im Suchraum gleichzeitig im Hauptspeicher benötigt. Um auch für große Datenbanken alle diese Transaktionsmengen im Hauptspeicher halten zu können, wird die Datenbank in n Teilmengen $p_1, \ldots, p_n$ zerlegt, die nacheinander separat bearbeitet werden. Die Zerlegung wird so gewählt, daß die jeweils benötigten Transaktionsmengen für jede Teilmenge $p_i, i = 1, \ldots, n$, vollständig im Hauptspeicher gehalten werden können.

Die für mindestens eine Teilmenge p_i häufigen Ereigniskombinationen sind eine Obermenge der auf der gesamten Datenbank häufigen Ereigniskombinationen. Daher ist es notwendig, in einem zweiten Durchlauf für jede dieser Ereigniskombinationen deren genauen Support zu bestimmen. Die abschließende Supportbestimmung wird indirekt über Schnittmengenbildung separat für jede Teilmenge $p_i, i = 1, \ldots, n$, durchgeführt und die Ergebnisse zu den globalen Supportwerten summiert.

Optimierungen dieses Ansatzes sind SPINC aus [Mueller, 1995] und AS-CPA aus [Lin und Dunham, 1998]. Beiden Algorithmen liegt die Idee zugrunde, schon im ersten Durchlauf für jede Teilmenge $p_i, i = 1, \ldots, n$, die in den bereits betrachteten Teilmengen $p_j, j = 1, \ldots, i-1$, als lokal häufig erkannten Ereigniskombinationen auszuzählen. Dadurch muß in der Regel nur ein Teil der $n-1$ Teilmengen ein zweites Mal betrachtet werden. AS-CPA als Erweiterung von SPINC versucht, mit verschiedenen Heuristiken die Anzahl der im zweiten Durchlauf zu betrachtenden Partitionen weiter zu reduzieren.

PreSample [Toivonen, 1996a], [Toivonen, 1996b]

In [Toivonen, 1996a], [Toivonen, 1996b] wird ein Algorithmus eingeführt, im folgenden als *PreSample* bezeichnet, der durch das vorangestellte Ziehen einer Stichprobe die Kandidatengenerierung optimiert. Das Ergebnis des Algorithmus PreSample ist die vollständige Menge häufiger Ereigniskombinationen zusammen mit den exakten Supportwerten.

Für die Häufigkeitsbestimmung in der Stichprobe wird der Apriori-Algorithmus verwendet, wobei ein Schwellenwert $\mathsf{minsupp}' < \mathsf{minsupp}$ vorgegeben wird. Die zugrundeliegende Annahme ist, daß die Menge der auf der Stichprobe häufigen Ereigniskombinationen weitgehend der Menge der auf der Datenbank häufigen Ereigniskombinationen entspricht. Durch den gegenüber $\mathsf{minsupp}$ herabgesetzten Schwellenwert $\mathsf{minsupp}'$ werden auch die auf der Stichprobe im Sinne von $\mathsf{minsupp}$ nahezu häufigen Ereigniskombinationen gefunden und ebenfalls auf der gesamten Datenbank ausgezählt.

Um sicherzustellen, daß die Stichprobe hinreichend repräsentativ gewählt, daß heißt die Menge der häufigen Ereigniskombinationen vollständig bestimmt wurde, wird nachträglich der negative Rand (vgl. Abschnitt 3.1.2) der Menge der erkann-

ten häufigen Ereigniskombinationen überprüft. Sind die Supportwerte sämtlicher Ereigniskombinationen aus dem negativen Rand bekannt, dann wurde die Menge der häufigen Ereigniskombinationen vollständig erzeugt. Andernfalls sind weitere Durchläufe über die Datenbank notwendig, bis die Supportwerte sämtlicher häufiger Ereigniskombinationen bestimmt sind.

Der Algorithmus PreSample ist kein eigenständiges Verfahren, sondern kann ergänzend mit vielen der in dieser Arbeit vorgestellten Algorithmen kombiniert werden.

DIC [Brin et al., 1997b]

Der Algorithmus *DIC*, „**d**irect **i**temset **c**ounting“, aus [Brin et al., 1997b] basiert wie Apriori auf der Kombination von Breitensuche mit direktem Zählen von Vorkommen. Anders als bei Apriori erfolgt bei DIC nicht ein kompletter Durchlauf über die Datenbank, bevor neue Kandidaten generiert werden, sondern bereits während des Auszählens von Kandidaten werden weitere Ereigniskombinationen zu der Kandidatenmenge hinzugefügt. Durch diese Verzahnung wird die Anzahl der für die Bestimmung der häufigen Ereigniskombinationen notwendigen Datenbankdurchläufe reduziert.

DIC unterbricht alle m Transaktionen den aktuellen Datenbankdurchlauf und bestimmt die Kandidaten, die mittlerweile den Schwellenwert für den Support erreicht haben. Der Support eines solchen Kandidaten ist zwar unter Umständen noch nicht vollständig bekannt, aber allein, daß ein Kandidat häufig ist, genügt, um bereits weitere Kandidaten auf Grundlage dieses Kandidaten und der Abgeschlossenheitseigenschaft des Supports zu generieren.

DIC zählt daher gleichzeitig Kandidaten unterschiedlicher Mächtigkeiten. Diese werden zusammen mit Zusatzinformationen, die jeweils über den aktuellen Status eines Kandidaten Auskunft geben, beispielsweise ob dieser bereits vollständig ausgezählt wurde, in einem Präfixbaum abgelegt[17] (vgl. Abschnitt 3.1.4).

Eclat: [Zaki et al., 1997a], [Zaki et al., 1997c]

Mit dem Algorithmus *Eclat* wurde die Generierung aller häufigen Ereigniskombinationen erstmals mittels einer Tiefensuche realisiert. Die Bestimmung der Supportwerte erfolgt indirekt durch Schneiden von Transaktionsmengen, wobei sogenannte Schnellschnitte (vgl. Abschnitt 3.1.4) verwendet werden.

Während des Absteigens im Suchraum ist es bei der Tiefensuche lediglich notwendig, die Transaktionsmengen der Ereigniskombinationen aus den Äquivalenzklas-

[17] In einem Hashbaum können nur Kandidaten einer gemeinsamen Mächtigkeit zugleich gespeichert werden.

sen auf dem Weg von der Wurzel bis zu der aktuell bearbeiteten Äquivalenzklasse gleichzeitig im Hauptspeicher zu halten. Daher können auch für sehr große Datenbanken sämtliche zugleich für die Schnitte benötigten Transaktionsmengen im Hauptspeicher abgelegt werden, ohne beispielsweise die Transaktionsdatenbank aufteilen zu müssen, wie es für den Algorithmus Partition notwendig ist. In [Zaki et al., 1997a], [Zaki et al., 1997c] werden außerdem verschiedene Varianten von Eclat vorgestellt.

Zu Eclat bleibt anzumerken, daß dieser Algorithmus in [Zaki et al., 1997a], [Zaki et al., 1997c] auf die Generierung häufiger k-Ereigniskombinationen mit $k \geq 3$ beschränkt ist. Die häufigen 1- und 2-Ereigniskombinationen werden als gegeben vorausgesetzt.

FP-Growth [Han et al., 2000], [Han und Pei, 2000]

Mit *FP-Growth* wird in [Han et al., 2000], [Han und Pei, 2000] der mit AIS eingeführte Ansatz, den Suchraum datenorientiert zu durchlaufen, aufgegriffen. Anders als AIS basiert FP-Growth auf einer Tiefensuche sowie einer als FP-Baum aufbereiteten Datenbasis.

Grundlage der Tiefensuche bildet die Menge aller häufigen Ereignisse als häufige 1-Ereigniskombinationen. In im Sinne der Relation $<$ absteigender Reihenfolge wird ausgehend von jedem dieser Ereignisse rekursiv im Suchraum abgestiegen, indem zu der jeweils bearbeiteten Ereigniskombination ein einzelnes Ereignis hinzugenommen wird, welches zusammen mit dieser in mindestens einer Transaktion enthalten ist. Um Ereigniskombinationen nicht mehrfach zu zählen, werden diese nur um solche Ereignisse erweitert, die im Sinne der Relation $<$ kleiner sind als sämtliche in der Ereigniskombination bereits enthaltenen Ereignisse. Der Suchraum aus Abbildung 3.1 wird folglich strikt von rechts nach links und von oben nach unten durchlaufen, ohne die Äquivalenzklassenstruktur zu berücksichtigen. Dabei wird nur zu tatsächlich in den Transaktionen vorkommenden Ereigniskombinationen abgestiegen.

Für die Häufigkeitsbestimmung zählt FP-Growth das direkte Vorkommen von Ereigniskombinationen. Dazu greift der Algorithmus nicht jeweils auf die gesamte Datenbank zurück, sondern schränkt mit jedem rekursiven Aufruf schrittweise die Datenbasis ein. Wird beispielsweise die Häufigkeit der Ereigniskombination $\{y, z\}$ bestimmt, dann werden nur die Transaktionen betrachtet, die Ereignis z enthalten. Wird weiter rekursiv zu Ereigniskombination $\{x, y, z\}$ abgestiegen, so werden lediglich die Transaktionen, die $\{y, z\}$ enthalten, betrachtet etc. Dieses Vorgehen wird von den in Abschnitt 3.1.4 vorgestellten FP-Bäumen effizient implementiert.[18]

[18] In [Wang et al., 2002] wird eine Variante von FP-Growth vorgestellt, die den Hauptspeicher effizienter nutzt, indem die bedingten FP-Bäume nicht mehr explizit erzeugt werden.

dEclat: [Zaki und Gouda, 2001]

Als Weiterentwicklung von Eclat wird in [Zaki und Gouda, 2001] der Algorithmus *dEclat*, „**d**iffset **Eclat**“, vorgestellt. dEclat erzeugt die häufigen Ereigniskombinationen ebenfalls mittels Tiefensuche und basierend auf Transaktionsmengen. Im Unterschied zu Eclat werden jedoch Differenzmengen (vgl. Abschnitt 3.1.4) für die Supportbestimmung zumindest der Kandidaten höherer Mächtigkeiten verwendet. Als sinnvoller Schwellenwert für den Wechsel von Transaktionsmengen zu Differenzmengen werden in [Zaki und Gouda, 2001] die Mächtigkeiten 2 oder 3 angegeben, in Abhängigkeit von den zugrundeliegenden Daten.

3.2.2 Generierung taxonomer Assoziationsregeln

Für die Generierung taxonomer Assoziationsregeln (vgl. Abschnitt 2.2.3) finden verschiedene Varianten des Apriori-Algorithmus Anwendung. Apriori wird dazu so erweitert, daß dieser die Taxonomie als zusätzliches Mittel zur Beschneidung des Suchraums nutzen kann. Da sämtliche Nachkommen eines nichthäufigen Ereignisses ebenfalls nichthäufig sind, ermöglichen die als Baum oder DAG vorliegenden Taxonomien dieses Vorgehen.

ML [Han und Fu, 1995], [Fu und Han, 1995]

ML (multiple level) bezeichnet eine Familie von Algorithmen, die verschiedene Varianten eines direkt an den einzelnen Ebenen der Taxonomie ausgerichteten Vorgehens implementieren. Für jede Ebene wird ein separater Durchlauf mit den auf die jeweilige Ebene transformierten Transaktionen durchgeführt. Um die Häufigkeiten zu bestimmen, findet das Apriori-Verfahren Verwendung. Die Verfahren der ML-Familie beginnen in der obersten Ebene der Taxonomie und steigen dann ebenenweise ab, bis keine häufigen Ereigniskombinationen mehr gefunden werden. Während des Abstiegs können die bereits aus den höheren Ebenen bekannten Informationen über Häufigkeiten genutzt werden, um den Suchraum zu beschneiden.

Basic [Srikant und Agrawal, 1995]

Der Algorithmus *Basic* realisiert die Generierung taxonomer Assoziationsregeln auf Basis des ursprünglichen Apriori-Algorithmus, wobei jedoch keine Optimierungen genutzt werden, welche auf die Taxonomie zurückgreifen. Um mit Apriori taxonome Assoziationsregeln generieren zu können, werden die vorliegenden Transaktionen jeweils mit den Ereignissen angereichert, die Vorfahr mindestens eines der bereits in der Transaktion enthaltenen Ereignisse sind. Die auf diese Weise erzeugten häufigen Ereigniskombinationen ermöglichen die Ableitung taxonomer Asso-

ziationsregeln. Um das Kopieren der unter Umständen sehr großen zugrundeliegenden Datenmengen zu vermeiden, werden die Transaktionen in einem zusätzlichen Schritt während der Generierung der häufigen Ereigniskombinationen dynamisch aufbereitet. Dazu wird jede Transaktion vor dem Auszählen der Häufigkeiten, das heißt direkt vor dem Absteigen im Hashbaum, um die entsprechenden Ereignisse erweitert. Unmittelbar nachdem eine angereicherte Transaktion bearbeitet ist, wird der von dieser belegte Speicherbereich wieder freigegeben, so daß die Anreicherung der Transaktionen in jedem Durchlauf über die Datenbank erneut erfolgt. Der durch das wiederholte Aufbereiten der Transaktionen entstehende Mehraufwand ist vergleichsweise gering.

Cumulate [Srikant und Agrawal, 1995]

Der Algorithmus *Cumulate* erweitert den Basic-Algorithmus, indem die Anreicherung der Transaktionen optimiert und die Taxonomie teilweise für das Beschneiden des Suchraums herangezogen wird. Im einzelnen handelt es sich um die folgenden Optimierungen:

- Es werden nicht mehr sämtliche Vorfahren der Ereignisse zu den Transaktionen hinzugenommen, sondern lediglich solche, die in mindestens einem der gerade gültigen Kandidaten vorkommen. Es werden außerdem die Ereignisse selbst, sofern sie ebenfalls in keinem der Kandidaten enthalten sind, aus den Transaktionen entfernt.

- Zu jedem Ereignis können dessen Vorfahren im voraus ermittelt werden, so daß der die Taxonomie repräsentierende Baum oder Graph während des Auszählens nicht aufwendig durchlaufen werden muß. Werden die Vorfahren vor jedem Datenbankdurchlauf neu ermittelt, dann können an dieser Stelle effizient die ungültigen, das heißt die in keinem der Kandidaten vorkommenden Ereignisse ausgefiltert werden.

- Die Kandidaten müssen nicht ausgezählt werden, die ein Ereignis x sowie einen Vorfahr $\hat{x}$ dieses Ereignisses enthalten, da die Häufigkeit eines Kandidaten $K \cup \{x, \hat{x}\}$ der des Kandidaten $K \cup \{x\}$ entspricht.[19] Weiterhin genügt es, vor dem Auszählen der Kandidaten der Mächtigkeit 2, die Kandidaten zu streichen und als nichthäufig zu markieren, die aus einem Ereignis und einem

[19] Sei $\hat{x}$ Vorfahr von x. Offensichtlich wird jede Transaktion, die den Zähler eines Kandidaten $K = \{x, \hat{x}, \ldots\}$ um 1 erhöht, auch den Zähler des Kandidaten $K \setminus \{\hat{x}\}$ um genau 1 erhöhen, und umgekehrt. Selbst wenn zugelassen wird, daß eine Transaktion nur $\hat{x}$ nicht aber x enthält, dann erhöht diese Transaktion weder den Zähler für K noch für $K \setminus \{\hat{x}\}$.

Vorfahr dieses Ereignisses bestehen, um auch in den folgenden Durchläufen keinen der im obigen Sinn redundanten Kandidaten auszuzählen.[20]

Stratify, Estimate, EstMerge [Srikant und Agrawal, 1995]

Der Algorithmus *Stratify* ist eine Erweiterung von Cumulate, welche das Beschneiden des Suchraums mit Hilfe einer vorhandenen Taxonomie weiterentwickelt. Stratify liegt zugrunde, daß eine Ereigniskombination X nur häufig sein kann, wenn sämtliche Ereigniskombinationen $\hat{X}$, die dadurch entstehen, daß mindestens eines der Ereignisse aus X durch einen Vorfahren dieses Ereignisses ersetzt wird, ebenfalls häufig sind. $\hat{X}$ wird auch als Vorfahr von X bezeichnet.[21] Werden die Häufigkeiten der aus Ereignissen der oberen Taxonomieebenen bestehenden Ereigniskombinationen zuerst bestimmt und wird dann sukzessive zu den tieferen Ebenen der Taxonomie abgestiegen, können Kandidaten gezielt von der Häufigkeitsbestimmung ausgeschlossen werden. Da bei diesem Vorgehen die Kandidaten der gleichen Mächtigkeit nicht mehr zu einer Kandidatenmenge zusammengefaßt werden können, sind zusätzliche Durchläufe über die Transaktionen der Datenbank notwendig.

Stratify basiert darauf, daß die Ereigniskombinationen, die aus Ereignissen der oberen und mittleren Ebenen der Taxonomie bestehen, bereits selten in den Daten vorkommen, und daher viele der aus Ereignissen der unteren Ebenen bestehenden Ereigniskombinationen nicht ausgezählt werden müssen. Ist hingegen von den Ereigniskombinationen, die aus Ereignissen der oberen und mittleren Ebenen bestehen, ein großer Teil häufig, werden im Vergleich zu Cumulate zusätzliche Durchläufe über die Transaktionen notwendig.

Von den Varianten *Estimate* und *EstMerge* aus [Srikant und Agrawal, 1995] kann diese Problematik teilweise durch Sampling umgangen werden.

3.2.3 Systematisierung

Im folgenden wird basierend auf den in Abschnitt 3.1 identifizierten algorithmischen Grundlagen eine Systematisierung der vorgestellten etablierten Verfahren entwickelt. Neben den speziellen Optimierungen auf Datenstrukturebene, die an

[20] Bei der Kandidatengenerierung entfernt Apriori und damit auch Cumulate die Kandidaten aus dem Suchraum, die eine nichthäufige Teilmenge enthalten. Da sämtliche Kandidaten, die aus einem Ereignis und einem Vorfahr dieses Ereignisses bestehen, bereits im zweiten Durchlauf über die Transaktionen als nichthäufig markiert wurden (unabhängig von ihrer tatsächlichen Häufigkeit), werden wie gefordert in den folgenden Schritten die Kandidaten vor dem Auszählen entfernt, die sowohl ein Ereignis als auch gleichzeitig einen Vorfahr dieses Ereignisses enthalten.

[21] Bisher war eine Vorfahrenbeziehung nicht zwischen Ereigniskombinationen definiert, sondern lediglich zwischen einzelnen Ereignissen.

dieser Stelle als Details nicht zu der angestrebten Allgemeingültigkeit der Systematisierung beitragen würden, wurden in dieser Arbeit drei Bereiche herausgearbeitet, in denen sich die etablierten Verfahren grundlegend unterscheiden:

(a) Im Rahmen der Kandidatengenerierung werden entweder die zugrundeliegenden Transaktionen berücksichtigt, das heißt es wird *datenorientiert* vorgegangen, oder der Abstieg im Suchraum erfolgt ohne diese Zusatzinformation, das heißt *nicht datenorientiert*.

(b) Die Bestimmung von Häufigkeiten erfolgt entweder direkt durch das *Zählen* von Vorkommen oder alternativ indirekt durch das *Schneiden* von Transaktionsmengen.

(c) Als Suchstrategie findet entweder eine *Tiefensuche* oder eine *Breitensuche* Verwendung.

Anhand der sich daraus ergebenden drei Hierarchiestufen werden in Abbildung 3.15 die vorgestellten Verfahren systematisiert (vgl. dazu auch [Hipp et al., 2000a]). Es

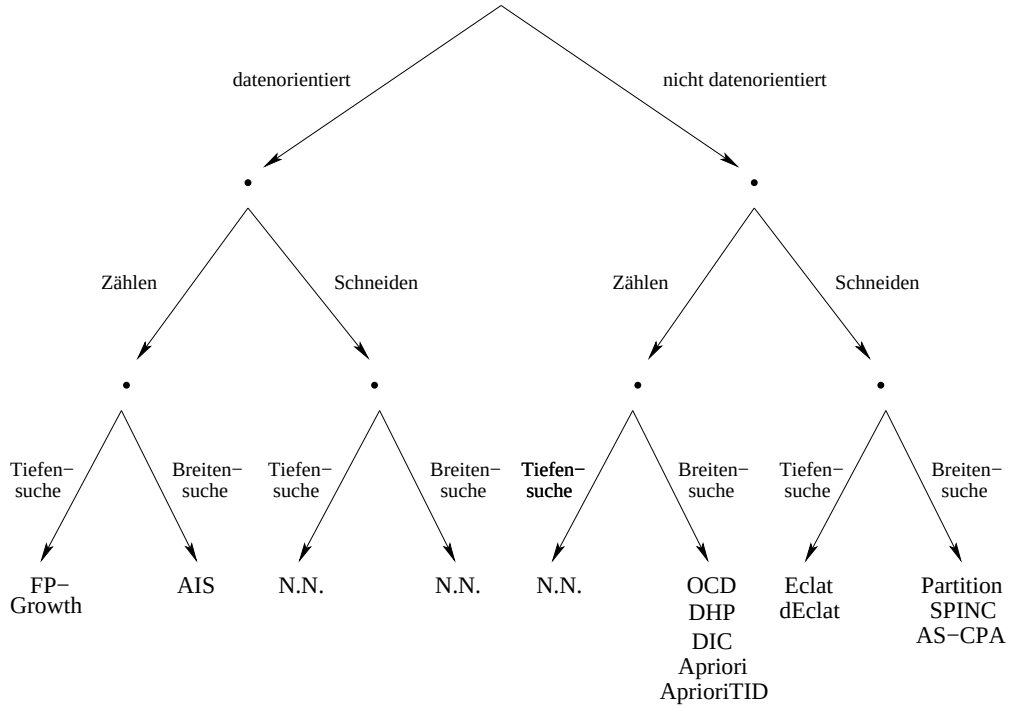


Abbildung 3.15: Systematisierung der vorgestellten Verfahren anhand der in Abschnitt 3.1 eingeführten Kriterien

fällt auf, daß zu drei der insgesamt acht möglichen Kombinationen keine algorithmische Entsprechung bekannt ist. Insbesondere ist keines der heute etablierten Verfahren datenorientiert und bestimmt außerdem die Häufigkeiten durch Schneiden. Ebenfalls scheint die Kombination von Zählen und Tiefensuche wenig vielversprechend zu sein, wenn nicht datenorientiert vorgegangen wird. Umgekehrt existieren fünf Varianten der datenorientierte Breitensuche mit Zählen von Häufigkeiten sowie zwei Verfahren, die das nicht datenorientierte Schneiden von Transaktionsmengen mit einer Tiefensuche kombinieren. Die im nächsten Abschnitt folgende Evaluierung untersucht unter anderem die Hintergründe für diese Unterschiede.

3.3 Evaluierung

In diesem Abschnitt werden das Laufzeitverhalten und der Speicherbedarf der vorgestellten etablierten Algorithmen untersucht. Bei vergleichbaren Evaluierungen liegt der Schwerpunkt zumeist auf synthetisch erzeugten Daten, und es sind gewöhnlich nur wenige der bekannten Algorithmen in die Experimente einbezogen.[22] Die folgende Evaluierung geht bezüglich der berücksichtigten Verfahren und Datensätzen über diese Evaluierungen hinaus. Außerdem wird erstmals der Hauptspeicherbedarf der Verfahren systematisch untersucht.

3.3.1 Grundlagen

Für die Evaluierung der Algorithmen wurden Implementierungen der Verfahren erstellt und sowohl die in der Literatur üblichen synthetischen Datensätze erzeugt als auch Datensätze aus verschiedenen realen Anwendungen beschafft. Auf diese Grundlagen der später im Rahmen der vorliegenden Arbeit durchgeführten Experimente wird im folgenden näher eingegangen:

Implementierungen und Experimentierumgebung

Die Verfahren wurden in C++ implementiert und sorgfältig bezüglich Laufzeit und Speicherallokation optimiert. Die im Rahmen dieser Arbeit erstellten Implementierungen konnten ihre Effizienz bereits in verschiedenen Projekten aus Anwendung und Forschung beweisen, wobei die erreichten Laufzeiten vielfach veröffentlicht wurden (vgl. u. a. [Hipp et al., 1998], [Hipp und Lindner, 1999], [Hipp et al.,

[22] Bei der Veröffentlichung neuer Verfahren beschränken sich die Autoren meist auf einen Vergleich der neu eingeführten Algorithmen mit dem Apriori-Algorithmus aus [Agrawal und Srikant, 1994a]. [Zheng et al., 2001] ist keine Ausnahme, da Apriori und FP-Growth die einzigen in die Evaluierung einbezogenen Algorithmen sind, die das dieser Arbeit zugrundeliegende Assoziationsproblem (vgl. Abschnitt 3.1.1) bearbeiten.

2000a], [Hipp et al., 2000b], [Wirth und Hipp, 2000], [Hipp et al., 2001a], [Hipp et al., 2001b], [Wirth et al., 2001], [Hipp und Güntzer, 2002], [Hipp et al., 2002a], [Hipp et al., 2002b]). In den folgenden Experimenten wird außerdem gegebenenfalls ein direkter Bezug zu den erreichten Laufzeiten anderer Implementierungen der Verfahren unter vergleichbaren Rahmenbedingungen hergestellt.

Als Rechnersystem wird für die Experimente eine Sun Workstation eingesetzt.[23] Die zu analysierenden Daten sind jeweils auf der lokalen Festplatte des Rechners gespeichert.

Die während der Experimente gemessenen Zeiten beziehen sich jeweils auf die komplette Laufzeit für die Generierung der häufigen Ereigniskombinationen. Es wird die tatsächlich vom Start des Algorithmus bis zum Abschluß der Analyse vergangene Zeit gemessen, das heißt der Datenzugriff und eventuelle systeminterne Vorgänge wie das Auslagern von Hauptspeicherseiten auf externe Speichermedien sind eingeschlossen. Die Zeiterfassung erfolgt durch das Betriebssystem, indem die unabhängig von der Ausführung der Algorithmen laufenden Systemuhr des eingesetzten Rechners ausgelesen wird.[24]

Die Messung des Hauptspeicherbedarfs der Algorithmen erfolgt ebenfalls über das Betriebssystem. Dazu wird fortlaufend während der Generierung der häufigen Ereigniskombinationen der momentan belegte Speicher abgefragt[25] und über den gesamten Ablauf der erreichte Spitzenwert ermittelt.

Daten

Grundlage der Experimente sind sowohl synthetisch generierte Daten als auch Daten aus verschiedenen realen Anwendungen.

Synthetische Daten Die verwendeten synthetischen Datensätze wurden mit einem am Forschungszentrum der IBM in Almaden, USA, entwickelten Datengenerator erzeugt (vgl. [Agrawal und Srikant, 1994a], [Srikant und Agrawal, 1995],

[23] Der Prozessor dieses Rechnersystems vom Typ UltraSPARC III ist mit 900 MHz getaktet, und als Betriebssystem kommt Solaris in Version 2.9 zum Einsatz. Auf dem System stehen 2 Gigabyte physikalischer Hauptspeicher zur Verfügung. Alle implementierten Verfahren sind mit dem GNU-Compiler `gcc` in Version 3.2.1 mit den Optimierungsoptionen `-O3` sowie `-mcpu=sun4u` übersetzt (vgl. u. a. [Griffith, 2002]).

[24] Die Zeitmessung erfolgt über die Funktion `clock_t times(struct tms *buf)`, welche unter dem verwendeten Betriebssystem in der Headerdatei `systimes.h` deklariert ist (vgl. z. B. [Mauro und McDougall, 2000]).

[25] Die Belegung des Hauptspeichers wird über das Proc-Pseudodateisystem bestimmt (vgl. z. B. [Mauro und McDougall, 2000]). Da die Bestimmung der Hauptspeicherbelegung vergleichsweise rechenzeitaufwendig ist, wurde bei der Implementierung der Verfahren sehr kritisch mit dieser Abfrage umgegangen.

[Srikant, 1996]). Durch ihre weite Verbreitung in der Literatur haben sich die mit diesem Datengenerator erzeugten Datensätze als ein Standard für die Evaluierung von Assoziationsregelverfahren etabliert. Grundlage der im Rahmen dieser Arbeit durchgeführten Evaluierung sind die in [Agrawal und Srikant, 1994a] eingeführten und beispielsweise in [Savasere et al., 1995a], [Zaki et al., 1997c], [Brin et al., 1997b], [Zheng et al., 2001], [Hipp et al., 2000a], [Hipp et al., 2000b] wieder aufgegriffenen Datensätze. Die Bezeichnungen der Datensätze sind im folgenden nach einem Schema aufgebaut, das die grundlegenden Charakteristika der Transaktionen widerspiegelt. So bezeichnet beispielsweise T10I4D1000K einen Datensatz, der pro Transaktion im Schnitt 10 Ereignisse enthält, auf vorgegebenen häufigen Ereigniskombinationen der durchschnittlichen Mächtigkeit 4 basiert und aus einer Million Transaktionen besteht.[26] Zusätzlich zu den üblichen Datensätzen wurden zwei Datensätze aus [Han et al., 2000], hier mit T25I10D10K und T25I20D100K bezeichnet, hinzugenommen, die ebenfalls mit dem Datengenerator erzeugt wurden.[27]

Die synthetischen Datensätze TmInDsK werden anhand der durchschnittlichen Anzahl von Ereignissen m in jeder Transaktion und der durchschnittlichen Mächtigkeit n der vorgegebenen häufigen Ereigniskombinationen in weniger anspruchsvolle Datensätze für kleinere Werte und anspruchsvolle Datensätze für größere Werte unterschieden. Die Werte für m liegen in den meisten Experimenten zwischen 5 und 20, die Werte für n zwischen 2 und 6. Die Anzahl der Transaktionen s geht in diese Unterscheidung nicht ein.

Mit dem Datengenerator kann außerdem eine Taxonomie zu den Ereignissen erzeugt werden. Die Generierung dieser Taxonomie läßt sich durch verschiedene Parameter steuern, auf die in Abschnitt 4.2 im Kontext der Generierung taxonomer Assoziationsregeln näher eingegangen wird.

Datensatz EINZELHANDEL Für die Evaluierung werden neben den synthetischen Daten auch anonym erfaßte Kundentransaktionen eines in Deutschland ansässigen Einzelhandelsunternehmens herangezogen. Das betrachtete Unternehmen verfügt über eine Verkaufsfläche von 6200m^2 und ein Warensortiment von

[26] Die den Experimenten in der Literatur zugrundeliegenden Datensätze bestehen üblicherweise aus lediglich 100.000 Transaktionen. Der Schritt zu 1.000.000 Transaktionen trägt zur Aussagekräftigkeit der Experimente auf der heutigen und seit den ersten vergleichbaren Evaluierungen aus [Agrawal und Srikant, 1994a] weiterentwickelten Hardware bei. Die andernfalls zu beobachtenden Meßergebnisse liegen teilweise in Bereichen von weniger als einer Sekunde. Ein Verzerrung der Ergebnisse ist nicht zu befürchten, da alle herangezogenen Verfahren annähernd linear mit der Anzahl zugrundeliegender Transaktionen skalieren (vgl. die folgenden Abschnitte).

[27] Die exakten Vorgaben für den Datengenerator sind für beide Datensätze anhand von [Han et al., 2000] nicht mit Sicherheit nachvollziehbar. Für die Durchführung der Experimente wurden die Datensätze jedoch direkt von den Autoren zur Verfügung gestellt.

19.535 Artikeln. Der Schwerpunkt der Verkäufe liegt im Bereich Lebensmittel, der 28% des Umsatzes ausmacht und einen Artikelanteil von 44% in den Kundentransaktionen aufweist. Die Daten wurden mittels eines Scannerkassensystems über einen Zeitraum von 4 vollständigen und aufeinander folgenden Wochen erfaßt. Die resultierende Datenbank besteht aus 77.588 Transaktionen, mit denen insgesamt ein Umsatz von ungefähr 2 Millionen Euro erwirtschaftet wurde. Durchschnittlich enthält jede Transaktion ungefähr 10,5 verschiedene Artikel. Die Kundentransaktionen belegen binär als Integerwerte aufbereitet (vgl. Abschnitt 3.1.4) 4,1 Megabyte. Zu den Transaktionen steht außerdem eine Taxonomie zur Verfügung, die das gesamte Warensortiment umfaßt und aus 5 Ebenen besteht.

Datensatz FAHRZEUG Des weiteren stehen für die Evaluierung Produktionsdaten von Fahrzeugen der Marke Mercedes-Benz zur Verfügung. Diese Daten enthalten Informationen zu den Sonderausstattungen von ungefähr 10 Millionen Fahrzeugen aus den Bereichen Personenkraftwagen und Nutzfahrzeuge. Jedes Fahrzeug entspricht einer Transaktion, wobei die Transaktionen im Durchschnitt ungefähr 22 aus einer Menge von fast 10.000 möglichen Sonderausstattungen enthalten. Die Daten stammen aus dem Qualitätsinformationssystem QUIS der DaimlerChrysler AG, auf das in Abschnitt 5.1 näher eingegangen wird.

Datensatz BMS-POS Der Datensatz BMS-POS aus [Zheng et al., 2001] enthält Kundentransaktionen eines im Internet tätigen Versandhandels. Jede Transaktion besteht aus den zu einem bestimmten Zeitpunkt von einem Kunden gemeinsam gekauften Artikeln. Den Transaktionen liegt ein zu 1.657 Artikelgruppen zusammengefaßtes Warensortiment zugrunde. Insgesamt besteht der Datensatz aus 515.597 Transaktionen, die das Einkaufverhalten der Kunden über einen Zeitraum von mehreren Jahren wiedergeben. Jede Transaktion enthält im Durchschnitt 6 bis 7 Ereignisse.

Datensätze BMS-WEBVIEW-1 und BMS-WEBVIEW-2 Die beiden Datensätze BMS-WEBVIEW-1 und BMS-WEBVIEW-2, ebenfalls aus [Zheng et al., 2001], spiegeln das Navigationsverhalten der Besucher auf den Internetseiten zweier verschiedener Unternehmen des E-Commerce wider. Jede Transaktion entspricht einer Benutzersitzung, die sämtliche Produktseiten als Ereignisse enthält, die der Besucher während dieser Sitzung betrachtet hat. Der Datensatz BMS-WEBVIEW-1 besteht aus 59.602 Transaktionen, denen insgesamt 497 verschiedene Ereignisse enthalten. Der Datensatz BMS-WEBVIEW-2 hingegen besteht aus 77.512 Transaktionen, die sich aus 3.340 Ereignissen zusammensetzen. Die Transaktionen umfassen im Durchschnitt 2 bis 3 beziehungsweise 5 Ereignisse.

3.3.2 Experimente

Anhand der beschriebenen Datensätze werden die Regelgenerierungsverfahren bezüglich Laufzeit und Speicherallokation evaluiert und verglichen. In einen typischen Experiment werden zu einem festen Datensatz die von den einzelnen Verfahren erreichten Werte für Laufzeit und Speicherallokation in Abhängigkeit von verschiedenen minimalen Schwellenwerten minsupp für die Häufigkeit ermittelt. Die Speicherzugriffe sämtlicher Verfahren sind nicht lokal auf einzelne Speicherbereiche beschränkt und außerdem schwer prognostizierbar, so daß das Auslagern von Hauptspeicherseiten den Generierungsprozeß annähernd zum Erliegen bringt. Es werden daher die Experimente vorzeitig abgebrochen, welche mehr Speicher benötigen als physikalischer Hauptspeicher verfügbar ist. Entsprechend sind für manche der Verfahren nicht zu jeder der minimalen Häufigkeiten die Laufzeit und der Speicherbedarf bekannt.

Um die Untersuchungsergebnisse geeignet aufzubereiten, werden die Verfahren in zählende und schneidende Ansätze unterteilt (vgl. dazu die Systematisierung der Verfahren in Abschnitt 3.2, insbesondere Abbildung 3.15).

Zählende Algorithmen

Von den in Abschnitt 3.2 als zählend klassifizierten Verfahren wurden im Rahmen der vorliegenden Arbeit die Algorithmen Apriori, DIC und FP-Growth implementiert.

Für den Algorithmus AIS wird bereits in [Agrawal und Srikant, 1994a], [Agrawal et al., 1996] gezeigt, daß diesem Apriori in jeder Hinsicht überlegen ist. AIS wurde daher im Rahmen dieser Arbeit nicht weiter berücksichtigt. Gleiches gilt für den Algorithmus OCD aus [Mannila et al., 1994a], [Mannila et al., 1994b], der sich als direkter Vorläufer von Apriori von diesem lediglich dadurch unterscheidet, daß keine speziellen Datenstrukturen, wie beispielsweise ein Hashbaum für das Auszählen der Kandidaten, beschrieben sind.

Apriori selbst wird hier als Stellvertreter für die Gruppe der in [Agrawal und Srikant, 1994a], [Agrawal et al., 1996] eingeführten Algorithmen ausgewählt. Seit seiner Veröffentlichung hat sich Apriori zu einem Referenzalgorithmus entwickelt, so daß fast jede bis heute veröffentlichte Evaluierung auf diesen Algorithmus als Bezugspunkt zurück greift.

Der Algorithmus DHP aus [Park et al., 1995a], [Park et al., 1997] als Variante von Apriori basiert auf einem expliziten Hashing der 2-Kandidaten. Gegenüber Apriori bietet dieses Vorgehen jedoch nur einen Vorteil, wenn die Häufigkeiten der 2-Kandidaten mittels eines Hashbaums bestimmt und nicht anhand einer zweidimensionalen Matrix ausgezählt werden. Die Optimierung des Apriori-Algorithmus

mittels einer Matrix wurde erst in [Srikant und Agrawal, 1995] veröffentlicht und fand daher keinen Eingang in [Park et al., 1995a].[28] Des weiteren beschneidet DHP die der Analyse zugrundeliegenden Transaktionen. Diese werden dazu nach [Park et al., 1995a], [Park et al., 1997] im Hauptspeicher gehalten, ein Vorgehen, das dem ursprünglichen Apriori-Algorithmus widerspricht. Vor diesem Hintergrund wird die Erweiterung DHP des Apriori-Algorithmus im Rahmen dieser Arbeit als von eingeschränkter praktischer Bedeutung betrachtet.

Vielversprechend erscheint DIC aus [Brin et al., 1997b] als Erweiterung von Apriori, die entsprechend in die im Rahmen der vorliegenden Arbeit durchgeführten Evaluierungen aufgenommen wurde.[29]

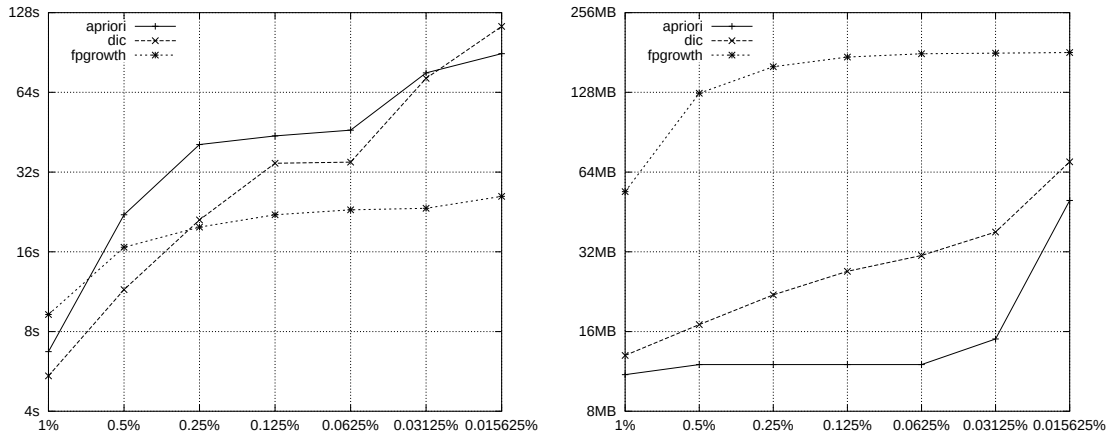
Vervollständigt wird die Evaluierung der etablierten Verfahren durch den Algorithmus FP-Growth aus [Han et al., 2000], der als datenorientiertes Verfahren ebenfalls auf dem Zählen von Häufigkeiten basiert.

Synthetische Daten Der Vergleich der von den Algorithmen Apriori und DIC erreichten Laufzeiten auf den synthetischen Datensätzen (siehe insbesondere die Abbildungen 3.16 bis 3.18 sowie die ergänzenden Messungen aus Anhang A.1.1) ergibt ein differenziertes Bild.

Auf dem Datensatz T5I2D1000K erreicht DIC bis zu 50% kürzere Laufzeiten als Apriori und ist in beinahe allen Experimenten effizienter. Auf dem Datensatz T10I4D1000K hingegen ist Apriori in fast allen Experimenten effizienter als DIC und die Laufzeiten von DIC sind bis zu Faktor 5 länger als die von Apriori. Der Abstand der Verfahren vergrößert sich für den Datensatz T20I6D1000K, auf dem Apriori in allen Experimenten effizienter als DIC ist und im Maximum um mehr als 90% kürzere Laufzeiten als DIC erreicht. Während also auf dem weniger anspruchsvollen Datensatz T5I2D1000K DIC effizienter als Apriori ist, ist auf den zunehmend anspruchsvolleren Datensätzen T10I4D1000K und T20I6D1000K umgekehrt Apriori das effizientere Verfahren.

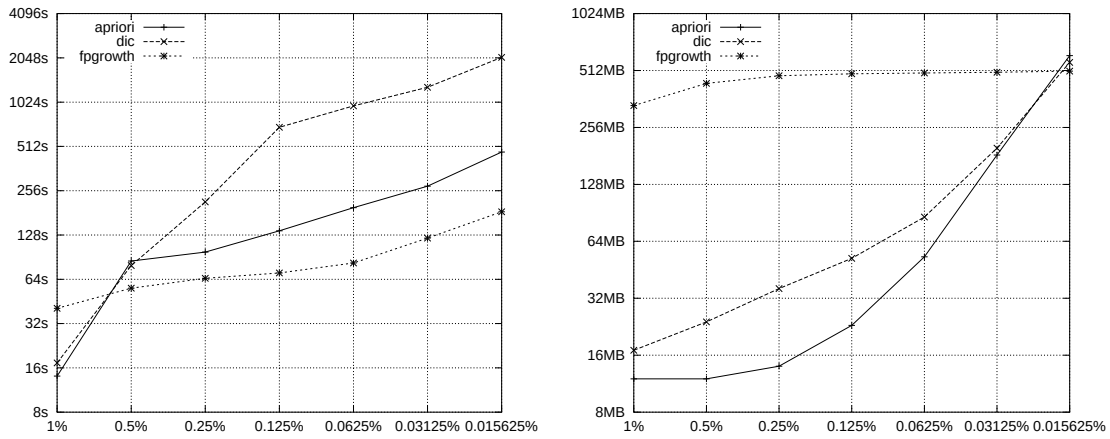
[28] Die in [Agrawal und Srikant, 1994a] für die Evaluierungen genutzten Implementierungen basieren bereits auf zweidimensionalen Matrizen anstelle von Hashbäumen für die Speicherung der 2-Kandidaten. Dieses Implementierungsdetail wurde laut Ramakrishnan Srikant aber bewußt erst in [Srikant und Agrawal, 1995] veröffentlicht (Quelle: persönliches Gespräch mit Ramakrishnan Srikant am Rande der PAKDD-Konferenz Anfang Mai 2002). Ramakrishnan Srikant und Rakesh Agrawal stehen daher den Ergebnissen in [Park et al., 1995a] skeptisch gegenüber (Quelle: erwähntes Gespräch mit Ramakrishnan Srikant, sowie Gespräch mit Rakesh Agrawal am Rande der KDD-Konferenz im August 1999).

[29] Unklar ist in [Brin et al., 1997b], welche Partitionsgröße für DIC auf den synthetischen Daten zu verwenden ist. In [Brin et al., 1997b] wird zwar für die dort analysierten jedoch nicht frei zugänglichen Zensusdaten erklärt, daß den Experimenten eine Partitionsgröße von 10.000 zugrunde liegt. In der hier vorliegenden Arbeit wurde auf den synthetischen Daten allerdings eine Partitionsgröße von 20.000 als optimal ermittelt und verwendet.



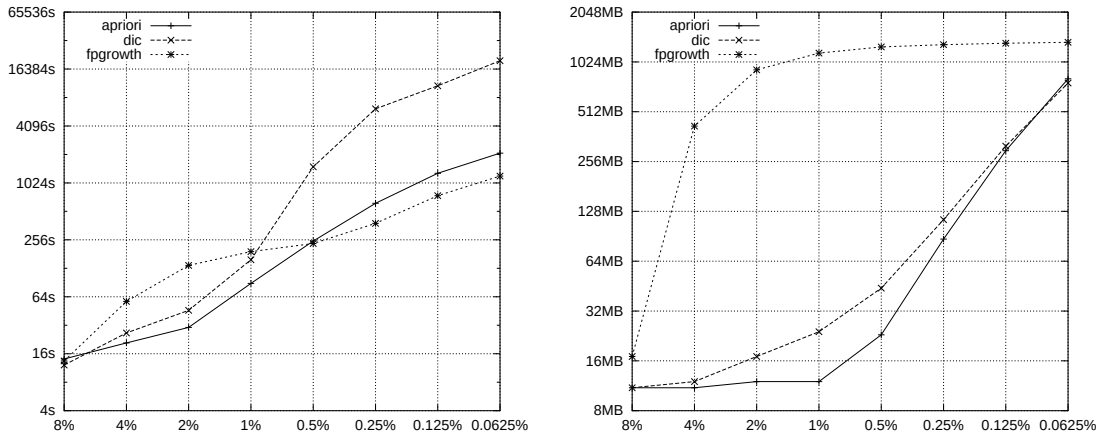
(a) Laufzeit in Sekunden (b) Speicherbedarf in Megabyte

Abbildung 3.16: Effizienz zählender Verfahren auf T5I2D1000K bei variierendem minsupp



(a) Laufzeit in Sekunden (b) Speicherbedarf in Megabyte

Abbildung 3.17: Effizienz zählender Verfahren auf T10I4D1000K bei variierendem minsupp



(a) Laufzeit in Sekunden

(b) Speicherbedarf in Megabyte

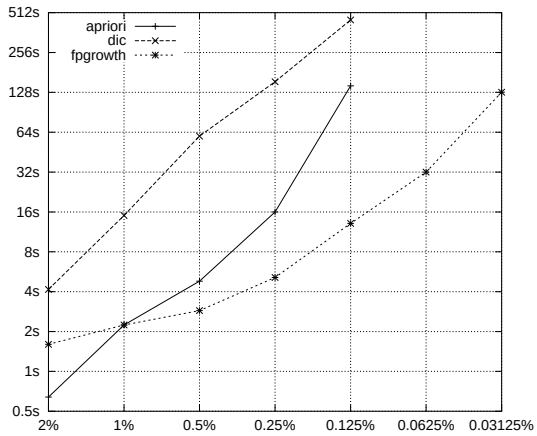
Abbildung 3.18: Effizienz zählender Verfahren auf T20I6D1000K bei variierendem minsupp

Bezüglich der Speichereffizienz ergibt sich ein eindeutiges Bild. Apriori ist das überlegene Verfahren und benötigt auf T5I2D1000K bis zu 60% weniger Hauptspeicher als DIC. Für geringere minimale Häufigkeiten minsupp und mit zunehmend anspruchsvolleren Datensätzen gleicht sich der Speicherbedarf beider Verfahren an, bis diese ab minsupp = 0,125% für T10I4D1000K und T20I6D1000K nahezu den gleichen Speicherbedarf aufweisen.

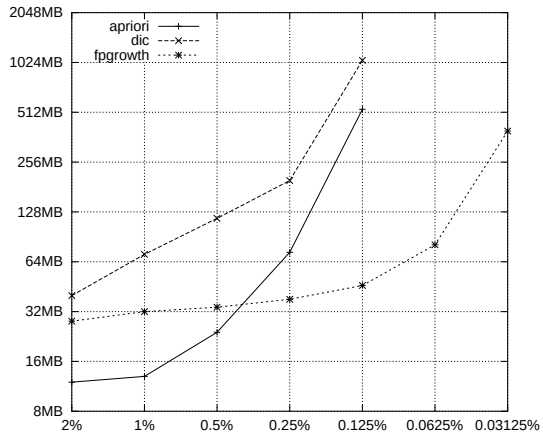
Der Vergleich der Laufzeiten von Apriori und FP-Growth ergibt, daß insgesamt FP-Growth das wesentlich effizientere der Verfahren ist. Dieser Vorteil von FP-Growth gegenüber Apriori wird jedoch mit zunehmend anspruchsvolleren Daten geringer. So erreicht FP-Growth auf T5I2D1000K bis zu 70% kürzere Laufzeiten als Apriori. Für T10I4D1000K ist FP-Growth noch um maximal 60% schneller, während auf dem Datensatz T20I6D1000K Apriori teilweise kürzere Laufzeiten als FP-Growth erreicht und FP-Growth im Maximum 40% kürzere Laufzeiten als Apriori erreicht.

Hinsichtlich des Hauptspeicherbedarfs ist Apriori gegenüber FP-Growth eindeutig das effizientere Verfahren. Auf allen drei Datensätzen steigt der Hauptspeicherbedarf von FP-Growth bereits für vergleichsweise große minimale Häufigkeiten steil an, während dieser Anstieg bei Apriori erst für vergleichsweise geringe Werte zu beobachten ist. Auf allen drei Datensätzen kann Apriori jeweils weit mehr als 90% des Hauptspeichers von FP-Growth einsparen.

Auf den Datensätzen T25I10D10K und T25I20D100K (vgl. die Abbildungen 3.19 und 3.20) ergibt sich ein ähnliches Bild, das allerdings zugunsten von FP-Growth verschoben ist. So kann FP-Growth seinen Laufzeitvorteil gegenüber Apriori ver-

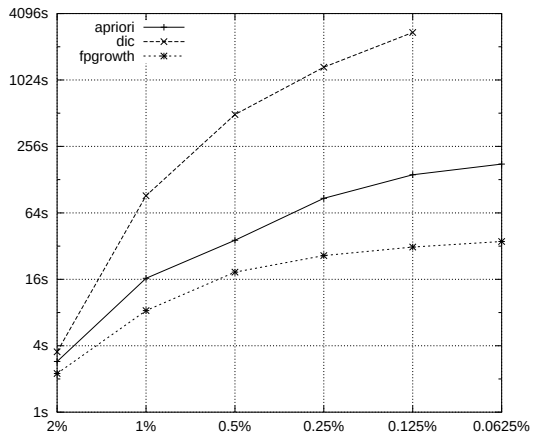


(a) Laufzeit in Sekunden

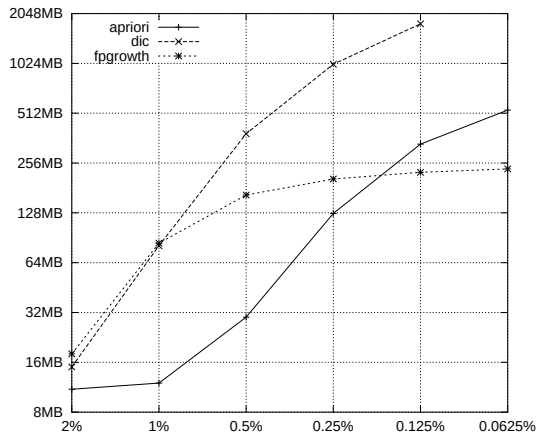


(b) Speicherbedarf in Megabyte

Abbildung 3.19: Effizienz zählender Verfahren auf T25I10D10K bei variierendem minsupp



(a) Laufzeit in Sekunden

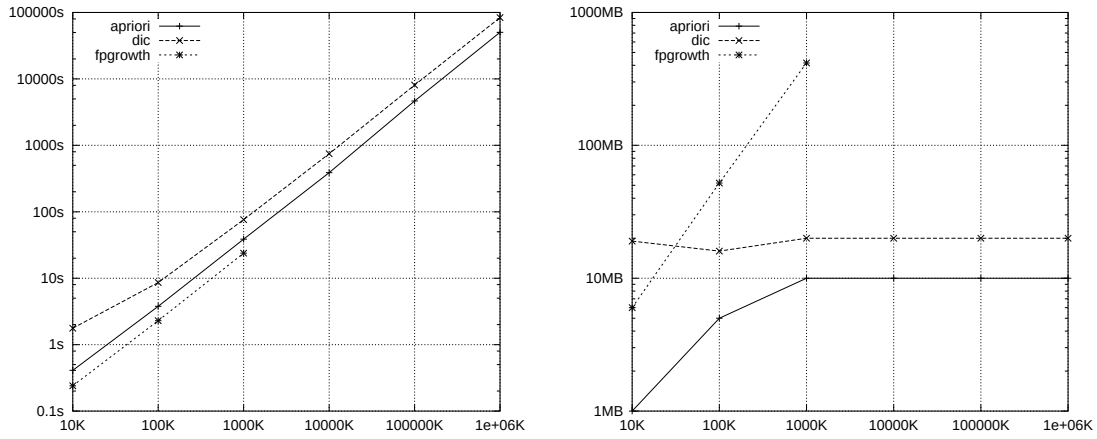


(b) Speicherbedarf in Megabyte

Abbildung 3.20: Effizienz zählender Verfahren auf T25I20D100K bei variierendem minsupp

größern und erreicht auf T25I10D10K bis zu 90% und auf T25I20D100K bis zu 80% kürzere Laufzeiten als Apriori, soweit die Zeiten gemessen wurden.[30] Auch der Hauptspeicherbedarf von FP-Growth ist auf diesen Daten zumindest für vergleichsweise geringe minimale Häufigkeiten bis zu 90% geringer als der von Apriori. DIC ist auf diesen Daten sowohl hinsichtlich der Laufzeiten als auch des Speicherbedarfs wie bisher wesentlich ineffizienter als die beiden anderen Verfahren.

In den Experimenten aus Abbildung 3.21 wird die Skalierbarkeit der Verfahren



(a) Laufzeit in Sekunden

(b) Speicherbedarf in Megabyte

Abbildung 3.21: Skalierbarkeit zählender Verfahren auf synthetischen Daten bei variierender Anzahl von Transaktionen in Tausend

evaluiert. Dazu wurden mit dem in Abschnitt 3.3.1 beschriebenen Datengenerator bei ansonsten gleichen Parametereinstellungen Datensätze mit wachsender Anzahl von Transaktionen generiert. Für jeden der erzeugten Datensätze T10I4DnK, wobei n für die Anzahl der Transaktionen in Tausend steht, werden bei festem Schwellenwert minsupp = 0,5% von den Verfahren alle häufigen Ereigniskombinationen generiert.[31] Bei ansonsten gleichbleibenden Einstellungen wird die Anzahl der zugrundeliegenden Transaktionen zwischen eintausend und einer Milliarde variiert. Abbildung 3.21(a) ist zu entnehmen, daß die Laufzeit für alle drei Verfahren nahezu linear mit der Anzahl Transaktionen skaliert. Während für Apriori und DIC

[30] Einige der Experimente wurden abgebrochen, da für diese der verfügbare Hauptspeicher nicht ausreichte.

[31] Anstelle der Sun Workstation kommt für die Experimente zur Skalierung ein auf Linux 2.4.18 basierendes PC-System zum Einsatz, das mit einem 1,8 GHz Pentium IV Prozessor und ebenfalls 2 Gigabyte Hauptspeicher ausgestattet ist. Hintergrund ist, daß auf der Sun Workstation für die mit zum Teil mehr als 50 Gigabyte sehr großen Dateien nicht ausreichend zusammenhängender lokaler Plattenplatz zur Verfügung stand.

eine obere Schranke für den benötigten Hauptspeicher existiert, skaliert gleichzeitig der Hauptspeicherbedarf von FP-Growth linear mit der Anzahl Transaktionen (vgl. Abbildung 3.21(b)).[32]

Interpretation Zunächst fällt auf, daß Apriori und DIC unterschiedliche Laufzeiten erreichen und insbesondere DIC als Weiterentwicklung von Apriori zumeist die schlechteren Ergebnisse erzielt. Der Grund ist darin zu sehen, daß DIC vor allem die Anzahl der benötigten Datenbankdurchläufe optimiert. Diese kann DIC zwar merklich verringern, in den Experimenten teilweise um mehr als 60%, insgesamt ist der Aufwand dafür jedoch zumeist größer als der Laufzeitgewinn.

Problematisch für die Laufzeit von DIC im Vergleich zu Apriori ist unter anderem die wesentlich schwierigere Kandidatengenerierung. So kann Apriori die n-Kandidatengenerierung basierend auf einer vollständigen und sortierten Liste der $(n-1)$-Kandidaten durch zwei ineinander verschachtelte Schleifen sehr effizient implementieren. DIC hingegen muß für jede neu als häufig erkannte Ereigniskombination deren direkte Obermengen auf Häufigkeit sämtlicher Teilmengen überprüfen. Dieser bereits sehr aufwendige Schritt wird zusätzlich erschwert, da unter Umständen die Häufigkeiten der Teilmengen des möglichen Kandidaten noch nicht bekannt sind. Dadurch müssen Kandidaten zurückgestellt werden. Ein geeignetes Ablegen der Zwischenergebnisse zusammen mit deren späterer Suche erscheint kaum effizient realisiertbar und in [Brin et al., 1997b] ist kein Hinweis auf einen solchen Ansatz zu finden. Somit werden die Teilmengen eines Kandidaten unter Umständen mehrfach erzeugt und auf Häufigkeit überprüft.

Weiterhin speichert DIC sämtliche aktiven Kandidaten und häufigen Ereigniskombinationen in einem einzigen Präfixbaum und nicht wie Apriori in einer Menge von Hashbäumen. Während bei Apriori bereits ausgezählte häufige Ereigniskombinationen im aktuellen Hashbaum nicht enthalten sind, sondern lediglich die tatsächlich zu zählenden Ereigniskombinationen, enthält der Präfixbaum von DIC auch die bereits bestimmten Ereigniskombinationen. Dadurch wird das Auszählen der aktiven Kandidaten erschwert, da zwangsläufig auch zu den inaktiven und bereits als häufig erkannten Kandidaten im Baum abgestiegen wird. Hier fällt insbesondere ins Gewicht, daß der Präfixbaum insbesondere die häufig in den Transaktionen vorkommenden Ereigniskombinationen enthält. Zum anderen ist es für DIC nicht möglich, für das Auszählen von Kandidaten der Mächtigkeit 2 auf die effizientere zweidimensionale Matrix auszuweichen (vgl. dazu auch Abschnitt 3.1.4), da der Präfixbaum sonst unvollständig aufgebaut würde. Des weiteren benötigt

[32] In Abbildung 3.21 brechen die Werte für FP-Growth bereits nach wenigen Messungen ab. Die Ursache dafür ist, daß auch die vergleichsweise üppige Ausstattung des für die Experimente eingesetzten Rechnersystems mit Hauptspeicher nicht ausreicht, um mit FP-Growth größere Datenmengen zu analysieren.

die Speicherung der häufigen Ereigniskombinationen in einem Präfixbaum wesentlich mehr Hauptspeicher als die für Apriori mögliche Speicherung in Form einer sortierten Liste. Als problematisch für den Hauptspeicherbedarf erweist sich außerdem, daß der Präfixbaum bereits aufgebaut wird, während die häufigen Ereignisse bestimmt werden. Anders als für Apriori ist es für DIC daher nicht möglich, den Präfixbaum so zu initialisieren, daß dieser in den einzelnen Knoten lediglich Einträge für die häufigen Ereignisse vorsieht, da diese noch nicht bekannt sind. Letztere Optimierung des Hashbaums wurde in Abschnitt 3.1.4 eingeführt.

Der Vergleich von Apriori und FP-Growth zeigt, daß die Aufbereitung der Transaktionen als FP-Baum kürzere Laufzeiten ermöglicht. Lediglich auf anspruchsvolleren Datensätzen wie T20I6D1000K wird die Erzeugung des initialen FP-Baums und der bedingten FP-Bäume so aufwendig, daß Apriori teilweise kürzere Laufzeiten erzielt. Der Hauptspeicherbedarf von FP-Growth ist wegen der als FP-Baum aufbereiteten Datenbasis jedoch wesentlich größer als der von Apriori.

Begünstigt wird FP-Growth gegenüber Apriori insbesondere auf den Datensätzen T25I10D10K und T25I20D100K (vgl. die Abbildungen 3.19 und 3.20). Beide Datensätze enthalten mit 10.000 beziehungsweise 100.000 vergleichsweise wenige Transaktionen. Der Aufbau des initialen FP-Baums sowie der bedingten FP-Bäume ist dadurch sowohl bezüglich der Laufzeit als auch des Speicherbedarfs wenig aufwendig. Zugleich wirken sich die vielen häufigen Ereigniskombinationen in den Daten bezüglich Laufzeit und Speicherbedarf negativ auf die Kandidatengenerierung von Apriori aus. Es bleibt allerdings anzumerken, daß diese beiden Datensätze in [Han et al., 2000] gerade für den Vergleich von Apriori mit dem dort neu eingeführten FP-Growth erzeugt wurden, und ein Vergleich auf den üblichen synthetischen Datensätzen, wie beispielsweise T10I4D1000K, in [Han et al., 2000] fehlt.

Vergleichbare Evaluierungen in der Literatur Eine Evaluierung von Apriori und DIC findet sich in [Brin et al., 1997b]. Im Widerspruch zu den Ergebnissen im Rahmen der vorliegenden Arbeit erreicht DIC in [Brin et al., 1997b] auf dem Datensatz T20I4D100K (vgl. Anhang A.1.1) zum Teil kürzere Laufzeiten als Apriori.[33] Die Messungen sind jedoch insoweit zu relativieren, als nach Auffassung der Autoren Apriori als eine spezielle Version von DIC verstanden werden kann, bei der die Partitionsgröße gleich der Anzahl Transaktionen gesetzt wird.[34] In diesem

[33] Beim Vergleich der absoluten Laufzeiten ist zu beachten, daß den Messungen in der vorliegenden Arbeit eine um Faktor 10 größere Transaktionsmenge als in [Brin et al., 1997b] zugrunde liegt. Auch sind in [Brin et al., 1997b] keinerlei Hinweise auf die Architektur und Leistungsfähigkeit des eingesetzten Rechnersystems zu finden.

[34] „The implementations were mostly the same since Apriori can be thought of as a special case of DIC – the case where the interval size, was the size of the datafile, not M." (siehe [Brin et al., 1997b]). Die Konstante M steht hier für die Partitionsgröße m.

Fall ist Apriori von den oben für DIC beschriebenen Nachteilen betroffen, ohne von einer Verringerung der Anzahl notwendiger Datenbankdurchläufe profitieren zu können und ist daher zwangsläufig weniger effizient als der den Messungen im Rahmen dieser Arbeit zugrundeliegende nach [Agrawal und Srikant, 1994a], [Agrawal und Srikant, 1994b], [Agrawal et al., 1996] mittels Hashbäumen implementierte Apriori-Algorithmus.

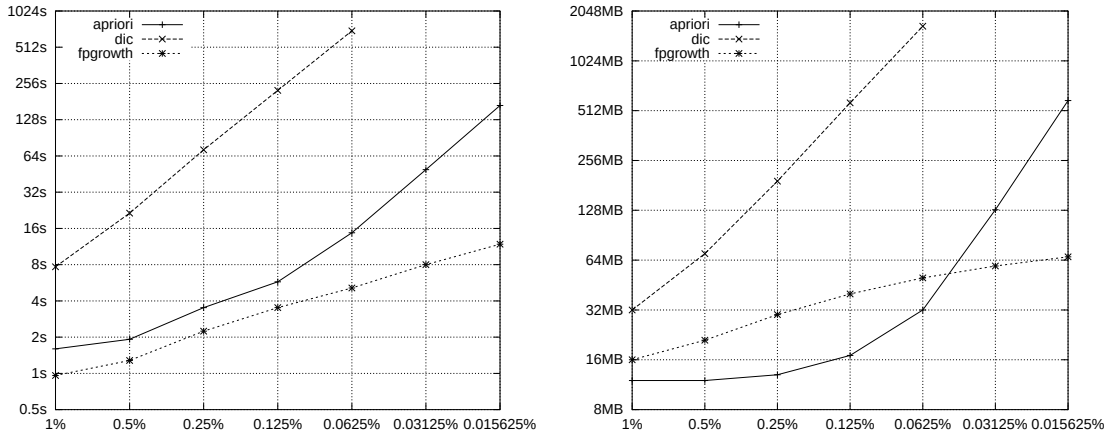
In [Han et al., 2000] werden Apriori und FP-Growth evaluiert, wobei aber lediglich die Datensätze T25I10D10K und T25I20D100K herangezogen werden, auf denen bis dahin keine Experimente in der Literatur beschrieben wurden. FP-Growth erreicht weit kürzere Laufzeiten als Apriori, ein Ergebnis, das grundsätzlich im Einklang mit den Messungen im Rahmen der vorliegenden Arbeit steht (vgl. die Abbildungen 3.19 und 3.20). Der Vorteil von FP-Growth gegenüber Apriori ist allerdings in [Han et al., 2000] wesentlich ausgeprägter als in den hier durchgeführten Experimenten. Um diesen Unterschied zu erklären, wurden ergänzende Messungen auf einem Rechnersystem durchgeführt, das mit dem in [Han et al., 2000] eingesetzten vergleichbar ist.[35] Auf diesem System erreicht die im Rahmen der vorliegenden Arbeit erstellte Implementierung von FP-Growth bei minsupp = 0.2% auf T25I10D10K eine Laufzeit von ungefähr 13,5 Sekunden im Vergleich zu 17-18 Sekunden,[36] welche FP-Growth in [Han et al., 2000] erreicht. Zugleich wurden für Apriori ungefähr 56 Sekunden gemessen, während in [Han et al., 2000] für Apriori bei minsupp = 0.2% kein Wert mehr angegeben wird, sondern die entsprechenden Abbildung eine fast als unendlich zu interpretierende Laufzeit suggeriert. Die letzte in [Han et al., 2000] nachvollziehbare Messung für Apriori auf T25I20D100K erreicht für minsupp = 1.5% einen Wert zwischen 60 und 70 Sekunden, während für die im Rahmen dieser Arbeit erstellte Implementierung ungefähr 21 Sekunden gemessen wurden. Für FP-Growth, ebenfalls auf T25I20D100K, erreicht die Implementierung in [Han et al., 2000] für minsupp = 1% ungefähr 25-30 Sekunden, während im Rahmen dieser Arbeit mit nur 15,3 Sekunden eine fast um Faktor 2 kürzere Laufzeit gemessen wurde. Die im Rahmen dieser Arbeit erstellten Implementierungen beider Verfahren sind somit effizienter als die in [Han et al., 2000] genutzten Implementierungen, wobei diese Effizienzsteigerung für Apriori größer ist. Damit ist der Vorteil von FP-Growth gegenüber Apriori zwar wesentlich, aber letztendlich deutlich geringer ausgeprägt als in [Han et al., 2000] gemessen.

[35] Zum Einsatz kam ein auf Linux 2.4.18 basierendes PC-System, welches über einem mit 150 MHz getakteten Pentium III Prozessor sowie 96 Megabyte Hauptspeicher verfügt. Um die absoluten Zeiten mit denen in [Han et al., 2000] auf einem 450 MHz Pentium System gemessenen vergleichen zu können, wurden die im folgenden angegebenen Laufzeiten durch Division mit 3 aus den tatsächlichen Laufzeiten ermittelt. Diese Abschätzung ist als ausreichend für die hier verfolgten Zwecke anzusehen.

[36] Diese und die weiteren Zeiten wurden aus verschiedenen Abbildungen in [Han et al., 2000] abgelesen und können daher oft nicht exakt angegeben werden.

[Zheng et al., 2001] sehen ebenfalls FP-Growth näher an Apriori als in [Han et al., 2000] dargestellt. Die Laufzeiten in [Zheng et al., 2001] konnten im Rahmen der vorliegenden Arbeit jedoch nicht nachvollzogen werden, zum einen weil kein Rechnersystem, das zu dem in [Zheng et al., 2001] eingesetzten vergleichbar gewesen wäre, zur Verfügung stand, zum anderen weil die verwendete Implementierung von Apriori nicht mehr mit dem herkömmlichen Apriori aus [Agrawal und Srikant, 1994a], [Agrawal und Srikant, 1994b], [Agrawal et al., 1996] vergleichbar ist, sondern unter anderem Elemente von FP-Growth beinhaltet (vgl. dazu [Borgelt und Kruse, 2002]).

Abschließender Vergleich auf Daten realer Anwendungen In den Abbildungen 3.22 bis 3.24 zeigt sich die Praxistauglichkeit der Verfahren anhand verschiedener ausgewählter realer Analyseanwendungen. Ergänzende Experimente auf realen Daten sind in Anhang A.1.1 zu finden.

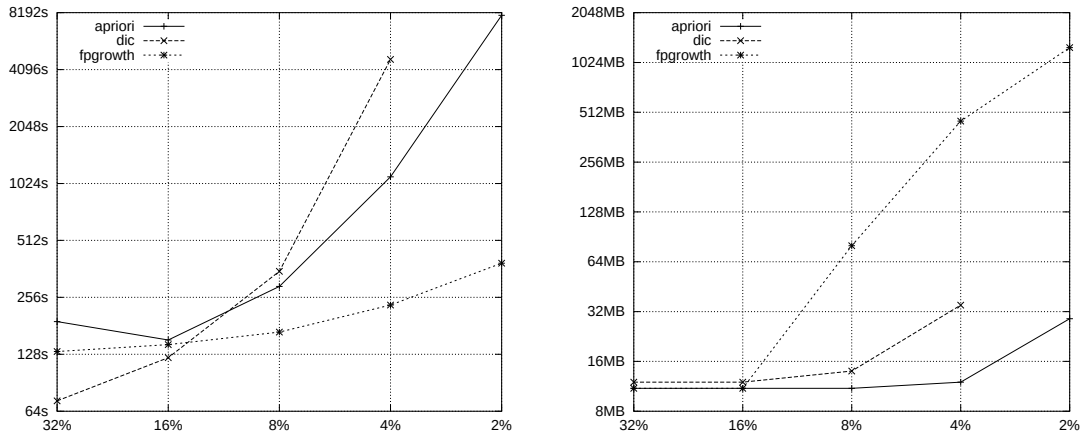


(a) Laufzeit in Sekunden

(b) Speicherbedarf in Megabyte

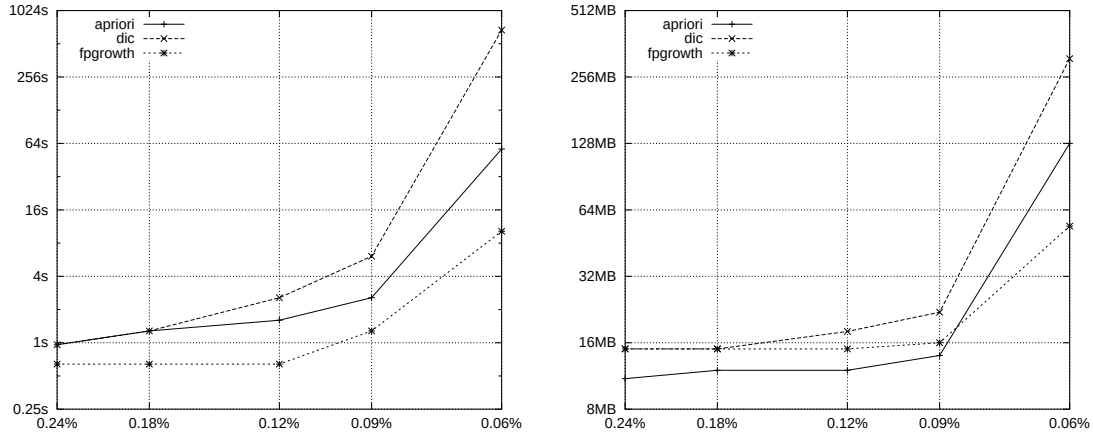
Abbildung 3.22: Effizienz zählender Verfahren auf EINZELHANDEL bei variierendem minsupp

Grundsätzlich bestätigen die Ergebnisse auf den realen Daten die auf den synthetischen Daten gewonnenen Einschätzungen der Verfahren. So bleibt Apriori gegenüber DIC das effizientere Verfahren und FP-Growth ist zumindest bezüglich der Laufzeit Apriori eindeutig überlegen. Insbesondere hinsichtlich der Analyse des Datensatzes FAHRZEUG wird jedoch auch deutlich, daß die in realen Szenarien unter Umständen sehr großen Datenmengen für den Algorithmus FP-Growth problematisch sind. Ähnlich lassen sich auch die Ergebnisse auf dem Datensatz EINZELHANDEL interpretieren. Die von FP-Growth erreichten Laufzeiten sind bis zu 90% kürzer als die von Apriori. Der Datensatz EINZELHANDEL enthält allerdings



(a) Laufzeit in Sekunden (b) Speicherbedarf in Megabyte

Abbildung 3.23: Effizienz zählender Verfahren auf FAHRZEUG bei variierendem minsupp



(a) Laufzeit in Sekunden (b) Speicherbedarf in Megabyte

Abbildung 3.24: Effizienz zählender Verfahren auf BMS-WEBVIEW1 bei variierendem minsupp

lediglich die Daten einer Filiale über einen Monat. Werden die Daten über den Zeitraum eines Jahres für n Filialen analysiert, dann bedeutet das für FP-Growth bei $\mathsf{minsupp} = 0{,}03125\%$ einen Speicherbedarf von ungefähr $12n \cdot 61$ Megabyte $= 732n$ Megabyte, während für Apriori der Speicherbedarf unabhängig von n bei ungefähr 130 Megabyte konstant bleibt.

Schneidende Algorithmen

Von den in Abschnitt 3.2 als schneidend klassifizierten Algorithmen wurden im Rahmen der vorliegenden Arbeit die Verfahren Partition, Eclat und dEclat implementiert.

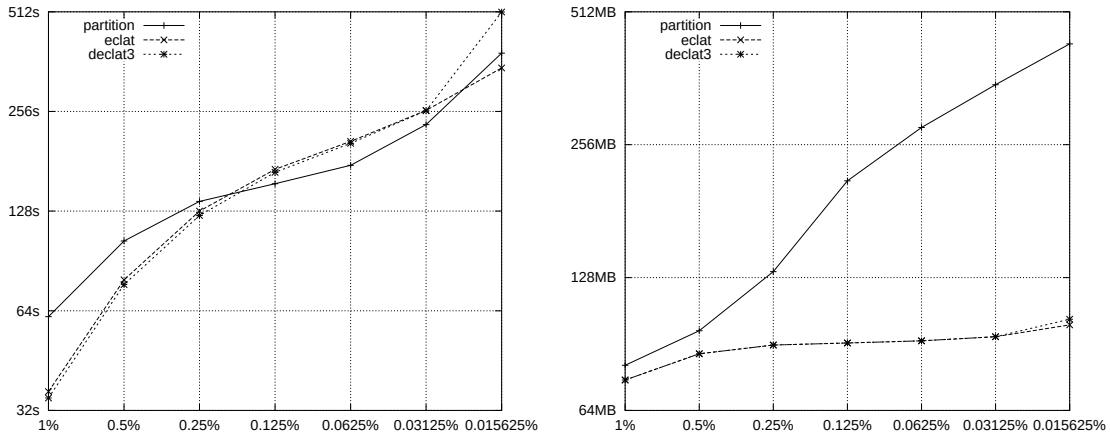
Die indirekte Bestimmung der Häufigkeiten von Ereigniskombinationen durch das Schneiden von Transaktionsmengen wurde in [Savasere et al., 1995a] mit dem Algorithmus Partition eingeführt. Die folgenden Evaluierungen basieren auf einer Implementierung dieses Algorithmus, welche die Daten nicht zerlegt. Damit kann der abschließende Lauf über die Transaktionen, der die tatsächlichen Häufigkeiten der Ereigniskombinationen bestimmt, entfallen. Diese Variante des Partition-Algorithmus ist bezüglich der Laufzeit effizienter als der ursprüngliche Algorithmus aus [Savasere et al., 1995a]. Die Ergebnisse der Hauptspeichermessungen sind jedoch nur eingeschränkt gültig. So skaliert der Hauptspeicherbedarf umgekehrt linear mit der Anzahl der Partitionen und kann auch für große Datenmengen nahezu beliebig klein gehalten werden. Diese Möglichkeit besteht entsprechend nicht für die hier verwendete Implementierung von Partition. Eine wachsende Anzahl von Partitionen bedeutet allerdings zugleich längere Laufzeiten. Mit SPINC und AS-CPA aus [Mueller, 1995] wurden zur Lösung dieses Problems zwei Varianten von Partition eingeführt, welche die Partitionierung bezüglich der Laufzeit optimieren. Beide Ansätze werden in der folgenden Evaluierung von dem hier herangezogenen Partition-Algorithmus bezüglich der Laufzeit mit abgedeckt, da dieser die zugrundeliegende Transaktionsdatenbank nicht zerlegt und daher am effizientesten ist.

Neben Partition wurde im Rahmen der vorliegenden Arbeit stellvertretend für die in [Zaki et al., 1997a], [Zaki et al., 1997c] eingeführte Algorithmenfamilie der Algorithmus Eclat implementiert.[37] Um Eclat mit den anderen Verfahren vergleichen zu können, muß dieser Algorithmus so erweitert werden, daß nicht nur häufige Ereigniskombinationen mit einer Mächtigkeit größer gleich 3 erzeugt werden, sondern ebenfalls die häufigen Ereigniskombinationen, die aus einem einzelnen Ereignis oder zwei Ereignissen bestehen. Dazu werden die Transaktionsmengen für die einzelnen Ereignisse generiert und Eclat für die Äquivalenzklasse $E(\emptyset)$ aufgerufen (vgl. Abschnitt 3.1.2 und [Hipp et al., 2000a]).

[37] Mohammed J. Zaki wählt ebenfalls für seine Experimente in [Zaki und Gouda, 2001] diesen Algorithmus aus [Zaki et al., 1997a], [Zaki et al., 1997c] als Benchmark.

Mit dEclat aus [Zaki und Gouda, 2001] wurde in die Evaluierung außerdem eine Variante von Eclat aufgenommen, welche für Ereigniskombinationen höherer Mächtigkeit anstelle der üblichen Transaktionsmengen auf Differenzmengen basiert. Als Schwellenwert für den Wechsel von Transaktionsmengen zu Differenzmengen wurde nach [Zaki und Gouda, 2001] der Wert 3 gewählt (vgl. auch Abschnitt 3.2). Entsprechend wird im folgenden dEclat auch als dEclat3 bezeichnet.

Synthetische Daten Der Vergleich der Verfahren auf den synthetischen Daten ergibt keine wesentlichen Unterschiede bezüglich der Laufzeiten (vgl. z. B. die Abbildungen 3.25(a) und 3.26(a) sowie die ergänzenden Evaluierungen aus Anhang A.1.2). Lediglich wenn für den minimalen Support sehr kleine Werte gewählt



(a) Laufzeit in Sekunden

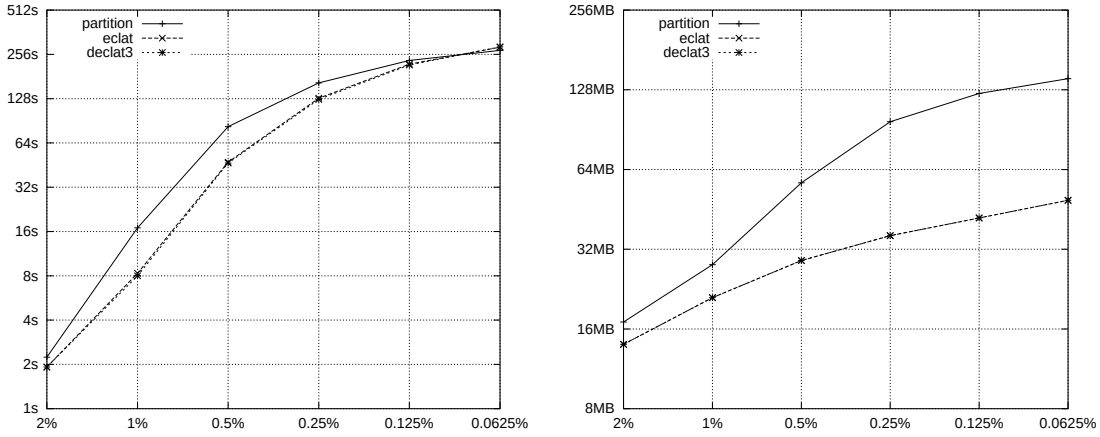
(b) Speicherbedarf in Megabyte

Abbildung 3.25: Effizienz schneidender Verfahren auf T10I4D1000K bei variierendem minsupp

werden, zeigen die Algorithmen Partition und dEclat schlechtere Laufzeiten als Eclat.

Größere Unterschiede ergeben sich bezüglich des Hauptspeicherbedarfs der Verfahren (vgl. z. B. die Abbildungen 3.25(b) und 3.26(b) sowie die ergänzenden Evaluierungen aus Anhang A.1.2). Während Eclat und dEclat ein nahezu identisches Speicherverhalten zeigen, übersteigt der Hauptspeicherbedarf von Partition die Werte der beiden anderen Verfahren bei weitem.

Des weiteren skalieren alle berücksichtigten Ansätze sowohl hinsichtlich der Laufzeit als auch des Hauptspeicherbedarfs nahezu linear mit der Anzahl Transaktionen



(a) Laufzeit in Sekunden

(b) Speicherbedarf in Megabyte

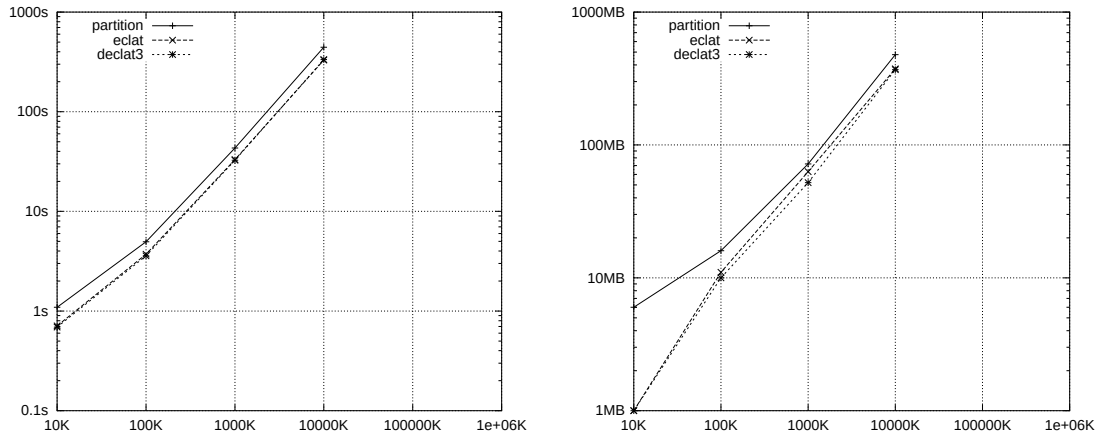
Abbildung 3.26: Effizienz schneidender Verfahren auf T25I20D100K bei variierendem minsupp

(vgl. Abbildung 3.27).[38] Entsprechend kann auf dem eingesetzten Rechnersystem keines der Verfahren für die mit 100 Millionen beziehungsweise einer Milliarde Transaktionen vergleichsweise großen Datensätze die häufigen Ereigniskombinationen generieren.

Interpretation Die Kandidatengenerierung von Partition ist effizienter als die von Eclat, so daß bei der Generierung häufiger Ereigniskombinationen mittels Partition im Vergleich zu Eclat weniger Schnitte über Transaktionsmengen notwendig sind. Umgekehrt basiert Eclat auf Schnellschnitten, welche die Schnittmengenbildung selbst effizienter machen. Beide Aspekte können sich in den durchgeführten Experimenten ungefähr ausgleichen. Es ist allerdings zu berücksichtigen, daß die hier evaluierte Variante des Partition-Algorithmus nicht vollständig ist. Es fehlt die Zerlegung der zugrundeliegenden Transaktionsdatenbank, welche zwar das Hauptspeicherproblem von Partition löst, sich jedoch wegen des zusätzlichen Analyselaufs negativ auf die Laufzeit auswirkt. Insgesamt ist somit Eclat als das gegenüber Partition überlegene Verfahren anzusehen.

Der Wechsel zu Differenzmengen von dEclat hat hingegen keinen bemerkenswerten Einfluß auf die Effizienz gegenüber Eclat.

[38] Anstelle der Sun Workstation kommt für die Experimente zur Skalierung erneut das dazu bereits eingesetzte PC-System zum Einsatz.



(a) Laufzeit in Sekunden

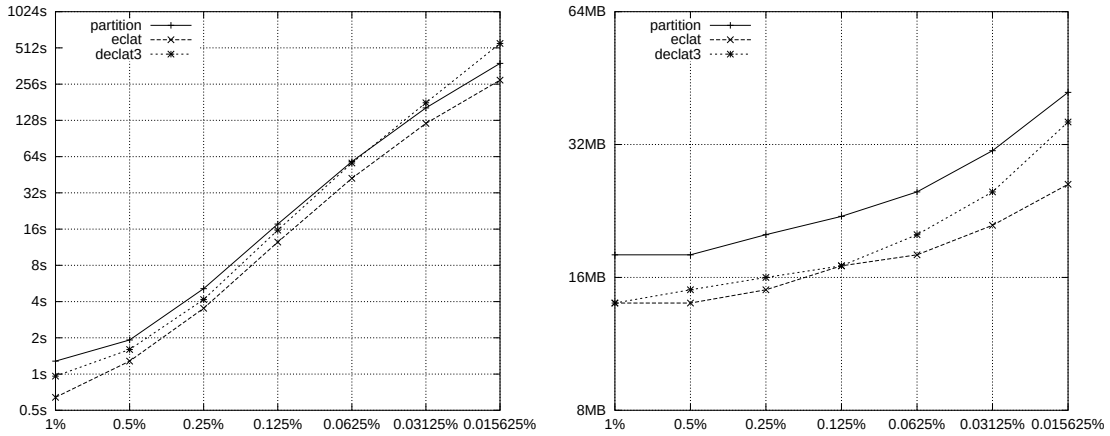
(b) Speicherbedarf in Megabyte

Abbildung 3.27: Skalierbarkeit schneidender Verfahren auf synthetischen Daten bei variierender Anzahl von Transaktionen in Tausend

Vergleichbare Evaluierungen in der Literatur Ein Vergleich der Laufzeiten von Eclat und dEclat ist in [Zaki und Gouda, 2001] zu finden. In Übereinstimmung mit den im Rahmen dieser Arbeit ermittelten Ergebnissen sind sich auch in [Zaki und Gouda, 2001] die Laufzeiten beider Verfahren auf den synthetischen Daten sehr ähnlich. Die Experimente in [Zaki und Gouda, 2001] zeigen außerdem einen deutlichen Vorteil für dEclat, wenn dieser Algorithmus auf besonders dichten Datensätzen, sogenannten „dense dataset“, eingesetzt wird. Diese Datensätze zeichnen sich dadurch aus, daß sie sehr viele häufige Ereigniskombinationen enthalten und in diesem Sinne „dicht“ sind. Aufgrund der sehr großen Anzahl häufiger Ereigniskombinationen ist es jedoch nicht mehr sinnvoll beziehungsweise praktisch möglich, die den Schwellenwerten genügenden Assoziationsregeln vollständig zu erzeugen. Entsprechend sind solche dichten Datensätze im Rahmen des Support-Konfidenz-Ansatzes nicht von Bedeutung (vgl. u. a. [Bayardo et al., 1999]) und werden in dieser Arbeit nicht weiter betrachtet.

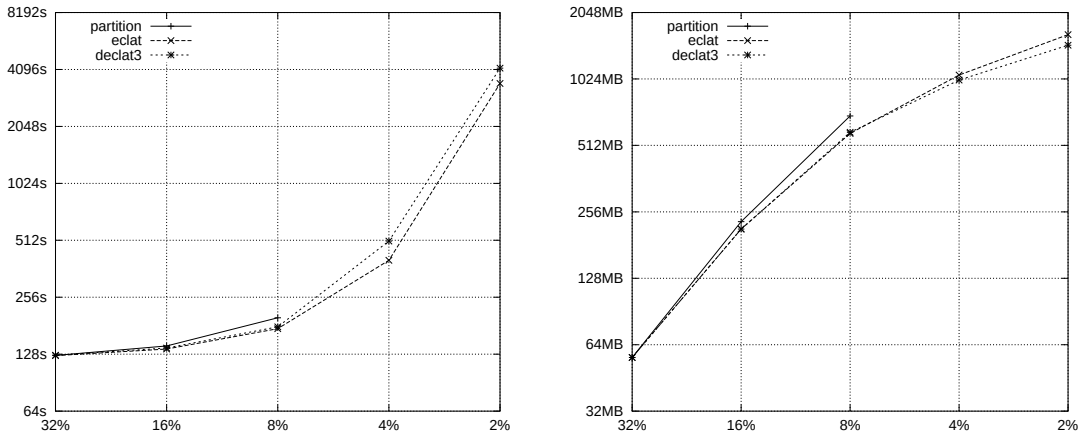
Abschließender Vergleich auf Daten realer Anwendungen In den Abbildungen 3.28 bis 3.30 werden die schneidenden Verfahren anhand von Datensätzen aus realen Anwendungen bezüglich ihrer Praxistauglichkeit evaluiert (für weitere Experimente siehe Anhang A.1.2).

Grundsätzlich werden die auf den synthetischen Daten gewonnenen Ergebnisse und Einschätzungen bestätigt. Die Laufzeiten der Verfahren sind sich ähnlich, wobei Eclat gegenüber Partition und dEclat die kürzeren Laufzeiten erreicht. Anhand



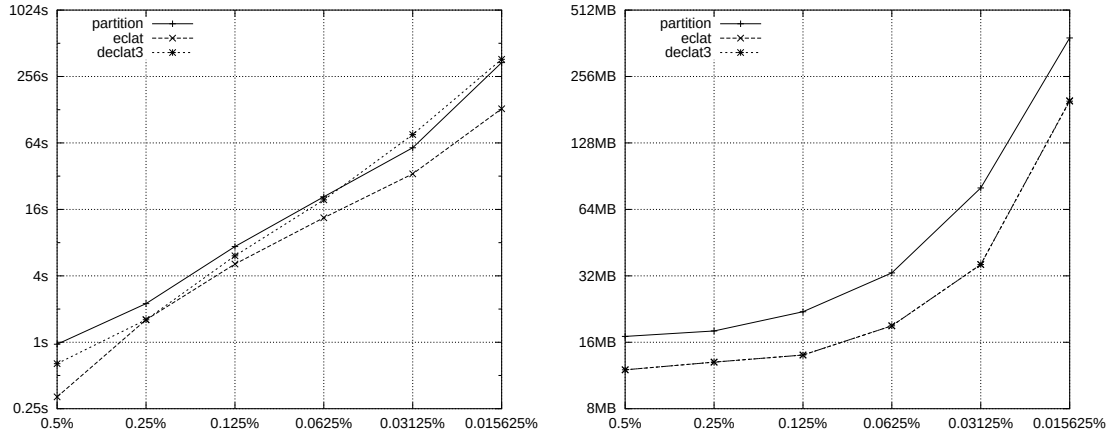
(a) Laufzeit in Sekunden (b) Speicherbedarf in Megabyte

Abbildung 3.28: Effizienz schneidender Verfahren auf EINZELHANDEL bei variierendem minsupp



(a) Laufzeit in Sekunden (b) Speicherbedarf in Megabyte

Abbildung 3.29: Effizienz schneidender Verfahren auf FAHRZEUG bei variierendem minsupp



(a) Laufzeit in Sekunden

(b) Speicherbedarf in Megabyte

Abbildung 3.30: Effizienz schneidender Verfahren auf BMS-WEBVIEW2 bei variierendem minsupp

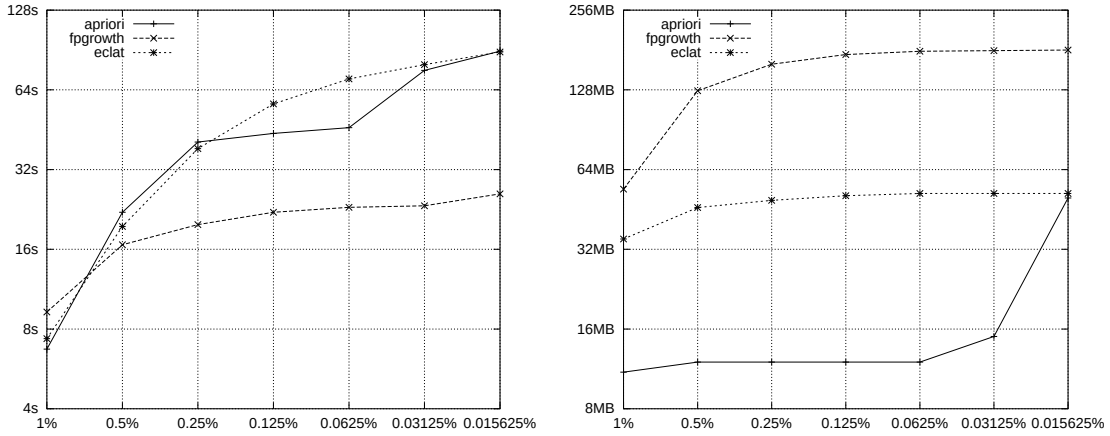
des Datensatzes FAHRZEUG (vgl. Abbildung 3.29) wird jedoch offensichtlich, daß in praktischen Anwendungen unter Umständen der von den Verfahren benötigte Hauptspeicher als kritisch einzuschätzen ist.

Zusammenfassender Vergleich

Im folgenden werden aufgrund der obigen Auswertungen die Algorithmen Apriori, FP-Growth, Eclat und dEclat als die effizientesten der vorgestellten etablierten Verfahren für einen abschließenden und zusammenfassenden Vergleich ausgewählt.

Synthetische Daten In den Abbildungen 3.31 bis 3.33 werden zählende und schneidende Verfahren direkt miteinander verglichen. Zunächst fällt auf, daß Apriori als zählendes Verfahren und Eclat sowie dEclat als schneidende Verfahren in vielen der Experimente sehr ähnliche Laufzeiten erreichen. Lediglich in einigen der Experimente liegen die Laufzeiten der Verfahren bis zu Faktor 2 auseinander. FP-Growth hingegen ist auf den weniger anspruchsvollen Datensätzen T5I2D1000K und T10I4D1000K deutlich das schnellste Verfahren. Auf T5I2D1000K kann FP-Growth um bis zu 70% kürzere Laufzeiten als die anderen Verfahren erreichen, auf T10I4D1000K um bis zu 60%. Auf den anspruchsvolleren Datensätzen, wie beispielsweise T20I6D1000K, relativiert sich dieser Vorteil und in einzelnen Ex-

perimenten erreicht Apriori Laufzeiten, die bis zu 75% kürzer sind als die von FP-Growth (siehe auch die ergänzenden Experimente auf synthetischen Daten in Anhang A.1.3).



(a) Laufzeit in Sekunden

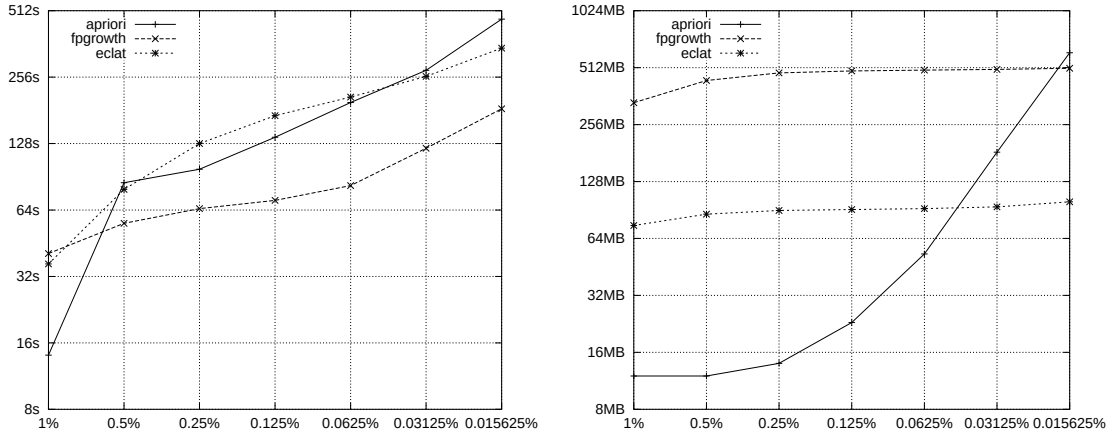
(b) Speicherbedarf in Megabyte

Abbildung 3.31: Effizienz ausgewählter etablierter Verfahren auf T5I2D1000K bei variierendem minsupp

Bezüglich des Hauptspeicherbedarfs sind die Unterschiede zwischen den Verfahren wesentlich ausgeprägter. So sind für Apriori bei vergleichsweise großen Schwellenwerten minsupp auf den drei Datensätzen 12 Megabyte Hauptspeicher ausreichend, während Eclat für die gleiche Aufgabe bis zu 49, 74 beziehungsweise 159 Megabyte benötigt. Mit sinkendem minsupp steigt der Hauptspeicherbedarf von Apriori jedoch steil an und liegt in vielen der Experimente weit über dem von Eclat. Auf T10I4D1000K benötigt Apriori bis zu 500% mehr Hauptspeicher als Eclat, auf T20I6D1000K bis zu 400%. Der Hauptspeicherbedarf von Eclat bleibt nach einem anfänglichen Anstieg nahezu konstant. Ebenso steigt der Hauptspeicherbedarf von FP-Growth zunächst steil an und wächst aber auf hohem Niveau nur wenig weiter. Im einzelnen liegt auf den drei Datensätzen der Hauptspeicherbedarf von FP-Growth um bis zu 240%, 500% beziehungsweise 430% über dem von Eclat.

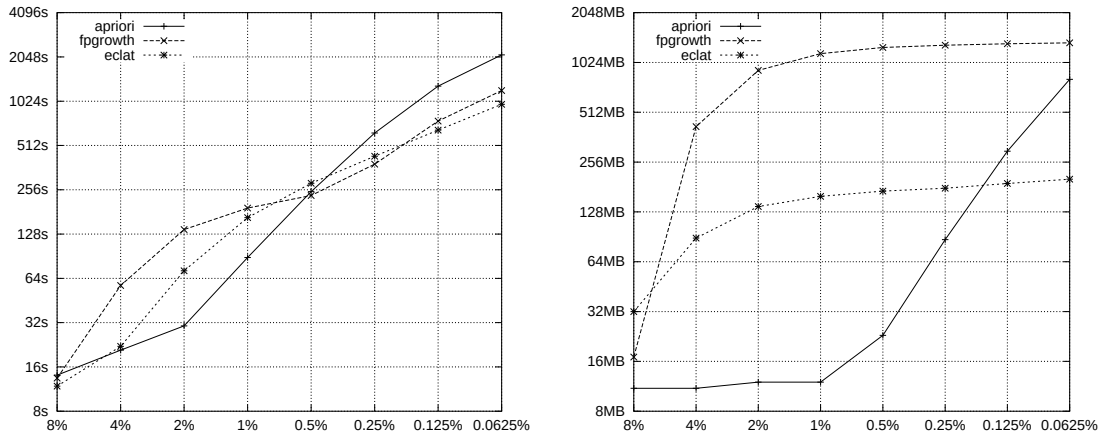
In Abbildung 3.34 werden die grundsätzlichen Unterschiede bezüglich der Skalierbarkeit der Verfahren deutlich, wobei wieder für den festen Schwellenwert minsupp = 0,5% alle häufigen Ereigniskombinationen generiert werden.[39] Während hin-

[39] Anstelle der Sun Workstation kommt für die Experimente zur Skalierung erneut das dazu bereits eingesetzte PC-System zum Einsatz.



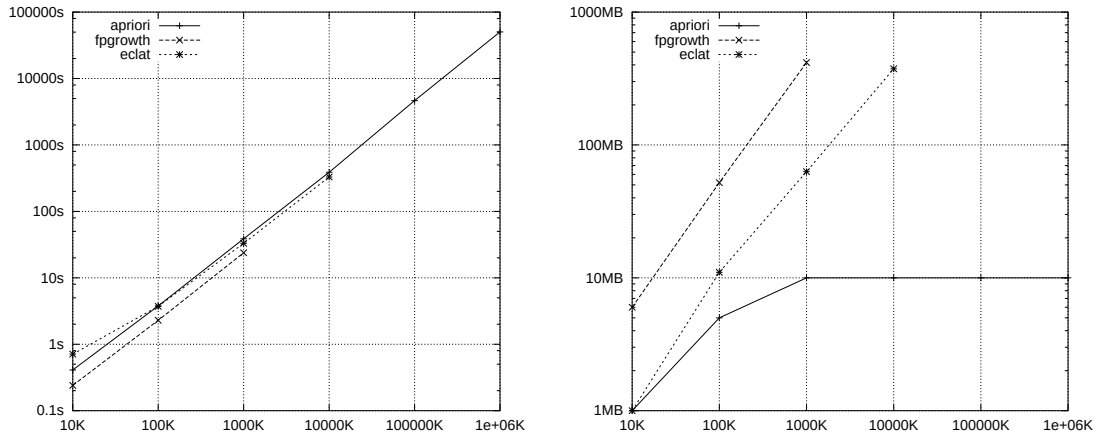
(a) Laufzeit in Sekunden (b) Speicherbedarf in Megabyte

Abbildung 3.32: Effizienz ausgewählter etablierter Verfahren auf T10I4D1000K bei variierendem minsupp



(a) Laufzeit in Sekunden (b) Speicherbedarf in Megabyte

Abbildung 3.33: Effizienz ausgewählter etablierter Verfahren auf T20I6D1000K bei variierendem minsupp



(a) Laufzeit in Sekunden

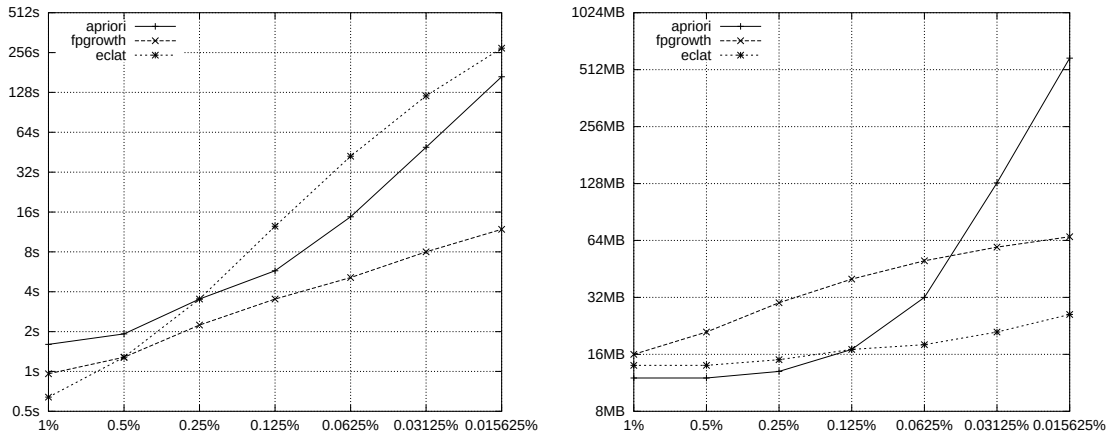
(b) Speicherbedarf in Megabyte

Abbildung 3.34: Skalierbarkeit ausgewählter etablierter Verfahren auf synthetischen Daten bei variierender Anzahl von Transaktionen in Tausend

sichtlich der Laufzeiten alle Verfahren linear mit der Anzahl der Transaktionen skalieren, ist der Speicherbedarf von Apriori insbesondere für größere Datensätze weitgehend konstant. Im Gegensatz dazu wächst der Speicherbedarf der anderen Verfahren linear mit der Anzahl der Transaktionen. Lediglich Apriori kann daher auf der gegebenen Hardware die Grenze von 10 Millionen Transaktionen überschreiten und die Menge der häufigen Ereigniskombinationen erfolgreich auf einer Milliarde Transaktionen generieren.

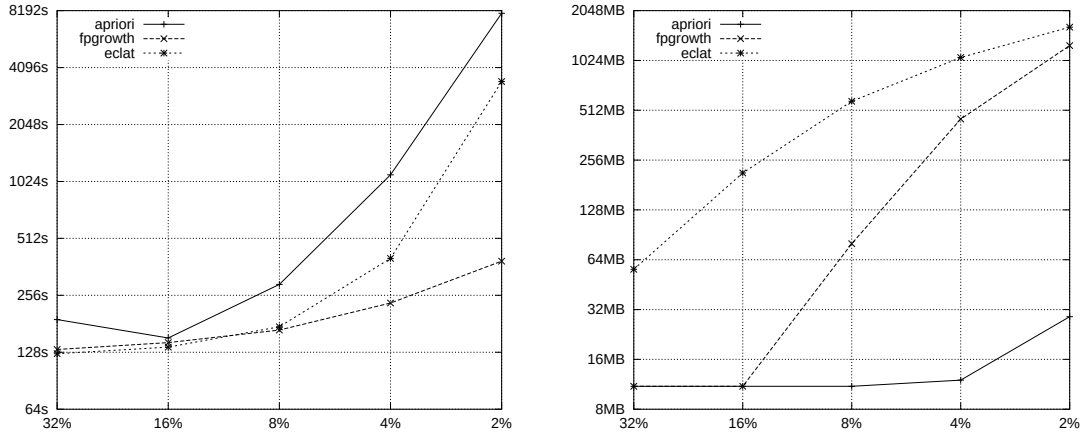
Abschließender Vergleich auf Daten realer Anwendungen In den Abbildungen 3.35 bis 3.39 werden die ausgewählten Verfahren auf sämtlichen in Abschnitt 3.3.1 eingeführten und auf realen Anwendungen basierenden Datensätzen hinsichtlich ihrer Praxistauglichkeit evaluiert. Die Experimente bestätigen FP-Growth als das bezüglich der Laufzeit deutlich effizienteste Verfahren, das jedoch in Abhängigkeit von der Anzahl und dem Umfang der zugrundeliegen Transaktionen schnell einen kritischen Hauptspeicherbedarf erreicht. Das zählende Verfahren Apriori zeigt gegenüber dem schneidenden Verfahren Eclat auf dem Datensatz EINZELHANDEL fast durchweg kürzere Laufzeiten, auf sämtlichen anderen Datensätzen ist hingegen Eclat das effizientere der beiden Verfahren.

Vergleichbare Evaluierungen in der Literatur Experimente, die Eclat mit Apriori vergleichen, sind in [Zaki et al., 1997a], [Zaki et al., 1997c] zu finden. Die Ergebnisse dort sind allerdings nur bedingt aussagekräftig, da lediglich die Zei-



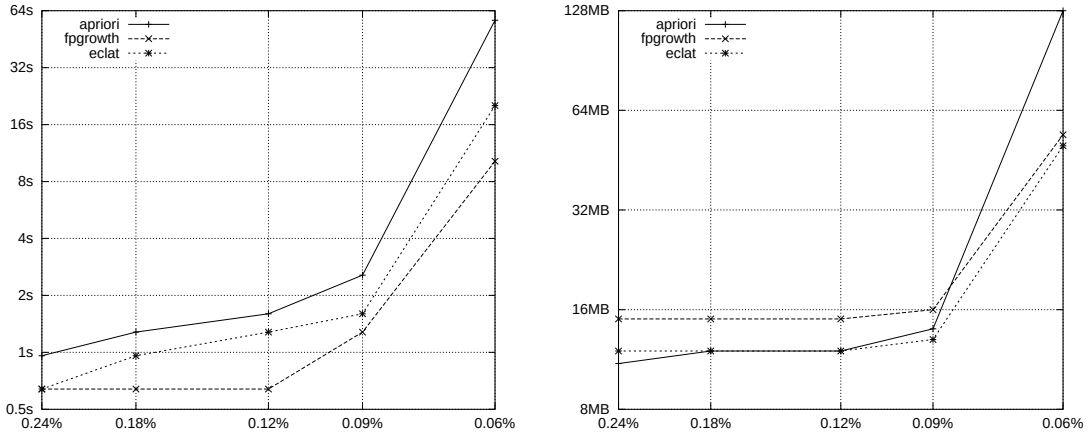
(a) Laufzeit in Sekunden (b) Speicherbedarf in Megabyte

Abbildung 3.35: Effizienz ausgewählter etablierter Verfahren auf EINZELHANDEL bei variierendem minsupp



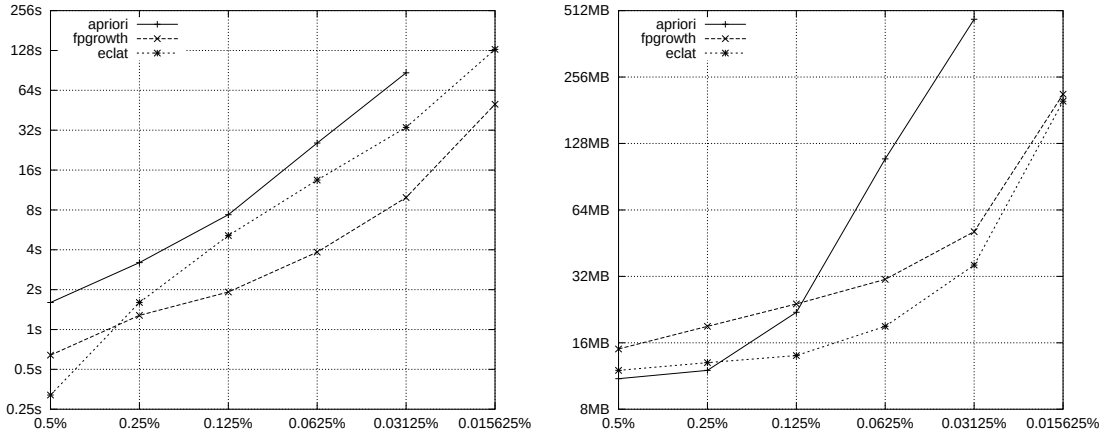
(a) Laufzeit in Sekunden (b) Speicherbedarf in Megabyte

Abbildung 3.36: Effizienz ausgewählter etablierter Verfahren auf FAHRZEUG bei variierendem minsupp



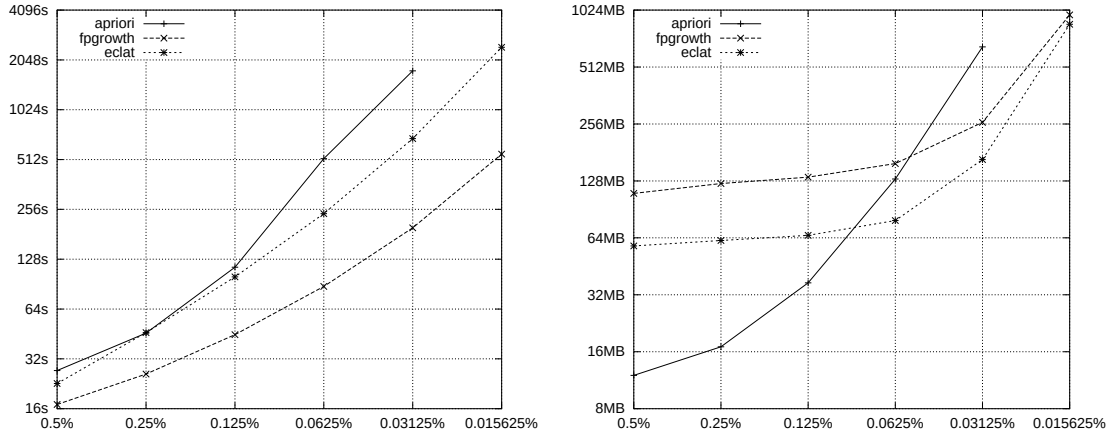
(a) Laufzeit in Sekunden (b) Speicherbedarf in Megabyte

Abbildung 3.37: Effizienz ausgewählter etablierter Verfahren auf BMS-WEBVIEW1 bei variierendem minsupp



(a) Laufzeit in Sekunden (b) Speicherbedarf in Megabyte

Abbildung 3.38: Effizienz ausgewählter etablierter Verfahren auf BMS-WEBVIEW2 bei variierendem minsupp



(a) Laufzeit in Sekunden

(b) Speicherbedarf in Megabyte

Abbildung 3.39: Effizienz ausgewählter etablierter Verfahren auf BMS-POS bei variierendem minsupp

ten für die Generierung der häufigen Ereigniskombinationen ab einschließlich der Mächtigkeit 3 angegeben werden. Die experimentellen Vergleiche von dEclat mit Apriori [Zaki und Gouda, 2001] betreffen fast ausschließlich dichte Datensätze, die im Rahmen dieser Arbeit nicht weiter untersucht werden. Eine systematische Betrachtung des Speicherbedarfs der Verfahren fehlt bisher in der Literatur.

3.3.3 Zusammenfassung und Bewertung

In diesem Abschnitt wurden die wichtigsten der heute als etabliert anzusehenden Verfahren zur Generierung häufiger Ereigniskombinationen evaluiert. Dazu wurde eine Vielzahl verschiedener Experimente sowohl auf synthetischen Datensätzen als auch auf Datensätzen aus realen Anwendungen durchgeführt. Die Eigenschaften der synthetischen Datensätze konnten gezielt verändert und auf diese Weise Aussagen bezüglich des Einflusses der Charakteristika der Datensätze abgeleitet werden. Die Experimente auf den realen Daten zeigten, inwieweit die Verfahren für praktische Anwendungen geeignet sind.

Die hier durchgeführte Evaluierung zeichnet sich unter anderem durch die vollständige Abdeckung der zusammen mit den etablierten Verfahren veröffentlichten Datensätze und Experimente aus. Des weiteren wurden die beschriebenen Verfahren erstmals gemeinsam und unter gleichen Rahmenbedingungen evaluiert. Insbesondere werden zum ersten Mal nicht nur die Laufzeiten, sondern auch der Hauptspeicherbedarf der Verfahren einbezogen. Letzterer entscheidet darüber, ob ein

Verfahren auf einen Datensatz anwendbar ist, da aufgrund der wenig lokalen Speicherzugriffe der Beginn des Auslagerns von Hauptspeicherseiten auf Sekundärspeicher die Verfahren nahezu zum Erliegen bringt.

Erstes Ergebnis der hier vorliegenden umfassenden Untersuchungen ist, daß, anders als verschiedene veröffentlichte Vergleiche im einzelnen den Eindruck erwecken (vgl. z. B. [Savasere et al., 1995a], [Zaki et al., 1997a], [Zaki et al., 1997c], [Han et al., 2000], [Zaki und Gouda, 2001]), ein allgemein „effizientestes Verfahren“ nicht existiert. Vielmehr ergibt sich insgesamt ein sehr differenziertes Bild. In Abhängigkeit von den zu analysierenden Daten erreichen unterschiedliche Verfahren die kürzesten Laufzeiten, wobei kurze Laufzeiten jedoch unter Umständen mit einem hohen Speicherbedarf einhergehen, so daß große Datenmengen oder besonders kleine Schwellenwerte für den minimalen Support gegebenenfalls nicht mehr bearbeitet werden können.

Als die vielversprechendsten Verfahren zur Assoziationsregelgenerierung wurden in der vorliegenden Arbeit die Algorithmen Apriori, FP-Growth, Eclat und dEclat identifiziert.

Im einzelnen zeichnet sich der Algorithmus Apriori dadurch aus, daß er für nicht zu klein vorgegebene minimale Häufigkeiten praktisch beliebig große Transaktionsdatenbanken analysieren kann. Die dabei erreichten Laufzeiten sind durchaus mit denen der anderen Verfahren vergleichbar, wenn Apriori auch für besonders anspruchsvolle Datensätze oder vergleichsweise geringe Werte für den minimalen Support den anderen Verfahren bezüglich der Laufzeit unterlegen ist. Der benötigte Hauptspeicher ist nahezu unabhängig von der Anzahl zugrundeliegender Transaktionen, skaliert allerdings in Abhängigkeit von den Daten teilweise exponentiell mit dem vorgegebenen Schwellenwert für den Support.

FP-Growth als datenorientiertes Verfahren erreicht auf vielen der Datensätze die mit Abstand kürzesten Laufzeiten. Für anspruchsvollere Datensätze mit Transaktionen und häufigen Ereigniskombinationen größerer Mächtigkeit nähern sich die Laufzeiten von FP-Growth allerdings zunehmend an die der anderen Verfahren an. Die Generierung häufiger Ereigniskombinationen mittels FP-Growth weist außerdem in den meisten Fällen einen sehr hohen Speicherbedarf auf, wobei dieser ungefähr linear mit der Anzahl zugrundeliegender Transaktionen skaliert. Für die Analyse sehr großer Datenbanken erscheint FP-Growth daher nicht geeignet.

Die Laufzeiten der schneidenden Verfahren Eclat und dEclat sind zumeist vergleichbar mit den von Apriori erreichten Laufzeiten beziehungsweise sogar kürzer als diese, wobei unter bestimmten Umständen dEclat auch wesentlich schlechtere Werte erreichen kann. Der Hauptspeicherbedarf beider Verfahren skaliert ebenfalls wie der Hauptspeicherbedarf von FP-Growth ungefähr linear mit der Anzahl zugrundeliegender Transaktionen, die gemessenen Werte liegen jedoch weit unter denen von FP-Growth.

Zusammenfassend ist FP-Growth als das effizienteste Verfahren bezüglich der Laufzeit anzusehen. Allerdings können in Abhängigkeit vom verfügbaren Hauptspeicher bereits Datensätze mittlerer Größenordnungen unter Umständen mit FP-Growth nicht mehr analysiert werden. Für solche Datensätze sind die Algorithmen Eclat beziehungsweise dEclat vorzuziehen, wobei diese Verfahren mit zunehmender Anzahl von Transaktionen ebenfalls die Grenzen des Hauptspeichers erreichen können. Letztlich ist für sehr große Datensätze lediglich der Algorithmus Apriori geeignet, dessen Einsatz jedoch zumeist mit einer weiteren Verschlechterung der Laufzeiten einhergeht.

Insgesamt sind die etablierten Verfahren mit den erwähnten Einschränkungen geeignet, auch große Datenmengen zu analysieren. Die erreichten Laufzeiten sind allerdings insbesondere vor dem Hintergrund des in Abschnitt 2.1.1 beschriebenen interaktiven Wissensentdeckungsprozesses als wenig befriedigend anzusehen.

Kapitel 4

Neu- und Weiterentwicklungen

In diesem Kapitel werden verschiedene Optimierungen der bereits vorgestellten Algorithmen und neue Ansätze zur Generierung einfacher und taxonomer Assoziationsregeln eingeführt. Grundlage bildet die Analyse der etablierten Verfahren aus Kapitel 3. In Abschnitt 4.1 werden zwei neue Ansätze zur Generierung einfacher Assoziationsregeln vorgestellt. Der erste Ansatz basiert auf einer optimierten Suchstrategie für die Tiefensuche. Mit dieser ist es möglich, den Suchraum anhand nichthäufiger Teilmengen von Ereigniskombinationen zu beschneiden und so die Häufigkeitsbestimmung für viele der Kandidaten einzusparen. Der zweite eingeführte Ansatz kombiniert Tiefensuche und Breitensuche sowie direkte und indirekte Häufigkeitsbestimmung. Das resultierende hybride Verfahren erreicht in vielen Experimenten weit kürzere Laufzeiten und weist einen geringeren Speicherbedarf auf als die zugrundeliegenden Ansätze im einzelnen. Die neuen Verfahren werden im Vergleich mit den etablierten Ansätzen ausführlich evaluiert. In Abschnitt 4.2 werden die etablierten Algorithmen dann erweitert, so daß diese Taxonomien in die Regelgenerierung einbeziehen können. Außerdem werden neue Verfahren eingeführt, welche die Taxonomie effizient zur Beschneidung des Suchraums nutzen. Eine Evaluierung zeigt, daß diese Verfahren wesentlich kürzere Antwortzeiten erreichen als die erweiterten etablierten Algorithmen. Das Kapitel schließt in Abschnitt 4.3 mit einer zusammenfassenden Bewertung der hier neu eingeführten Verfahren zur Generierung einfacher und taxonomer Assoziationsregeln.

4.1 Einfache Assoziationsregeln

In diesem Abschnitt werden die Erweiterungen und Optimierungen der Verfahren zur Generierung einfacher Assoziationsregeln eingeführt. Dazu wird zunächst die Tiefensuche erweitert und im Anschluß der hybride Ansatz vorgestellt. Es folgt die Evaluierung der neuen Ansätze gemeinsam mit den etablierten Verfahren.

4.1.1 Optimierte Tiefensuche für schneidende Verfahren

Das Verfahren Eclat durchläuft den Suchraum mittels einer Tiefensuche (vgl. Abschnitt 3.2). Für Eclat ist es daher nicht möglich, den Suchraum anhand nichthäufiger Teilmengen von Ereigniskombinationen vollständig zu beschneiden. Der Hintergrund dieser in [Zaki et al., 1997a], [Zaki et al., 1997c] nicht erwähnten Einschränkung ist, daß zum Zeitpunkt der Häufigkeitsbestimmung eines Kandidaten nicht immer die Häufigkeiten sämtlicher Teilmengen des Kandidaten bekannt sind. Beispielsweise kann eine Tiefensuche im Baum aus Abbildung 3.3 die Äquivalenzklasse $E(\{a\})$ und damit $E(\{a, c\})$ vor der Äquivalenzklasse $E(\{c\})$ besuchen. In diesem Fall ist zum Zeitpunkt der Häufigkeitsbestimmung des Kandidaten $\{a, c, d\} \in E(\{a, c\})$ nicht bekannt, daß dessen Teilmenge $\{c, d\} \in E(\{c\})$ nichthäufig ist. Die Häufigkeit des Kandidaten $\{a, c, d\}$ wird folglich aufwendig anhand der Datenbank bestimmt, obwohl dieser Kandidat aufgrund seiner nichthäufigen Teilmenge $\{c, d\}$ nicht häufig sein kann.

Die Anzahl von Vorkommen in der Datenbank wird lediglich für die häufigen der betrachteten Kandidaten während der Tiefensuche gespeichert, da betrachtete nichthäufige Ereignisse abzulegen einen großen zusätzlichen Speicherbedarf bedeuten würde. Ist während des Abstiegs im Suchraum zu der Häufigkeit einer Ereigniskombination nichts bekannt, dann ist diese Ereigniskombination folglich entweder nichthäufig oder wurde bisher nicht betrachtet. Daher ist auch ein teilweises Heranziehen nichthäufiger Ereigniskombinationen zur Beschneidung des Suchraums nicht ohne weiteres möglich.

Erweiterter Ansatz

Um die Beschneidung des Suchraums anhand nichthäufiger Ereigniskombinationen auch für die auf einer Tiefensuche beruhenden Verfahren möglich zu machen, müssen zwei Voraussetzungen erfüllt sein:

(a) Bevor bestimmt wird, ob ein Kandidat K häufig ist, müssen sämtliche Teilmengen von K bereits bearbeitet worden sein.

(b) Ein effizienter Zugriff auf die Häufigkeiten der bereits ausgezählten Ereigniskombinationen ist zu gewährleisten, damit die Häufigkeiten der Teilmengen eines Kandidaten schnell gefunden werden können.

Sei $X.\mathsf{item}_n,\ X \subseteq \mathcal{E}$, das n-te Ereignis der eineindeutig auf die natürlichen Zahlen abgebildeten und aufsteigend sortierten Ereignisse aus X (vgl. dazu auch Abschnitt 3.1.2). Nach [Hipp et al., 1998] sei außerdem die Relation $<_{\mathcal{L}}$ für Paare

X, Y von Ereigniskombinationen gleicher Mächtigkeit, $X, Y \in K_m$, $K_m = \{K \subseteq \mathcal{E} \mid |K| = m\}$, wie folgt definiert:

$$X <_{\mathcal{L}} Y :\Leftrightarrow \exists n \in \mathbb{N}, n \leq m : X.\mathsf{item}_n < Y.\mathsf{item}_n \quad \wedge$$
$$\forall n' \in \mathbb{N} \wedge n' < n : X.\mathsf{item}_{n'} = Y.\mathsf{item}_{n'}$$

Die Relation $<_{\mathcal{L}}$ definiert eine lexikographische Ordnung auf Ereigniskombinationen gleicher Mächtigkeit. Für jede Teilmenge K'_m von K_m existiert daher genau eine im Sinne der Relation $<_{\mathcal{L}}$ größte Ereigniskombination.
Sei $K = \{k_1, \ldots, k_n\}$ mit $k_i = K.\mathsf{item}_i$ eine Ereigniskombination. Sei außerdem $P = \{k_1, \ldots, k_{n-1}\}$ das $(|K|-1)$-Präfix von K. Für alle $S \subseteq K$ mit $|S| = |K| - 1$ gilt dann:

$$\begin{aligned} S \neq P &\Rightarrow \exists j \in \mathbb{N} : S = \{k_1, \ldots, k_{j-1}, k_{j+1}, \ldots, k_n\} \\ &\Rightarrow \exists j \in \mathbb{N} : P.\mathsf{item}_j < S.\mathsf{item}_j \wedge \forall n' \in \mathbb{N}, n' < j : P.\mathsf{item}_{n'} = S.\mathsf{item}_{n'} \\ &\Rightarrow P <_{\mathcal{L}} S. \end{aligned}$$

□

In Abschnitt 3.1.2 wurde der Suchraum in Äquivalenzklassen eingeteilt, wobei die Ereigniskombinationen einer Äquivalenzklasse $E(P)$ das gleiche $(|P|-1)$-Präfix P haben. Folglich gilt für alle Teilmengen S mit $|S| = |K| - 1, S \neq P$, der Ereigniskombinationen $K \in E(P)$ die Eigenschaft $P <_{\mathcal{L}} S$. Sind alle nichthäufigen Ereigniskombinationen S mit $P <_{\mathcal{L}} S$ bekannt, können folglich die Kandidaten K aus $E(P)$ anhand nichthäufiger Teilmengen beschnitten werden.[1] Um zu erreichen, daß die nichthäufigen Ereigniskombinationen S mit $P <_{\mathcal{L}} S$ bekannt sind, bevor die Häufigkeiten der Kandidaten in $E(P)$ bestimmt werden, müssen die Äquivalenzklassen in der Weise abgearbeitet werden, daß jeweils die im Sinne der Relation $<_{\mathcal{L}}$ größte Ereigniskombination ausgewählt wird, wenn die Wahl besteht zwischen:

- verschiedenen Kandidaten, deren Häufigkeiten zu bestimmen sind, oder
- verschiedenen Präfixen P, welche die nächste Äquivalenzklasse $E(P)$ bestimmen, zu der abgestiegen wird.

Vor dem Hintergrund der Darstellungen des Suchraums in den Abbildungen 3.1 und 3.3 wird diese Suchstrategie im folgenden als rechtsorientierten Tiefensuche bezeichnet. Für das Beispiel aus Abschnitt 3.1.2 ist die von der rechtsorientierten Tiefensuche für die Bearbeitung der Äquivalenzklassen implizierte Reihenfolge in Abbildung 4.1 veranschaulicht.

[1] Letztlich ist ausreichend, daß die Häufigkeiten der Ereigniskombinationen S mit $|S| = |K|-1$ bekannt sind.

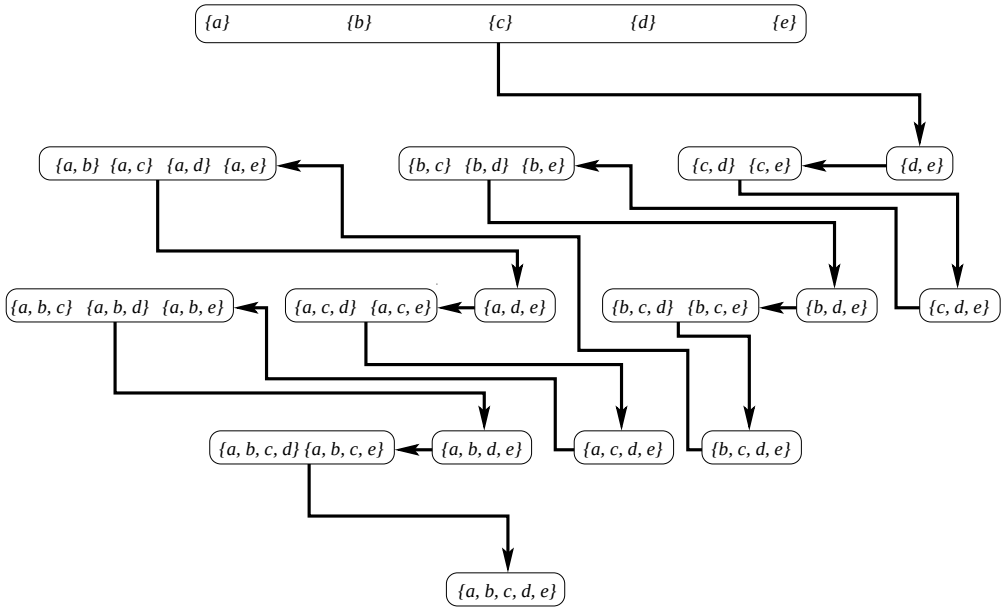


Abbildung 4.1: Rechtsorientierte Tiefensuche zur Ereignismenge $\mathcal{E} = \{a, b, c, d, e\}$ aus Abschnitt 3.1.2

Nachdem Voraussetzung (a) gegeben ist, bleibt Voraussetzung (b), der effiziente Zugriff auf die häufigen Ereigniskombinationen, geeignet umzusetzen. Dazu stellt Eclat als auf einer Tiefensuche und Schnittmengenbildung beruhendes Verfahren keine Datenstrukturen zur Verfügung.

Im folgenden wird vorgeschlagen, die häufigen Ereigniskombinationen anhand der eingeführten Relation $<_{\mathcal{L}}$ lexikographisch sortiert und für jede Mächtigkeit separat in einem Array abzulegen. Aufgrund der verwendeten rechtsorientierten Tiefensuche werden die häufigen Ereigniskombinationen absteigend sortiert erzeugt und können somit direkt nacheinander in das jeweilige Array geschrieben werden. Der Speicherplatz wird auf diese Weise effizient genutzt, und die Sortierung der häufigen Ereigniskombinationen erlaubt einen schnellen Zugriff auf die Häufigkeiten mittels einer Binärsuche.

Der aus Eclat anhand der beschriebenen Optimierung abgeleitete Algorithmus zur Generierung häufiger Ereigniskombinationen wird mit *Optra*, abgeleitet von „**op**timized lattice **tra**versal“, bezeichnet.

Erste Evaluierung

Der Algorithmus Optra wird hier zusammen mit Eclat exemplarisch auf dem Datensatz T10I4D1000K evaluiert. In Abbildung 4.2 ist zu verschiedenen Mindestvor-

gaben für die Häufigkeit in Prozent die Anzahl der Kandidaten, deren Häufigkeiten von Eclat und Optra bestimmt werden, zueinander in Bezug gesetzt. Mit sinkender

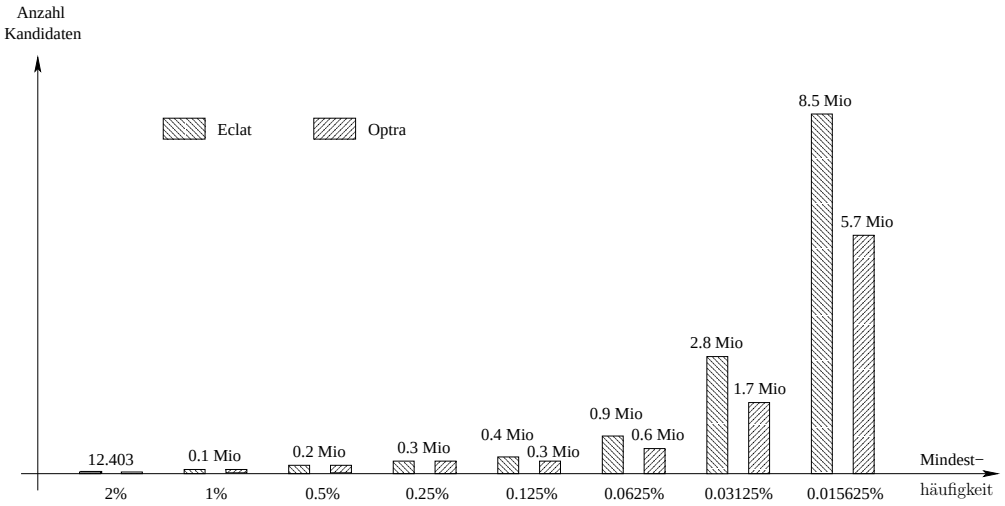
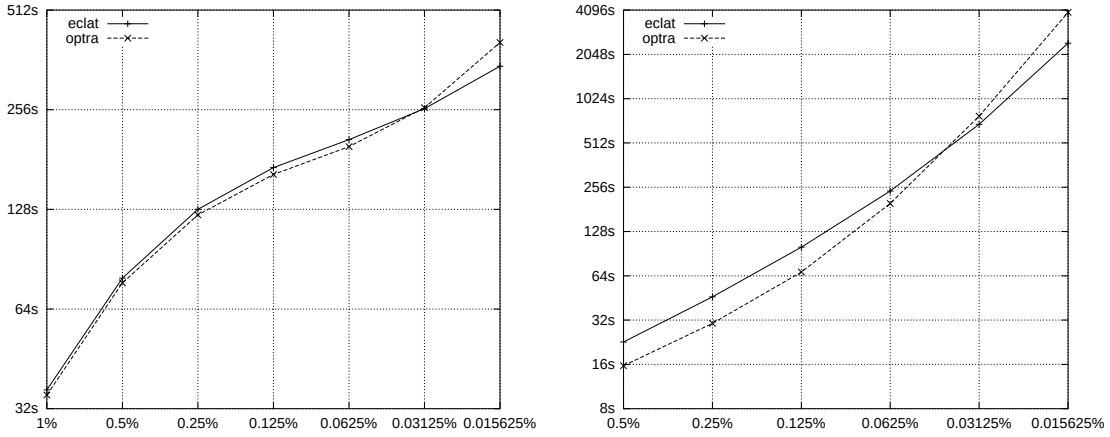


Abbildung 4.2: Anzahl von Kandidaten, deren Häufigkeiten Eclat und Optra bei variierender Mindesthäufigkeit auf dem Datensatz T10I4D1000K bestimmen (Anzahl zumeist gerundet)

Mindesthäufigkeit steigt die absolute Anzahl von Kandidaten für beide Verfahren, wobei dieser Anstieg für Optra im beschriebenen Experiment jedoch wesentlich geringer als für Eclat ausfällt. Schließlich kann Optra bei einer Mindesthäufigkeit von minsupp = 0,015625% mit 5.673.647 Kandidaten gegenüber Eclat mit 8.481.455 das Auszählen von fast einem Drittel der Kandidaten einsparen.

Bezüglich des Speicherbedarfs ist kein Vorteil von Optra gegenüber Eclat zu erwarten. Das optimierte Beschneiden des Suchraums impliziert nicht, daß Optra weniger Äquivalenzklassen als Eclat besucht, sondern lediglich, daß die Häufigkeiten einzelner Kandidaten nicht bestimmt werden müssen, um diese als nichthäufig zu erkennen.

In Abbildung 4.3 sind die Laufzeiten von Optra und Eclat auf den Datensätzen T10I4D1000K und BMSPOS dargestellt. Offensichtlich kann Optra den Vorteil durch das Beschneiden des Suchraums kaum direkt in kürzere Laufzeiten umsetzen. Für besonders geringe Mindesthäufigkeiten liegen die Zeiten von Optra sogar über denen von Eclat. Lediglich in einigen Experimenten auf dem Datensatz BMSPOS erreicht Optra um mehr als 30% kürzere Laufzeiten als Eclat. Daß die Zeiten von Eclat und Optra sehr ähnlich sind, ist in Anbetracht des hohen Anteils von Kandidaten, die Optra gegenüber Eclat aufgrund der rechtsorientierten Tiefensuche nicht auszählt, sondern direkt als nichthäufig verwirft, zunächst erstaunlich. Die Erklärung für dieses Verhalten ist in Abbildung 4.4 zu finden, wo auf dem Daten-



(a) Datensatz T10I4D1000K

(b) Datensatz BMSPOS

Abbildung 4.3: Laufzeiten in Sekunden von Optra und Eclat bei variierendem minsupp

satz T10I4D1000K bei ansonsten gleichen Parametereinstellungen die maximale Mächtigkeit der zu erzeugenden häufigen Ereigniskombinationen von 1 bis 10 variiert wird. Die angegebenen Werte entsprechen jeweils den summierten Laufzeiten für die Erzeugung der häufigen Ereigniskombinationen der Mächtigkeit 1 bis zur der jeweiligen maximalen Mächtigkeit. Aus den gemessenen Laufzeiten läßt sich ablesen, daß der Aufwand für die Erzeugung der häufigen Ereigniskombinationen der Mächtigkeit 2 den Gesamtaufwand dominiert. Der zusätzlich für die Generierung der häufigen Ereigniskombinationen der Mächtigkeit 3 erforderliche Aufwand ist wesentlich geringer, während für die Mächtigkeiten größer 3 fast kein weiterer Aufwand entsteht (zur Erklärung siehe auch den folgenden Abschnitt 4.1.2).

Für die Kandidaten der Mächtigkeit 2 kann Optra keine Häufigkeitsbestimmungen einsparen, da diese jeweils aus zwei einelementigen Teilmengen bestehen. Diese Teilmengen müssen immer häufig sein, da jeder Kandidat genau aus diesen beiden Teilmengen erzeugt wurde. Umgekehrt wirken sich eingesparte Häufigkeitsbestimmungen für die Kandidaten größerer Mächtigkeit kaum auf die Laufzeiten aus, da der Anteil dieser Kandidaten am Gesamtaufwand gering ist. In Extremfällen kann sich der Effekt sogar umkehren, wenn die zu schneidenden Transaktionsmengen sehr kleine Mächtigkeiten aufweisen und gleichzeitig die zugehörigen Kandidaten viele Ereignisse enthalten. In diesen Fällen kann der Aufwand für die Überprüfung der Teilmengen auf Häufigkeit den alternativen Aufwand für das Schneiden der Transaktionsmengen übersteigen. Auf dem Datensatz T10I4D1000K bleiben die Kandidaten der Mächtigkeit 3, 4 und 5, für welche die vorgestellte optimierte Beschneidung des Suchraums sich merklich auf die Laufzeit auswirken kann. Da

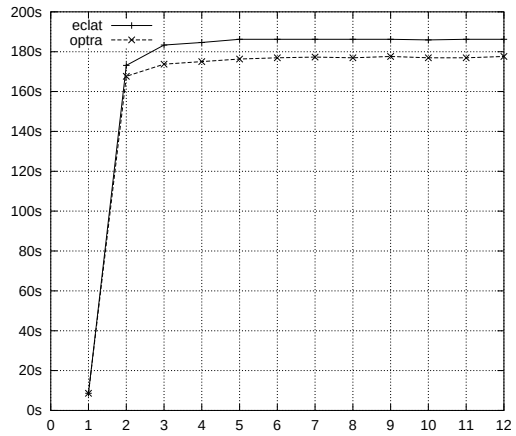


Abbildung 4.4: Laufzeiten von Optra und Eclat auf T10I4D1000K bei variierender maximaler Mächtigkeit der zu erzeugenden häufigen Ereigniskombinationen

diese Kandidaten die Gesamtlaufzeit nicht dominieren, ist die letztlich erreichte Verkürzung der Laufzeit durch die vorgestellte Optimierung gering bezüglich der Gesamtlaufzeit.

Insgesamt bleibt festzuhalten, daß die eingeführte optimierte Beschneidung des Suchraums zwar einen großen Teil der Schnitte über Transaktionsmengen einsparen kann, es sich jedoch vor allem um die weniger aufwendigen Schnitte über die Transaktionsmengen der Kandidaten höherer Mächtigkeiten handelt, die für die Gesamtlaufzeiten von untergeordneter Bedeutung sind. Eine umfassende Evaluierung von Optra zusammen mit einer Auswahl der etablierten Verfahren wird in Abschnitt 4.1.3 durchgeführt. Erst bei der Generierung taxonomer Assoziationsregeln kann die beschriebene Beschneidung des Suchraums in Laufzeitvorteile umgesetzt werden, wie in Abschnitt 4.2 gezeigt wird.

4.1.2 Hybrides Verfahren

Im folgenden wird ein hybrides Verfahren eingeführt, das auf etablierten Ansätzen beruht. Mit diesem Verfahren wird das Ziel verfolgt, durch geeignete Kombination zählender und schneidender Häufigkeitsbestimmung, die Laufzeiten gegenüber den etablierten Ansätzen zu verkürzen.

Motivation

In den Evaluierungen aus Abschnitt 3.3 ist zu beobachten, daß Apriori gegenüber Eclat und Partition auf den Datensätzen kürzere Laufzeiten erreicht, die vor allem

häufige Ereigniskombinationen kleinerer Mächtigkeit enthalten, wie beispielsweise T5I2D1000K, während auf den anspruchsvolleren Datensätzen, wie beispielsweise T20I6D1000K, Eclat und Partition die bezüglich der Laufzeit effizienteren Verfahren sind.

Apriori zählt die Häufigkeiten unter Verwendung eines Hashbaums (vgl. dazu Abschnitt 3.1). In diesem Hashbaum wird beim Zählen lediglich für jedes tatsächliche Vorkommen eines Kandidaten abgestiegen und der zugehörige Zähler erhöht. Das Zählen eines Kandidaten, der selten in den Transaktionen vorkommt, erzeugt somit einen vernachlässigbaren Aufwand. Die Laufzeiten von Apriori profitieren demzufolge von Ereigniskombinationen, die als Kandidaten erzeugt werden, jedoch trotzdem selten in den Daten vorkommen. Der Anteil häufiger Kandidaten an der Menge der erzeugten Kandidaten ist insbesondere für die von den Assoziationsregelalgorithmen erzeugten Kandidatenmengen, die aus Ereigniskombinationen der Mächtigkeit 2 und 3 bestehen, vergleichsweise gering, wie exemplarisch in Tabelle 4.1 für den Datensatz T10I4D1000K dargestellt.

	$n=1$	$n=2$	$n=3$	$n=4$	$n=5$	$n=6$
minsupp = 0,5%	28,1%	0,3%	67%	88,9%	87,5%	92,8%
minsupp = 0,125%	39,9%	2,2%	28,7%	94,1%	96,3%	92,8%
minsupp = 0,03125%	43%	9,1%	1,1%	98,3%	99%	96,7%

Tabelle 4.1: Anteil häufiger Kandidaten in den Kandidatenmengen zu den Mächtigkeiten n und verschiedenen Mindesthäufigkeiten minsupp auf dem Datensatz T10I4D1000K (bei Beschneidung des Suchraums anhand nichthäufiger Teilmengen)

Allerdings wird das Zählen der Häufigkeiten unter Verwendung eines Hashbaums zunehmend problematisch mit sinkenden Schwellenwerten für die minimale Häufigkeit. Sinkende Schwellenwerte für die minimale Häufigkeit bedeuten in der Regel, daß wesentlich mehr Ereigniskombinationen häufig sind und gleichzeitig die durchschnittliche Mächtigkeit der häufigen Ereigniskombinationen zunimmt. Entsprechend wachsen die während des Algorithmenlaufs erzeugten Hashbäume sowohl in die Breite als auch in die Höhe, so daß der Aufwand für das Absteigen im Baum anhand der Teilmengen der Transaktionen sehr aufwendig werden kann.

Eclat und Partition bestimmen die Häufigkeiten indirekt mittels Schnittmengenbildung (vgl. Abschnitt 3.1). Im Gegensatz zu dem Apriori zugrundeliegenden Ansatz entsteht bei diesem Vorgehen auch für Kandidaten, die selten in den Transaktionen vorkommen, ein Mindestaufwand, da die Häufigkeitsbestimmung durch Schnittmengenbildung erfordert, daß zumindest die weniger mächtige der beiden Transaktionsmengen durchlaufen wird. Dies gilt auch, wenn ein Kandidat nicht in den Transaktionen vorkommt. Selbst der Aufwand für die Schnellschnitte, auf denen Eclat basiert (vgl. Abschnitt 3.1), ist unabhängig von der Häufig-

keit der Kandidaten. Umgekehrt hat die Mächtigkeit der Kandidaten für Eclat und Partition als schneidende Verfahren auf die Laufzeiten keinen Einfluß, da bei der indirekten Häufigkeitsbestimmung durch Schnittmengenbildung lediglich deren zugehörige Transaktionsmengen durchlaufen werden, anstatt die Kandidaten in einem Hashbaum zu suchen.

Folglich ist insgesamt das Zählen von Häufigkeiten unter Verwendung eines Hashbaums für Kandidaten kleinerer Mächtigkeit effizienter, während für Kandidaten größerer Mächtigkeit die Schnittmengenbildung kürzere Laufzeiten ermöglicht. Es stellt sich daher die Frage nach einem hybriden Verfahren, welches die Vorteile beider Ansätze verbinden kann.

Umsetzung

Das hier vorgestellte hybride Verfahren basiert zunächst auf der Häufigkeitsbestimmung durch Zählen, der eine Breitensuche zugrunde liegt. Sobald eine festgelegte Mächtigkeit s erreicht ist, findet ein Wechsel zu einer mit Tiefensuche kombinierten Häufigkeitsbestimmung durch Schnittmengenbildung statt. Die Häufigkeiten der Kandidaten kleinerer Mächtigkeit werden also durch Zählen anhand eines Hashbaums ermittelt, während die Häufigkeiten der Kandidaten größerer Mächtigkeit indirekt durch Schnittmengenbildung bestimmt werden. Der Wechsel von Breitensuche zu Tiefensuche ist notwendig, da weder für die Tiefensuche mit Häufigkeitsbestimmung durch Zählen noch für die Breitensuche mit Häufigkeitsbestimmung durch Schnittmengenbildung effiziente Ansätze bekannt sind (vgl. auch die Systematisierung in Abbildung 3.15 und die Evaluierung in Abschnitt 3.3).[2]

Der Übergang von zählender Häufigkeitsbestimmung zu Schnittmengenbildung ist problematisch, da diesen Ansätzen jeweils unterschiedliche Aufbereitungen der Daten zugrunde liegen (vgl. Abschnitt 3.1). Das hier vorgeschlagene Verfahren bestimmt die Häufigkeiten der Kandidaten bis einschließlich der Mächtigkeit s durch Zählen der tatsächlichen Vorkommen anhand von Hashbäumen. Sind die häufigen Ereigniskombinationen der Mächtigkeit s bekannt, dann werden zu diesen die

[2]Der Algorithmus FP-Growth kombiniert eine Tiefensuche mit dem Zählen von Häufigkeiten, basiert jedoch auf zu einem FP-Baum aufbereiteten Transaktionen. Neben der damit einhergehenden Hauptspeicherproblematik (vgl. Abschnitt 3.3) können aus dieser aufbereiteten Form die ursprünglichen Transaktionen nicht mehr ohne weiteres abgeleitet werden (vgl. Abschnitt 3.1). Es ist außerdem nicht offensichtlich, ob FP-Growth derart erweitert werden kann, daß dieser Algorithmus häufige Ereigniskombinationen nur bis zu einer maximalen Mächtigkeit effizient erzeugt.

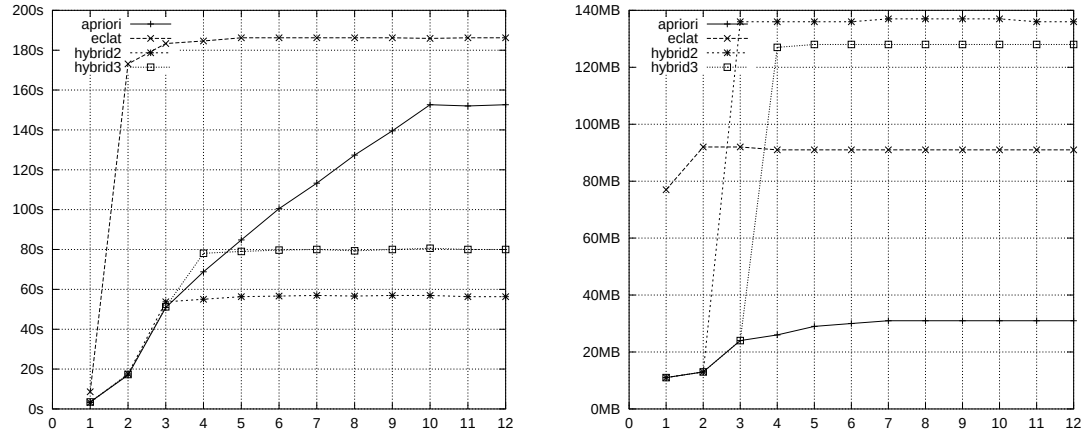
Partition basiert auf einer Tiefensuche und der Häufigkeitsbestimmung durch Schneiden von Transaktionsmengen. Partition muß aber die Transaktionen wegen des ansonsten problematischen Hauptspeicherbedarfs aufteilen und ist daher weniger effizient als der auf Tiefensuche basierende Algorithmus Eclat.

Transaktionsmengen erzeugt. Dazu wird wie zur Häufigkeitsbestimmung ein Hashbaum verwendet, wobei, anstatt Zähler zu speichern und diese hochzuzählen, der Hashbaum die Transaktionsmengen speichert und für jede in einer Transaktion gefundene häufige Ereigniskombination die Transaktionsnummer der gerade aktuellen Transaktion angefügt wird. Die Transaktionsmengen direkt zu sämtlichen Kandidaten der Mächtigkeit s zu erzeugen, daß heißt ohne zuvor die häufigen der Kandidaten separat zu bestimmen, ist in der Regel wegen des zu hohen Speicherbedarfs für die dadurch hinzukommenden Transaktionsmengen der nichthäufigen Kandidaten unmöglich. Basierend auf den häufigen Ereigniskombinationen der Mächtigkeit s und den zugehörigen Transaktionsmengen wird dann für jede Äquivalenzklasse die rekursive Tiefensuche auf Tiefe $s + 1$ gestartet.

Das neue Verfahren wird als Hybrid2 beziehungsweise Hybrid3 bezeichnet, wobei 2 und 3 die Konstante s angeben, welche die Mächtigkeit für den Wechsel der zugrundeliegenden Suchstrategie bestimmt.

Evaluierung

Im folgenden Experiment wird auf dem Datensatz T10I4D1000K die maximale Mächtigkeit der zu erzeugenden häufigen Ereigniskombinationen von 1 bis 12 bei ansonsten gleichen Parametereinstellungen variiert. Die in Abbildung 4.5 ange-



(a) Laufzeit in Sekunden

(b) Speicherbedarf in Megabyte

Abbildung 4.5: Effizienz ausgewählter Verfahren auf T10I4D1000K bei variierender maximaler Mächtigkeit der zu erzeugenden häufigen Ereigniskombinationen

gebenen Werte summieren folglich die Laufzeit und den Speicherbedarf für die Erzeugung der häufigen Ereigniskombinationen der Mächtigkeit 1 bis zur jeweili-

gen maximalen Mächtigkeit (vgl. auch Abbildung 4.4). Die größte Mächtigkeit, die von einer häufigen Ereigniskombination erreicht wird, ist 10.

Für die Algorithmen Apriori und Eclat ergibt sich das aus obigen Überlegungen erwartete Bild. Während der Übergang von der maximalen Mächtigkeit 1 zur maximalen Mächtigkeit 2 für das zählende Verfahren Apriori nur einen moderaten Anstieg der Laufzeit bedeutet, ist dieser Anstieg für das schneidende Verfahren Eclat am stärksten ausgeprägt. Zugleich ist die Zunahme der Laufzeit mit wachsender maximaler Mächtigkeit für Eclat wesentlich stärker abnehmend als für Apriori.

Neben Eclat und Apriori sind in Abbildung 4.5 auch die Ergebnisse für den hybriden Ansatz mit $s = 2$ und $s = 3$, Hybrid2 und Hybrid3, wiedergegeben. Die beiden hybriden Varianten zeigen ebenfalls das erwartete Verhalten.[3] Bis zum Wechsel der Methode zur Häufigkeitsbestimmung verhalten sich Hybrid2 und Hybrid3 weitgehend wie der Apriori-Algorithmus und steigen entsprechend nur mäßig beim Übergang von Mächtigkeit 1 zu Mächtigkeit 2, um später die ebenfalls moderate Steigung von Eclat zu zeigen. Die für die Generierung der Transaktionsmengen benötigte Zeit ist in den Übergängen der maximalen Mächtigkeiten von 2 nach 3 beziehungsweise 3 nach 4 zu erkennen.

Bezüglich des Hauptspeicherbedarfs fällt auf, daß die beiden hybriden Verfahren bis zum Wechsel der Such- und Zählstrategie zwar Apriori folgen, danach aber jeweils wesentlich mehr Speicher benötigen als Eclat. Der Grund dafür ist, daß die Transaktionsmengen sämtlicher häufiger Ereigniskombinationen der Mächtigkeit s gleichzeitig im Hauptspeicher gehalten werden. Würden die Transaktionsmengen nur für die jeweils aktuelle Äquivalenzklasse generiert, wäre für jede dieser Äquivalenzklassen ein separater Durchlauf über die Datenbank notwendig, wodurch das Verfahren insgesamt ineffizient werden würde (vgl. auch Abschnitt 3.1).

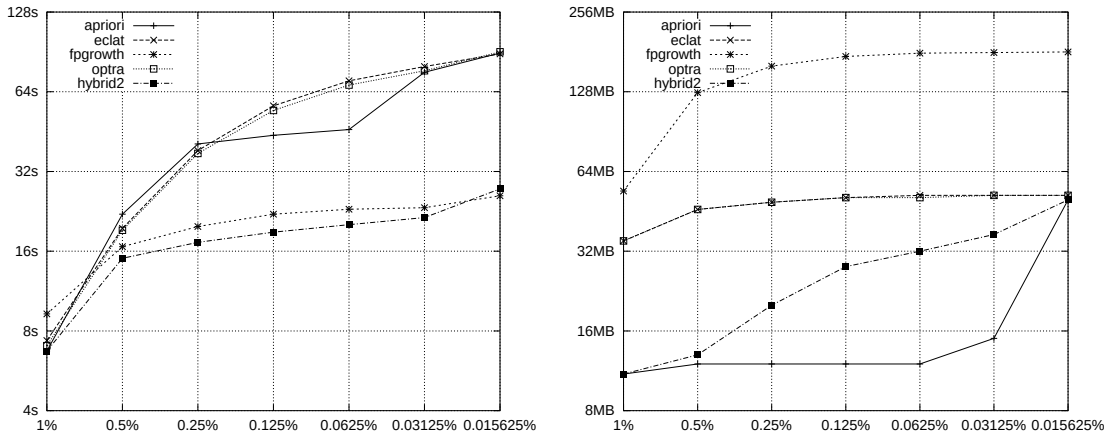
4.1.3 Vergleich mit den etablierten Verfahren

Das neu vorgestellte Verfahren Optra sowie die ebenfalls neuen hybriden Ansätze werden im folgenden entsprechend Abschnitt 3.3 auf verschiedenen synthetischen Datensätzen und Datensätzen aus realen Anwendung evaluiert. Außerdem werden die Algorithmen Apriori, Eclat und FP-Growth als die effizientesten der etablierten Verfahren in die Evaluierung aufgenommen (vgl. dazu ebenfalls Abschnitt 3.3).

[3] Experimente mit der eingeführten optimierten Tiefensuche ergeben für das hybride Vorgehen keine Verbesserung und werden im folgenden nicht weiterverfolgt.

Synthetische Daten

In den Abbildungen 4.6 bis 4.9 sind die Ergebnisse von Experimenten auf den in Abschnitt 3.3 eingeführten Datensätzen wiedergegeben.



(a) Laufzeit in Sekunden

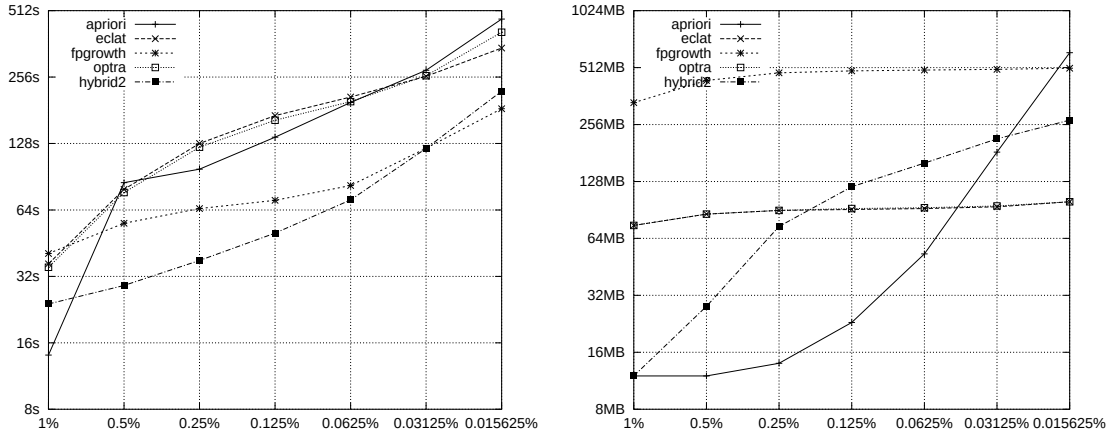
(b) Speicherbedarf in Megabyte

Abbildung 4.6: Effizienz ausgewählter Verfahren auf T5I2D1000K bei variierendem minsupp

Wie in Abschnitt 4.1.1 zeigen Eclat und Optra sowohl bezüglich der Laufzeiten als auch des Speicherbedarfs ein sehr ähnliches Verhalten.

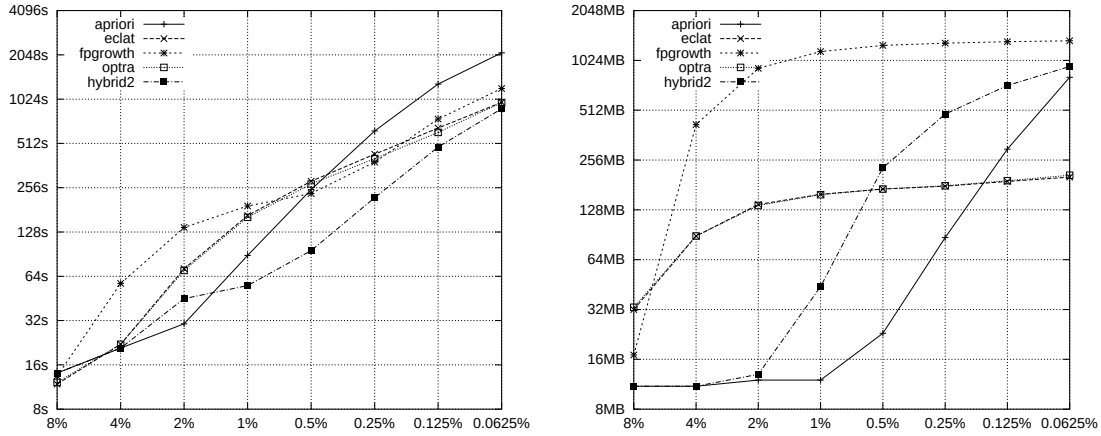
Der neue hybride Ansatz ist hinsichtlich der Laufzeiten in fast allen Experimenten den anderen Verfahren überlegen. Selbst auf dem Datensatz T5I2D1000K, auf dem FP-Growth das bisher bei weitem effizienteste Verfahren war, erreicht Hybrid zumeist kürzere Laufzeiten als FP-Growth. Lediglich bei besonders kleinen Mindestvorgaben für die Häufigkeit bleibt FP-Growth auf den Datensätzen T5I2D1000K und T10I4D1000K in den Experimenten das effizienteste Verfahren. Im einzelnen erreicht Hybrid2 auf den Datensätzen T5I2D1000K, T10I4D1000K und T20I6D1000K um bis zu 15%, 50% beziehungsweise fast 60% kürzere Laufzeiten als das bisher effizienteste Verfahren. Auch auf den speziell für FP-Growth erzeugten Daten T25I20D100K ist Hybrid2 dem Algorithmus FP-Growth zumeist überlegen oder erreicht vergleichbare Laufzeiten. Für weitere Experimente sei auf Anhang A.2 verwiesen.

Auf dem weniger anspruchsvollen Datensatz T5I2D1000K und bei moderaten Vorgaben für die Mindesthäufigkeit auf den anderen Datensätzen liegt in den durchgeführten Experimenten der Speicherbedarf von Hybrid zwischen dem von Apriori und dem von Eclat. Hybrid benötigt lediglich für geringe Mindesthäufigkeiten auf den anspruchsvolleren Datensätzen T10I4D1000K und T20I6D1000K mehr Hauptspeicher als Apriori und Eclat, jedoch noch wesentlich weniger als FP-Growth.



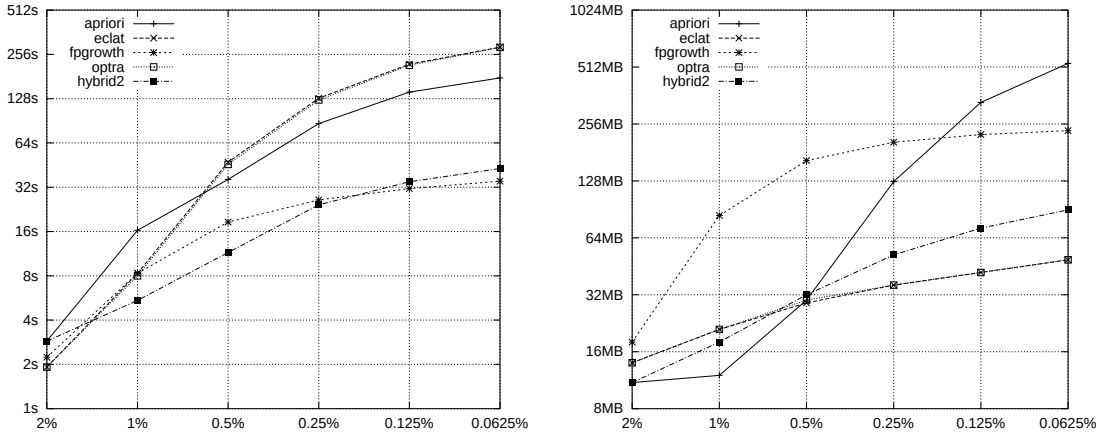
(a) Laufzeit in Sekunden (b) Speicherbedarf in Megabyte

Abbildung 4.7: Effizienz ausgewählter Verfahren auf T10I4D1000K bei variierendem minsupp



(a) Laufzeit in Sekunden (b) Speicherbedarf in Megabyte

Abbildung 4.8: Effizienz ausgewählter Verfahren auf T20I6D1000K bei variierendem minsupp



(a) Laufzeit in Sekunden

(b) Speicherbedarf in Megabyte

Abbildung 4.9: Effizienz ausgewählter Verfahren auf T25I20D100K bei variierendem minsupp

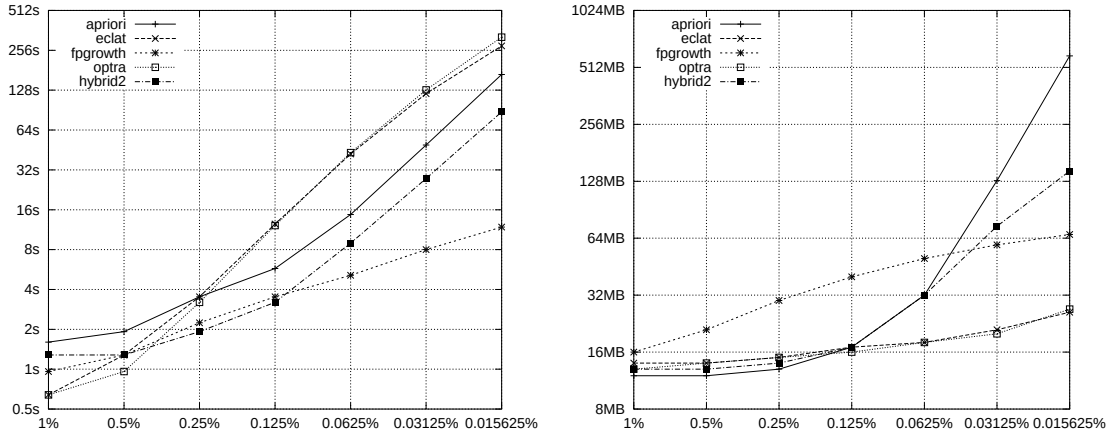
Interpretation

Wie in Abschnitt 4.1.2 exemplarisch für den Datensatz T10I4D1000K bei fester Vorgabe einer minimalen Häufigkeit und variierender maximaler Mächtigkeit gezeigt, kann das eingeführte hybride Verfahren die Vorteile von Apriori und Eclat bezüglich der erreichten Laufzeiten miteinander verbinden. Entsprechend effizient ist der resultierende Algorithmus, der nicht nur kürzere Laufzeiten als die ihm zugrundeliegenden Ansätze erreicht, sondern hinsichtlich der Laufzeit zumeist auch effizienter als das bisher effizienteste Verfahren FP-Growth ist.

Hinsichtlich des Speicherbedarfs stellt sich die Frage, ob die von Hybrid gemeinsam im Hauptspeicher zu haltenden Transaktionsmengen gegenüber Apriori und Eclat von Nachteil sind. Zwar ist das hybride Verfahren weit vom Speicherbedarf des Algorithmus FP-Growth entfernt, „erbt“ jedoch die jeweils negativen Aspekte des Speicherverhaltens von Apriori und Eclat. So steigt der Speicherbedarf von Hybrid für kleine Vorgaben für die minimale Häufigkeit stark an, da ähnlich wie bei Apriori sehr viele Zähler für Kandidaten zum selben Zeitpunkt im Hauptspeicher gehalten werden müssen. Zugleich skaliert der Hauptspeicherbedarf von Hybrid wie der von Eclat linear mit der Anzahl zugrundeliegender Transaktionen, das heißt ist nicht wie bei Apriori unabhängig von dieser Größe.

Abschließender Vergleich auf Daten realer Anwendungen

Im folgenden werden die neu eingeführten Ansätze zusammen mit den ausgewählten etablierten Verfahren auf Datensätzen aus realen Anwendungen evaluiert. Die Ergebnisse der Evaluierung sind in den Abbildungen 4.10 bis 4.14 dargestellt.



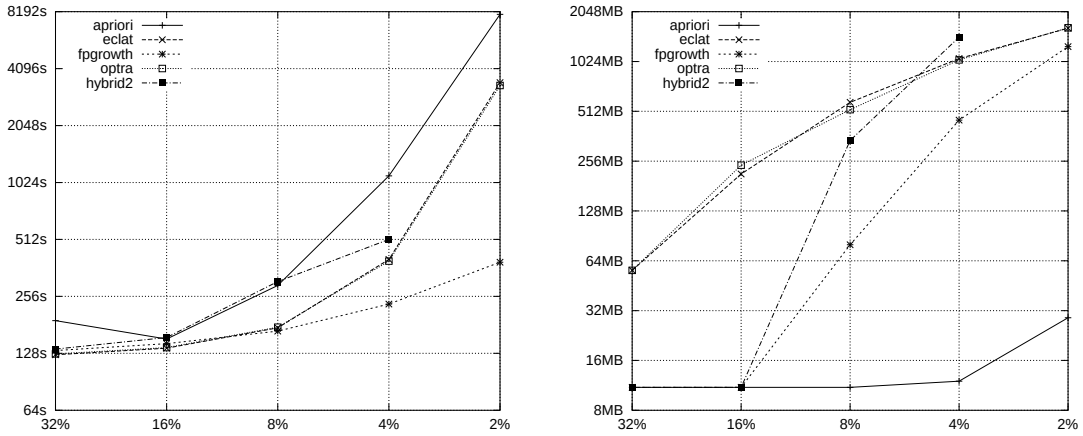
(a) Laufzeit in Sekunden

(b) Speicherbedarf in Megabyte

Abbildung 4.10: Effizienz ausgewählter Verfahren auf EINZELHANDEL bei variierendem minsupp

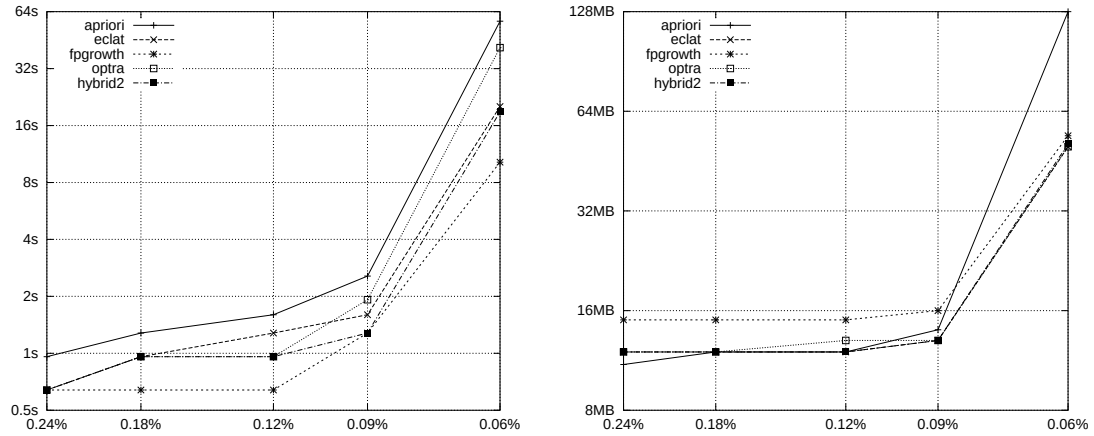
Insgesamt bestätigen die Messungen die auf den synthetischen Daten gewonnenen Erkenntnisse. Die Laufzeiten und der Speicherbedarf der Algorithmen Eclat und Optra liegen weiterhin sehr nahe beieinander. Hybrid erreicht auf den Datensätzen EINZELHANDEL, BMS-WEBVIEW1 und BMS-WEBVIEW2 bis zu annähernd 50%, 20% beziehungsweise ebenfalls 50% kürzere Laufzeiten als der jeweils effizienteste der beiden zugrundeliegenden Algorithmen Apriori und Eclat. Hybrid zeigt außerdem bezüglich des Hauptspeicherbedarfs die negativen Eigenschaften dieser beiden Algorithmen. Im Vergleich mit FP-Growth verschieben sich allerdings auf den Daten aus realen Anwendungen die von Hybrid erzielten Ergebnisse. FP-Growth erreicht in fast allen Experimenten die kürzeren Laufzeiten, wobei außerdem Hybrid unter bestimmten Bedingungen für die gleiche Aufgabe mehr Hauptspeicher als FP-Growth benötigt.

Letztlich ergibt sich ein deutlicher Vorteil für Hybrid auf den synthetisch erzeugten Datensätzen, jedoch nicht auf den Datensätzen aus realen Anwendungen. Optra hingegen kann die verbesserte Kandidatenbestimmung nur in wenigen Experimenten auf den synthetischen Daten und den Datensätzen aus realen Anwendungen in kürzere Laufzeiten umsetzen.



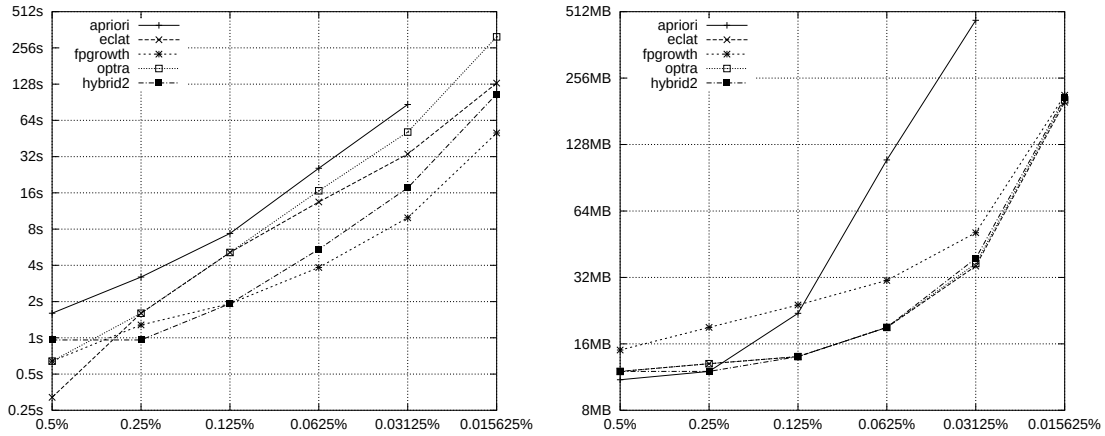
(a) Laufzeit in Sekunden (b) Speicherbedarf in Megabyte

Abbildung 4.11: Effizienz ausgewählter Verfahren auf FAHRZEUG bei variierendem minsupp



(a) Laufzeit in Sekunden (b) Speicherbedarf in Megabyte

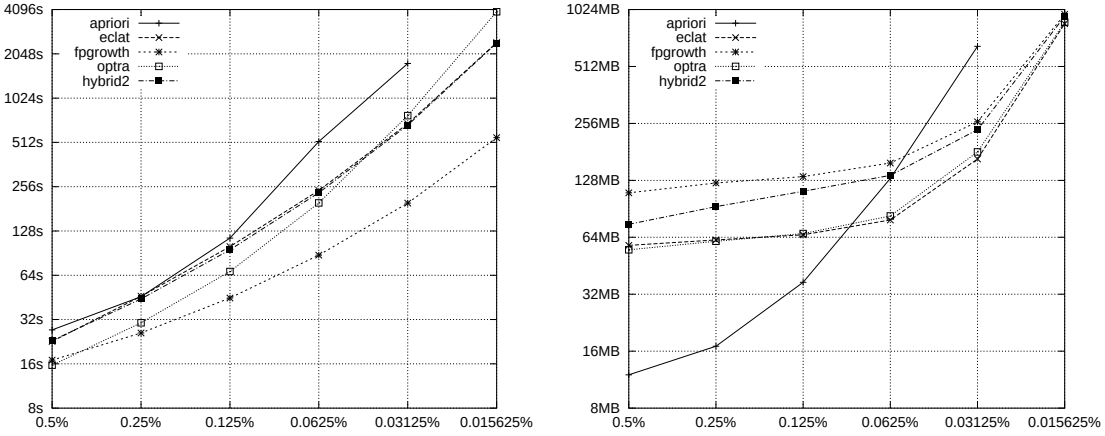
Abbildung 4.12: Effizienz ausgewählter Verfahren auf BMS-WEBVIEW1 bei variierendem minsupp



(a) Laufzeit in Sekunden

(b) Speicherbedarf in Megabyte

Abbildung 4.13: Effizienz ausgewählter Verfahren auf BMS-WEBVIEW2 bei variierendem minsupp



(a) Laufzeit in Sekunden

(b) Speicherbedarf in Megabyte

Abbildung 4.14: Effizienz ausgewählter Verfahren auf BMSPOS bei variierendem minsupp

4.2 Taxonome Verfahren

In Abschnitt 2.2.3 wurden taxonome Assoziationsregeln als eine Erweiterung der einfachen Assoziationsregeln vorgestellt. In diesem Abschnitt folgt nun die Einführung von Verfahren zur Generierung dieser Regeln. Dazu werden zunächst die etablierten Verfahren so erweitert, daß diese taxonome Assoziationsregeln erzeugen können. Anschließend werden neue Ansätze eingeführt, die auf den etablierten Verfahren beruhen und durch Einbeziehung der Taxonomie kürzere Laufzeiten als die erweiterten etablierten Verfahren erreichen können. Der Abschnitt schließt mit einer Evaluierung der im Rahmen dieser Arbeit erweiterten etablierten Ansätze und der hier neu vorgestellten Verfahren auf verschiedenen synthetischen Datensätzen sowie einem Datensatz aus einer realen Anwendung.

4.2.1 Erweiterung der etablierten Verfahren

Um mit den Ansätzen zur Generierung einfacher Assoziationsregeln auch taxonome Assoziationsregeln erzeugen zu können, müssen die Verfahren die auf den Ereignissen definierte Taxonomie mit in die Regelgenerierung einbeziehen. Zu diesem Zweck ist es nicht erforderlich, die im Taxonomiegraphen enthaltene Information vollständig zu nutzen, sondern es genügt, wenn zu jedem Ereignis sämtliche Vorfahren bekannt sind. Damit ist es möglich, jede Transaktion mit all den Ereignissen zu erweitern, die Vorfahr mindestens eines der bereits in der Transaktion enthaltenen Ereignisse sind (vgl. auch Abschnitt 3.2.2). Die Transaktionen sind weiterhin Mengen von Ereignissen, das heißt, jede Transaktion kann ein Ereignis nur einmal enthalten. Anhand derart aufbereiteter Transaktionen werden dann mit einem der in den Abschnitten 3.1 und 4.1 beschriebenen Verfahren häufige Ereigniskombinationen erzeugt. Die resultierenden häufigen Ereigniskombinationen enthalten nicht nur die Ereignisse aus den Blättern des Taxonomiebaums, sondern ebenfalls die Ereignisse aus den höheren Ebenen. Durch die Anreicherung trägt jede Transaktion ohne weitere Anpassung der Verfahren auch zur Häufigkeit der Ereigniskombinationen bei, die Vorfahren der Ereignisse aus der ursprünglichen Transaktion enthalten. Die aus diesen häufigen Ereigniskombinationen mit dem in Abschnitt 3.1.1 beschriebenen Verfahren abgeleiteten Assoziationsregeln sind taxonome Assoziationsregeln, wie in [Srikant und Agrawal, 1995], [Srikant, 1996] gezeigt wird.

Im folgenden werden die Taxonomien jeweils als Menge sortierter Arrays von Vorfahren zu den Ereignissen vorausgesetzt. Die Aufbereitung der einzelnen Transaktionen wird implementiert, indem für jedes in einer Transaktion vorkommende Ereignis dessen sämtliche Vorfahren zur aktuellen Transaktion hinzugefügt werden. Die resultierende Transaktion wird sortiert und eventuell enthaltene Dubletten werden entfernt.

Von den etablierten Verfahren werden auf diese Weise Apriori, Eclat und FP-Growth für die Evaluierungen erweitert, so daß diese Verfahren auch taxonome Assoziationsregeln erzeugen können. Apriori reichert dazu in jedem Durchlauf über die Datenbank die Transaktionen erneut mit den Vorfahren der Ereignisse an. Der zusätzlich Aufwand für das wiederholte Aufbereiten der Transaktionen ist im Zusammenhang mit dem Einlesen der Transaktionen von externem Speicher sowie dem Generieren und Auszählen der Kandidaten vernachlässigbar. Da bei diesem Vorgehen keine Transaktionen im Hauptspeicher gehalten werden müssen, bleibt das im Vergleich mit den anderen untersuchten Verfahren optimale Speicherverhalten von Apriori (vgl. Abschnitt 3.3.2) erhalten. Bei dem in Abschnitt 3.2.2 vorgestellten Algorithmus Basic handelt es sich um eine ähnliche Variante von Apriori. Allerdings berechnet Basic die Vorfahren der Ereignisse aus der als Graph vorausgesetzten Taxonomie zur Laufzeit und ist damit weniger effizient als der hier eingeführte erweiterte Apriori-Algorithmus.

Im Gegensatz zu Apriori bestimmen die Algorithmen Eclat und FP-Growth die Häufigkeiten der Ereigniskombinationen nicht direkt auf den Transaktionen, sondern bereiten die Transaktionen im Hauptspeicher als Transaktionsmengen oder FP-Bäume auf. Sobald diese Aufbereitung abgeschlossen ist, wird auf die mittels der Taxonomie erweiterten Transaktionen nicht mehr zugegriffen und diese können verworfen werden. Ein wiederholtes Aufbereiten der Transaktionen im Verlauf der Generierung der häufigen Ereigniskombinationen, wie bei Apriori, ist nicht notwendig.

Die Erweiterungen der Algorithmen Eclat und FP-Growth für die Generierung taxonomer Assoziationsregeln werden im Rahmen der vorliegenden Arbeit erstmals eingeführt. Entsprechend ist eine Evaluierung dieser erweiterten etablierten Verfahren bisher nicht in der Literatur zu finden.

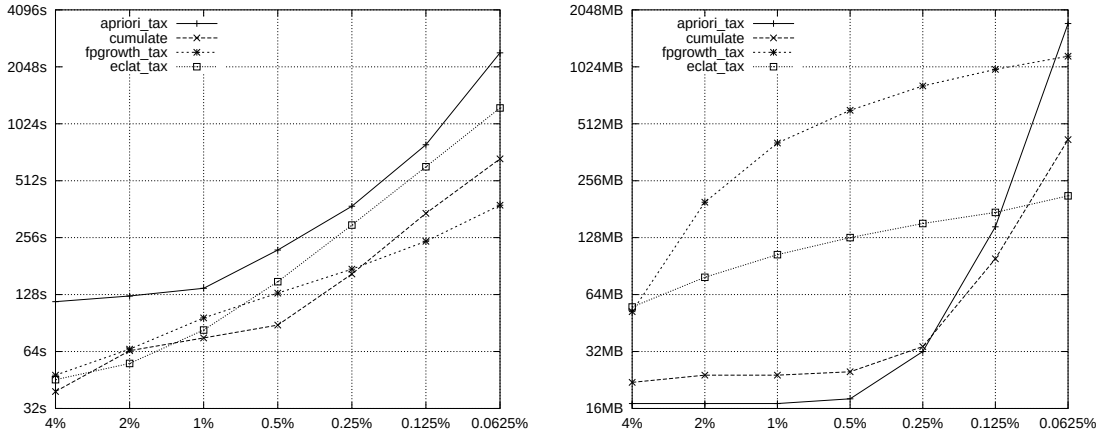
Erste Evaluierung

In die folgende Evaluierung wird neben den erweiterten etablierten Verfahren der speziell für die Generierung taxonomer Assoziationsregeln entwickelte Algorithmus Cumulate aus [Srikant und Agrawal, 1995] (vgl. auch Abschnitt 3.2.2) einbezogen. Dieser Algorithmus steht stellvertretend für die ebenfalls in [Srikant und Agrawal, 1995] eingeführten Varianten Stratify, Estimate und EstMerge, die dort ähnliche Evaluierungsergebnisse wie Cumulate erreichen.

Die Erzeugung und Bezeichnung der synthetischen Datensätze erfolgt wie in Abschnitt 3.3.1 beschrieben. Neben den Transaktionen wird für die Experimente außerdem eine Taxonomie mit dem Datengenerator erzeugt.[4] In den Abbildungen 4.15 bis 4.17 sind die von den Verfahren auf den in Abschnitt 3.3 eingeführten

[4]Die Generierung der Taxonomie erfolgt auf Basis der Standardvorgaben des eingesetzten Datengenerators, die auch den Evaluierungen in [Srikant und Agrawal, 1995], [Hipp et al., 1998] zugrunde liegen. Zu den Einzelheiten der synthetischen Generierung der Taxonomien sei auf den folgenden Abschnitt 4.2.3 verwiesen.

Standarddatensätzen benötigten Laufzeiten und der erreichte Speicherbedarf dargestellt.



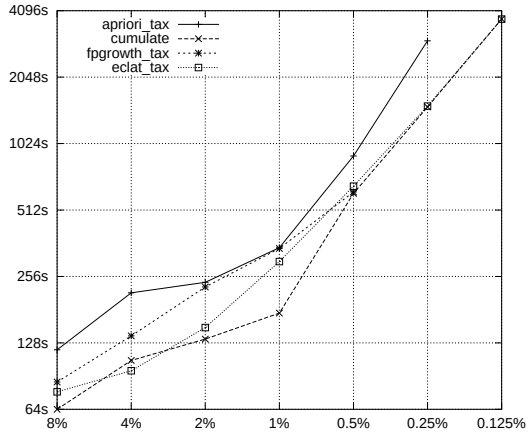
(a) Laufzeit in Sekunden

(b) Speicherbedarf in Megabyte

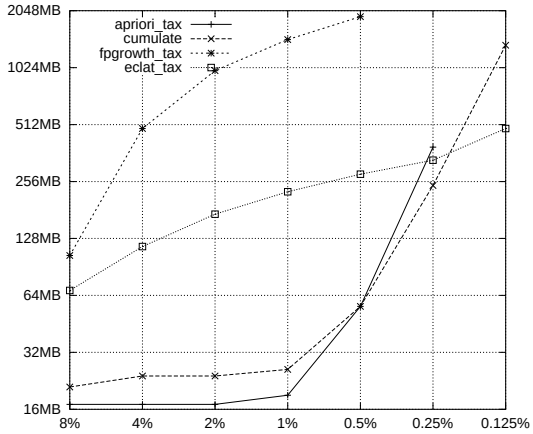
Abbildung 4.15: Effizienz erweiterter etablierter Verfahren auf T5I2D1000K bei variierendem minsupp

Während der für die Generierung taxonomer Assoziationsregeln erweiterte Algorithmus Apriori in sämtlichen Experimenten bezüglich der Laufzeit das ineffizienteste der Verfahren ist, kann für die anderen Verfahren keine eindeutige Aussage abgeleitet werden. So erreichen auf dem Datensatz T5I2D1000K die beiden zählenden Verfahren Cumulate und FP-Growth gegenüber Eclat um bis zu 50% beziehungsweise 70% kürzere Laufzeiten. Auf dem Datensatz T10I4D1000K sind sich die Laufzeiten der Verfahren sehr ähnlich, während auf zunehmend anspruchsvolleren Daten, beispielsweise auf T20I6D1000K, Eclat als schneidendes Verfahren um bis zu 55% effizienter als Cumulate ist und FP-Growth aufgrund des Hauptspeicherbedarfs nur sehr beschränkt eingesetzt werden kann (vgl. auch die ergänzenden Evaluierungen in Anhang A.3.1). Ein ähnlicher Effekt wurde in Abschnitt 3.3.2 bereits bei der Generierung einfacher Assoziationsregeln beobachtet.

Hinsichtlich des Hauptspeicherbedarfs ergibt sich kein wesentlicher Unterschied zu den Ergebnissen bei der Generierung einfacher Assoziationsregeln (vgl. Abschnitt 3.3.2). Apriori und der auf Apriori basierende Algorithmus Cumulate zeigen zunächst die mit Abstand günstigsten Werte, die aber mit zunehmend geringeren minimalen Häufigkeiten aufgrund der vielen zu generierenden Kandidaten stark ansteigen und schließlich über denen der anderen Verfahren liegen. Cumulate benötigt zumindest für geringere minimale Häufigkeiten etwas weniger Hauptspeicher als Apriori. Der Hauptspeicherbedarf von Eclat liegt zumeist zwischen dem

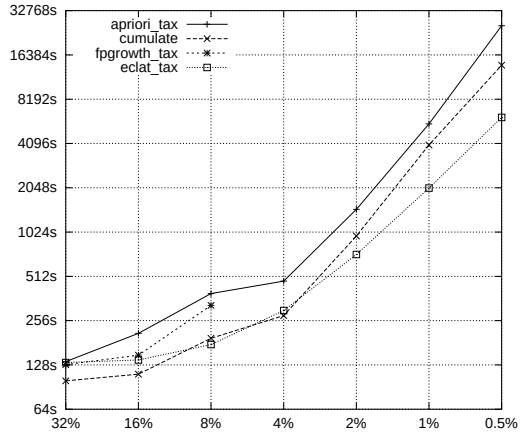


(a) Laufzeit in Sekunden

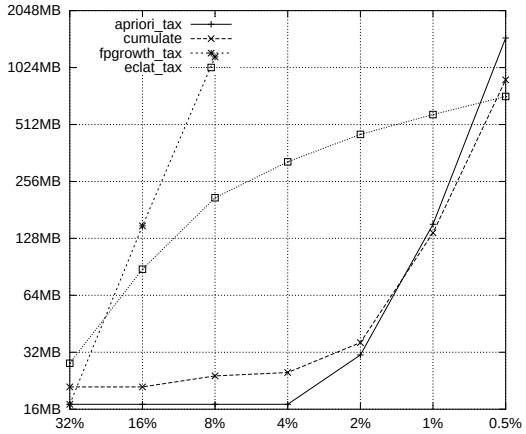


(b) Speicherbedarf in Megabyte

Abbildung 4.16: Effizienz erweiterter etablierter Verfahren auf T10I4D1000K bei variierendem minsupp



(a) Laufzeiten in Sekunden



(b) Speicherbedarf in Megabyte

Abbildung 4.17: Effizienz erweiterter etablierter Verfahren auf T20I6D1000K bei variierendem minsupp

von Apriori beziehungsweise Cumulate und dem von FP-Growth. FP-Growth zeigt, wie im Fall der Generierung einfacher Assoziationsregeln, den mit Abstand größten Hauptspeicherbedarf, so daß mit diesem Algorithmus auf dem zugrundeliegenden Rechnersystem für anspruchsvollere Datensätze nur eingeschränkt Experimente möglich waren (vgl. insbesondere Abbildung 4.17).

Somit sind Eclat und mit Einschränkungen Cumulate für die Generierung taxonomer Assoziationsregeln als besonders geeignet anzusehen. FP-Growth ist nicht nur wegen seines hohen Speicherbedarfs wenig geeignet, sondern kann außerdem, anders als im Fall der Generierung einfacher Assoziationsregeln (vgl. Abschnitt 3.3.2), bei der Generierung taxonomer Assoziationsregeln nicht durch kürzere Laufzeiten überzeugen.

4.2.2 Eigene Ansätze

Vor dem Hintergrund der bisher in dieser Arbeit vorgestellten Ergebnisse ist die Erweiterung von Eclat als der vielversprechendste Ansatz zur Generierung taxonomer Assoziationsregeln einzuschätzen. Im folgenden wird daher ein Verfahren vorgestellt, das auf einer Tiefensuche kombiniert mit der Häufigkeitsbestimmung durch Schnittmengenbildung beruht, wobei außerdem erstmals spezielle Optimierungen zur Generierung taxonomer Assoziationsregeln verwendet werden. Des weiteren wird ein auf dem in Abschnitt 4.1.2 neu eingeführten hybriden Verfahren basierender Algorithmus vorgestellt, der die Häufigkeiten der Kandidaten der Mächtigkeit 2 mittels Breitensuche und Zählen bestimmt und für die verbleibenden Kandidaten eine Tiefensuche kombiniert mit der Häufigkeitsbestimmung durch Schnittmengenbildung und ebenfalls den speziellen Optimierungen verwendet. Mit diesen beiden neuen Verfahren gelingt es erstmals, die Taxonomie effizient zur Beschneidung des Suchraums heranzuziehen.

Vorgehensweise

Der Algorithmus Cumulate optimiert die Bestimmung der Häufigkeiten unter anderem dadurch, daß von der Häufigkeitsbestimmung die Kandidaten ausgeschlossen werden, die Ereignisse und zugleich einen oder mehrere Vorfahren der enthaltenen Ereignisse enthalten. Solche Kandidaten sind als redundant zu betrachten, da ihre Häufigkeit mit der Häufigkeit der Ereigniskombination übereinstimmt, welche aus dem Kandidaten abgeleitet werden kann, indem die Vorfahren aller enthaltenen Ereignisse entfernt werden (vgl. auch Abschnitt 3.2.2).

Cumulate überprüft nicht für jeden der Kandidaten, ob dieser ein Ereignis sowie einen Vorfahr dieses Ereignisses enthält, denn es genügt, dies genau für die Kandidaten der Mächtigkeit 2 sicherzustellen. Werden die Kandidaten der Mächtigkeit 2,

die ein Ereignis und einen Vorfahr dieses Ereignisses enthalten, als nichthäufig markiert, dann werden wegen der Abgeschlossenheitseigenschaft des Supports (vgl. Abschnitt 3.1.1) von Cumulate keine redundanten Kandidaten der Mächtigkeit größer 2 erzeugt (vgl. dazu Abschnitt 3.1.2 und [Srikant und Agrawal, 1995]).

Für auf einer Tiefensuche beruhende Verfahren, wie beispielsweise Eclat, ist dieses Vorgehen nicht möglich, da dazu sichergestellt sein muß, daß kein Kandidat ausgezählt wird, der als Teilmenge eine als nichthäufig markierte, das heißt gegebenenfalls redundante Ereigniskombination enthält. Erst das in Abschnitt 4.1 eingeführte Verfahren Optra, das auf der optimierten Tiefensuche beruht, bietet diese Voraussetzung.

Weiteres Optimierungspotential ergibt sich wie folgt aus der Taxonomie: Sei ein *Vorfahr (ancestor)* der Ereigniskombination X definiert als die Ereigniskombination $\hat{X}$ mit $|\hat{X}| = |X|$, wobei $\hat{X}$ aus X erzeugt werden kann, indem ein beliebiges Ereignis x in X durch einen der Vorfahren $\hat{x} \in \mathsf{anc}(x)$ von x ersetzt wird. $\hat{X}$ sei ein *direkter Vorfahr (direct ancestor)* von X, wenn kein X' existiert, das Vorfahr von X ist und zugleich $\hat{X}$ Vorfahr von X' ist. Offensichtlich können die Kandidaten, zu denen ein nichthäufiger direkter Vorfahr bekannt ist, ohne Häufigkeitsbestimmung als nichthäufig verworfen werden.

Da Ereigniskombinationen, die nach obiger Definition in einem Vorfahrenverhältnis stehen, die gleiche Mächtigkeit aufweisen, kann die Beschneidung des Suchraums anhand nichthäufiger Vorfahren für eine Breitensuche, auf der beispielsweise der Algorithmus Cumulate basiert, nicht effizient umgesetzt werden. Der Grund ist, daß die Breitensuche voraussetzt, daß sämtliche Kandidaten gleicher Mächtigkeit gemeinsam oder zumindest in Gruppen und nicht einzeln ausgezählt werden, indem für eine Menge von Kandidaten jeweils sämtliche Datenbanktransaktionen durchlaufen werden (vgl. auch Abschnitt 3.1). Die im Sinne der Vorfahrendefinition oberen beziehungsweise allgemeineren Ereigniskombinationen zuerst auszuzählen und dann sukzessive abzusteigen, ist aufgrund der vielen zusätzlich erforderlichen Datenbankdurchläufe ineffizient. Für die Tiefensuche wurde hingegen in Abschnitt 4.1.1 gezeigt, daß die Kandidaten einzeln nacheinander ausgezählt werden können.

Voraussetzung für das Beschneiden des Suchraums anhand nichthäufiger direkter Vorfahren ist, daß zum Zeitpunkt der Häufigkeitsbestimmung eines Kandidaten die Häufigkeiten seiner sämtlichen direkten Vorfahren bereits bekannt sind. Entsprechend wird eine Ordnung auf den Ereigniskombinationen einer Mächtigkeit benötigt, welche die Vorfahrenbeziehung widerspiegelt und beim Durchlaufen des Suchraums einzuhalten ist. Dazu wird die *Tiefe (depth)* einer Ereigniskombination

definiert (vgl. [Srikant und Agrawal, 1995], [Hipp et al., 1998]):

$$\mathsf{depth} : \mathcal{P}(\mathcal{E}) \to \mathbb{N} : X \mapsto \begin{cases} 0, \text{ falls } \{\hat{X} \mid \hat{X} \text{ ist Vorfahr von } X\} = \emptyset \\ \mathsf{max}(\{\mathsf{depth}(\hat{X} \mid \hat{X} \text{ ist Vorfahr von } X\}) + 1, \text{ sonst} \end{cases}$$

Für jeden Vorfahr $\hat{K}$ eines Kandidaten K gilt dann $\mathsf{depth}(\hat{K}) < \mathsf{depth}(K)$. Um sicherzustellen, daß sämtliche direkten und nichthäufigen Vorfahren eines Kandidaten bekannt sind, wenn dessen Häufigkeit bestimmt werden soll, genügt es, die Häufigkeiten der Kandidaten K mit $\mathsf{depth}(K) = d$ zu bestimmen, bevor die Häufigkeiten der Kandidaten K' mit $\mathsf{depth}(K') = d + 1$ und $|K'| = |K|$ bestimmt werden.

Im Widerspruch dazu setzt die in Abschnitt 4.1.1 vorgestellte optimierte Tiefensuche voraus, daß die von der dort ebenfalls eingeführten Abbildung map induzierte Ordnung $<_{\mathcal{L}}$ beim Durchlaufen des Suchraums eingehalten wird. Dieser Widerspruch läßt sich dahingehend auflösen, daß zwar die von depth erzeugte Ordnung durch die Taxonomie fest vorgegeben ist, die Abbildung map aber lediglich als eine beliebige Basis für die Relation $<_{\mathcal{L}}$ auf den Ereigniskombinationen dient. Welche Ordnung die Abbildung map in Form von $<_{\mathcal{L}}$ induziert, ist letztlich jedoch unerheblich. Wichtig ist für die optimierte Tiefensuche nur, daß für das Durchlaufen des Suchraums genau eine verbindliche Ordnung definiert wird. Sei x_n das n-te Element des Vektors aller Ereignisse $x \in \mathcal{E}$, die anhand der Tiefe der Ereignisse innerhalb der Taxonomie, das heißt der Entfernung der Ereignisse von der Wurzel des Taxonomiebaums, absteigend sortiert wurden. Die Abbildung map sei so gewählt, daß $\mathsf{map}(x_n) = n$ für alle n gilt:[5]

$$\mathsf{map}(x) < \mathsf{map}(\hat{x}) \Leftrightarrow \mathsf{depth}(\{x\}) \geq \mathsf{depth}(\{\hat{x}\}).$$

Für beliebige $X, \hat{X} \subseteq \mathcal{E}$ mit $|X| = |\hat{X}|$ folgt dann:

$$\begin{aligned} X <_{\mathcal{L}} \hat{X} &\Rightarrow \exists n \in \mathbb{N} : X.\mathsf{item}_n < \hat{X}.\mathsf{item}_n \\ &\Rightarrow \exists n \in \mathbb{N} : \mathsf{depth}(\{X.\mathsf{item}_n\}) \geq \mathsf{depth}(\{\hat{X}.\mathsf{item}_n\}) \\ &\Rightarrow X \text{ ist kein Vorfahr von } \hat{X}. \end{aligned}$$

Damit gilt $K <_{\mathcal{L}} \hat{K}$ für alle Vorfahren $\hat{K}$ von K.[6] Wird der optimierten Tiefensuche aus Abschnitt 4.1.1 die oben definierte Abbildung map zugrunde gelegt, ist für jeden Kandidaten K sichergestellt, daß sämtliche nichthäufigen Vorfahren $\hat{K}$ von

[5] Da die verwendete Sortierung auf Basis der Entfernungen von der Wurzel nicht eindeutig ist, gibt es im allgemeinen mehrere Möglichkeiten, die Abbildung map zu wählen.

[6] Die Umkehrung, aus $K <_{\mathcal{L}} K'$ folgt K' ist Vorfahr von K, gilt nicht.

K vor der Bestimmung der Häufigkeit von K bekannt sind und der Suchraum daher anhand der Taxonomie beschnitten werden kann.

Das resultierende Verfahren, das auf einer optimierten Tiefensuche beruht und den Suchraum wie beschrieben anhand redundanter Ereigniskombinationen sowie der Taxonomie beschneidet, wird mit *Prutax*, abgeleitet von „**pru**ne by **tax**onomy“ bezeichnet.

Neben dem ausschließlich auf der Häufigkeitsbestimmung mittels Schnittmengenbildung beruhenden Verfahren Prutax wird im folgenden ein hybrider Ansatz eingeführt, der entsprechend dem in Abschnitt 4.1.2 vorgestellten Vorgehen die Häufigkeiten der Kandidaten der Mächtigkeit 2 durch direktes Zählen bestimmt und zu den dabei als häufig erkannten Kandidaten die zugehörigen Transaktionsmengen anhand der Datenbank erzeugt. Basierend auf diesen Transaktionsmengen wird dann das als Prutax bezeichnete Verfahren zur Bestimmung der verbleibenden häufigen Ereigniskombinationen verwendet. Dieser hybride Ansatz wird mit *Prutax+* bezeichnet.

Erste Evaluierung

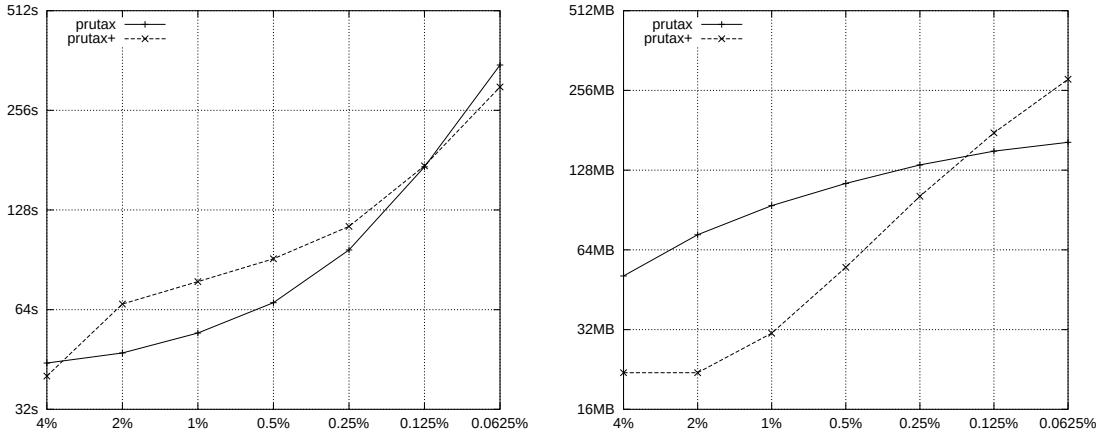
In den Abbildungen 4.18 bis 4.20 werden die Laufzeiten und der Speicherbedarf von Prutax und Prutax+ auf verschiedenen Datensätzen miteinander verglichen, wobei die Laufzeiten dieser neuen Ansätze insgesamt vergleichsweise ähnlich sind. Anders als bei der Generierung einfacher Assoziationsregeln erreicht jedoch nicht das hybride Verfahren die kürzeren Laufzeiten, sondern Prutax ist insbesondere auf den anspruchsvolleren Datensätzen das effizientere der beiden Verfahren und erreicht um bis zu 30% kürzere Laufzeiten.

Bei sinkender minimaler Häufigkeit fällt der hybride Ansatz durch einen stark ansteigenden Hauptspeicherbedarf auf. Während für größere minimale Häufigkeiten Prutax+ wesentlich weniger Hauptspeicher benötigt als Prutax, ist für geringe minimale Häufigkeiten die Situation umgekehrt. Beispielsweise benötigt Prutax+ auf dem Datensatz T20I6D1000K bei minsupp = 16% nur 25% des Hauptspeichers von Prutax, während bei minsupp = 1% der Hauptspeicherbedarf von Prutax ungefähr 25% des Hauptspeicherbedarfs von Prutax+ beträgt. Dieser Effekt verstärkt sich mit zunehmend anspruchsvolleren Datensätzen.

Insgesamt ist Prutax auf den angeführten Datensätzen sowohl bezüglich der Laufzeiten als auch des Hauptspeicherbedarfs effizienter als die hybride Erweiterung Prutax+.

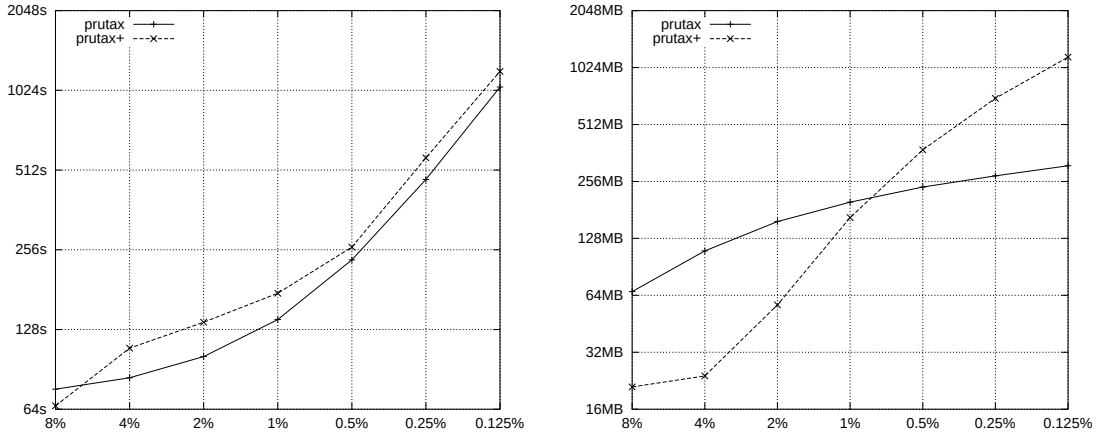
4.2.3 Abschließender Vergleich

Zunächst ist zu betrachten, wie sich die Hinzunahme der Taxonomie auf die Eigenschaften der angereicherten Daten auswirkt. In einer angereicherten Transakti-



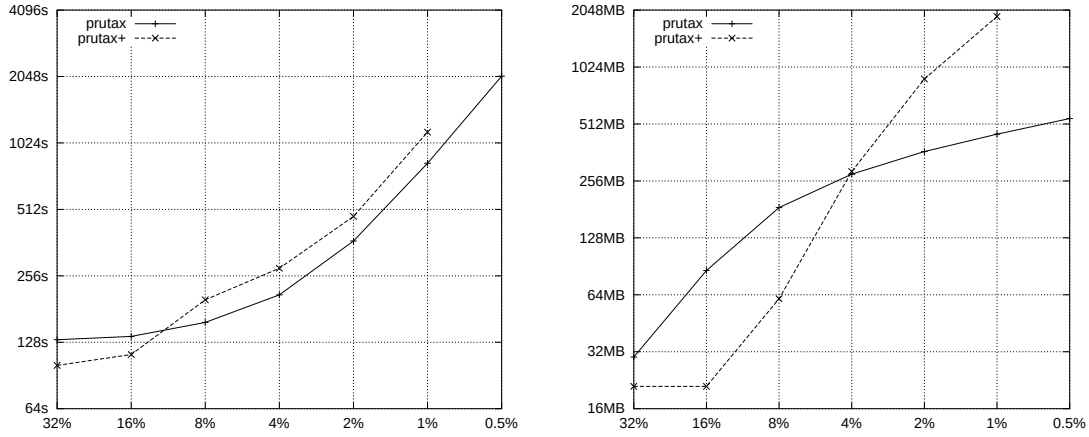
(a) Laufzeit in Sekunden (b) Speicherbedarf in Megabyte

Abbildung 4.18: Effizienz der Verfahren Prutax und Prutax+ auf T5I2D1000K bei variierendem minsupp



(a) Laufzeit in Sekunden (b) Speicherbedarf in Megabyte

Abbildung 4.19: Effizienz der Verfahren Prutax und Prutax+ auf T10I4D1000K bei variierendem minsupp



(a) Laufzeiten in Sekunden

(b) Speicherbedarf in Megabyte

Abbildung 4.20: Effizienz der Verfahren Prutax und Prutax+ auf T20I6D1000K bei variierendem minsupp

on kommt ein Ereignis immer dann vor, wenn es entweder bereits in der Ausgangstransaktion enthalten ist oder die Ausgangstransaktion mindestens einen Nachkommen dieses Ereignisses enthält. Entsprechend sind die aus den oberen Ebenen des Taxonomiebaums zu den Transaktionen hinzukommenden Ereignisse meist weit häufiger als die bereits enthaltenen Ereignisse. Unter anderem durch die Anzahl der Ebenen des Taxonomiebaums und die mittlere Anzahl von Sohnknoten pro Vaterknoten, das heißt durch Breite und Höhe, hat offensichtlich die Taxonomie großen Einfluß auf die Eigenschaften der angereicherten Transaktionen.[7]

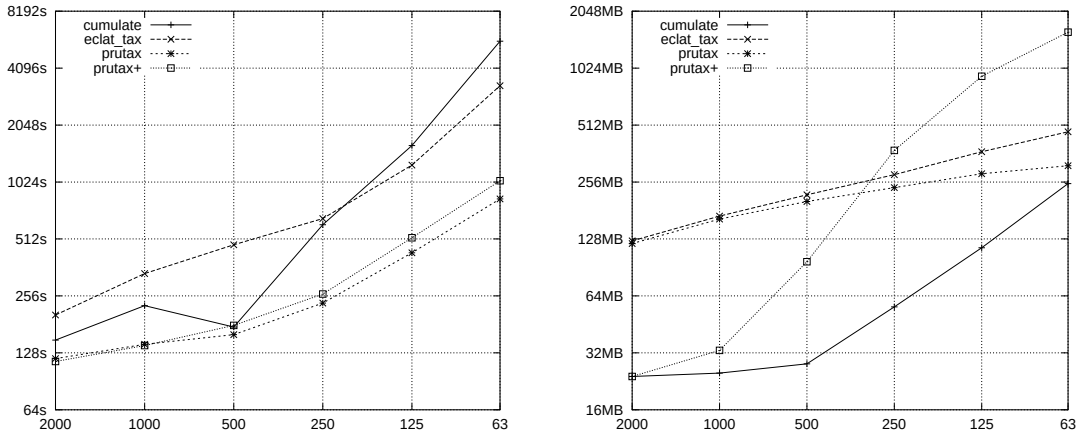
Mit dem in Abschnitt 3.3.1 vorgestellten Datengenerator werden im folgenden Taxonomien zu den Transaktionsmengen erzeugt. Ergänzend zu den Parametern, welche die Generierung der Transaktionen steuern, kann die Erzeugung einer Taxonomie durch die Vorgabe weiterer Parameter beeinflußt werden. Zum einen ist es möglich, die Anzahl der Ereignisse auf der zweiten Ebene der Taxonomie sowie die durchschnittliche Anzahl von Sohnknoten für die Ereignisse vorzugeben. Zum anderen kann das sogenannte Tiefenverhältnis angegeben werden, das beeinflußt, ob die Ereignisse der häufigen Ereigniskombinationen eher aus den oberen Ebenen der Taxonomie stammen oder aus den unteren. Wenn nicht anders angegeben, dann ist die Anzahl der Ereignisse auf der zweiten Ebene der Taxonomie auf 250 gesetzt, die Vorgabe für die durchschnittliche Anzahl von Söhnen pro Ereignis ist

[7] Dies gilt sinngemäß auch für eine als gerichteter azyklischer Graph modellierte Taxonomie.

mit 5 initialisiert und das Tiefenverhältnis beträgt 1.[8] Eine ausführliche Diskussion der Parameter sowie der Datengenerierung ist in [Srikant und Agrawal, 1995], [Srikant, 1996] zu finden.

Für die abschließende Evaluierung werden von den im Rahmen dieser Arbeit erweiterten etablierten Verfahren die Algorithmen Cumulate und Eclat, die sich in den ersten Experimenten für die Generierung taxonomer Assoziationsregeln als die effizientesten Verfahren erwiesen haben, ausgewählt. Außerdem werden die beiden Neuentwicklungen Prutax und Prutax+ aufgenommen.

In Abbildung 4.21 wird die Anzahl von Ereignissen auf der zweiten Ebene der Taxonomie variiert. Mit abnehmender Anzahl von Ereignissen auf der zweiten Ebene



(a) Laufzeit in Sekunden

(b) Speicherbedarf in Megabyte

Abbildung 4.21: Effizienz taxonomer Verfahren bei variierender Anzahl Ereignisse auf der zweiten Ebene der Taxonomie

wird der Taxonomiebaum schlanker und höher, da die Anzahl von Ereignissen insgesamt beschränkt ist. Aufgrund der dadurch vergleichsweise häufigen Ereignisse aus den oberen Ebenen der Taxonomie, ist die Generierung der häufigen Ereigniskombinationen entsprechend aufwendiger.

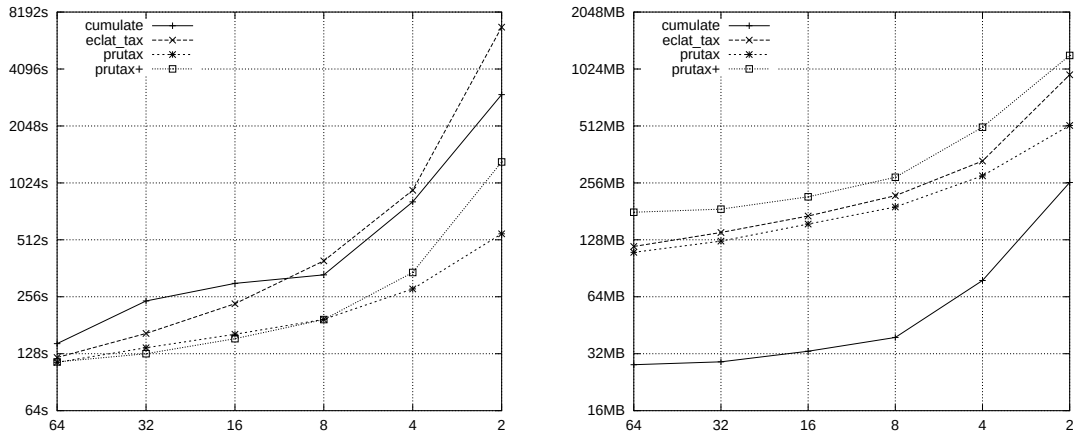
Die beiden neuen Verfahren Prutax und Prutax+ erreichen in fast allen Experimenten wesentlich kürzere Laufzeiten als Cumulate und Eclat. Die Laufzeiten von Prutax und Prutax+ sind sich in den Experimenten recht ähnlich, wobei Prutax

[8] Im Unterschied zur Generierung von Transaktionen ohne Taxonomie (vgl. Abschnitt 3.3.1) wird im Fall der Generierung von Transaktionen mit Taxonomie nach [Srikant und Agrawal, 1995], [Srikant, 1996] die Anzahl der potentiell häufigen Ereigniskombinationen von 2.000 auf 10.000 und die Anzahl der Ereignisse von 1.000 auf 10.000 heraufgesetzt.

nahezu in jedem Fall das effizientere der beiden Verfahren ist. Prutax erreicht um bis zu 75% kürzere Laufzeiten als das effizienteste der erweiterten etablierten Verfahren.

Hinsichtlich des Hauptspeicherbedarfs ist Cumulate das bei weitem effizienteste Verfahren. Eclat und Prutax benötigen mehr Hauptspeicher, wobei Prutax mit zunehmend schlankerer und höherer Taxonomie gegenüber Eclat etwas bessere Werte erreicht. Prutax benötigt in den Experimenten mindestens 20% mehr Hauptspeicher als Cumulate. Prutax+ hingegen zeichnet sich zwar durch einen moderaten Hauptspeicherbedarf für vergleichsweise viele Ereignisse in der zweiten Ebene der Taxonomie aus, der Hauptspeicherbedarf steigt aber mit abnehmender Anzahl von Ereignissen in Ebene 2 steil an und liegt schließlich weit über dem der anderen Verfahren.

In Abbildung 4.22 wird die vorgegebene mittlere Anzahl von Söhnen pro Ereignis variiert. Für kleinere Werte nimmt die Höhe des Taxonomiebaums zu. Anders



(a) Laufzeit in Sekunden

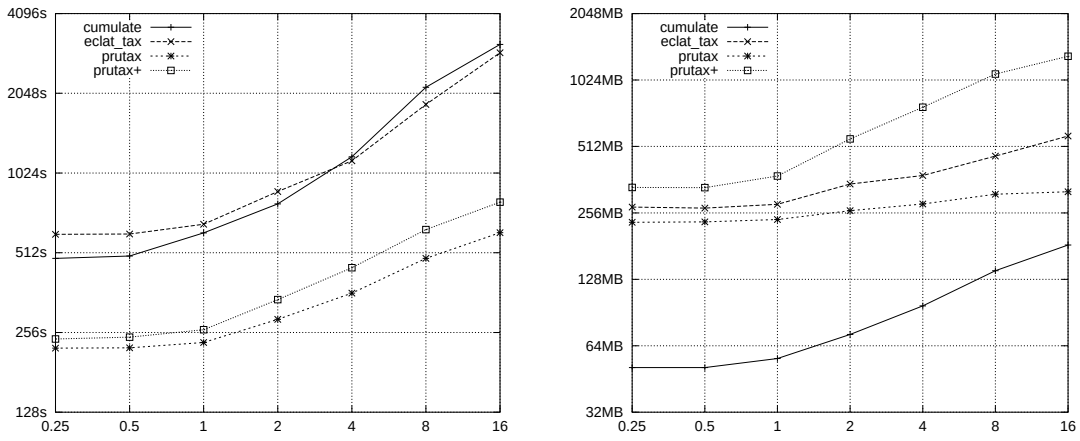
(b) Speicherbedarf in Megabyte

Abbildung 4.22: Effizienz taxonomer Verfahren bei variierender mittlerer Anzahl von Söhnen pro Ereignis

als die Verringerung der Anzahl von Ereignissen auf der zweiten Ebene der Taxonomie wirkt sich die durchschnittliche Anzahl der Söhne pro Ereignis auf die Struktur des gesamtem Baums aus. Die Ergebnisse sind trotzdem mit denen aus dem vorherigen Experiment vergleichbar, das heißt, Prutax und Prutax+ sind bezüglich der Laufzeiten im Vergleich mit den erweiterten etablierten Verfahren die bei weitem effizienteren Ansätze, und Prutax erreicht in den Experimenten um bis zu 80% kürzere Laufzeiten als die erweiterten etablierten Verfahren. Zugleich bleibt Cumulate das hinsichtlich des Hauptspeicherbedarfs effizienteste Verfahren,

auf das direkt Prutax und anschließend Eclat und Prutax+ folgen, wobei bereits Prutax in den Experimenten um mindestens 100% mehr Hauptspeicher als Cumulate benötigt.

In Abbildung 4.23 wird das der Datengenerierung zugrundeliegende Tiefenverhältnis variiert. Ein hoher Wert für das Tiefenverhältnis begünstigt die Häufigkeit von



(a) Laufzeit in Sekunden

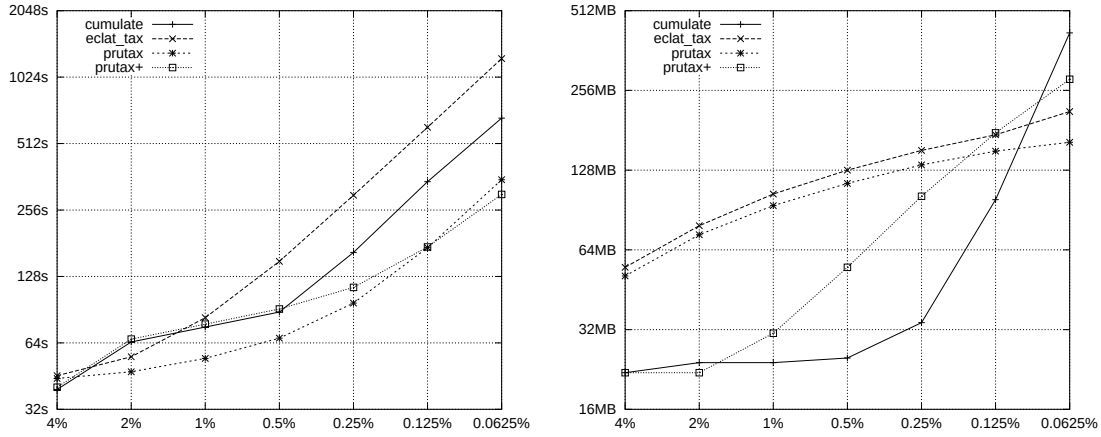
(b) Speicherbedarf in Megabyte

Abbildung 4.23: Effizienz taxonomer Verfahren bei variierendem Wert für das Tiefenverhältnis

Ereigniskombinationen aus den unteren Ebenen der Taxonomie, deren Vorfahren aus den oberen Ebenen dadurch zugleich häufiger werden. Entsprechend nimmt der Aufwand für die Generierung der häufigen Ereigniskombinationen mit steigendem Wert für das Tiefenverhältnis zu. Die Laufzeiten der neu entwickelten Verfahren sind erneut wesentlich kürzer als die der erweiterten etablierten Verfahren. Prutax ist in jedem der Experimente effizienter als die hybride Erweiterung Prutax+ und erreicht zwischen 50% und 80% kürzere Laufzeiten als die erweiterten etablierten Verfahren. Hinsichtlich des Hauptspeicherbedarfs bleibt Cumulate das mit Abstand effizienteste der Verfahren, gefolgt von Prutax und Eclat und in einigem Abstand von Prutax+, wobei Prutax ebenfalls um mindestens 100% mehr Hauptspeicher als Cumulate benötigt.

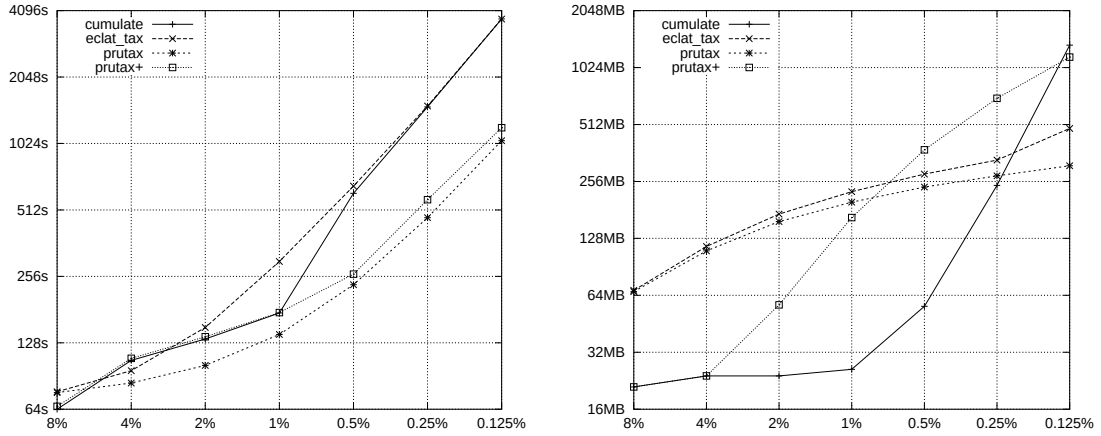
Nach der expliziten Untersuchung des Einflusses der Taxonomie auf die Laufzeiten und den Hauptspeicherbedarf der Verfahren werden in den Abbildungen 4.24 bis 4.26 die Algorithmen auf den bekannten synthetischen Datensätzen evaluiert (für weitere Experimente auf synthetischen Daten vgl. Anhang A.3.2). Die Taxonomie wird jeweils auf Basis der Standardwerte für die Parameter erzeugt.

Hinsichtlich der Laufzeiten bestätigen sich die bisherigen Ergebnisse. Auf dem Datensatz T5I2D1000K erreicht Cumulate nahezu in jedem Fall kürzere Laufzeiten als



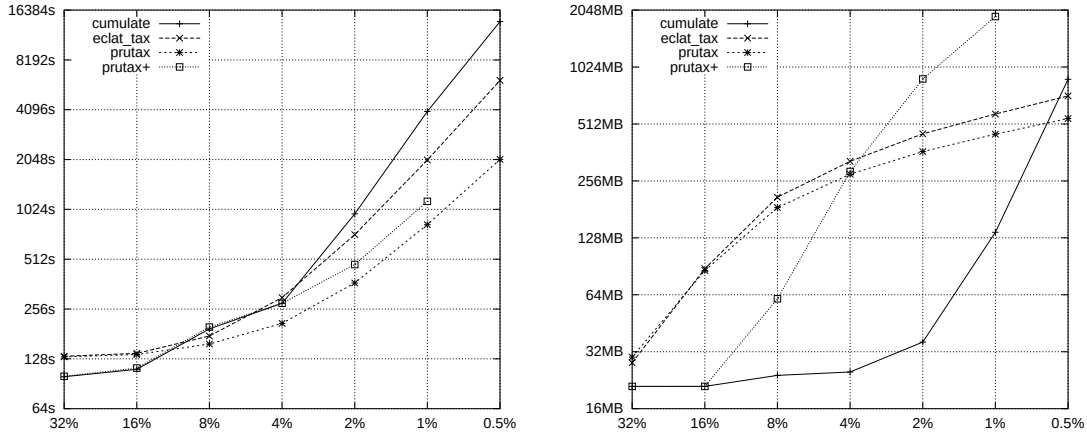
(a) Laufzeit in Sekunden (b) Speicherbedarf in Megabyte

Abbildung 4.24: Effizienz ausgewählter taxonomer Verfahren auf T5I2D1000K bei variierendem minsupp



(a) Laufzeit in Sekunden (b) Speicherbedarf in Megabyte

Abbildung 4.25: Effizienz ausgewählter taxonomer Verfahren auf T10I4D1000K bei variierendem minsupp



(a) Laufzeit in Sekunden

(b) Speicherbedarf in Megabyte

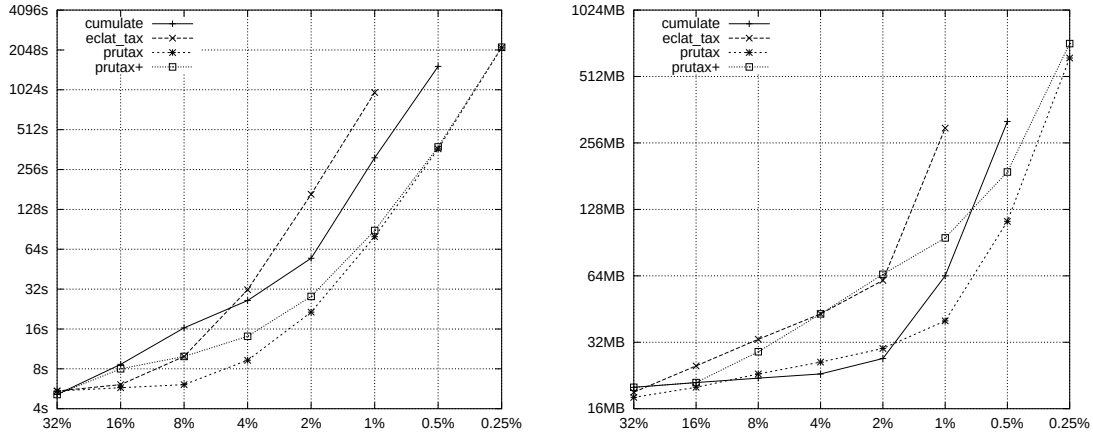
Abbildung 4.26: Effizienz ausgewählter taxonomer Verfahren auf T20I6D1000K bei variierendem minsupp

Eclat, während beide Verfahren auf T10I4D1000K für vergleichsweise geringe minimale Häufigkeiten ähnliche Zeiten erreichen. Auf T20I6D1000K ist Eclat effizienter als Cumulate. Wieder erreichen Prutax und Prutax+ gegenüber den erweiterten etablierten Verfahren deutlich kürzere Laufzeiten. Im einzelnen erreicht Prutax auf den Datensätzen T5I2D1000K, T10I4D1000K und T20I6D1000K gegenüber den erweiterten etablierten Verfahren um bis zu 50%, 70% beziehungsweise ebenfalls bis zu 70% kürzere Laufzeiten.

Auch bezüglich des Hauptspeicherbedarfs der Verfahren werden die bisherigen Ergebnisse bestätigt. Cumulate ist der mit Abstand effizienteste Ansatz, solange die Menge der zu untersuchenden Kandidaten nicht zu groß ist. Mit zunehmend kleineren Vorgaben für die minimale Häufigkeit geht der Vorteil von Cumulate gegenüber den anderen Verfahren jedoch verloren. Für besonders geringe minimale Häufigkeiten benötigt Cumulate schließlich mehr Hauptspeicher als die anderen Verfahren. Dieser Effekt ist noch ausgeprägter für Prutax+, das als hybrides Verfahren meist sehr große Transaktionsmengen und zugleich sehr große Kandidatenmengen erzeugt. Der Hauptspeicherbedarf von Eclat und Prutax ist sehr ähnlich, wobei Prutax das effizientere der beiden Verfahren ist.

Für die Evaluierung der Verfahren auf Daten aus einer realen Anwendungen wurde der in Abschnitt 3.3.1 eingeführte Datensatz EINZELHANDEL herangezogen. Zu diesem Datensatz stand die ebenfalls in Abschnitt 3.3.1 vorgestellte Taxonomie zur Verfügung.

Die Laufzeiten und der Speicherbedarf der ausgewählten taxonomen Verfahren sind für den Datensatz EINZELHANDEL in Abbildung 4.27 wiedergegeben. Eclat



(a) Laufzeit in Sekunden

(b) Speicherbedarf in Megabyte

Abbildung 4.27: Effizienz taxonomer Verfahren auf EINZELHANDEL bei variierendem minsupp

erreicht auf diesen Daten sowohl hinsichtlich der Laufzeiten als auch des Speicherbedarfs die schlechtesten Ergebnisse. Bereits für minimale Häufigkeiten von 0,5% und 0,25% genügt der Hauptspeicher des eingesetzten Rechnersystems nicht mehr, um die häufigen Ereigniskombinationen mit Eclat zu generieren. Cumulate erreicht kürzere Laufzeiten als Eclat, kann aber ebenfalls aufgrund des Hauptspeicherbedarfs für minimale Häufigkeiten von 0,25% und weniger keine häufigen Ereigniskombinationen erzeugen. Die beiden neu entwickelten Verfahren schneiden sowohl hinsichtlich der Laufzeiten als auch des Hauptspeicherbedarfs wesentlich besser ab als die erweiterten etablierten Ansätze. Selbst bei einer minimalen Häufigkeit von 0,25% benötigen Prutax und Prutax+ weit weniger als die Hälfte des verfügbaren Hauptspeichers. Die Laufzeiten von Prutax und Prutax+ liegen zumeist weit unter denen von Eclat und Cumulate, und Prutax erreicht in den Experimenten bis zu 80% kürzere Laufzeiten als die erweiterten etablierten Verfahren.

Zusammenfassend kann festgehalten werden, daß die hybride Erweiterung Prutax+ insbesondere wegen des teilweise hohen Hauptspeicherbedarfs nicht durchweg überzeugen kann. Hingegen ist das ebenfalls neu entwickelte Verfahren Prutax bezüglich der Laufzeiten und des Hauptspeicherbedarfs in den durchgeführten Experimenten weit effizienter als die erweiterten etablierten Verfahren sowie Cumulate. In vielen der Experimente erreicht Prutax um bis zu 80% kürzere Laufzeiten.

4.3 Zusammenfassung und Bewertung

In diesem Kapitel wurden verschiedene Erweiterungen bekannter Verfahren vorgestellt und neu entwickelte Ansätze zur Generierung einfacher und taxonomer Assoziationsregeln eingeführt.

Mit den bisher bekannten Verfahren zur Generierung häufiger Ereigniskombinationen, die auf einer Tiefensuche basieren, ist es nicht möglich, den Suchraum anhand nichthäufiger Teilmengen von Kandidaten zu beschneiden. Der Nachteil von Verfahren wie beispielsweise Eclat besteht darin, daß beim Abstieg im Suchraum nicht sichergestellt ist, daß vor der Häufigkeitsbestimmung eines Kandidaten die Häufigkeiten zu dessen sämtlichen Teilmengen bekannt sind. Außerdem kann aufgrund der vielen Kandidaten nicht mitgeführt werden, ob die Häufigkeit einer Ereigniskombination bereits bestimmt wurde oder nicht. Damit ist während des Durchlaufens des Suchraums nicht bekannt, ob eine Ereigniskombination nichthäufig oder häufig aber noch nicht ausgezählt ist. Der Suchraum kann folglich auch nicht teilweise über nichthäufige Teilmengen von Kandidaten beschnitten werden.

Mit dem hier neu vorgestellten Ansatz konnte erstmals eine Ordnung auf den Ereigniskombinationen definiert werden, die beim Abstieg im Suchraum garantiert, daß die Häufigkeiten sämtlicher Teilmengen eines Kandidaten bekannt sind, sobald dieser Kandidat selbst ausgezählt werden soll. Die auf dieser Ordnung basierende spezielle Tiefensuche wird als rechtsorientierte Tiefensuche bezeichnet. Erfolgt der Abstieg im Suchraum als rechtsorientierte Tiefensuche, dann ist die Beschneidung des Suchraums anhand nichthäufiger Ereigniskombinationen ohne Einschränkungen möglich. Das resultierende Verfahren Optra ermöglicht in verschiedenen Experimenten, daß bis zu 30% der Kandidaten direkt verworfen werden können, ohne aufwendig deren Häufigkeiten anhand der Daten zu bestimmen. Trotzdem gelingt es nur beschränkt, diesen Vorteil in kürzere Laufzeiten umzusetzen. Die eigentliche Bedeutung von Optra wird erst bei der Generierung taxonomer Assoziationsregeln deutlich, da hier die vollständige Beschneidung des Suchraums anhand nichthäufiger Teilmengen neue Optimierungsmöglichkeiten eröffnet.

Neben der Verbesserung der Suchstrategie wurde außerdem ein hybrider Ansatz zur Generierung einfacher Assoziationsregeln eingeführt. Dieser Ansatz beruht auf Erkenntnissen aus der Analyse der etablierten Verfahren in Kapitel 3. Dort wurde unter anderem gezeigt, daß in Abhängigkeit von den Eigenschaften der zugrundeliegenden Daten jeweils andere Verfahren die kürzeren Laufzeiten erreichen. Auf Basis der verschiedenen Experimente wurde hier in diesem Kapitel die These abgeleitet, daß die Häufigkeitsbestimmung durch direktes Zählen insbesondere für Kandidaten kleinerer Mächtigkeiten vorteilhaft ist, während das Schneiden von Transaktionsmengen für Kandidaten größerer Mächtigkeiten eine höhere Effizienz verspricht. Diese These wurde anhand von Laufzeitmessungen überprüft und ein

entsprechender hybrider Ansatz vorgestellt. Die Effizienz des neuen Verfahrens Hybrid wurde in verschiedenen Experimenten gezeigt, wobei Hybrid oft wesentlich kürzere Laufzeiten als die bisherigen Verfahren aufwies, größtenteils ohne daß diese Zeiten mit dem problematischen Hauptspeicherbedarf einhergehen, der beispielsweise für FP-Growth typisch ist. Im einzelnen wurden für das neu entwickelte Verfahren auf verschiedenen Datensätzen zwischen 15% und 60% kürzere Laufzeiten als für die etablierten Verfahren gemessen.

Für die Generierung taxonomer Assoziationsregeln wurden zunächst die vielversprechendsten der etablierten Verfahren derart erweitert, daß diese eine Taxonomie einbeziehen und taxonome Assoziationsregeln generieren können. In den folgenden Experimenten fiel insbesondere auf, daß FP-Growth, anders als es aufgrund der Laufzeiten bei der Generierung einfacher Assoziationsregeln zu erwarten gewesen wäre, nicht mehr wesentlich kürzere Laufzeiten als die anderen Verfahren erreicht. Zugleich stieg der bereits für die Generierung einfacher Assoziationsregeln problematische Hauptspeicherbedarf von FP-Growth bei der Generierung taxonomer Assoziationsregeln weiter an.

Mit Prutax und Prutax+ wurden anschließend zwei neue Verfahren eingeführt, die erstmals eine Taxonomie auf den Ereignissen umfassend zur Beschneidung des Suchraums einsetzen können. Mit den bisher üblichen Verfahren zur Generierung taxonomer Assoziationsregeln ist die Beschneidung des Suchraums anhand einer Taxonomie nur eingeschränkt möglich, da der Algorithmus Cumulate und seine Varianten auf dem Zählen von Häufigkeiten beruhen und damit aus Effizienzgründen die Kandidaten gruppenweise auszählen müssen. Auch Verfahren, die wie Eclat auf einer Tiefensuche beruhen, sind nur beschränkt für die Generierung taxonomer Assoziationsregeln geeignet, da eine wichtige Optimierung zur Vermeidung des Auszählens von im Sinne einer Taxonomie redundanten häufigen Ereigniskombinationen darauf basiert, daß der Suchraum anhand der nichthäufigen Teilmengen von Kandidaten beschnitten wird. Erst mit der zuvor für die Generierung der einfachen Assoziationsregeln eingeführten rechtsorientierten Tiefensuche wird es möglich, die im Sinne einer Taxonomie redundanten häufigen Ereigniskombinationen auch bei einer Tiefensuche effizient auszuschließen. Schließlich wurden aufgrund dieser Überlegungen und der anhand verschiedener Experimente gewonnenen Erkenntnisse zwei neue Algorithmen abgeleitet. Der Algorithmus Prutax basiert ausschließlich auf indirekter Häufigkeitsbestimmung durch Schneiden von Transaktionsmengen, während der hybride Algorithmus Prutax+ außerdem Häufigkeiten ausgewählter Ereigniskombinationen durch direktes Zählen von Häufigkeiten bestimmt. Beide Algorithmen können erstmals effizient eine Taxonomie für die Beschneidung des Suchraums umfassend nutzen. In verschiedenen Experimenten konnte insbesondere für Prutax gezeigt werden, daß dieses Verfahren durchweg wesentlich kürzere Laufzeiten und einen günstigeren Speicherbedarf als die bisherigen Verfahren er-

reicht. Im einzelnen wurden für Prutax auf verschiedenen Datensätzen bis zu 80% kürzere Laufzeiten als für die erweiterten etablierten Verfahren beziehungsweise die bisher speziell für die Generierung taxonomer Assoziationsregeln entwickelten Verfahren ermittelt.

Kapitel 5

Integration in den Wissensentdeckungsprozeß

Der Prozeß der Wissensentdeckung in Datenbanken wurde in Abschnitt 2.1 als nicht trivial, iterativ und hochgradig interaktiv eingeführt. Als für den Erfolg eines Wissensentdeckungsprozesses entscheidend wurde der Analyst mit seinen Fähigkeiten, seiner Motivation und Kreativität identifiziert. Im Kontext dieses „human centered process“ kommt der Integration der Analyseverfahren in den Wissensentdeckungsprozeß eine entscheidende Bedeutung zu. In der Forschung im Bereich der Assoziationsregelgenerierung wurde diesem Aspekt bisher allerdings wenig Beachtung zuteil, auch wenn die grundsätzliche Problematik im Umfeld der Wissensentdeckung in Datenbanken bekannt ist (vgl. u. a. [Dalkilic et al., 1995], [Johnson und Dasu, 1998], [Sarawagi et al., 1998a], [Quinlan, 2000]).

Bei der Wissensentdeckung mit Assoziationsregeln auf Basis großer Datenmengen addieren sich die Antwortzeiten für die Regelgenerierung (vgl. Kapitel 3 und 4) und die Extraktion und Aufbereitung der relevanten Daten typischerweise zu mehreren Minuten oder sogar Stunden. Im Kontext des Wissensentdeckungsprozesses sind solche Laufzeiten als äußerst problematisch anzusehen, da bereits die Verfolgung einer auch nur spekulativen Analyseidee potentiell einen erneuten Regelgenerierungslauf impliziert. Zumindest mittelfristig ist vor diesem Hintergrund anzunehmen, daß der Analyst sich während des Prozesses eher kritisch und zurückhaltend verhalten wird, statt spontan zu handeln. Letztlich wird daher die in praktischen Anwendungen für den Erfolg der Wissensentdeckung in Datenbanken entscheidende Kreativität und Inspiration des Analysten durch die unzureichende Effizienz der zugrundeliegenden Technologie entscheidend gehemmt.

In diesem Kapitel werden drei für die Integration der Assoziationsregelverfahren in den Wissensentdeckungsprozeß zentrale Aspekte identifiziert. Zu diesen Aspekten werden jeweils eigene Ansätze entwickelt und mit den bisher bekannten Ansätzen

verglichen. Zu Beginn des Kapitels wird in Abschnitt 5.1 die Problematik des Datenzugriffs untersucht und die Anbindung der Verfahren an relationale Datenbanksysteme vorgeschlagen. Eine konkrete Implementierung wird beschrieben und diese gemeinsam mit verfügbaren kommerziellen Lösungen evaluiert. In Abschnitt 5.2 wird ein sogenannter Regelcache eingeführt, der es in Verbindung mit einer ebenfalls vorgestellten Retrievalsprache ermöglicht, Iterationen des Wissensentdeckungsprozesses auf Basis bereits erzeugter Regeln durchzuführen, ohne die aufwendige Regelgenerierung erneut anstoßen zu müssen. Des weiteren wird in Abschnitt 5.3 der auf dem Regelcache basierende Ansatz erweitert, so daß auch Anfragen aus dem Cache beantwortet werden können, die auf Selektionen der zugrundeliegenden Transaktionen beruhen. Auf diese Weise können Regeln direkt aus dem Cache abgeleitet werden, die beispielsweise darauf basieren, daß die zugrundeliegende Transaktionsmenge auf einzelne Wochentage oder Fahrzeugtypen eingeschränkt wird.

5.1 Anbindung an relationale Datenbanksysteme

Die prototypischen Implementierungen der Verfahren zur Generierung von Assoziationsregeln basieren im Bereich der Forschung zumeist auf proprietären Dateiformaten zur Speicherung der Analysedaten. Soweit die Entwicklung und Evaluierung neuer Algorithmen das eigentliche Ziel ist, sind einfach zu implementierende und trotzdem effiziente Lösungen für den Datenzugriff naheliegend und angemessen. Auch im Kontext forschungsgetriebener Pilotanwendungen, die von den Entwicklern der Algorithmen selbst oder ähnlich technisch ausgerichteten Personen durchgeführt werden, sind solche Anbindungen noch als ausreichend anzusehen. Sobald allerdings die Verfahren in Geschäftsprozesse integriert und von technisch weniger versierten Anwendern eingesetzt werden, ist dieser Ansatz nicht mehr befriedigend und wird zu einem oft unterschätzten Hemmnis für den Transfer von Data Mining-Technologien in operative Umgebungen (vgl. [Hipp und Lindner, 1999], [Hipp et al., 2001b]).

In der Praxis werden für die permanente Datenspeicherung zunehmend relationale Datenbanksysteme eingesetzt. Die Daten sind demzufolge für die Data Mining-Algorithmen nicht direkt zugänglich, sondern müssen für Analysen erst aus den Datenbanksystemen exportiert werden. Da damit zumeist die Aufbereitung der Daten in ein proprietäres Dateiformat verbunden ist, gestaltet sich ein Exportieren oft als aufwendig und fehleranfällig (vgl. auch [Hipp und Lindner, 1999], [Hipp et al., 2001b]). Insbesondere im Kontext des interaktiven und iterativen Prozesses der Wissensentdeckung ist diese Problematik nicht zu unterschätzen. So bedeutet

jede Iteration, die zu einer der frühen Phasen des Wissensentdeckungsprozesses zurückkehrt, ein erneutes Exportieren und eine erneute Aufbereitung der Daten.

Anstatt die Daten den Verfahren zuzuführen, wird in dieser Arbeit mit der Integration der Algorithmen in das relationale Datenbanksystem ein entgegengesetzter Ansatz verfolgt. Dieser Ansatz kann als Weiterführung der Forderung von Quinlan angesehen werden, daß die Verfahren der Wissensentdeckung mit relationalen Daten effizient umgehen können sollen (vgl. [Quinlan, 2000]).

5.1.1 Anforderungsanalyse

Die klassische Anwendungsdomäne der Assoziationsregelgenerierung ist die Warenkorbanalyse. Während dieses Szenario in der wissenschaftlichen Literatur als anschauliche Beispielanwendung sehr populär ist, geht in der Praxis die Komplexität der meisten Anwendungen weit über die Analyse von zu Warenkörben zusammengefaßten Artikeln hinaus (vgl. z. B. Abschnitt 2.3). Im folgenden werden anhand einer solchen erweiterten Anwendung, der Analyse des Qualitätsinformationssystems *QUIS* aus dem Bereich Fahrzeugproduktion, Garantie und Kulanz der DaimlerChrysler AG, verschiedene Anforderungen an die Integration der Assoziationsregelverfahren mit relationalen Datenbanksystemen abgeleitet.

Die Datenbank QUIS im Vergleich

In Tabelle 5.1 ist die typische relationale Repräsentation von Warenkörben an-

trans_id	item_id
⋮	⋮
1147	4553
1147	5596
1148	8933
1148	12780
1148	27453
⋮	⋮

Tabelle 5.1: Relationale Repräsentation von Warenkörben

hand eines Beispiels dargestellt (vgl. ebenfalls [Agrawal und Shim, 1996], [Sarawagi et al., 1998a], [Sarawagi et al., 1998b]). Jede Kundentransaktion und jeder Artikel aus dem Sortiment wird eindeutig durch einen ganzzahligen Schlüssel trans_id beziehungsweise item_id identifiziert. Die Modellierung durch zwei Spalten ist notwendig, da es sich bei den Kundentransaktionen um Mengen von Artikeln handelt

und die Anzahl der Artikel in einer Transaktion stark variieren kann sowie eine maximale Transaktionsgröße im voraus nicht bekannt ist.

Die Datenbank QUIS wurde in den frühen achtziger Jahren bei der damaligen Daimler-Benz AG in Betrieb genommen und speichert seitdem Informationen über jedes produzierte Fahrzeug der Marke Mercedes-Benz aus den Bereichen Personenkraftwagen und Nutzfahrzeuge. QUIS enthält neben detaillierten Informationen über die Produktion auch die komplette Historie der Werkstattaufenthalte während der Garantie- und Kulanzzeit (vgl. auch [Hipp und Lindner, 1999], [Hotz et al., 1999]). Heute sind Informationen über weit mehr als 10 Millionen Fahrzeuge in der Datenbank gespeichert. In Abbildung 5.1 ist ein vereinfachter Ausschnitt des Entity-Relationship-Modells von QUIS wiedergegeben. Die Tabellen 5.2 und 5.3 enthalten zwei Beispiele für typische Tabellen aus QUIS.

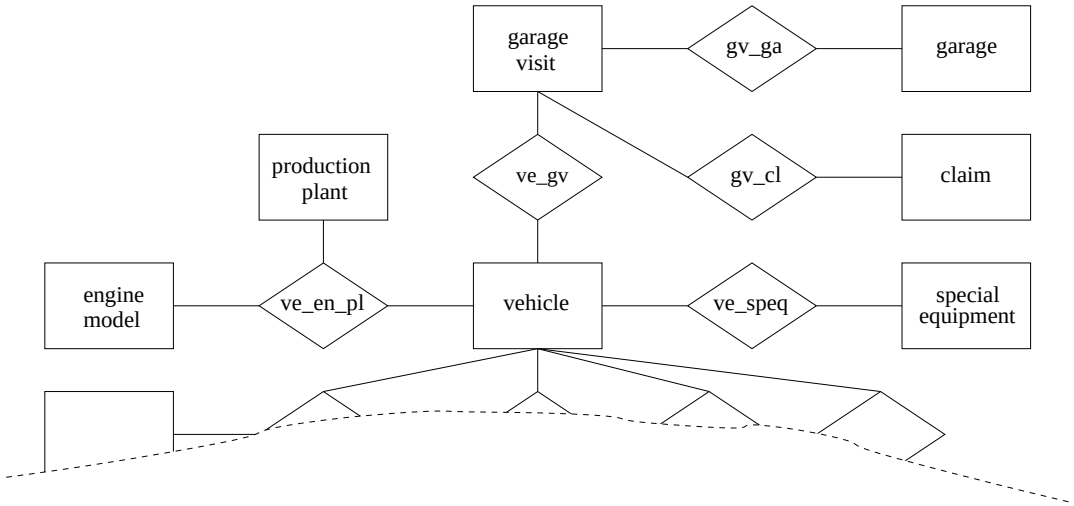


Abbildung 5.1: Vereinfachter Ausschnitt des Entity-Relationship-Modells zu QUIS

vehicle_id	model_type	type	prod_date	...
⋮	⋮	⋮	⋮	⋮
A23943CI	W202	R34	Jan 10 2003	...
C33467CI	W202	R56	Feb 12 2003	...
C45940CI	W220	R56	Feb 12 2003	...
C59612CI	W220	R56	Feb 12 2003	...
C59613CI	W220	R56	Feb 12 2003	...
⋮	⋮	⋮	⋮	⋮

Tabelle 5.2: Fahrzeuginformationen in Tabelle vehicle

vehicle_id	equip_id
⋮	⋮
A23943CI	A48543
A23943CI	A59032
C33467CI	B89494
C33467CI	A59032
C33467CI	B89569
C45940CI	A48543
⋮	⋮

Tabelle 5.3: Sonderausstattungen in Tabelle ve_speq

Im Vergleich zur Warenkorbanalyse fallen insbesondere drei Unterschiede auf:

(a) Anstatt mit einer einzelnen Tabelle, die sämtliche für die Erzeugung der Transaktionen relevanten Daten enthält, ist der Analyst mit mehreren Tabellen konfrontiert, über welche die zu analysierenden Daten verteilt sind. Beispielsweise enthält die Datenbanktabelle vehicle aus Tabelle 5.2 Daten über die Fahrzeuge selbst, wie Modelltyp, Typ oder Produktionsdatum. Hingegen sind die Daten über die Werkstattaufenthalte der Fahrzeuge in Tabelle garage_visit abgelegt, während Details über die in den Fahrzeugen verbauten Sonderausstattungen in der Datenbanktabelle special_equipment zu finden sind. Außerdem werden die Beziehungen zwischen den Entitäten teilweise in eigenen Tabellen abgelegt, wie beispielsweise in Tabelle ve_sqeq, die in Tabelle 5.3 wiedergegeben ist und die n:m-Beziehung zwischen Fahrzeugen und in diesen Fahrzeugen verbauten Sonderausstattungen modelliert. Diese Verteilung von Information ist eine Folge der Datenbanknormalisierung und damit typisch für relationale Datenbanken (zur Normalisierung vgl. u. a. [Date, 2000, S. 348ff.]).

(b) In QUIS speichern nur wenige der Tabellen Mengen beliebiger Mächtigkeit, wie beispielsweise die Datenbanktabelle ve_speq (vgl. Tabelle 5.3), die der kanonischen Modellierung der Einzelhandelstransaktionen in Tabelle 5.1 entspricht. Die meisten Tabellen enthalten hingegen genau einen Datensatz für jedes Objekt, wie etwa die Datenbanktabelle vehicle aus Tabelle 5.2, die für jedes Fahrzeug genau eine Zeile speichert.

(c) Anders als die Einzelhandelstransaktionen aus Tabelle 5.1 enthält eine Datenbank wie QUIS typischerweise nicht ausschließlich als ganzzahlige Schlüssel kodierte Ereignisse, sondern ist aus vielen verschiedenen Datentypen aufgebaut. Häufig sind für symbolische Werte Zeichenketten wie „W202“, in

QUIS der Fahrzeugtyp, oder für Sonderausstattungen beispielsweise eine Zeichenkette wie „AirCondition“. Selbst Schlüssel enthalten zumeist nicht ausschließlich Ziffern und werden daher als Zeichenketten gespeichert (vgl. u. a. Tabelle 5.3).

Nicht nur die Datenmodellierung ist in QUIS komplexer als im Kontext der klassischen Warenkorbanalyse, sondern auch der Analysebedarf ist deutlich vielschichtiger. Während bei der Warenkorbanalyse die Aufgabe typischerweise darin besteht, zu einer Menge von Warenkörben alle Assoziationsregeln zu generieren, die bestimmten Schwellenwerten für die Gütemaße der Regeln genügen, und die Regeln allenfalls anhand der Artikel eingeschränkt werden, sind die Freiheitsgrade im Zusammenhang mit der Analyse einer Datenbank wie QUIS deutlich höher:

Zum einen sind die Analyseanfragen nicht mehr auf isolierte Tabellen beschränkt. So können beispielsweise Abhängigkeiten zwischen Sonderausstattungen der Fahrzeuge und späteren Werkstattaufenthalten von Interesse sein. Zum anderen ist die Wahl der Bezugsobjekte nicht mehr vorgegeben, sondern abhängig von der Anwendung. Beispielsweise können Zusammenhänge zwischen den Eigenschaften der Fahrzeuge analysiert werden oder auch Zusammenhänge zwischen Eigenschaften der Werkstätten und den von diesen ausgeführten Arbeiten, beispielsweise den eingereichten Garantiekosten oder häufig durchgeführten Reparaturen.

Die Warenkorbanalyse, so wie sie heute zumeist im Mittelpunkt der Arbeiten zur Assoziationsregelgenerierung steht, ist demzufolge lediglich als ein Spezialfall der Wissensentdeckung mit Assoziationsregeln anzusehen.

Abgeleitete Anforderungen

Im folgenden werden verschiedene Anforderungen an einen Ansatz zur Integration der Assoziationsregelverfahren mit relationalen Datenbanksystemen abgeleitet:

1. Die Anzahl zu analysierender Transaktionen ist oft groß und unterliegt nahezu keinen Einschränkungen seitens der Analyseverfahren (vgl. Abschnitt 3.3). Große Datenmengen als Text in das Dateisystem zu exportieren, ist hinsichtlich der Laufzeiten und des notwendigen Plattenplatzes sehr aufwendig, so daß ein direkter Zugriff der Verfahren auf die Daten erstrebenswert ist.

2. Die üblichen Implementierungen der Assoziationsregelalgorithmen arbeiten intern auf natürlichen Zahlen (vgl. Abschnitt 3.1). Die dazu notwendige Abbildung der in der Datenbank beispielsweise als Zeichenketten abgelegten symbolischen Werte auf natürliche Zahlen sollte für den Benutzer transparent erfolgen.

3. Die Wissensentdeckung ist typischerweise eine Anwendung, die nachträglich zu einem bereits eingeführten System hinzugefügt wird (vgl. auch Abschnitt 2.1). Die Verfahren sollten daher keine Änderungen der Datenbankstruktur erforderlich machen.

4. In der Datenbank sind zum einen Objekte abgelegt, denen jeweils eine fixe Menge von Attributen zugewiesen ist, das heißt jedem Objekt entspricht genau ein Datensatz aus einer Tabelle. Zum anderen sind zugleich auch Objekte gespeichert, die durch eine variierende Anzahl von Attributen beschrieben werden, die unter Umständen in mehreren Datensätzen einer Tabelle abgelegt und über einen gemeinsamen Schlüssel identifiziert werden. Die Verfahren zur Assoziationsregelgenerierung sollten auf beide Aufbereitungen zugreifen können.

5. Die für eine Analyse relevanten Daten sind im allgemeinen über verschiedene Tabellen verteilt und liegen nicht als Transaktionen aufbereitet vor. Demzufolge müssen zu jedem der in eine Analyse einbezogenen Objekte die entsprechenden Attributwerte in der Datenbank zunächst gesucht und dann als Transaktion, das heißt als Menge von Ereignissen, zusammengefaßt werden.

6. Analysen können sich auf verschiedene Objekttypen beziehen. In obigem Beispiel aus QUIS können unter anderem Fahrzeuge oder Werkstätten als Bezug für die Regelerzeugung sinnvoll sein, so daß die angestrebte Integration diesbezüglich flexibel sein muß.

5.1.2 Bewertung bekannter Ansätze

Für die Integration von Assoziationsregelverfahren mit relationalen Datenbanken lassen sich drei grundsätzliche Ansätze unterscheiden (vgl. u. a. [Sarawagi et al., 1998a], [Sarawagi et al., 1998b], [Nestorov und Tsur, 1999]). Die einfachste Lösung ist die Anbindung über Zusatzprogramme, welche die Aufbereitung des Exports einer Datenbank in proprietäre Algorithmenformate übernehmen. Neben dieser als *ungekoppelt (uncoupled)* bezeichneten Integration werden außerdem *lose gekoppelte (loosly coupled)* und *eng gekoppelte (tightly coupled)* Integrationen unterschieden. Bei der lose gekoppelten Integration haben die Verfahren über Mechanismen, die vom Datenbanksystem zur Verfügung gestellt werden, direkten Zugriff auf die Daten. Die eigentliche Regelgenerierung findet jedoch außerhalb der Datenbank statt. Bei der engen Koppelung hingegen werden die Verfahren selbst in die Datenbank transferiert. Beispielsweise erlaubt das Datenbanksystem DB2 von IBM die Ausführung von Funktionen im Adressraum der Datenbank in Form sogenannter

„stored procedures“ (vgl. z. B. [Chamberlin, 1998, S. 551ff.]). Das Datenbanksystem Oracle des gleichnamigen Anbieters ist mit einem vergleichbaren Mechanismus für sogenannte „PL/SQL-Prozeduren“ ausgestattet (vgl. u. a. [Urman, 1997], [Feuerstein und Pribyl, 2002]). Mit Hilfe solcher datenbankinterner Prozeduren und SQL können spezielle Varianten der Verfahren zur Assoziationsregelgenerierung implementiert (vgl. auch Abschnitt 3.2) und dann direkt im Adressraum des Datenbanksystems ausgeführt werden.

Ungekoppelte Lösungen sind aus den bereits dargestellten Gründen als nicht befriedigend anzusehen. Enge Koppelungen wurden bisher verschiedentlich beschrieben (vgl. u. a. [Agrawal und Shim, 1995], [Agrawal und Shim, 1996], [Sarawagi et al., 1998a], [Sarawagi et al., 1998b], [Rajamani und Cox, 1999]). Die dort eingeführten Verfahren zielen aber lediglich auf Aspekte der Effizienz und nicht des Datenzugriffs. Entsprechend werden von den im vorigen Abschnitt aufgestellten Anforderungen lediglich die Punkte 1 und 2 erfüllt. Die hinsichtlich des Datenzugriffs wesentlich umfassenderen Ansätze aus [Han et al., 1996b], [Meo et al., 1998] erfüllen zwar die aufgestellten Anforderungen, sind bezüglich der technischen Umsetzung aber nicht ausreichend konkretisiert. Eine Implementierung der Ansätze allein auf Basis der zugänglichen Literatur scheint daher nicht möglich. Erschwerend kommt hinzu, daß diese Ansätze nicht auf die Assoziationsregelgenerierung beschränkt sind, sondern mit dem Anspruch der Allgemeingültigkeit für eine Vielzahl von Data Mining-Verfahren entwickelt wurden und entsprechend komplex sind.

Auch seitens kommerzieller Softwareanbieter sind verschiedene Integrationslösungen erhältlich. Laut einer Umfrage in [Piatetsky-Shapiro, 2000] sind Clementine von SPSS, Enterprise Miner von SAS und Intelligent Miner von IBM mit Anteilen von 20%, 14% beziehungsweise 9% die meistgenutzten Data Mining-Werkzeuge.[1] Im Sinne von [Sarawagi et al., 1998a], [Nestorov und Tsur, 1999] implementieren diese kommerziellen Lösungen eine lose gekoppelte Integration, wobei keine der dieser Arbeit zugrundeliegenden Versionen[2] direkt die Punkte 4, 5 und 6 aus dem oben aufgestellten Anforderungskatalog erfüllt.

[1] Ein Überblick über die wichtigsten kommerziellen Data Mining-Werkzeuge ist beispielsweise in [Krahl et al., 1998, S. 564ff.], [Han und Kamber, 2001], [Klösgen und Zytkow, 2002, S. 539ff.] zu finden.

[2] Die dieser Arbeit zugrundeliegenden kommerziellen Data Mining-Werkzeuge sind SPSS Clementine in Version 5.1, SAS Enterprise Miner in Version 2.0 und IBM Intelligent Miner in Version 6.1.

5.1.3 Eigener Ansatz und Evaluierung

Der im folgenden vorgestellte Ansatz wurde mit dem Ziel entwickelt, eine effiziente Implementierung zu ermöglichen, welche die im vorigen Abschnitt aufgestellten Anforderungen erfüllt.

Grundidee

Assoziationsregelverfahren setzen die zu analysierenden Daten als zu Transaktionen aufbereitete Ereigniskombinationen voraus (vgl. Abschnitt 3.1). Die Generierung solcher Transaktionen erfordert die Kombination von Daten, die über die gesamte Datenbank verteilt sein können. Anstatt diese Funktionalität neu zu implementieren, wird im folgenden auf das die Daten speichernde relationale Datenbanksystem zurückgegriffen. Dabei soll die angestrebte Lösung allgemeingültig, das heißt unabhängig von einem konkreten Datenbankdesign oder Analyseszenario sein.

Die hier vorgeschlagene Lösung basiert darauf, daß der Analyst eine SQL-Abfrage an das Datenbanksystem absetzt, die sämtliche in die Analyse einzubeziehenden Attributwerte aus der Datenbank selektiert und entsprechend verknüpft. Die Formulierung solcher Anfragen ist in praktischen Anwendungen zumeist naheliegend (vgl. [Hipp et al., 2001b]). In Abbildung 5.2 ist ein Beispiel für eine vergleichswei-

```
select  vehicle_id, model_type,
        equipment_name, engine_type
from    vehicle as ve,
        ve_speq as vesp,
        special_equipment as sp,
        ve_en_pl as veen,
        engine_model as en
where   ve.prod_year = 2003 and
        ve.vehicle_id = vesp.vehicle_id and
        vesp.equip_id = sp.equip_id and
        ve.vehicle_id = veen.vehicle_id and
        en.engine_id = veen.engine_id
```

Abbildung 5.2: Beispiel für eine Anfrage auf QUIS

se komplexe Anfrage wiedergegeben. Die zugrundeliegende Analyse zielt darauf, Abhängigkeiten zwischen dem Modelltyp, dem Motortyp und den verbauten Sonderausstattungen für Fahrzeuge, die bisher im Jahr 2003 produziert wurden, in den Daten aus QUIS (vgl. Abbildung 5.1) zu entdecken.

Durch die Verknüpfung der verschiedenen Tabellen wird die Normalisierung teilweise rückgängig gemacht. Entsprechend enthält die aus einer solchen Anfrage resultierende Tabelle im allgemeinen einen großen Anteil redundanter Informationen. Da auf die Ergebnistabelle lediglich lesend zugegriffen wird, sind Konflikte im Sinne von Änderungs-, Lösch- oder Einfügeanomalien nicht relevant.

Als Ergebnistabelle der Anfrage aus Abbildung 5.2 könnte beispielsweise Tabelle 5.4 resultieren. Keines der hier betrachteten Data Mining-Werkzeuge kann mit

vehicle_ id	model_ type	equipment_ name	engine_ type	... ...
⋮	⋮	⋮	⋮	⋮
A23943CI	W202	AirCond.	D	...
A23943CI	W202	2nd Airbag	D	...
C33467CI	W202	Clutch	P	...
C33467CI	W202	2nd Airbag	P	...
C33467CI	W202	Hardtop	P	...
C45940CI	W220	AirCond.	P	...
⋮	⋮	⋮	⋮	⋮

Tabelle 5.4: Beispiel für die Ergebnismenge einer SQL-Anfrage zur Datenanalyse

einer derartigen Tabelle direkt umgehen, daß heißt das Ergebnis einer solchen Anfrage ist mit den kommerziellen Werkzeugen nicht ohne weiteres analysierbar.

Um obige Tabelle zu Transaktionen aufzubereiten, wird diese zunächst nach ihrem Schlüssel vehicle_id geordnet, so daß Zeilen mit gleichen Schlüsselwerten aufeinander folgen. Durch das Anfügen von order by vehicle_id an die SQL-Anfrage wird dies durch eine vom Datenbanksystem ausgeführte Sortierung erreicht. In der Ergebnismenge sind sowohl Attribute enthalten, die für jedes Objekt der realen Welt genau einen Wert aufweisen, beispielsweise der Motortyp engine_type, als auch Attribute, die jedes Objekt durch eine beliebige Menge von Attributwerten beschreiben, wie beispielsweise die Ausstattungen im Attribut equipment_name.

Von solchen sortierten Ergebnistabellen werden Transaktionen abgeleitet, indem die Zeilen der Ergebnismenge nacheinander durchlaufen werden. Wechselt der Schlüsselwert, dann ist die aktuelle Transaktion vollständig erfaßt, und mit der Erzeugung der nächsten Transaktion kann fortgefahren werden. Mehrfach vorkommende Werte dürfen jeweils nur einmal in jede Transaktion eingetragen werden.

Bei dem hier beschriebenen Ansatz entscheidet die Wahl des Schlüsselattributs darüber, bezüglich welchen Objekttyps die Regeln generiert werden. Wird beispielsweise model_type statt vehicle_id als Schlüssel gewählt, werden dadurch anstelle der Fahrzeuge die verschiedenen Modelltypen zu den Objekten der Analyse.

Umsetzung

Für eine Umsetzung ist zunächst zu entscheiden, ob die Integration lose oder eng gekoppelt implementiert werden soll. Von den verschiedenen Alternativen, die in [Sarawagi et al., 1998a], [Sarawagi et al., 1998b] untersucht werden, ist bezüglich der Effizienz der sogenannte Cache Mine-Ansatz am vielversprechendsten. Dieser basiert auf einem Cache, der aus der Datenbank initialisiert wird und dann den Regelgenerierungsverfahren, die außerhalb des Datenbanksystems ausgeführt werden, zur Verfügung steht. Die lose gekoppelte Integration greift lediglich einmal auf die Datenbank zu. Im Rahmen der Regelgenerierung nachfolgende Iterationen über die Transaktionen (vgl. Abschnitt 3.1) werden effizient ohne Rückgriff auf die Datenbank anhand des Caches bearbeitet.

Die in der vorliegenden Arbeit vorgeschlagene Implementierung erweitert den Cache Mine-Ansatz, so daß dieser die oben beschriebenen Ergebnistabellen verarbeiten kann. Der Zugriff auf das Datenbanksystem DB2 von IBM erfolgt über das CLI (vgl. [Chamberlin, 1998, S. 462ff.]).[3] Die vom Benutzer erstellte SQL-Anfrage wird zur Ausführung an das Datenbanksystem übergeben. Die resultierende Ergebnistabelle wird direkt in das in Abschnitt 3.1.4 eingeführte Binärformat transformiert und in dieser kompakten Form im Dateisystem abgelegt. Wie oben beschrieben werden dazu die Zeilen der Ergebnismenge sukzessive durchlaufen und anhand der Schlüsselwerte die Grenzen zwischen den einzelnen Transaktionen bestimmt. Jeder vorgefundene Attributwert wird als eine Zeichenkette interpretiert und mit dem Spaltennamen als Präfix versehen. Die Ergänzung um den Spaltennamen ist notwendig, da in der Regel Attributwerte nur in Verbindung mit einer Spalte eindeutig sind.[4] Die resultierenden Zeichenketten werden eineindeutig auf ganzzahlige Werte abgebildet. Bevor eine Transaktion in das Dateisystem geschrieben wird, werden die enthaltenen Ereignisse aufsteigend sortiert und eventuelle Dubletten entfernt.

Evaluierung

Eine Implementierung der oben beschriebenen Integrationslösung in C++ wird im folgenden auf einer SUN Ultra-10 Workstation unter dem Betriebssystem Solaris 2.7 sowie Version 6.1 des Datenbanksystems IBM DB2 gemeinsam mit kommerziellen Werkzeugen evaluiert. Datenbasis sind die Sonderausstattungen von

[3] In verschiedenen Experimenten konnten keine bemerkenswerten Laufzeitunterschiede zwischen einer Implementierung der Datenextraktion als „Stored Procedure“, die im Adressraum des Datenbanksystems ausgeführt wird (vgl. [Chamberlin, 1998, S. 561ff.]), und einer außerhalb des Datenbanksystems ausgeführten Prozedur, die über das CLI auf die Daten zugreift (vgl. [Chamberlin, 1998, S. 551ff.]), ermittelt werden.

[4] Beispielsweise kann der Attributwert „schwarz“ sowohl die Lackierung eines Fahrzeugs beschreiben als auch für die Polsterfarbe stehen.

Fahrzeugen aus QUIS, die bereits in Abschnitt 3.3 für die Evaluierung der Datengenerierungsverfahren herangezogen wurden. Die selektierte Tabelle besteht aus ungefähr 200 Millionen Zeilen. Im Rahmen der Evaluierung wurde außerdem die maximale Mächtigkeit der erzeugten häufigen Ereigniskombinationen auf drei Ereignisse beschränkt und als Schwellenwert für die Mindesthäufigkeit minsupp = 2% gewählt.

Für die Evaluierung stehen zwei der in Abschnitt 5.1 genannten, kommerziellen Data Mining-Werkzeuge auf der Solaris-Plattform zur Verfügung. Eines der beiden Werkzeuge überschreitet allerdings bereits bei einigen hunderttausend Zeilen seine Kapazität und bricht die Analyse ab. Das andere Werkzeug und die im Rahmen dieser Arbeit vorgestellte Implementierung skalieren wie erwartet annähernd linear mit der Anzahl der Zeilen (vgl. auch Abschnitt 3.3).

Die Ergebnisse der Evaluierung sind in Abbildung 5.3. wiedergegeben. Die An-

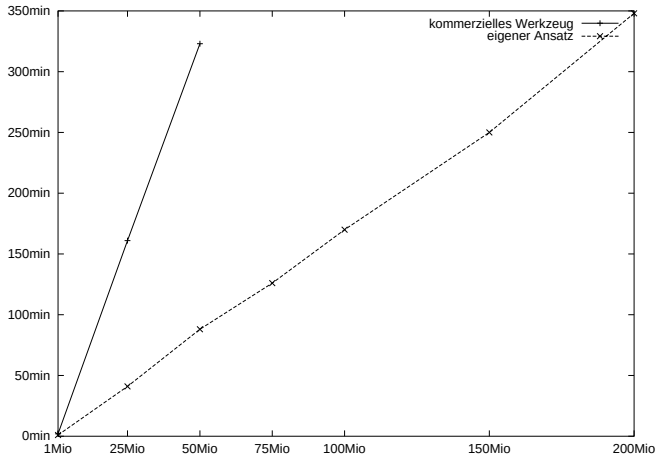


Abbildung 5.3: Evaluierung auf Sonderausstattungen von Fahrzeugen mit variierender Anzahl von Zeilen in Millionen und Laufzeit in Minuten

zahl der in die Generierung einbezogenen Zeilen wird zwischen 1 Million und 200 Millionen variiert, und die zugehörigen Laufzeiten sind in Minuten angegeben. Offensichtlich ist der hier vorgestellte Ansatz bezüglich der Laufzeiten dem kommerziellen Werkzeug überlegen, wobei die Messungen nur einen ersten Eindruck vermitteln können. Für ein abschließendes Urteil ist eine wesentlich umfassendere Evaluierung notwendig. Im Rahmen dieser Arbeit soll als Ergebnis genügen, daß der hier vorgestellte Ansatz mit kommerziellen Werkzeugen bezüglich der Laufzeit grundsätzlich vergleichbar ist. Der entscheidende Vorteil gegenüber den kommerziellen Lösungen ist jedoch, daß mit dem neu eingeführten Ansatz die in Abschnitt 5.1.1 aufgestellten Anforderungen an eine Integration von Assoziationsregelverfahren mit relationalen Datenbanksystemen erfüllt werden.

5.2 Effizientes Retrieval von Assoziationsregeln

Die Anbindung an relationale Datenbanksysteme ist nur ein Aspekt der Integration der Assoziationsregelverfahren in den Wissensentdeckungsprozeß. Wie in Abschnitt 2.1 beschrieben, hat der Prozeß der Wissensentdeckung einen hochgradig iterativen und interaktiven Charakter. Der Analyst als „human in the loop“ (vgl. [Brachman und Anand, 1996]) entscheidet auf Basis der augenblicklichen Ergebnisse, ob zu einer früheren Phase zurückgekehrt, die aktuelle Phase wiederholt oder mit einer der folgenden Phasen fortgefahren wird.

Ein auf Assoziationsregeln, die aus einem vorgelagerten Generierungslauf stammen, basierender Ansatz wurde in [Imielinski und Mannila, 1996], [Imielinski et al., 1999], [Imielinski und Virmani, 1999] erstmals beschrieben. Im folgenden wird diese Grundidee in den Kontext des Wissensentdeckungsprozesses übertragen. Dazu wird der Ansatz insbesondere durch die Einführung eines expliziten *Regelcaches (rule cache)* aufgewertet. Dieser Cache wird mit einer erweiterten Anfragesprache als Schnittstelle ergänzt und eine Implementierung auf Basis eines relationalen Datenbanksystems vorgestellt. Die kurzen Antwortzeiten für Anfragen, die aus dem Cache beantwortet werden können, ermöglichen dem Analysten ein interaktives Arbeiten, unabhängig vom Volumen der zugrundeliegenden Daten. Die erweiterte Anfragesprache unterstützt ihn zugleich adäquat bei der Formulierung seiner Analysehypothesen. Vergleichbare, aber weit weniger mächtige Ansätze beruhen auf dem sogenannten „rule browsing“, daß heißt dem interaktiven und benutzergesteuerten Durchsuchen von Regelmengen, unter anderem anhand vorgegebener *Schablonen (templates)* (vgl. z. B. [Klemettinen et al., 1996], [Liu et al., 2000]).[5]

5.2.1 Eigener Ansatz versus fokussierte Wissensentdeckung

Heutige Ansätze zielen vor allem darauf, die Effizienz der Regelgenerierung zu steigern. Trotz signifikanter Fortschritte sind die etablierten Verfahren, insbesondere für vergleichsweise große Datenmengen, jedoch nicht ausreichend effizient, um einem Analysten interaktives Arbeiten auch nur annähernd zu ermöglichen (vgl. u. a. Kapitel 3).

Als ein Ansatz zur Lösung des Laufzeitproblems tritt zunehmend die sogenannte *fokussierte Wissensentdeckung (constrained mining)*[6] in den Mittelpunkt des Forschungsinteresses (vgl. z. B. [Srikant et al., 1997], [Ng et al., 1998], [Bayardo, 2002],

[5] In [Nag et al., 1999], [Goethals und Bussche, 2000] sind Ansätze beschrieben, die einen Wissensentdeckungsprozeß begleitend die häufigen Ereigniskombinationen und teilweise ebenfalls die generierten Regeln in einem Cache ablegen. Dieser Cache wird sukzessive mit dem Ziel aufgebaut, die Algorithmen während der Generierung häufiger Ereigniskombinationen zu unterstützen. Der Analyst hat weder direkt noch indirekt unmittelbaren Zugriff auf den Cache.

[6] „Constrained“ steht in diesem Zusammenhang für die gezielte Einschränkung der Analysen, wird also im Sinne einer fokussierten Wissensentdeckung gebraucht.

[Pei und Han, 2002], [Leung et al., 2002]). Die fokussierte Wissensentdeckung basiert darauf, daß die zu erzeugenden Regeln bereits vor der eigentlichen Regelgenerierung sinnvoll eingeschränkt werden und diese Einschränkung während der Regelgenerierung dazu genutzt werden kann, die Effizienz der Verfahren zu steigern. Typische Beschränkungen bestehen aus Bedingungen für die An- oder Abwesenheit von Ereignissen in Prämisse beziehungsweise Konklusion der Regeln (vgl. u. a. [Srikant et al., 1997], [Lakshmanan et al., 1998], [Ng et al., 1999], [Goethals und Bussche, 2000]).

Beschränkungen dieser Art sind außerdem geeignet, die oft sehr großen Regelmengen zu filtern und dadurch dem Analysten einfacher zugänglich zu machen. Da außerdem Verfahren zur Verfügung stehen, welche die Beschränkungen zur Laufzeitoptimierung nutzen können, scheint die fokussierte Wissensentdeckung als ein Schlüssel zur Verwirklichung des interaktiven und iterativen Wissensentdeckungsprozesses. Zum einen sind die erreichten Laufzeiten (vgl. u. a. [Srikant et al., 1997], [Ng et al., 1999], [Pei und Han, 2002], [Leung et al., 2002]) jedoch noch weit von den tatsächlichen Anforderungen entfernt. Zum anderen ist die fokussierte Wissensentdeckung grundsätzlich kritisch zu sehen, da die Assoziationsregelgenerierung darauf zielt, neues, bisher unvermutetes Wissen in Daten zu entdecken. Die Ergebnismenge der Regelgenerierung bereits im voraus einzuschränken, steht diesem Ziel grundsätzlich entgegen. Somit ist die Fokussierung einerseits ein hilfreiches Mittel, Ergebnismengen sinnvoll zu beschränken, andererseits birgt die Fokussierung die Gefahr, die Wissensentdeckung auf ein Überprüfen von Hypothesen zu reduzieren (vgl. auch [Hipp und Güntzer, 2002]).

Insbesondere während der initialen Iterationen eines Wissensentdeckungsprozesses ist eine Fokussierung kritisch. Der Analyst soll in dieser Orientierungsphase einen ersten allgemeinen Eindruck von den Daten erhalten und nicht aufgrund subjektiver Einschätzungen die Analyse frühzeitig in bestimmte Bahnen lenken. Zwar werden für die Gütemaße üblicherweise relativ hohe Schwellenwerte gewählt, um nicht unübersichtlich viele Regeln zu erzeugen, aber eine Beschränkung der generierten Regeln auf bestimmte Ereignisse oder Ereigniskombinationen wird möglichst vermieden. Das Ziel dieser ersten Phase ist nicht, bereits konkrete Ergebnisse zu erzeugen, sondern vielversprechende Ansatzpunkte für tiefergehende, das heißt fokussierte Analysen zu identifizieren.

Vor diesem Hintergrund setzt der im folgenden vorgestellte Ansatz bewußt nicht auf fokussierte Regelgenerierungsverfahren, da diese für die initialen Phasen der Wissensentdeckung wenig geeignet sind und auch für die späteren Phasen die Laufzeitprobleme nicht zufriedenstellend lösen können (vgl. auch [Hipp et al., 2002a], [Hipp et al., 2002b], [Hipp und Güntzer, 2002]). Im Gegensatz zu der heute populären Fokussierung (vgl. z. B. [Bayardo, 2002]) verfolgt der nachstehend beschriebene Ansatz die konträre Idee, die Regelgenerierung möglichst von jeglichen

Einschränkungen zu befreien und die Fokussierung als nachgelagerte Filterung umzusetzen.

Ausgangspunkt ist ein einzelner, weitgehend unbeschränkter Regelgenerierungslauf. Dieser Lauf dient dazu, einen Regelcache zu initialisieren, wobei gegebenenfalls eine vergleichsweise lange Unterbrechung des Wissensentdeckungsprozesses resultiert. Der initialisierte Regelcache kann den Analysten sowohl in der Orientierungsphase unterstützen, als auch tiefergehende, das heißt fokussierte Analyseanfragen beantworten, wobei dazu in der Regel nicht erneut auf die zugrundeliegenden Daten zurückgegriffen werden muß. Der Effekt der vergleichsweise teuren initialen Regelgenerierung ist Interaktivität für die nachfolgenden Iterationen des Wissensentdeckungsprozesses, da diese unabhängig vom Volumen der zugrundeliegenden Daten auf Basis des Caches bearbeitet werden können. Dieser Ansatz hat den Vorteil, daß viele kürzere Unterbrechungen zu einer initialen Regelgenerierung zusammengefaßt werden und damit die Arbeit des Analysten lediglich einmal unterbrochen werden muß. Außerdem ist die für die initiale Regelgenerierung benötigte Zeit oft kürzer als die Summe der Zeiten für die alternativ notwendigen fokussierten Regelgenerierungsläufe (vgl. u. a. die in Abschnitt 5.2.3 folgende Evaluierung).

5.2.2 Retrievalsprache

Grundlegend für den oben eingeführten Ansatz ist der Zugriff auf die gespeicherten Regeln durch den Analysten. Dieser Zugriff muß nicht nur effizient bezüglich der Antwortzeiten sein, sondern vor allem den Analysten bei der Regelauswahl angemessen unterstützen, das heißt die Fokussierung als nachgelagerte Filterung geeignet umsetzen. In der Literatur werden verschiedene Ansätze zur Auswahl von Regeln beschrieben, die auch im Zusammenhang mit dem hier vorgestellten Regelcache Verwendung finden können. Diese Ansätze reichen von einfachen Schablonen (vgl. u. a. [Klemettinen et al., 1994], [Fu und Han, 1995], [Kamber et al., 1997a], [Kamber et al., 1997b], [Liu et al., 2000]) bis zu mächtigen Anfragesprachen (vgl. u. a. [Han et al., 1996b], [Han et al., 1996a], [Meo et al., 1996], [Imielinski und Virmani, 1999], [Boulicaut, 1999], [Boulicaut et al., 1999]). Teilweise beinhalten diese Anfragesprachen auch Aspekte des Datenzugriffs und der Datenaufbereitung, welche für den hier vorgestellten Ansatz bereits in Abschnitt 5.1 mit der Anbindung an relationale Datenbanksysteme abgedeckt wurden.

Der im folgenden vorgestellte Ansatz geht über die bekannten Anfragesprachen hinaus und basiert auf einer Erweiterung des in Abschnitt 2.2 eingeführten Assoziationsmodells. Ziel dieses Ansatzes ist nicht, die bekannten Anfragesprachen zu ersetzen, sondern vielmehr diese um nützliche und durch praktischen Anwendungen motivierte Erweiterungen sinnvoll zu ergänzen. Zur Veranschaulichung wird

im folgenden eine einfache Anfragesprache eingeführt, anhand derer durch verschiedene Anwendungsbeispiele die grundlegenden Konzepte und Möglichkeiten des erweiterten Assoziationsmodells für das Retrieval von Regeln aus einem Regelcache verdeutlicht werden.

Erweiterung

In Abschnitt 2.2 wurden Ereignisse als Literale eingeführt. Dieses Verständnis der Ereignisse ist grundlegend für die Assoziationsregelgenerierung (vgl. z. B. [Agrawal et al., 1993], [Agrawal und Srikant, 1994a], [Hipp et al., 2000a]). Obwohl die Regelgenerierung selbst die Ereignisse als Literale behandelt, weisen diese in praktischen Anwendungen jedoch meist eine Struktur auf, und insbesondere für das Retrieval von Assoziationsregeln aus einem Regelcache stellen diese Zusatzinformationen ein mächtiges Hilfsmittel dar.

Zunächst gilt es, die Struktur der Ereignisse zugänglich zu machen. So können beispielsweise den Artikeln aus dem Warensortiment eines Einzelhandelsunternehmens Verkaufspreise oder Deckungsbeiträge zugeordnet sein. Entsprechend können Produktionsdaten, Herstellerinformationen oder ähnliches zu Fahrzeugteilen bekannt sein und zur Formulierung von Anfragen an einen Regelcache genutzt werden. Solche domänenspezifischen Attribute können auch mittels Taxonomien (vgl. Abschnitt 2.2.3) oder quantitativen Assoziationsregelverfahren (vgl. Abschnitt 2.2.4) in die Regelgenerierung eingebracht werden. Hier in dieser Arbeit wird jedoch ein neuer Ansatz eingeführt. Anstatt die generierten Regeln zu erweitern, wird die Struktur der Ereignisse dazu herangezogen, den Zugriff auf den Regelcache für den Analysten geeigneter zu gestalten.

Formal wird das in Abschnitt 2.2 eingeführte Assoziationsmodell wie folgt erweitert: Sei $\mathcal{E} \subseteq \mathbb{N} \times \mathbb{A}_1 \times \cdots \times \mathbb{A}_m$ eine Menge von Ereignissen. Jedes Ereignis wird eindeutig durch einen Schlüssel $S \in \mathbb{N}$ identifiziert und außerdem durch weitere Attribute $(a_1, \ldots, a_m) \in \mathbb{A}_1 \times \cdots \times \mathbb{A}_m$ beschrieben. Die einzelnen Attribute können anwendungsabhängig aus verschiedenen Domänen stammen. So kann es sich beispielsweise um Preise, Kosten oder Zeitstempel handeln. Basierend auf dieser Erweiterung können mit den üblichen Verfahren Assoziationsregeln generiert werden, indem die Ereignisse über ihre Schlüsselwerte identifiziert und als Literale behandelt werden. Zugleich stehen für das Retrieval von Regeln aus einem Regelcache die entsprechenden Zusatzinformationen für die Formulierung von Anfragen zur Verfügung.

Retrieval von Regeln

Die im folgenden vorgestellte einfache Anfragesprache für das Retrieval von Assoziationsregeln aus einem Regelcache demonstriert das Potential der oben beschriebenen Erweiterung. Diese Anfragesprache ist bereits ein mächtiges Werkzeug in den Händen des Analysten, kann aber auch als Ergänzung zu umfassenden Analysesystemen oder etablierten Anfragesprachen verstanden werden. Die Anfragesprache wird im folgenden anhand verschiedener Beispiele aus dem Umfeld der QUIS-Datenbank (vgl. Abschnitt 5.1) eingeführt.

Jede Anfrage an den Regelcache besteht aus dem Schlüsselwort SelectRulesFrom, gefolgt vom Namen des abzufragenden Regelcaches sowie einer sogenannten Where-Bedingung, welche die zurückgelieferten Regeln einschränkt. Die einfachste Anfrage selektiert die Regeln anhand minimaler Schwellenwerte für die Gütemaße. Beispielsweise ist die Ergebnismenge der folgenden Anfrage an einen Cache rulecache auf die Regeln beschränkt, die für die Konfidenz einen Wert von mehr als 75% aufweisen und für das Maß Lift mindestens den Wert 10 erreichen:

```
SelectRulesFrom rulecache                                    (Anfrage 1)
Where conf > 0.75 and lift >= 10;
```

Oft sind es einzelne Ereignisse oder Gruppen von Ereignissen, anhand deren Vorkommen beziehungsweise Nichtvorkommen in Prämisse oder Konklusion die Regelmengen weiter eingeschränkt werden. Beispielsweise kann der Analyst an solchen Regeln interessiert sein, welche die Ausstattung eines Fahrzeugs mit einem Fahrerairbag anhand anderer gewählter Ausstattungen erklären. Die zurückgelieferten Regeln sollen in diesem Fall das Ereignis Airbag in der Regelkonklusion enthalten, die im folgenden kurz als head bezeichnet wird:

```
SelectRulesFrom rulecache                                    (Anfrage 2)
Where 'Airbag' in head and conf > 0.75 and lift >= 10;
```

Wenn beispielsweise bekannt ist, daß ein Beifahrerairbag CoAirbag stets einen Fahrerairbag impliziert, dann ist es sinnvoll, durch die Erweiterung der Where-Bedingung mit not 'CoAirbag' in body, wobei body die Regelprämisse bezeichnet, in Anfrage 2 triviale Regeln in der Ergebnismenge im voraus auszuschließen.

Die soeben anhand von Beispielen skizzierte Anfragesprache ist vergleichsweise einfach und trotzdem bereits geeignet, den Analysten vielfältig zu unterstützen (vgl. [Hipp et al., 2002a], [Hipp et al., 2002b]). Als erste Erweiterung gegenüber den etablierten Anfragesprachen werden Aggregatfunktionen auf den Regelgütemaßen eingeführt. So ist es in vielen Szenarien hilfreich, Schwellenwerte nicht absolut anzugeben, sondern relativ zu den von den Regeln im Regelcache erreichten Werten. Beispielsweise selektiert die folgende Anfrage Regeln, die einen im Verhältnis zur

Gesamtheit der erzeugten Regeln vergleichsweise geringen Support, weniger als 1% größer als der geringste im Regelcache vorgefundene Support, und gleichzeitig einen vergleichsweise hohen Wert für die Konfidenz, mehr als 99% der höchsten im Regelcache zu findenden Konfidenz, aufweisen. Der kleinste Support beziehungsweise die größte im Regelcache abgelegte Konfidenz wird mit min(supp) beziehungsweise max(conf) bezeichnet und umgekehrt:[7]

```
SelectRulesFrom rulecache                                    (Anfrage 3)
Where supp < 1.01*min(supp) and conf > 0.99*max(conf);
```

Die zweite Erweiterung gegenüber den etablierten Ansätzen nutzt die oben eingeführte Zugriffsmöglichkeit auf die Struktur der Ereignisse. Sei x.attname der Wert des Attributs attname für das Ereignis x. So gilt beispielsweise x.type = 'SpEquip' für alle Ereignisse x, die eine Sonderausstattung kodieren. Damit können unter anderem die Regeln selektiert werden, die bestimmte Vorgaben für die Gütemaße erfüllen und zugleich Sonderausstattungen erklären, denen jeweils Kosten von mindestens c zugerechnet werden können. Solche Anfragen werden durch die Einführung von Existenz- und Allquantoren auf den Attributwerten der Ereignisse möglich, beispielsweise:

```
SelectRulesFrom rulecache                                    (Anfrage 4)
Where supp > 0.25 and conf > 0.975
and exists x in head (x.type = 'SpEquip' and x.costs > c );
```

Eine etwas komplexere, aber gleichwohl sehr nützliche Anfrage ist die Selektion der Regeln, die mindestens eine Sonderausstattung in der Konklusion enthalten, die von einem Hersteller, Attribut manu, stammt, der auch mindestens eine der Sonderausstattungen aus der Prämisse produziert hat:

```
SelectRulesFrom rulecache                                    (Anfrage 5)
Where exists x in head (x.type = 'SpEquip' and
exists y in body (y.type = 'SpEquip' and x.manu = y.manu))
and supp > 0.25 and conf > 0.975;
```

Die hier erstmals für das Retrieval von Regeln eingeführten Quantoren in Verbindung mit der Erweiterung des Assoziationsmodells um Attribute erlauben auf intuitive Weise die Formulierung von in praktischen Anwendungen sehr hilfreichen Anfragen (vgl. auch [Hipp et al., 2002a], [Hipp et al., 2002b]).

[7] Andere Aggregatfunktionen, wie beispielsweise das arithmetische Mittel, sind ebenfalls sinnvolle Ergänzungen.

5.2.3 Umsetzung und Evaluierung

Im folgenden wird eine prototypische Implementierung eines Regelcaches sowie der soeben eingeführten Retrievalsprache vorgestellt und anhand einer praktischen Anwendung evaluiert.

Umsetzung

Für die Umsetzung des Regelcaches wird keine spezielle Datenstruktur implementiert, sondern es wird auf ein konventionelles relationales Datenbanksystem zurückgegriffen. Zum einen ist die Speicherung der Regeln in einer Datenbank naheliegend, da die Regeln typischerweise mit hohem und teilweise manuellem Aufwand erzeugt werden (vgl. u. a. Kapitel 2) und demzufolge als entsprechend wertvoll anzusehen sind. Zum anderen profitiert die Implementierung des Systems direkt von der Verwendung des Datenbanksystems. Anstatt die Anfragelogik vollständig neu zu implementieren, genügt es, die Anfragen für das Retrieval von Regeln in SQL zu übersetzen und diese dann direkt auf dem Datenbanksystem auszuführen.

Eine wichtige Frage im Zusammenhang mit der Implementierung des Caches ist, ob anstelle der Regeln nicht häufige Ereigniskombinationen gespeichert werden sollten, aus denen die Regeln bei Bedarf abgeleitet werden können (vgl. dazu Abschnitt 3.1.1). Für die Implementierung eines Regelcaches, so wie er oben eingeführt wurde, wird diese Entscheidung in der vorliegenden Arbeit zugunsten der Speicherung der Regeln getroffen. Hintergrund ist, daß der Support in der Praxis erfahrungsgemäß weniger im Sinne eines Gütemaßes genutzt wird, sondern als eine der wenigen Möglichkeiten, um die Antwortzeiten der Verfahren zur Regelgenerierung auf im Rahmen des Wissensentdeckungsprozesses tolerierbare Werte zu verkürzen (vgl. auch [Hipp und Güntzer, 2002]). Die Einführung des Regelcaches entbindet den Analysten jedoch weitgehend von den Restriktionen durch die Laufzeiten der Regelgenerierungsverfahren. Entsprechend wird der Analyst vergleichsweise niedrige Schwellenwerte für die minimalen Häufigkeiten ansetzen und gleichzeitig die Regeln anhand weiterer Gütemaße strenger filtern, um aussagekräftige Regeln zu erhalten. Als Folge nähern sich die Anzahl gespeicherter Regeln und die Anzahl alternativ zu speichernder häufiger Ereigniskombinationen einander an, da letztere lediglich anhand der minimalen Häufigkeit eingeschränkt werden können, nicht aber wie die Regeln auch anhand der weitereren Gütemaße (vgl. Abschnitt 3.1.1 und [Brin et al., 1997a]). Damit ist die Speicherersparnis als Argument für das Ablegen von häufigen Ereigniskombinationen anstelle von Regeln nicht mehr gültig. Außerdem wird die Generierung von Regeln aus häufigen Ereigniskombinationen bei entsprechend kleinem Schwellenwert für den Support vergleichsweise aufwendig, da die große Anzahl häufiger Ereigniskombinationen eine weit größere Anzahl von Regeln impliziert, die während der Regelerzeugung

bezüglich der verschiedenen Gütemaße auf das Erreichen der Schwellenwerte überprüft werden müssen (vgl. auch Abschnitt 3.1.1).

Jede zu speichernde Regel besteht aus Regelprämisse, Regelkonklusion und einem Vektor mit den Ausprägungen der verschiedenen Gütemaße. Eine häufige Ereigniskombination kann in mehreren Regeln als Prämisse wie auch als Konklusion vorkommen. Zur Vermeidung von Redundanz werden die Ereigniskombinationen in einer Tabelle separat von den gespeicherten Regeln gehalten, wobei jede Ereigniskombination genau einmal abgelegt wird, sobald sie in mindestens einer der Regeln vorkommt. Die Regeltabelle enthält lediglich Referenzen auf die häufigen Ereigniskombinationen. Dadurch kann Speicher in erheblichem Umfang eingespart werden. So sinkt der Speicherbedarf durch die Vermeidung redundanter Einträge für die verschiedenen in Abschnitt 3.3.1 eingeführten synthetischen Datensätze um 50% bis 90% des ursprünglichen Bedarfs (vgl. [Hipp et al., 2002b]).

Die in Abbildung 5.4 wiedergegebenen Tabellendefinitionen implementieren den

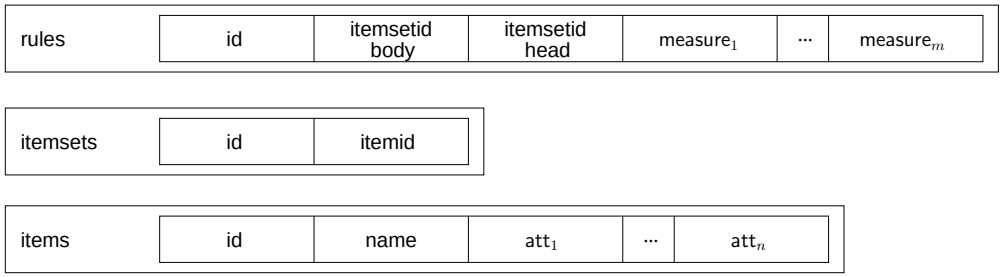


Abbildung 5.4: Datenbanktabellen für die Implementierung des Regelcaches

oben beschriebenen Regelcache. Zu jeder Regel werden in der Regeltabelle rules ein Regelschlüssel id und zwei Fremdschlüssel itemsetidbody und itemsetidhead für die Prämisse beziehungsweise die Konklusion der Regel abgelegt. Die Attribute measure$_1$, ..., measure$_m$ speichern die zugehörigen Werte für die Gütemaße als Fließkommazahlen. Die in der Tabelle itemsets gespeicherten Ereigniskombinationen bestehen aus einem Schlüssel id, der die jeweilige Ereigniskombination identifiziert, und einem Fremdschlüssel itemid in die Tabelle items, welche Informationen zu den einzelnen Ereignissen enthält. Die Tabelle items speichert die Bezeichnung der Ereignisse als Attribut name. Außerdem werden dort applikationsabhängige Attributwerte att$_1$, ..., att$_n$ abgelegt, welche die Ereignisse näher beschreiben.

Für den Zugriff auf den Regelcache genügt es, die in der oben dargestellten Retrievalsprache formulierten Anfragen entsprechend der relationalen Umsetzung des Caches in SQL zu übersetzen und diese dann auf der Datenbank auszuführen.[8] Als

[8] Für die Implementierung wurde die Programmiersprache C++ verwendet.

Beispiel ist in Abbildung 5.5 eine Übersetzung von Anfrage 4 aus dem vorherigen

```
CREATE VIEW extended_itemsets (itemsetid, itemid, type, costs)
AS (    SELECT itemsets.id, itemid, type, costs
        FROM itemsets INNER JOIN items
        ON itemsets.itemid = items.id )

SELECT * FROM rules
WHERE  supp > 0.25 AND conf > 0.975
AND    itemsetidhead in (
        SELECT itemsetid FROM extended_itemsets
        WHERE type = 'SpEquip' and costs > c )
```

Abbildung 5.5: SQL-Übersetzung von Anfrage 4 aus Abschnitt 5.2.2

Abschnitt wiedergegeben.[9]

Evaluierung

Für die Evaluierung der beschriebenen Implementierung des Regelcaches wird exemplarisch die in Abschnitt 5.1 vorgestellte QUIS-Datenbank herangezogen. Gegenstand der Analyse sind die in den Fahrzeugen verbauten Sonderausstattungen zusammen mit weiteren Attributen, wie dem Fahrzeugmodell oder dem Produktionsdatum. Die der Evaluierung zugrundeliegende Transaktionsmenge umfaßt ungefähr 10 Millionen Fahrzeuge. Die Erzeugung der Assoziationsregeln benötigt mehr als 4 Stunden auf der in Abschnitt 5.1.3 beschriebenen SUN Workstation.[10] Offensichtlich bedeutet in diesem Szenario jeder erneute Regelgenerierungslauf eine Unterbrechung der Analyse, die einen Prozeß im Sinne der interaktiven Wissensentdeckung in Datenbanken aus Abschnitt 2.1 dieser Arbeit unmöglich macht.

Für die folgende Evaluierung werden auf Basis der Transaktionen und verschiedener minimaler Schwellenwerte für den Support vier separate Regelcaches initialisiert. Die Regeln werden außerdem auf maximal drei Ereignisse in der Prämisse

[9] Die Anzahl und die Typinformationen der Attribute, die in Tabelle items die Ereignisse näher beschreiben, sind anwendungsabhängig und daher in der Implementierung des Regelcaches nicht fest kodiert. Während der Übersetzung der Applikation durch den Compiler sowie später während der Übersetzung der Retrievalanfragen müssen die Datentypen noch nicht bekannt zu sein. Erst sobald eine übersetzte Anfrage an die Datenbank zur Ausführung weitergegeben wird, müssen die Attributtypen zur Verfügung stehen.

[10] Wie in Abschnitt 5.1.3 beobachtet, wird die Laufzeit der Regelgenerierung vom Datenbankzugriff dominiert.

und ein Ereignis in der Konklusion beschränkt.[11] Die Caches enthalten 10.000, 100.000, 200.000 beziehungsweise 300.000 verschiedene Regeln. Die Schwellenwerte für die Gütemaße in den Anfragen aus dem vorherigen Abschnitt werden angepaßt, so daß für jede der Anfragen maximal 1.000 Regeln zurückgegeben werden. Die Where-Bedingung aus Anfrage 2 basiert auf einem Ereignis, das jeder dritten Transaktion zugewiesen wurde. Für Anfrage 5 wurden außerdem fünf verschiedene Attributwerte gleichmäßig auf die Sonderausstattungen verteilt.

In Abbildung 5.6 sind die Antwortzeiten für die verschiedenen Anfragen wiederge-

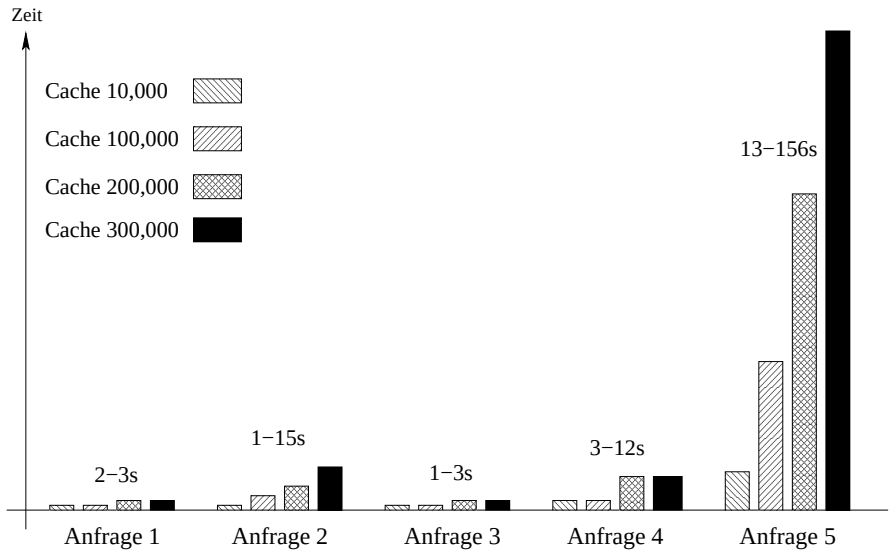


Abbildung 5.6: Antwortzeiten für fünf verschiedene Anfragen an vier verschiedene Regelcaches

geben. Diesen Zeiten liegt die beschriebene Implementierung zugrunde, wobei für die Laufzeitmessungen das Datenbanksystem IBM DB2 V7.1 auf einem 500MHz Pentium III Linuxsystem in Version 2.4.18 zum Einsatz kam.

Die erreichten Laufzeiten von einzelnen Sekunden bis zu wenigen Minuten ermöglichen das interaktive Arbeiten mit den im Cache gespeicherten Regeln. Insbesondere ist zu erwähnen, daß die Antwortzeiten nicht von der Anzahl zugrundeliegender Transaktionen abhängen. Während die Regelgenerierung nahezu linear mit der Anzahl Transaktionen skaliert (vgl. Abschnitt 3.3), sind die Laufzeiten für das Regelretrieval lediglich von der Anzahl im Cache gespeicherter Regeln abhängig. Die Erfahrungen aus verschiedenen Anwendungen (vgl. z. B. [Hipp und Lindner, 1999], [Hipp et al., 2001a], [Hipp et al., 2002b]) zeigen, daß typischerweise eine wachsende

[11] Regeln mit mehr als drei Ereignissen in der Prämisse oder mit mehr als einem Ereignis in der Konklusion werden für das angenommene Analyseszenario als wenig sinnvoll erachtet.

Anzahl Transaktionen keine wachsende Anzahl Regeln impliziert, konstante relative Schwellenwerte für die Gütemaße vorausgesetzt. Demzufolge ist das Retrieval von Regeln aus einem Cache, der aus mehreren Millionen Transaktionen erzeugt wurde, gewöhnlich kaum aufwendiger, als wenn dieser lediglich auf einer Teilmenge von einigen tausend Transaktionen basiert, da sich in praktischen Anwendungen die Anzahl der im Cache gespeicherten Regeln kaum unterscheidet.

5.3 Datenselektion als Ergebnisnachbearbeitung

Während die im vorherigen Abschnitt dem Retrieval von Assoziationsregeln zugrundegelegten Anfragen sich vergleichsweise einfach als Filter implementieren lassen, sind Selektionen, die nicht die Regeln selbst, sondern vielmehr die zugrundeliegenden Transaktionsmengen betreffen, nach bisherigem Stand der Forschung als äußerst problematisch anzusehen. Datenbezogenen Selektionen, das heißt Beschränkungen auf Teilmengen der ursprünglichen Daten, gehören im Rahmen der üblichen Fokussierung zu einem Wissensentdeckungsprozeß (vgl. u. a. [Fayyad et al., 1996a], [Wirth und Hipp, 2000], [Hipp und Güntzer, 2002]). Impliziert jede Selektion einer Teilmenge der Transaktionen jedoch eine Neuinitialisierung des Caches, dann ist ein interaktives Arbeiten nicht mehr vorstellbar. Dies gilt insbesondere im Hinblick darauf, daß für die Regelgenerierung im Rahmen des auf einem Regelcache basierenden Ansatzes besonders niedrige Schwellenwerte für die Gütemaße anzunehmen sind und demzufolge mit entsprechend langen Laufzeiten zu rechnen ist. Da heute keine Verfahren bekannt sind, die bezüglich der Selektionen eine Alternative zur vollständigen Neugenerierung des Regelcaches aufzeigen, ist der im vorangehenden Kapitel beschriebene, auf einem Regelcache basierende Ansatz demnach grundsätzlich in Frage zu stellen.

Im folgenden wird das geschilderte Problem zunächst anhand verschiedener Beispiele vertieft, wobei als Motivation der Prozeßcharakter der Wissensentdeckung im Mittelpunkt steht. Darauf aufbauend wird eine entsprechende Lösung für das Selektionsproblem entwickelt. Mit dem beschriebenen Ansatz ist es erstmals möglich, aus einem für eine Gesamtheit von Transaktionen generierten Regelcache die Regeln abzuleiten, die bisher dadurch erzeugt werden, daß eine Teilmenge der Transaktionen selektiert und auf dieser Teilmenge ein Regelgenerierungsalgorithmus gestartet wird. Das Kapitel schließt mit einer Evaluierung des beschriebenen Ansatzes anhand verschiedener realer Anwendungen. Es zeigt sich, daß der auf einem Cache basierende Ansatz aus dem vorherigen Kapitel mittels der hier vorgestellten Erweiterung auch für Datenselektionen die geforderte Interaktivität des Wissensentdeckungsprozesses garantieren kann.

5.3.1 Problematisierung

Um auch für sehr große Datenbanken einen interaktiven Wissensentdeckungsprozeß im Sinne von Abschnitt 2.1 zu ermöglichen, wurde im vorangehenden Abschnitt 5.2 ein Regelcache eingeführt, der nach seiner Initialisierung sehr kurze Antwortzeiten garantiert. Dafür ist es erforderlich, daß sich die während des Wissensentdeckungsprozesses formulierten Anfragen durch Filterung aus dem Regelcache ableiten lassen. Während die üblichen Einschränkungen durch Gütemaße oder Bedingungen auf den Ereignissen und den zugehörigen Attributen vergleichsweise einfach umzusetzen sind, bedeutet eine Änderung der zugrundeliegenden Transaktionsdaten, daß eine Realisierung als Filterung auf dem Cache nicht mehr möglich ist. Kommen lediglich neue Transaktionen zu den zugrundeliegenden Daten hinzu, können sogenannte inkrementelle Verfahren herangezogen werden, um den Aufwand für die Neugenerierung der Regeln gering zu halten (vgl. u. a. [Cheung et al., 1996], [Cheung et al., 1997], [Feldman et al., 1997], [Thomas et al., 1997], [Hafez et al., 1999]). Für den typischen Wissensentdeckungsprozeß ist jedoch anzunehmen, daß die zu analysierende Datenbasis nicht wächst. Entsprechend sind inkrementelle Verfahren in dem hier beschriebenen Zusammenhang nicht anwendbar.

Typisch für den in Abschnitt 2.1 beschriebenen Prozeß der Wissensentdeckung ist die zunehmende Fokussierung während der Durchführung. So zielen die ersten Iterationen des Prozesses darauf, ein allgemeines Bild der Daten abzuleiten, und ausgehend von diesem werden vielversprechende Details in den folgenden Iterationen eingehender untersucht. Solche Details können beispielsweise bestimmte Attribute, das heißt Ereignisse oder Ereigniskombinationen sein. Eine zentrale Rolle kommt jedoch der Fokussierung auf vielversprechende Teilmengen der Daten im Sinne der oben beschriebenen Datenselektion zu.

Ein Beispiel aus dem Umfeld der in Abschnitt 5.1 vorgestellten QUIS-Datenbank ist die Unterscheidung verschiedener Fahrzeugbaureihen. Während zu Anfang eines Wissensentdeckungsprozesses beispielsweise allgemeine Zusammenhänge zwischen Sonderausstattungen analysiert werden, können in den späteren Iterationen baureihenspezifische Abhängigkeiten zunehmend in den Mittelpunkt rücken. So werden für die Fahrzeuge der Mercedes S-Klasse aus dem Premiumsegment andere Sonderausstattungskombinationen typisch sein als für die Einstiegsmodelle der Mercedes A-Klasse oder Modelle aus dem Bereich der Nutzfahrzeuge. Weitere Möglichkeiten der Fokussierung ergeben sich beispielsweise aus den Produktionsdaten, die es unter anderem erlauben, Produktionsquartale einzeln zu betrachten oder separate Analysen für die verschiedenen Werke durchzuführen.

Ein Beispiel aus der klassischen Warenkorbanalyse ist die Betrachtung von Transaktionen nach Wochentagen. Zu Beginn einer Analyse wird typischerweise eine allgemeine Auswertung sämtlicher Transaktionen stehen. Im Anschluß ist jedoch

die Betrachtung der Transaktionen nach Wochentagen sinnvoll, da zum Beispiel anzunehmen ist, daß sich die Eigenschaften der Einkäufe an Samstagen von denen zu Wochenbeginn unterscheiden. Stehen außerdem Informationen über die Kunden zur Verfügung, dann sind Fokussierungen auf einzelne Kundengruppen, beispielsweise nach dem Familienstatus, denkbar.

Solche Fokussierungen lassen sich während der Regelgenerierung dadurch berücksichtigen, daß die entsprechenden Daten selektiert und nur auf diesen selektierten Daten Regeln generiert werden. Werden die Regeln aus dem Cache abgeleitet, müssen jedoch die Ausprägungen der Regelgütemaße jeweils so angepaßt werden, daß diese die Eigenschaften der ausgewählten Teilmengen korrekt widerspiegeln. So kann im Kontext der QUIS-Datenbank beispielsweise allgemein über sämtliche Baureihen der Einbau der Sonderausstattungen a und b mit Wahrscheinlichkeit beziehungsweise Konfidenz p den Einbau der Sonderausstattung c bedingen. Dieser Wert p kann aber für jede der Fahrzeugbaureihen verschieden sein. Eine komplette Neugenerierung des Regelcaches oder auch nur von Teilen des Caches kommt nicht in Betracht, da die zu erwartenden Laufzeiten einen interaktiven Wissensentdeckungsprozeß unmöglich machen würden.

5.3.2 Eigener Ansatz

Der im folgenden beschriebene Ansatz bereitet die im Regelcache abgelegten Informationen so auf, daß Datenselektionen sehr effizient als Ergebnisnachbearbeitung durchgeführt werden können.

Grundidee

Eine Voraussetzung für die Realisierung von Selektionen als Ergebnisnachbearbeitung ist, daß sämtliche Informationen, die potentiell für die Selektionen herangezogen werden, bereits während der Generierung des Regelcaches in Form von Ereignissen kodiert in den Transaktionen enthalten sind. Diese sogenannten *Pseudoereignisse (pseudo items)* sind nicht im eigentlichen Sinn Teil einer Transaktion, sondern beschreiben eine Transaktion als Ganzes (vgl. [Hipp und Güntzer, 2002]). Attribute, die als ein beschreibendes Pseudoereignis kodiert zu jeder Transaktion hinzugenommen werden, sind beispielsweise der Familienstatus eines Kunden oder der Wochentag, an dem eine Transaktion stattgefunden hat. Im Kontext der QUIS-Datenbank sind vergleichbare selektionsrelevante Pseudoereignisse beispielsweise die verschiedenen Baureihen, wie A-Klasse oder C-Klasse, aber auch sämtliche Datumsangaben, üblicherweise auf Monats- oder Quartalsbasis aggregiert, oder die verschiedenen Produktionswerke.

Sei $\mathcal{D}'$ Teilmenge der Transaktionsdatenbank $\mathcal{D}$, wobei $\mathcal{D}'$ aus genau den Trans-

aktionen aus $\mathcal{D}$ besteht, die jeweils eine bestimmte Ereigniskombination R enthalten. $\mathcal{D}'$ ist die anhand von R aus $\mathcal{D}$ selektierte Datenbank, daß heißt $\mathcal{D}' = \{T \in \mathcal{D} \mid R \subseteq T\}$. Die Häufigkeit einer Ereigniskombination A bezüglich der Datenbank $\mathcal{D}'$, $\mathsf{supp}_{\mathcal{D}'}(A)$, kann dann wie folgt aus der Häufigkeit von A bezüglich $\mathcal{D}$, $\mathsf{supp}_{\mathcal{D}}(A)$, abgeleitet werden:

$$\mathsf{supp}_{\mathcal{D}'}(A) = \mathsf{supp}_{\mathcal{D}}(A \cup R) \cdot \frac{|\mathcal{D}|}{|\mathcal{D}'|}.$$

Die Transaktionen in $\mathcal{D}'$, die A enthalten, sind genau die Transaktionen in $\mathcal{D}$, die A und zugleich R enthalten. Die absolute Häufigkeit von A in der selektierten Datenbank $\mathcal{D}'$ entspricht damit der absoluten Häufigkeit von $A \cup R$ in der ursprünglichen Datenbank $\mathcal{D}$. Entsprechend ergibt sich durch Multiplikation mit dem Faktor $|\mathcal{D}|/|\mathcal{D}|'$ die relative Häufigkeit von A in $\mathcal{D}'$.

Um auf diese Weise die Häufigkeit von A bezüglich $\mathcal{D}'$ ableiten zu können, muß die Häufigkeit von $A \cup R$ bezüglich $\mathcal{D}$ bereits bekannt sein. Im Kontext des beschriebenen auf einem Regelcache basierenden Szenarios ist dies genau dann der Fall, wenn die Ereigniskombination $A \cup R$ in $\mathcal{D}$ häufig ist.[12]

Zunächst scheint diese Voraussetzung problematisch zu sein. Es zeigt sich jedoch, daß in praktischen Anwendungen kaum relevante Einschränkungen zu erwarten sind. Obige Voraussetzung bedeutet lediglich, daß $\mathcal{D}'$ eine hinreichend große Teilmenge von $\mathcal{D}$ sein muß. Diese Eigenschaft ist für typische Selektionen zumeist erfüllt, vergleiche dazu die am Ende dieses Abschnitts beschriebenen Experimente.

Ableitung verschiedener Regelgütemaße

Die soeben beschriebene Eigenschaft der Häufigkeiten von Ereigniskombinationen läßt sich dazu nutzen, Selektionen von der eigentlichen Regelgenerierung in das Retrieval aus dem Regelcache als Ergebnisnachbearbeitung zu verlagern. Im folgenden werden entsprechende Berechnungsvorschriften für die wichtigsten der in

[12] Formal reicht es nicht aus, daß $A \cup R$ in $\mathcal{D}$ häufig ist, sondern mindestens eine der auf $A \cup R$ basierenden Regeln muß auch in den Cache aufgenommen worden sein. Werden für die Regelgütemaße wie Konfidenz oder Lift ungewöhnlich hohe Schwellenwerte angenommen, dann werden unter Umständen sämtliche auf einer Ereigniskombination $A \cup R$ basierenden Regeln ausgefiltert, obwohl $A \cup R$ eine in $\mathcal{D}$ häufige Ereigniskombination ist. In diesem Fall genügt es jedoch, zusätzlich zu den Regeln auch die häufigen Ereigniskombinationen im Regelcache zu speichern. Da sämtliche in $\mathcal{D}$ häufigen Ereigniskombinationen vom Analysealgorithmus bereits erzeugt wurden, stehen diese ohne zusätzlichen Rechenaufwand zur Verfügung. Für die im vorherigen Kapitel vorgeschlagene Umsetzung des Regelcaches auf einem relationalen Datenbanksystem können die vom Analysealgorithmus zwar erzeugten, aber keiner gültigen Regel zugeordneten Ereigniskombinationen zusätzlich in die Tabelle itemsets eingetragen werden.

den Abschnitten 2.2.1 und 2.2.2 eingeführten Regelgütemaße vorgestellt, indem jedes der Maße auf Häufigkeiten von Ereigniskombinationen in $\mathcal{D}'$ zurückgeführt wird. Wie bisher bezeichnet $\mathcal{D}'$ die mittels R aus $\mathcal{D}$ selektierte Teilmenge.

Der Support einer Regel ergibt sich aus der Häufigkeit der Vereinigung von Prämisse und Konklusion:

$$\begin{aligned}\mathsf{supp}_{\mathcal{D}'}(A \to B) &= \mathsf{supp}_{\mathcal{D}'}(A \cup B) \\ &= \mathsf{supp}_{\mathcal{D}}(A \cup B \cup R) \cdot \frac{|\mathcal{D}|}{|\mathcal{D}'|}.\end{aligned}$$

Die Konfidenz einer Regel basiert auf dem Verhältnis der Häufigkeit der Vereinigung von Prämisse und Konklusion zur Häufigkeit der Prämisse:

$$\begin{aligned}\mathsf{conf}_{\mathcal{D}'}(A \to B) &= \frac{\mathsf{supp}_{\mathcal{D}'}(A \cup B)}{\mathsf{supp}_{\mathcal{D}'}(A)} \\ &= \frac{\mathsf{supp}_{\mathcal{D}}(A \cup B \cup R)}{\mathsf{supp}_{\mathcal{D}}(A \cup R)}.\end{aligned}$$

Der Lift einer Regel läßt sich definitionsgemäß wie folgt durch die Häufigkeiten von Prämisse, Konklusion und deren Vereinigung ausdrücken:

$$\begin{aligned}\mathsf{lift}_{\mathcal{D}'}(A \to B) &= \frac{\mathsf{supp}_{\mathcal{D}'}(A \cup B)}{\mathsf{supp}_{\mathcal{D}'}(A) \cdot \mathsf{supp}_{\mathcal{D}'}(B)} \\ &= \frac{\mathsf{supp}_{\mathcal{D}}(A \cup B \cup R)}{\mathsf{supp}_{\mathcal{D}}(A \cup R) \cdot \mathsf{supp}_{\mathcal{D}}(B \cup R)} \cdot \frac{|\mathcal{D}'|}{|\mathcal{D}|}.\end{aligned}$$

Für die Berechnung der Konviktion aus den Häufigkeiten in $\mathcal{D}$ gilt:

$$\begin{aligned}\mathsf{conv}_{\mathcal{D}'}(A \to B) &= \frac{\mathsf{supp}_{\mathcal{D}'}(A) \cdot \mathsf{supp}_{\mathcal{D}'}(\neg B)}{\mathsf{supp}_{\mathcal{D}'}(A, \neg B)} \\ &= \frac{\mathsf{supp}_{\mathcal{D}'}(A) \cdot (1 - \mathsf{supp}_{\mathcal{D}'}(B))}{\mathsf{supp}_{\mathcal{D}'}(A) - \mathsf{supp}_{\mathcal{D}'}(A \cup B)} \\ &= \frac{\mathsf{supp}_{\mathcal{D}}(A \cup R) \cdot (1 - \mathsf{supp}_{\mathcal{D}}(B \cup R) \cdot \frac{|\mathcal{D}'|}{|\mathcal{D}|})}{\mathsf{supp}_{\mathcal{D}}(A \cup R) - \mathsf{supp}_{\mathcal{D}}(A \cup B \cup R)}.\end{aligned}$$

Voraussetzung für den Transfer eines Maßes von $\mathcal{D}$ nach $\mathcal{D}'$ ist, daß dieses Maß auf die Häufigkeiten von Prämisse, Konklusion und der Vereinigung von Prämisse und Konklusion zurückgeführt werden kann. Da auch von den in Kapitel 3 und 4 vorgestellten Verfahren über diese drei Werte hinaus keine weiteren Informationen generiert werden, sind Maße, die zusätzliche Informationen benötigen, basierend auf den heute üblichen Assoziationsregelverfahren nicht direkt berechenbar und damit unabhängig von der Verwendung des vorgestellten Regelcaches als problematisch anzusehen.

5.3.3 Evaluierung

Im folgenden wird der beschriebene Ansatz anhand einer typischen Anwendung evaluiert. Grundlage ist die Implementierung des Apriori-Algorithmus aus Kapitel 3 und die Datenbankanbindung aus Abschnitt 5.1.[13] Als Datensatz werden die bekannten Einzelhandelsdaten EINZELHANDEL (vgl. Abschnitt 3.3) herangezogen, wobei die Transaktionen mit dem jeweiligen Wochentag als Pseudoereignis ergänzt wurden.

Szenario

Für die Experimente wird das folgende Analyseszenario angenommen: Ein Analyst untersucht die Einzelhandelstransaktionen im Rahmen eines Wissensentdeckungsprozesses, wie in Abschnitt 2.1 beschrieben. Zu Beginn steht die allgemeine Explorationsphase. Die Ergebnisse dieser ersten Orientierung dienen als Einstiegspunkte für tiefergehendere Untersuchungen. Im angenommenen Szenario sind insbesondere die zusätzlich zu den Artikeln in den Transaktionen abgelegten Wochentage für eine genauere Untersuchung von Interesse. Es stellt sich die naheliegende Frage, inwieweit sich das Kaufverhalten der Kunden an den verschiedenen Wochentagen unterscheidet. Beispielsweise werden typische Versorgungskäufe eher vor den Wochenenden zu beobachten sein als in der Wochenmitte. Es ist außerdem anzunehmen, daß von den verschiedenen Kundengruppen, wie Familien, Alleinstehende, Rentner oder Berufstätige, unterschiedliche Wochentage oder Tageszeiten für Einkäufe bevorzugt werden.

Typischerweise wird der Analyst damit beginnen, auf Basis sämtlicher Transaktionen Assoziationsregeln zu generieren. Er wird dann, nachdem er die ersten Ergebnisse gesichtet hat, gezielt nacheinander für die verschiedenen Wochentage Daten selektieren und auf diesen Teilmengen die Regelgenerierung erneut anstoßen. In Abhängigkeit von der Anzahl zugrundeliegender Transaktionen und den gewählten Schwellenwerten für die Gütemaße werden diese Generierungsläufe die Arbeit des Analysten für einige Minuten oder sogar für Stunden unterbrechen.

Durch die Möglichkeit, Selektionen im Rahmen des Retrievals der Regeln aus dem Cache zu bearbeiten, sind solche wiederholten Regelgenerierungsläufe nicht länger notwendig. Nach Initialisierung des Regelcaches auf Basis angemessen niedriger Schwellenwerte für die Gütemaße betragen die Antwortzeiten für Selektionen im allgemeinen nur noch wenige Sekunden.

[13] Die Experimente werden auf dem 500MHz Pentium III Linuxsystem durchgeführt, das bereits in Abschnitt 5.2 eingesetzt wurde.

Experimente und Ergebnisse

Für die folgenden Experimente werden zwei Generierungsläufe mit 0,1% beziehungsweise 0,7% als Mindestschwellenwert für den Support auf der gesamten Datenbank durchgeführt. Die minimale Konfidenz wird jeweils auf 75% gesetzt und keine Schwellenwerte für weitere Gütemaße vorgegeben. Auf die Regelmenge, die mit einem minimalen Support von 0,1% erzeugt wird, werden die nachgelagerten Selektionen angewendet. Das heißt, ohne den Generierungsalgorithmus erneut zu starten, werden die Regeln aus dem Cache abgeleitet, die generiert worden wären, wenn der Analyst die Daten für die einzelnen Wochentage im voraus selektiert und jeweils für diese Teilmengen der Transaktionen separat die Regelgenerierung angestoßen hätte.

Um die Ergebnisse vergleichen zu können, wurden außerdem parallel die Datensätze für jeden Wochentag tatsächlich selektiert und jeweils ein separater Generierungslauf durchgeführt. Als Schwellenwert für den minimalen Support wurde dazu mit 0,7% ein um Faktor 7 größerer Schwellenwert gewählt. Dieser erhöhte Schwellenwert soll der Tatsache Rechnung tragen, daß bei gleichen absoluten Häufigkeiten die relative Häufigkeit einer Ereigniskombination in $\mathcal{D}'$ in der mächtigeren Menge $\mathcal{D}$ entsprechend kleiner ist (vgl. auch den vorherigen Abschnitt).[14]

In den durchgeführten Experimenten stimmen sämtliche mittels nachgelagerter Selektion generierten Regelmengen mit den direkt durch einen separaten Generierungslauf auf den selektierten Transaktionen erzeugten Regelmengen überein, beziehungsweise enthalten sogar einige zusätzliche Regeln. Wird der Schwellenwert für die Mindesthäufigkeit angepaßt, dann können mit dem beschriebenen Ansatz demnach Selektionen ohne Verluste von der Datenaufbereitung in das Retrieval von Regeln nachgelagert werden.

Der Vorteil des neuen Ansatzes zeigt sich bei Betrachtung der erreichten Laufzeiten in Abbildung 5.7, wobei die Laufzeiten jeweils den Datenbankzugriff einschließen. Zunächst fällt auf, daß für den gesamten Monat die Laufzeit nach dem neuen Ansatz mit 122 Sekunden um fast 30% über den 96 Sekunden des konventionellen Ansatzes liegt. Diese große Differenz erklärt sich durch die unterschiedlichen minimalen Schwellenwerte für den Support von 0,1% beziehungsweise 0,7%, die sich direkt auf die Laufzeit auswirken. Der Generierungslauf zur Initialisierung des Regelcaches ist damit aufwendiger als der Orientierungslauf nach dem konventionellen Ansatz. Zugleich zeigt sich jedoch für die nachgelagerte Selektion ein deutlicher Laufzeitvorteil, sobald die einzelnen Wochentage in die Betrachtung einbezogen werden. Während ein konventioneller Analyselauf für einen Wochentag zwischen

[14] Der Faktor 7 entspricht sechs Werktagen und einem zusätzlichen Tag als geschätzte Sicherheitskonstante. In dem angenommenen Szenario ergaben sich bei einem minimalen Support von 0,7% jeweils 50 bis 90 Regeln pro Wochentag, die im Regelcache abgelegt wurden.

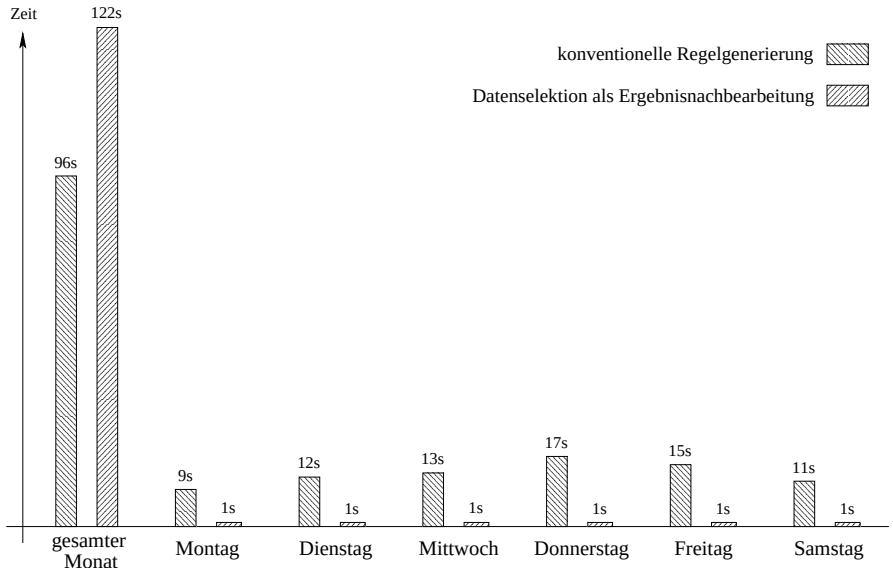


Abbildung 5.7: Laufzeiten für den Gesamtmonat und die einzelnen Wochentage auf Einzelhandelsdaten (jeweils inklusive Datenbankzugriff)

9 und 17 Sekunden benötigt, wird das entsprechende Ergebnis durch den neuen Ansatz in maximal einer Sekunde erzeugt.[15]

Der Laufzeitvorteil des neuen Ansatzes wird noch deutlicher, wenn größere Datenmengen analysiert werden. Da die Regelgenerierungsverfahren nahezu linear mit der Anzahl zugrundeliegender Transaktionen skalieren (vgl. Kapitel 3), würden sich bei der Analyse der Transaktionen eines vollständigen Jahres die Zeiten für die einzelnen Regelgenerierungsläufe auf den selektierten Daten der Wochentage und für den initialen Regelgenerierungslauf ungefähr um Faktor 12 verlängern. Da sich aber gleichzeitig die Anzahl erzeugter Regeln nicht wesentlich verändern wird (vgl. dazu Abschnitt 5.2.3), bleiben die bereits sehr kurzen Antwortzeiten des neuen Ansatzes auch auf den Daten eines gesamten Jahres nahezu die gleichen. Dies ist darin begründet, daß die Antwortzeiten für das Retrieval der Regeln aus dem Cache unabhängig von der Anzahl zu analysierenden Transaktionen sind.

[15] Im Gegensatz zu den im vorherigen Abschnitt beschriebenen Experimenten wurden die Regeln nicht im Datenbanksystem zwischengespeichert, sondern im Hauptspeicher gehalten.

Kapitel 6

Zusammenfassung und Ausblick

Im folgenden sollen die Ergebnisse dieser Arbeit abschließend zusammengefaßt und ein Ausblick auf verschiedene vielversprechende Ansatzpunkte für weiterführende Forschungen im Bereich der Wissensentdeckung in Datenbanken mit Assoziationsregeln gegeben werden.

Zusammenfassung der Ergebnisse

Zu Beginn der vorliegenden Arbeit wurde eine Einführung in das Gebiet der Wissensentdeckung in Datenbanken und der Assoziationsregelgenerierung als Data Mining-Methode gegeben. Dabei lag der Schwerpunkt auf dem iterativen und interaktiven Prozeßcharakter der Wissensentdeckung in Datenbanken. Mit CRISP-DM folgte die Beschreibung eines konkreten Prozeßmodells, an dem exemplarisch auf die einzelnen Phasen der Wissensentdeckung eingegangen wurde (vgl. auch [Hipp und Lindner, 1999], [Wirth und Hipp, 2000], [Hipp et al., 2002a]). Als kritisch wurden in diesem Zusammenhang die Laufzeiten der Verfahren zur Generierung von Assoziationsregeln sowie die Integration der Verfahren in den Wissensentdeckungsprozeß identifiziert.

Um abgesicherte Erkenntnisse hinsichtlich der Vergleichbarkeit und Effizienz der heute etablierten Verfahren zur Assoziationsregelgenerierung zu gewinnen, wurden diese eingehend analysiert und evaluiert (vgl. auch [Hipp et al., 1998], [Hipp et al., 2000a], [Hipp et al., 2000b]).

Im Vordergrund der Analyse stand zunächst die Identifizierung der Prinzipien und Vorgehensweisen, auf denen die etablierten Verfahren basieren. Das Ergebnis dieser Aufarbeitung ist eine einheitliche Darstellung der Verfahren, aus der schließlich eine Systematisierung der bekannten Algorithmen abgeleitet werden konnte, welche diese hinsichtlich der Gemeinsamkeiten und Unterschiede der zugrundeliegenden Ansätze einordnet.

Im einzelnen wurden die Verfahren anhand folgender Kriterien differenziert: Zunächst wurde anhand der verwendeten Suchstrategie unterschieden, das heißt ob der Suchraum mittels einer Breiten- oder einer Tiefensuche durchlaufen wird, des weiteren anhand der Kandidatengenerierung, das heißt ob diese datenorientiert erfolgt oder nicht, und schließlich anhand der eingesetzten Methode zur Häufigkeitsbestimmung, das heißt ob diese durch direktes Zählen erfolgt oder indirekt mittels Schnittmengenbildung.

Die Evaluierung der Verfahren wurde sowohl auf Basis von gezielt mit bestimmten Eigenschaften synthetisch erzeugten Datensätzen als auch anhand von Datensätzen aus realen Anwendungen durchgeführt. Allein aufgrund der vielen einbezogenen Verfahren und den verschiedenen Datensätzen gingen die Experimente über vergleichbare Evaluierungen in der Literatur hinaus. Das wichtigste Ergebnis dieser Untersuchungen war, daß, anders als die bisherigen Evaluierungen zumeist suggerieren, ein insgesamt als effizientestes Verfahren zur Generierung von Assoziationsregeln anzusehender Algorithmus nicht existiert. Vielmehr zeigte sich, daß in Abhängigkeit von den Eigenschaften der zu analysierenden Daten jeweils andere Ansätze die kürzeren Antwortzeiten aufweisen. Diese Beobachtung war nicht auf Datensätze mit ungewöhnlichen Eigenschaften beschränkt, sondern galt auch für die Analysedaten, die den Evaluierungen in der Literatur zugrunde liegen.

Neben den neuen Erkenntnissen bezüglich des Laufzeitverhaltens war es der bisher kaum systematisch untersuchte Hauptspeicherbedarf der verschiedenen Ansätze, der zu einer Neubewertung der Algorithmen im Rahmen dieser Arbeit führte. Insbesondere für Verfahren, die in vielen Experimenten sehr kurze Laufzeiten erreichten, ging die Generierung der häufigen Ereigniskombinationen meist mit einem sehr hohen Speicherbedarf einher. Da ein Auslagern von Hauptspeicherseiten auf Sekundärspeicher durch das Betriebssystem für alle untersuchten Algorithmen die Regelgenerierung nahezu zum Erliegen bringt, können diese Verfahren nur eingeschränkt auf großen Analysedaten eingesetzt werden.

Die Algorithmen Apriori, Eclat und FP-Growth wurden als die leistungsfähigsten Verfahren zur Generierung von Assoziationsregeln identifiziert. Wird allein die Laufzeit betrachtet, so ist insgesamt FP-Growth als das effizienteste Verfahren anzusehen. Allerdings können in Abhängigkeit vom verfügbaren Hauptspeicher bereits Datensätze mittlerer Größenordnungen mit FP-Growth unter Umständen nicht mehr analysiert werden. Eclat nutzt den verfügbaren Hauptspeicher effizienter, weist aber längere Laufzeiten als FP-Growth auf. Letztlich skaliert der Hauptspeicherbedarf von Eclat jedoch wie der von FP-Growth linear mit der Anzahl von Transaktionen. Für große Analysedaten ist daher lediglich Apriori geeignet, dessen Speicherbedarf nahezu unabhängig von der Anzahl zu analysierender Transaktionen ist. Der Einsatz von Apriori geht allerdings zumeist mit einer weiteren Verschlechterung der Laufzeiten gegenüber Eclat einher.

Auf Basis der umfassenden Evaluierung und des besseren Verständnisses der etablierten Verfahren konnten außerdem im Rahmen dieser Arbeit die bekannten Algorithmen optimiert beziehungsweise neue Ansätze entwickelt werden. So wurde die Tiefensuche dahingehend erweitert, daß für diese auf die gleiche Art und Weise der Suchraum beschnitten werden kann, wie es bisher allein für die Breitensuche möglich war. Das Verfahren Optra – ***op**timized **tr**aversal* –, welches auf der eingeführten rechtsorientierten Tiefensuche basiert, erlaubte es in verschiedenen Experimenten, die Häufigkeitsbestimmung von teilweise mehr als einem Drittel der Kandidaten einzusparen. Dieser Vorteil kann allerdings lediglich eingeschränkt in kürzere Laufzeiten umgesetzt werden. Außerdem wurde der Algorithmus Hybrid eingeführt, der durch geeignete Kombination bekannter Ansätze zur Häufigkeitsbestimmung und zum Durchlaufen des Suchraums in verschiedenen Experimenten nahezu ohne Ausnahme weit kürzere Laufzeiten und einen günstigeren Speicherbedarf als die etablierten Algorithmen erreichte. Im einzelnen waren die für verschiedene Datensätze ermittelten Laufzeiten des neu entwickelten Verfahrens Hybrid zwischen 15% und 60% kürzer als die der etablierten Verfahren.

Neben den Verfahren zur Generierung einfacher Assoziationsregeln wurden in der vorliegenden Arbeit auch Verfahren evaluiert und weiterentwickelt, die unter Einbeziehung von Taxonomien sogenannte taxonome Assoziationsregeln erzeugen. In diesem Zusammenhang wurden zunächst die etablierten Verfahren erweitert, so daß diese ebenfalls Taxonomien auf den Ereignissen berücksichtigen können. Das Ergebnis der Evaluierung war, daß diese Erweiterungen, welche für die vergleichsweise neuen Verfahren hier erstmals vorgestellt wurden, in vielen Fällen effizienter sind als die bisher bekannten, auf die Generierung taxonomer Assoziationsregeln ausgerichteten Verfahren.[1]

Zusätzlich gelang es, die im Rahmen der vorliegenden Arbeit gewonnenen Erkenntnisse zur Generierung einfacher Assoziationsregeln auf die Generierung taxonomer Assoziationsregeln zu übertragen. Auf Basis der vorgestellten rechtsorientierten Tiefensuche konnten Ansätze neu entwickelt werden, welche außerdem erstmals effizient den Suchraum anhand der Taxonomie beschneiden. Das resultierende Verfahren Prutax – ***pru**ne by **tax**onomy* – und seine hybride Variante Prutax+ sind bezüglich der Laufzeiten wesentlich effizienter als die erweiterten etablierten Verfahren FP-Growth, Eclat und Apriori sowie das Verfahren Cumulate, das stellvertretend für die bisher speziell zur Generierung taxonomer Assoziationsregeln entwickelten Ansätze steht. Im einzelnen wurden für die neu entwickelten Verfahren auf verschiedenen Datensätzen bis zu 80% kürzere Laufzeiten als für die erweiterten etablierten Verfahren beziehungsweise die bisher speziell zur Generie-

[1] Die hier erstmals erweiterten Verfahren basieren auf Ansätzen zur Generierung einfacher Assoziationsregeln, die bei der Entwicklung der speziell auf die Generierung taxonomer Assoziationsregeln ausgelegten Verfahren noch nicht bekannt waren.

rung taxonomer Assoziationsregeln entwickelten Verfahren ermittelt. Insbesondere Prutax erreicht diese Laufzeiten, ohne den häufig kritischen Hauptspeicherbedarf der bisherigen Verfahren aufzuweisen.

Eng verbunden mit der Frage nach der Effizienz der Verfahren ist die Frage nach deren Integration in den beschriebenen iterativen und interaktiven Wissensentdeckungsprozeß. Im Rahmen dieser Arbeit wurden die für diese Integration zentralen Aspekte identifiziert und entsprechende Ansätze vorgestellt. So wurde für den Datenzugriff die Anbindung der Verfahren an ein relationales Datenbanksystem vorgeschlagen und eine entsprechende Implementierung evaluiert. Nicht nur hinsichtlich der Effizienz konnte die Umsetzung dieses Ansatzes im Vergleich zu kommerziellen Produkten überzeugen, sondern vielmehr wurde es durch diesen Ansatz möglich, wesentlich komplexere Analysefragestellungen direkt aus der Datenbank zu beantworten (vgl. auch [Hipp et al., 2001b], [Hipp et al., 2002a]).

Um trotz der in vielen Fällen noch immer unbefriedigenden Laufzeiten der Algorithmen einen iterativen und interaktiven Wissensentdeckungsprozeß zu ermöglichen, wurde außerdem ein Regelcache zusammen mit einer entsprechenden Anfragesprache zur Integration der Verfahren eingeführt (vgl. auch [Hipp et al., 2002b]). Ist der Cache initialisiert, werden mit diesem Ansatz Antwortzeiten von wenigen Sekunden erreicht, während die Laufzeit für den bisher notwendigen, erneuten Regelgenerierungslauf oft in Bereichen von einigen Minuten oder Stunden liegt. Problematisch im Zusammenhang mit dem beschriebenen Ansatz sind allerdings Veränderungen der zugrundeliegenden Datenbasis, da in diesem Fall die im Cache abgelegten Regeln ungültig werden und ein erneuter Regelgenerierungslauf notwendig wird. Dies gilt insbesondere auch für Selektionen von Teilmengen der Analysedaten, die beispielsweise der Beschränkung auf Transaktionen eines bestimmten Typs entsprechen. Solche Selektionen sind grundlegend für den im Verlauf gewöhnlich zunehmend fokussierten Wissensentdeckungsprozeß. Der beschriebene Cache wurde im Rahmen der vorliegenden Arbeit jedoch erweitert, so daß auch im Falle von Selektionen auf den Analysedaten die dann gültigen Regeln aus dem vorhandenen Cache sehr effizient abgeleitet werden können (vgl. auch [Hipp und Güntzer, 2002], [Hipp et al., 2002b]). Dazu wird nicht auf die Datensätze selbst zurückgegriffen, das heißt, der Cache bleibt gültig, auch wenn sich die Datenbasis durch Selektionen verändert.

Ausblick

Die Forschung auf dem Gebiet der Assoziationsregelgenerierung hat sich in den letzten Jahren sehr dynamisch entwickelt. Entsprechend ist zu erwarten, daß auch in Zukunft eine Vielzahl neuer Algorithmen publiziert wird, die in Wissenschaft und Praxis weitere Verbreitung finden werden. Während mit der vorliegenden Ar-

beit der Anspruch erhoben wird, die grundlegenden Aspekte der heute als etabliert anzusehenden Verfahren weitgehend abschließend betrachtet zu haben, erscheint es insbesondere lohnenswert, die im Rahmen dieser Arbeit begonnene Analyse und Systematisierung fortzuführen, das heißt um zukünftig entwickelte und sich in Forschung und Anwendung bewährende Algorithmen zu erweitern.

Neben den hier behandelten Verfahren zur Generierung einfacher und taxonomer Assoziationsregeln sind auch die Verfahren zur Generierung weiterer Regeltypen, wie beispielsweise quantitativer Assoziationsregeln (vgl. z. B. [Srikant und Agrawal, 1996a], [Büchter und Wirth, 1998], [Hong et al., 1999]), die Verfahren zur Generierung sequentieller Muster (vgl. u. a. [Agrawal und Srikant, 1995], [Srikant, 1996], [Mannila et al., 1997]) oder Verfahren zu verwandten Fragestellungen, die nicht direkt mit der Regelerzeugung im Zusammenhang stehen (vgl. z. B. [Bayardo, 1998], [Tsur et al., 1998], [Lin und Kedem, 1998] [Pasquier et al., 1999], [Bykowski und Rigotti, 2001]), vielversprechende Kandidaten zur Fortführung beziehungsweise Erweiterung der Systematik.

Des weiteren ist anzunehmen, daß die anhand der Analyse der Verfahren gewonnenen Erkenntnisse, welche in dieser Arbeit der Ausgangspunkt für die Optimierung und Neuentwicklung von Ansätzen zur Generierung einfacher und taxonomer Assoziationsregeln bildeten, auf weitere Gebiete übertragbar sind. So sind ähnliche Effizienzsteigerungen bezüglich der Laufzeiten und des Speicherbedarfs auch im Zusammenhang mit anderen Regeltypen, sequentiellen Mustern und verwandten Problemstellungen zu erwarten. Ebenso können die in dieser Arbeit nicht berücksichtigten Ansätze zur parallelisierten Generierung von Assoziationsregeln (vgl. etwa [Agrawal und Shafer, 1996], [Zaki et al., 1997d], [Skillicorn, 1998], [Agarwal et al., 2001]) von den vorgestellten Optimierungen und Neuentwicklungen profitieren. Auch sind die im Rahmen der vorliegenden Arbeit untersuchten und neu eingeführten Ansätze selbst mögliche Ziele weiterer Optimierungen. In diesem Kontext wurde vor allem mit der systematischen Untersuchung des Speicherbedarfs Potential für grundlegende Verbesserungen aufgezeigt. Das Gebiet der Assoziationsregelgenerierung weist außerdem Bezüge zur formalen Begriffsanalyse und Galois-Korrespondenzen auf (vgl. u. a. [Wille, 1992], [Ganter und Wille, 1999], [Cristofor et al., 2000], [Pasquier et al., 2001], [Stumme et al., 2001]), die ein vielversprechendes Thema zukünftiger Arbeiten sein können.

Neben den Algorithmen zur Regelgenerierung bedarf die Integration der Verfahren in den Wissensentdeckungsprozeß noch weitergehender Untersuchungen. Hier ist die vorliegende Arbeit so weit fortgeschritten, daß zunächst Erfahrungen aus praktischen Anwendungen abzuwarten sind, um die Ausrichtung und Schwerpunkte weiterer Forschungsarbeiten festlegen zu können. Offene Fragen sind beispielsweise, wie sich der Regelcache in verschiedenen Anwendungsszenarien bewährt oder wie die Anfragesprache weiter zu entwickeln ist, um den Analysten adäquat

unterstützen zu können. Offen ist auch die Umsetzung einer geeigneten Benutzerschnittstelle, welche über die hier vorgestellte textbasierte Anfrageeingabe hinaus geht. In diesem Zusammenhang ist ebenso die Evaluierung und Weiterentwicklung von Gütemaßen zur Bewertung der Regeln zu nennen (vgl. z. B. [Ramaswamy et al., 1998], [Shah et al., 1999], [Tan und Kumar, 2000], [Hilderman und Hamilton, 2002]).

Eine weiterer wichtiger Aspekt der Integration der Algorithmen zur Regelgenerierung in den Wissensentdeckungsprozeß ist, daß die Laufzeiten der Verfahren bisher nicht im voraus bekannt sind. Die Möglichkeit, vor einem Analyselauf die zu erwartende Laufzeit ungefähr abzuschätzen, würde den Analysten jedoch in die Lage versetzen, anhand der für ihn maximal tolerierbaren Laufzeit des initialen Regelgenerierungslaufs für die Regelgenerierung möglichst wenige beziehungsweise nur schwache Einschränkungen vorgeben zu können (vgl. Abschnitt 5.2.1 und [Hipp und Güntzer, 2002] zu dieser Problematik).

Weitere vielversprechende Arbeiten ergeben sich auf dem Gebiet der Ausreißererkennung mittels Assoziationsregeln, die in [Hipp et al., 2001a] vorgestellt wurde. In diesem Zusammenhang ist ebenfalls die Entwicklung einer geeigneten Benutzerschnittstelle und entsprechender Gütemaße von zentralem Interesse, wobei die Forschung in diesem Zusammenhang wieder zunächst vor allem auf Erfahrungen aus praktischen Anwendungen angewiesen ist.

Die Integration der Verfahren zur Assoziationsregelgenerierung in den Wissensentdeckungsprozeß ist heute sicherlich als eine der drängendsten Fragestellungen einzuschätzen. Während in den letzten Jahren vor allem algorithmische Aspekte die Aufmerksamkeit auf sich zogen und auf diesem Gebiet große Fortschritte erzielt wurden, können die Verfahren zur Generierung von Assoziationsregeln doch nur dann mit Erfolg von der Forschung in praktische Anwendungen transferiert werden, wenn die Integration der Verfahren in den Wissensentdeckungsprozeß auch tatsächlich den Anforderungen dieses „human centered process“ genügt.

Anhang A

Ergänzende Evaluierungsergebnisse

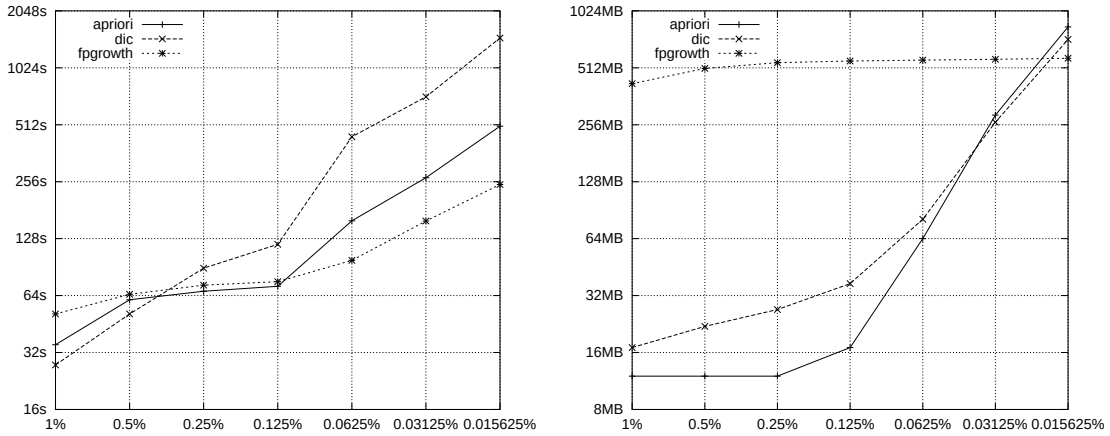
Im folgenden sind ergänzend zu den Messungen aus Kapitel 3 und 4 weitere Evaluierungsergebnisse angeführt. Zusammen mit den bereits beschriebenen Experimenten werden damit im Rahmen dieser Arbeit sämtliche in [Agrawal und Srikant, 1994a], [Savasere et al., 1995a], [Zaki et al., 1997c], [Han et al., 2000], [Hipp et al., 2000a], [Hipp et al., 2000b] verwendeten synthetischen Testdatensätze abgedeckt (vgl. auch Abschnitt 3.3.1 zum Hintergrund der synthetischen Datensätze). Gleiches gilt für die verbleibenden, aus realen Anwendungen stammenden Datensätze, die in [Zheng et al., 2001] erstmals zur Evaluierung von Verfahren zur Generierung von Assoziationsregeln verwendet wurden.

Zusätzlich werden auch die Experimente auf den ausschließlich für diese Arbeit zur Verfügung stehenden Datensätzen aus realen Anwendungen (vgl. auch Abschnitt 3.3.1) mit diesem Anhang vervollständigt.

Die Laufzeit- und Speichermessungen finden unter den gleichen Bedingungen statt, die auch den Messungen in den Kapiteln 3 und 4 zugrunde liegen. Die Experimente bestätigen die bisherigen Ergebnisse und sind hier vor allem im Sinne der Vollständigkeit angeführt.

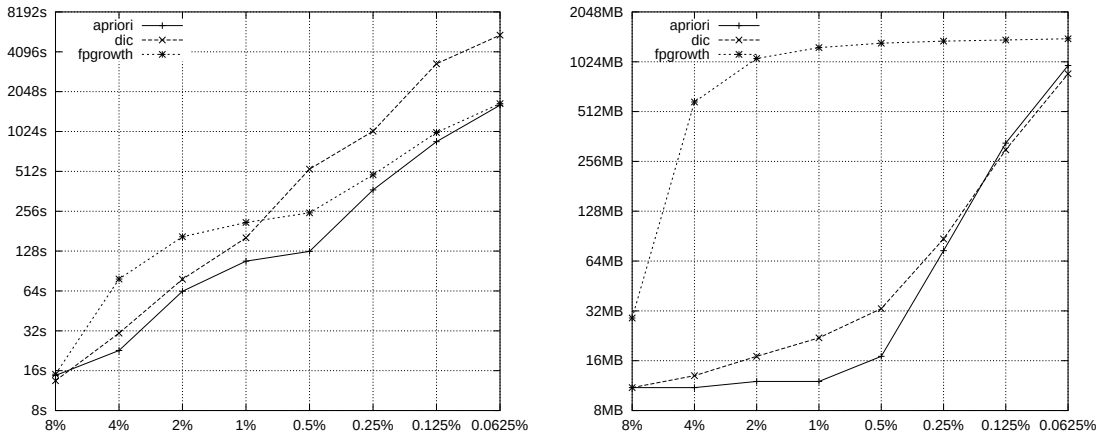
A.1 Etablierte Verfahren

A.1.1 Zählende Verfahren



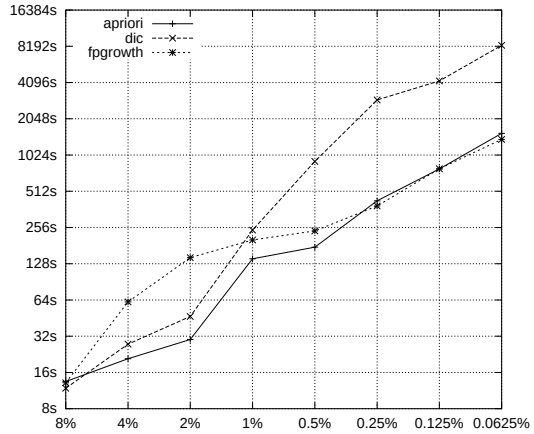
(a) Laufzeit in Sekunden (b) Speicherbedarf in Megabyte

Abbildung A.1: Effizienz zählender Verfahren auf T10I2D1000K bei variierendem minsupp

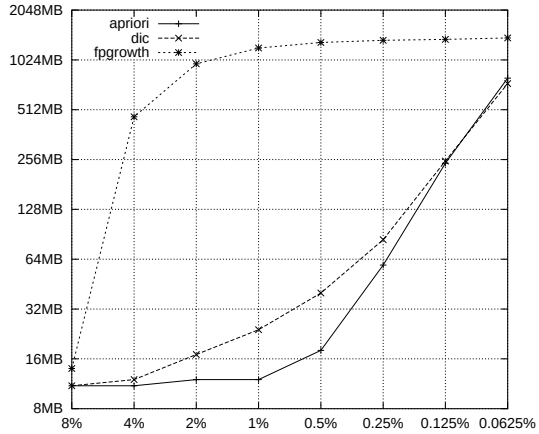


(a) Laufzeit in Sekunden (b) Speicherbedarf in Megabyte

Abbildung A.2: Effizienz zählender Verfahren auf T20I2D1000K bei variierendem minsupp

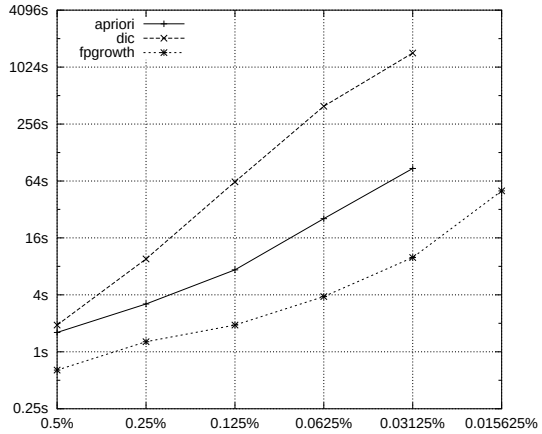


(a) Laufzeit in Sekunden

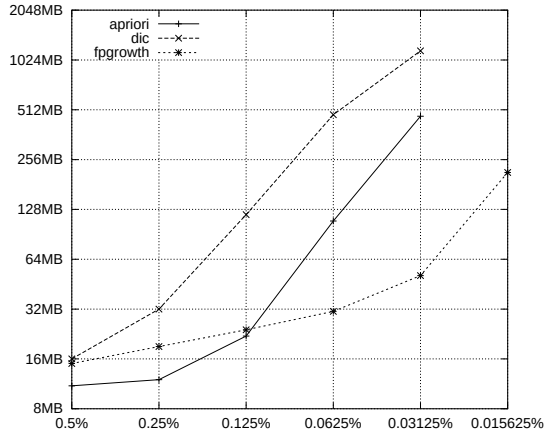


(b) Speicherbedarf in Megabyte

Abbildung A.3: Effizienz zählender Verfahren auf T20I4D1000K bei variierendem minsupp

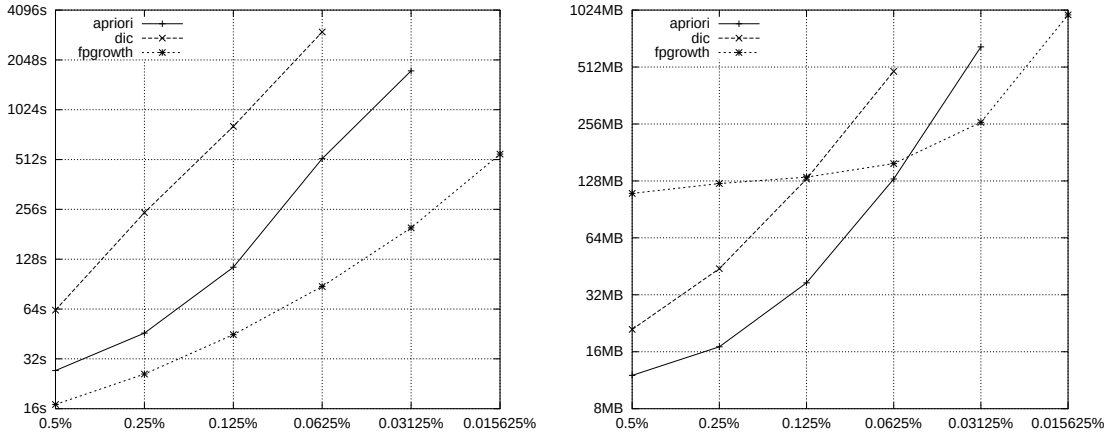


(a) Laufzeit in Sekunden



(b) Speicherbedarf in Megabyte

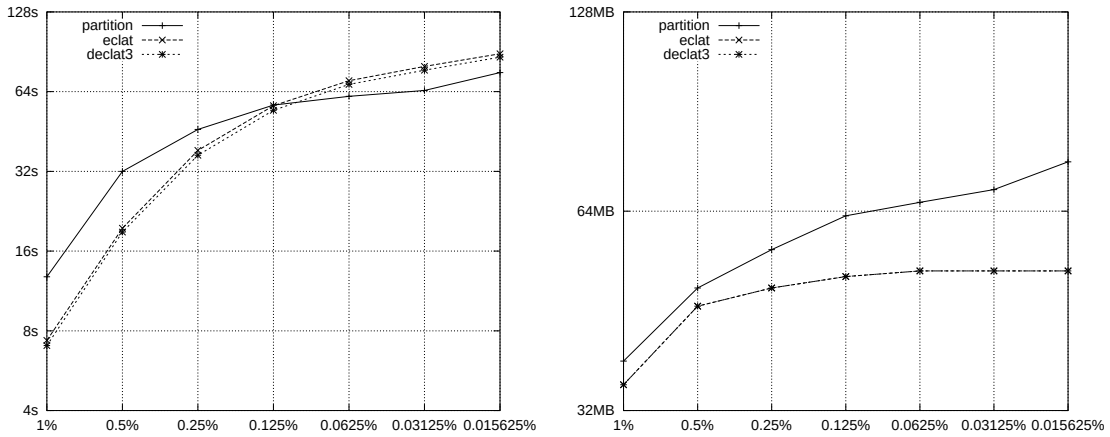
Abbildung A.4: Effizienz zählender Verfahren auf BMS-WEBVIEW2 bei variierendem minsupp



(a) Laufzeit in Sekunden (b) Speicherbedarf in Megabyte

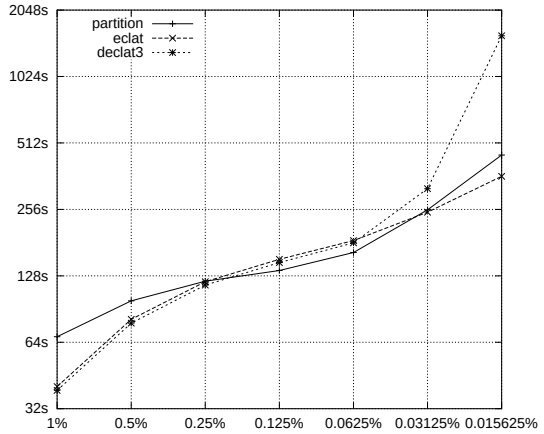
Abbildung A.5: Effizienz zählender Verfahren auf BMS-POS bei variierendem minsupp

A.1.2 Schneidende Verfahren

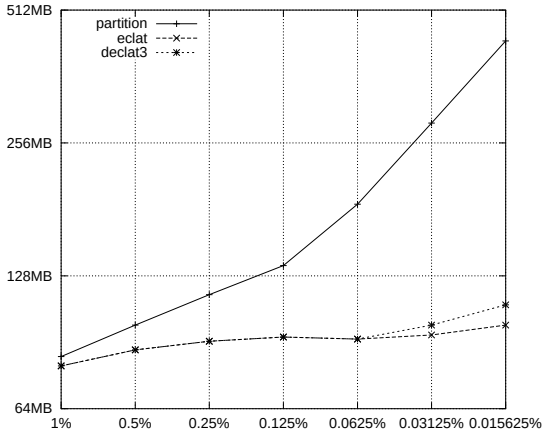


(a) Laufzeit in Sekunden (b) Speicherbedarf in Megabyte

Abbildung A.6: Effizienz schneidender Verfahren auf T5I2D1000K bei variierendem minsupp

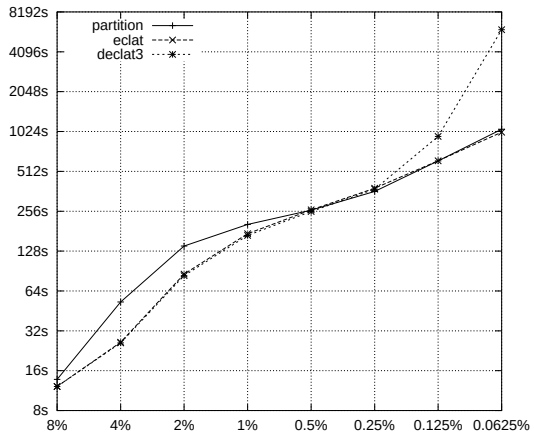


(a) Laufzeit in Sekunden

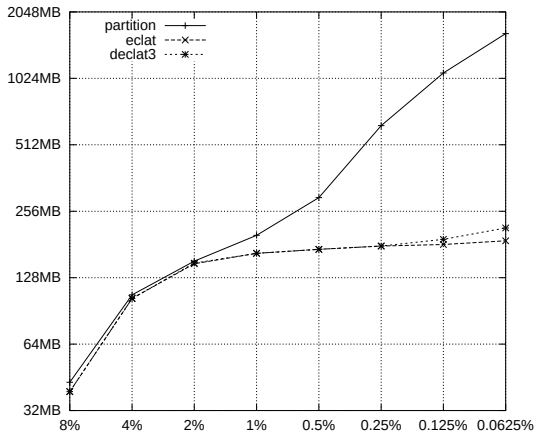


(b) Speicherbedarf in Megabyte

Abbildung A.7: Effizienz schneidender Verfahren auf T10I2D1000K bei variierendem minsupp

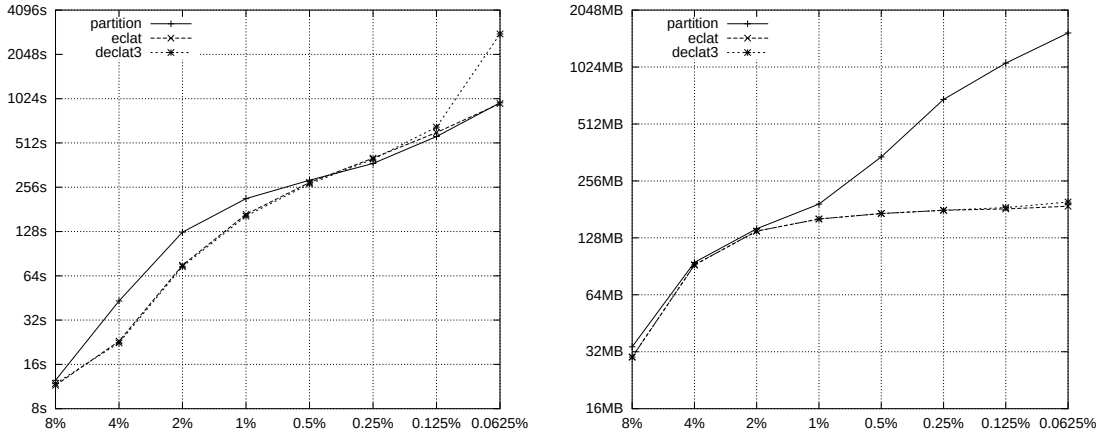


(a) Laufzeit in Sekunden



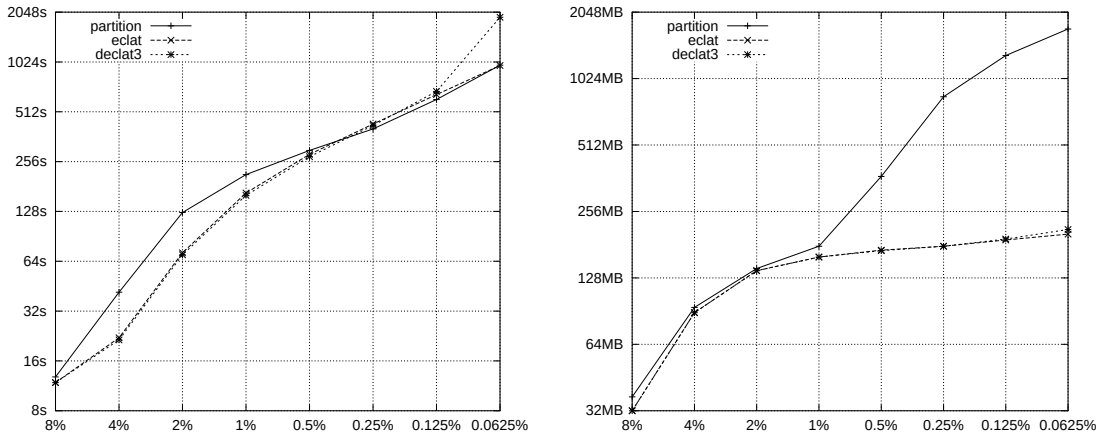
(b) Speicherbedarf in Megabyte

Abbildung A.8: Effizienz schneidender Verfahren auf T20I2D1000K bei variierendem minsupp



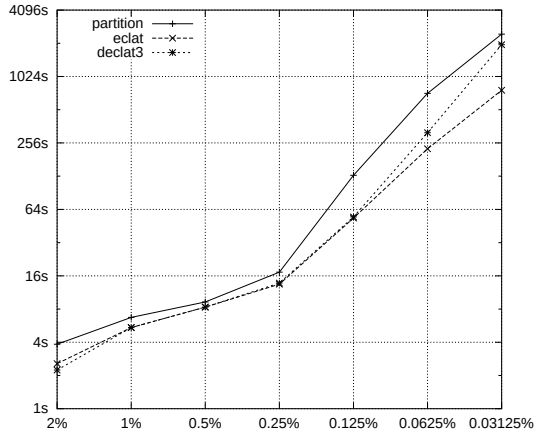
(a) Laufzeit in Sekunden (b) Speicherbedarf in Megabyte

Abbildung A.9: Effizienz schneidender Verfahren auf T20I4D1000K bei variierendem minsupp

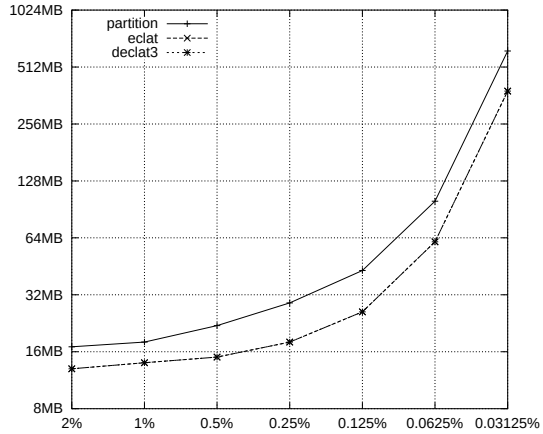


(a) Laufzeit in Sekunden (b) Speicherbedarf in Megabyte

Abbildung A.10: Effizienz schneidender Verfahren auf T20I6D1000K bei variierendem minsupp

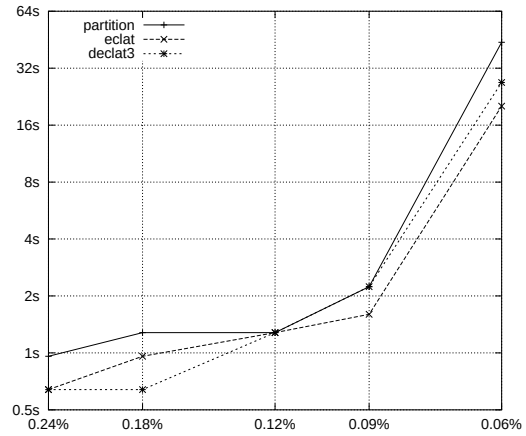


(a) Laufzeit in Sekunden

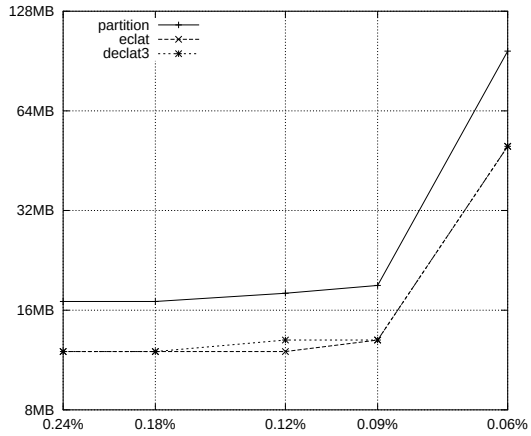


(b) Speicherbedarf in Megabyte

Abbildung A.11: Effizienz schneidender Verfahren auf T25I10D10K bei variierendem minsupp

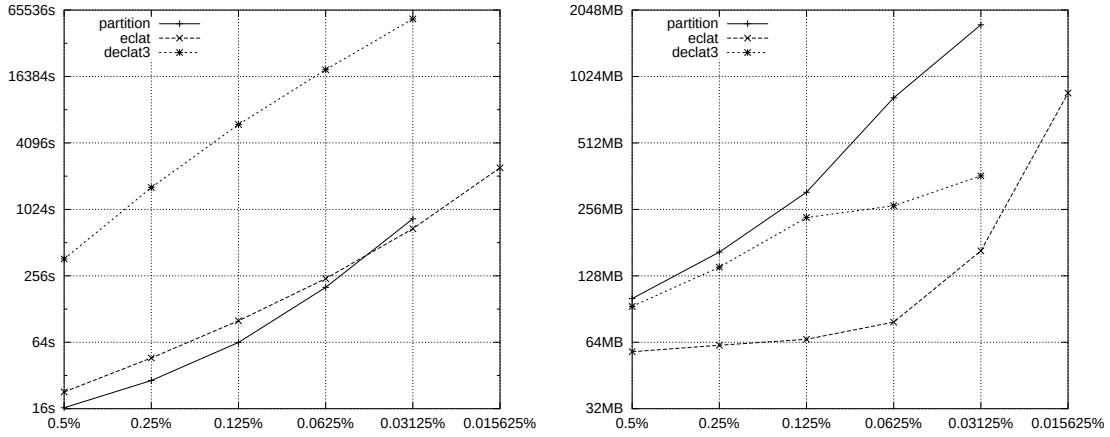


(a) Laufzeit in Sekunden



(b) Speicherbedarf in Megabyte

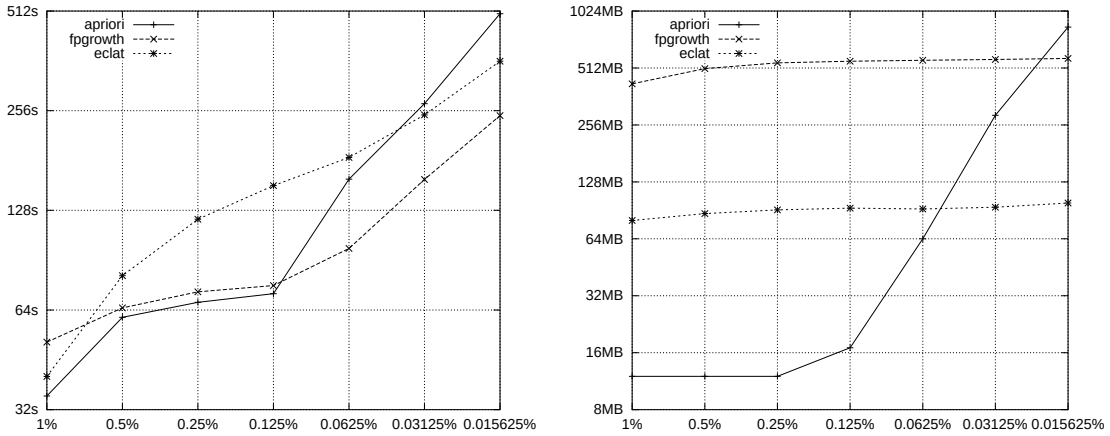
Abbildung A.12: Effizienz schneidender Verfahren auf BMS-WEBVIEW1 bei variierendem minsupp



(a) Laufzeit in Sekunden (b) Speicherbedarf in Megabyte

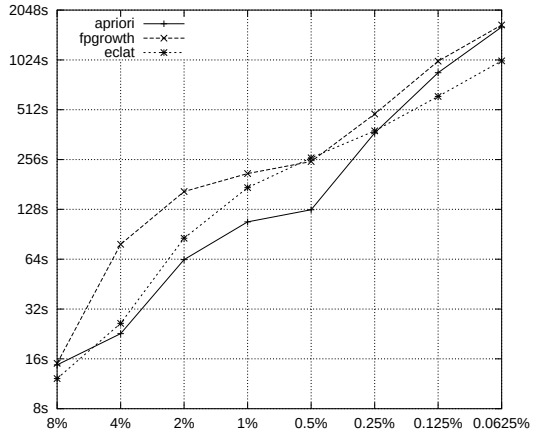
Abbildung A.13: Effizienz schneidender Verfahren auf BMS-POS bei variierendem minsupp

A.1.3 Ausgewählte etablierte Verfahren

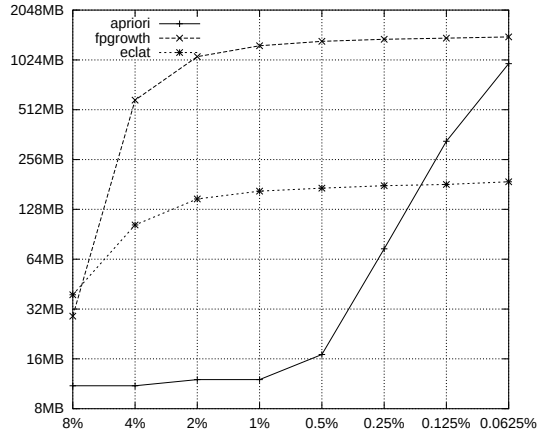


(a) Laufzeit in Sekunden (b) Speicherbedarf in Megabyte

Abbildung A.14: Effizienz ausgewählter etablierter Verfahren auf T10I2D1000K bei variierendem minsupp

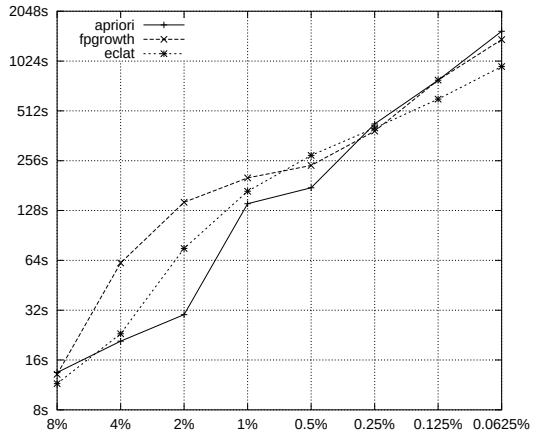


(a) Laufzeit in Sekunden

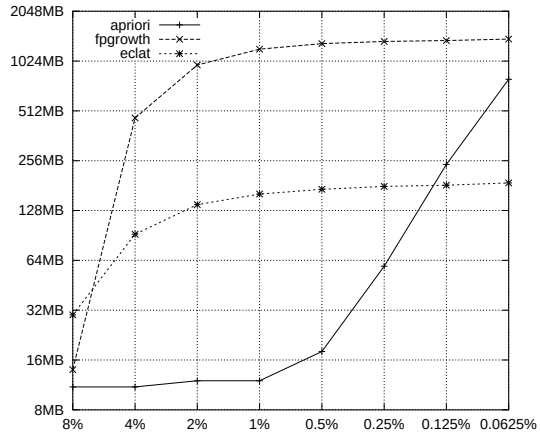


(b) Speicherbedarf in Megabyte

Abbildung A.15: Effizienz ausgewählter etablierter Verfahren auf T20I2D1000K bei variierendem minsupp

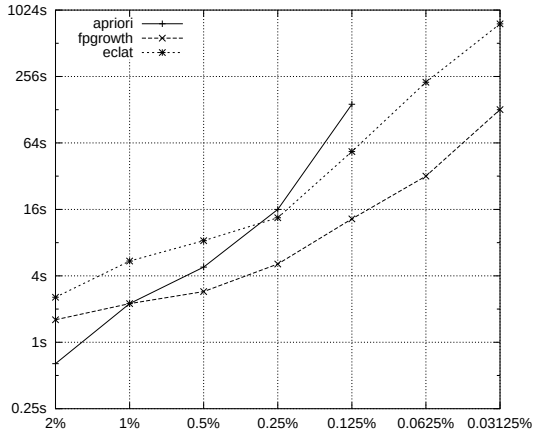


(a) Laufzeit in Sekunden

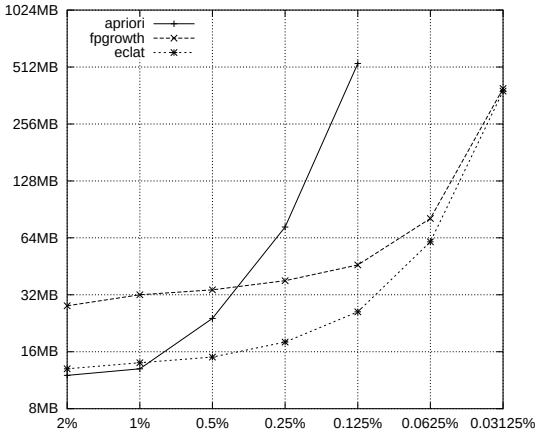


(b) Speicherbedarf in Megabyte

Abbildung A.16: Effizienz ausgewählter etablierter Verfahren auf T20I4D1000K bei variierendem minsupp

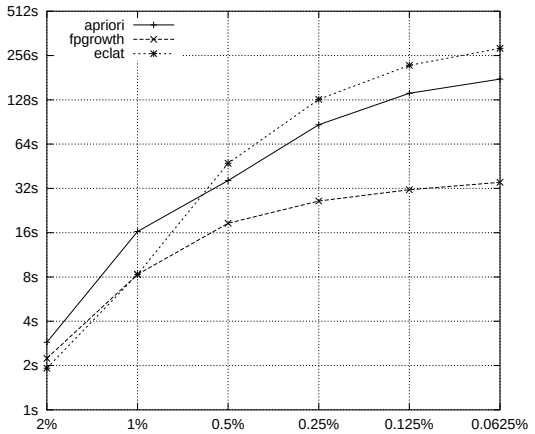


(a) Laufzeit in Sekunden

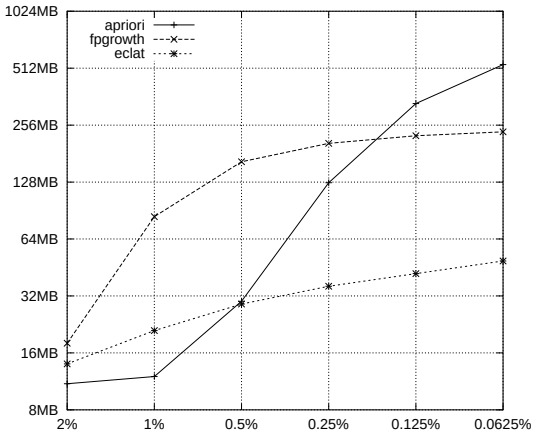


(b) Speicherbedarf in Megabyte

Abbildung A.17: Effizienz ausgewählter etablierter Verfahren auf T25I10D10K bei variierendem minsupp



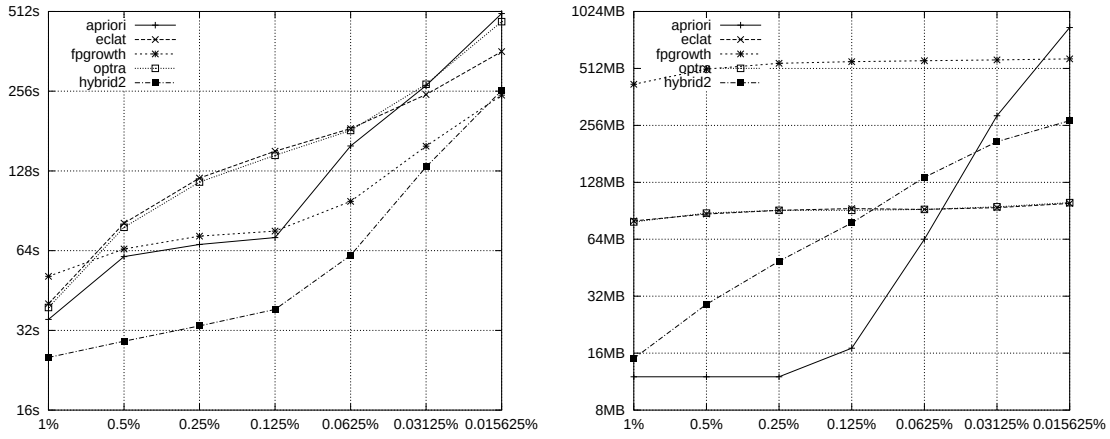
(a) Laufzeit in Sekunden



(b) Speicherbedarf in Megabyte

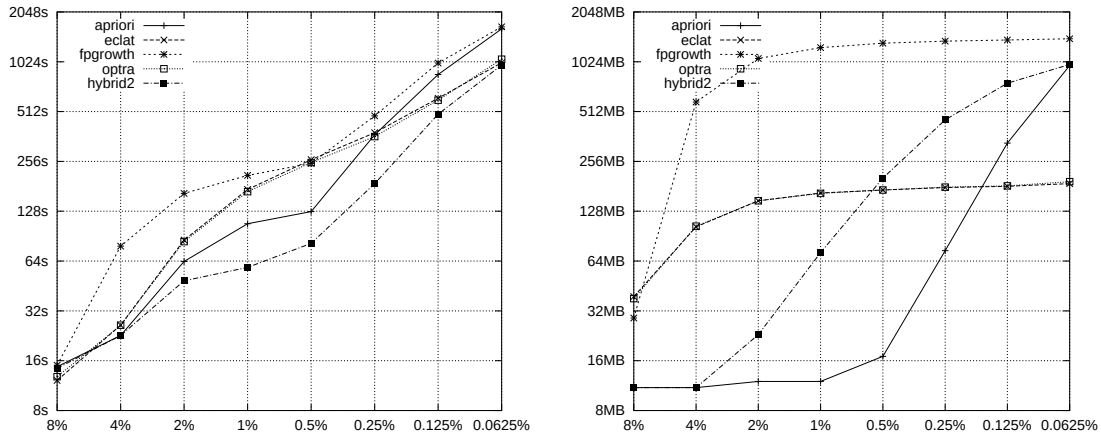
Abbildung A.18: Effizienz ausgewählter etablierter Verfahren auf T25I20D100K bei variierendem minsupp

A.2 Neu entwickelte und etablierte Verfahren zur Generierung einfacher Assoziationsregeln



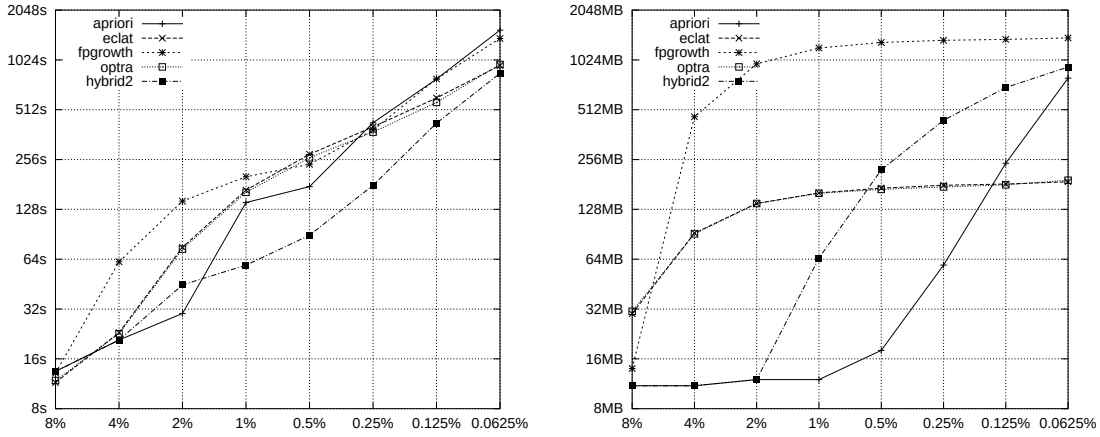
(a) Laufzeit in Sekunden (b) Speicherbedarf in Megabyte

Abbildung A.19: Effizienz ausgewählter Verfahren auf T10I2D1000K bei variierendem minsupp



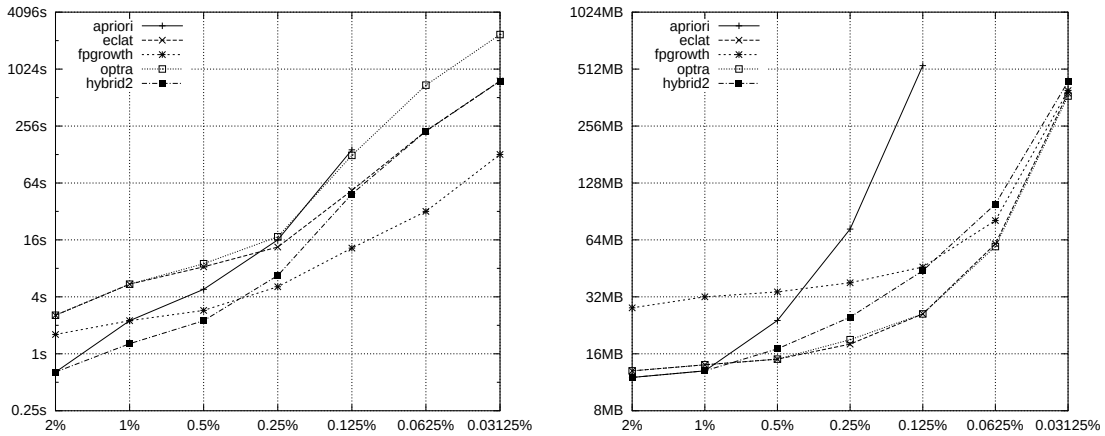
(a) Laufzeit in Sekunden (b) Speicherbedarf in Megabyte

Abbildung A.20: Effizienz ausgewählter Verfahren auf T20I2D1000K bei variierendem minsupp



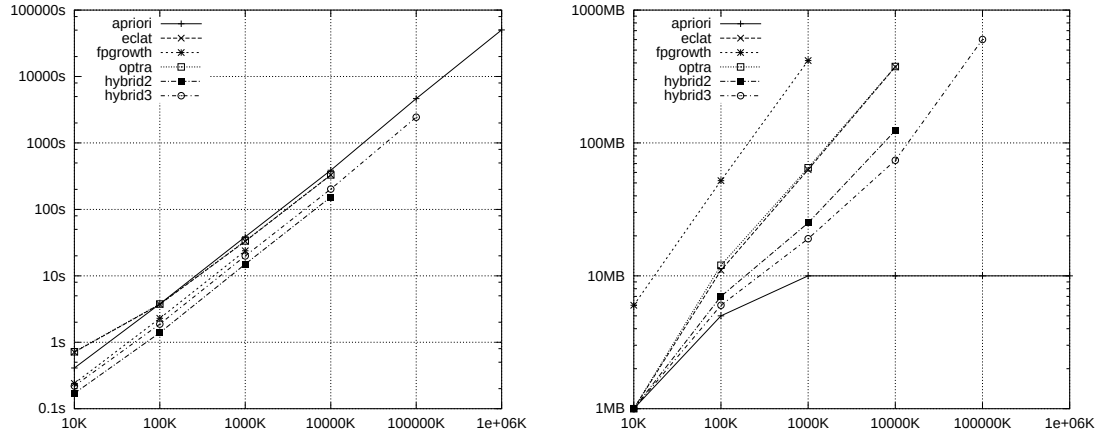
(a) Laufzeit in Sekunden (b) Speicherbedarf in Megabyte

Abbildung A.21: Effizienz ausgewählter Verfahren auf T20I4D1000K bei variierendem minsupp



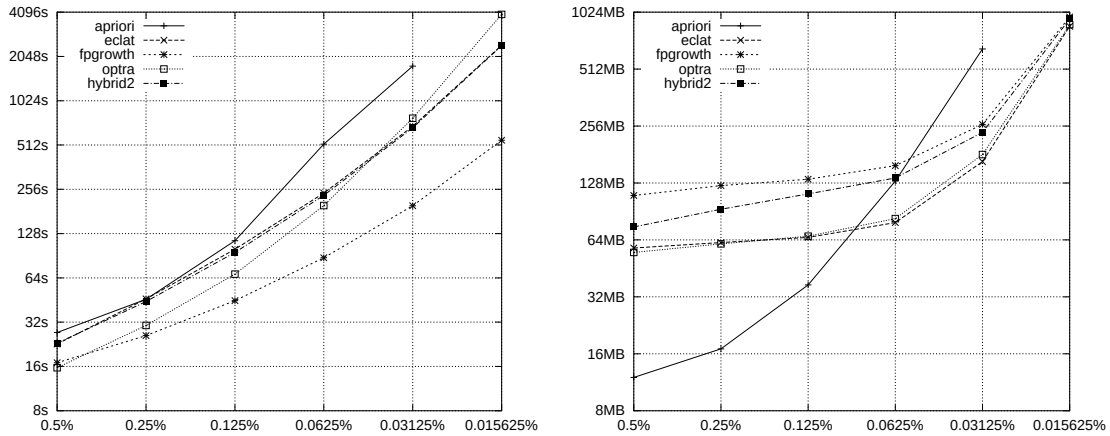
(a) Laufzeit in Sekunden (b) Speicherbedarf in Megabyte

Abbildung A.22: Effizienz ausgewählter Verfahren auf T25I10D10K bei variierendem minsupp



(a) Laufzeit in Sekunden (b) Speicherbedarf in Megabyte

Abbildung A.23: Skalierbarkeit ausgewählter Verfahren auf synthetischen Daten bei variierender Anzahl von Transaktionen in Tausend

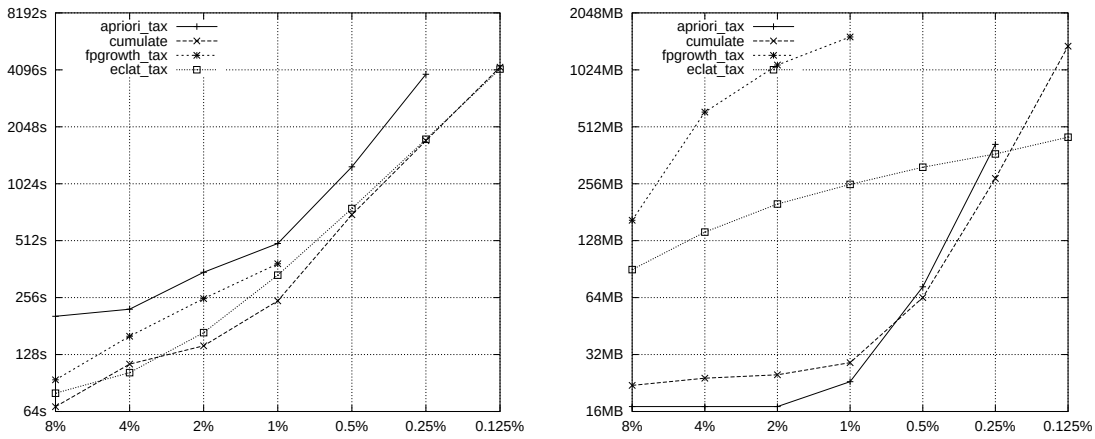


(a) Laufzeit in Sekunden (b) Speicherbedarf in Megabyte

Abbildung A.24: Effizienz ausgewählter Verfahren auf BMSPOS bei variierendem minsupp

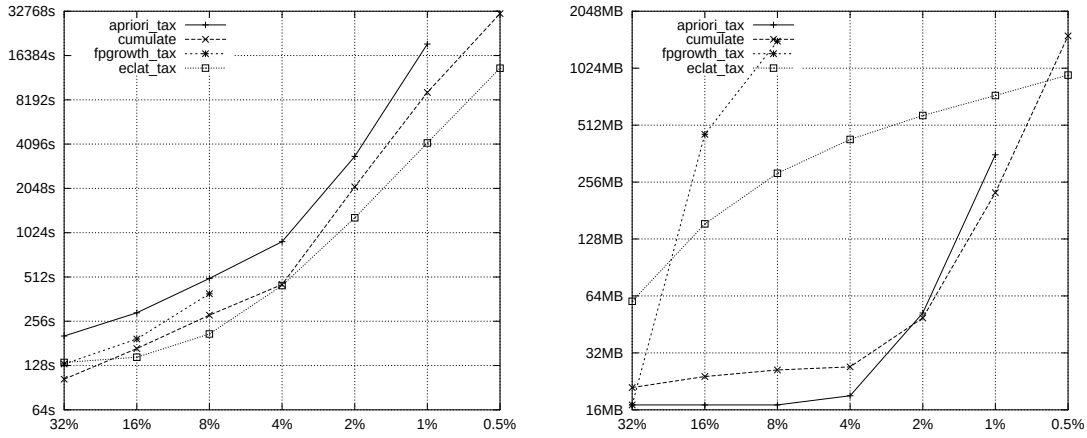
A.3 Taxonome Verfahren

A.3.1 Erweiterung der etablierten Verfahren



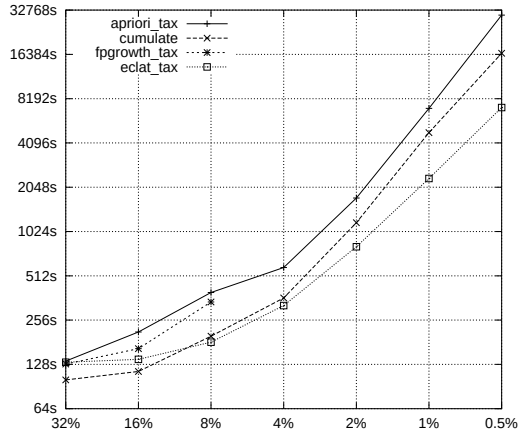
(a) Laufzeit in Sekunden (b) Speicherbedarf in Megabyte

Abbildung A.25: Effizienz erweiterter etablierter Verfahren auf T10I2D1000K bei variierendem minsupp

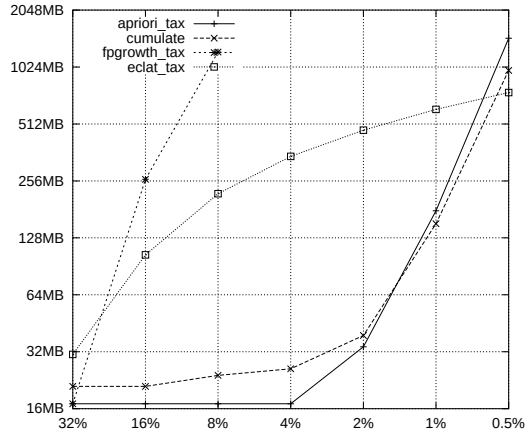


(a) Laufzeit in Sekunden (b) Speicherbedarf in Megabyte

Abbildung A.26: Effizienz erweiterter etablierter Verfahren auf T20I2D1000K bei variierendem minsupp

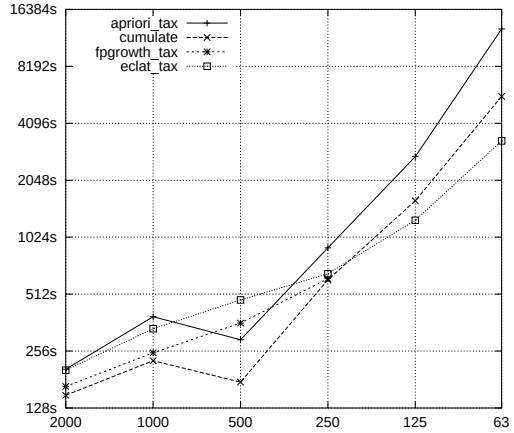


(a) Laufzeit in Sekunden

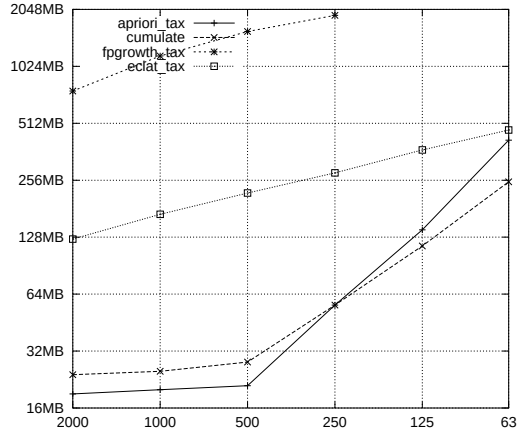


(b) Speicherbedarf in Megabyte

Abbildung A.27: Effizienz erweiterter etablierter Verfahren auf T20I4D1000K bei variierendem minsupp

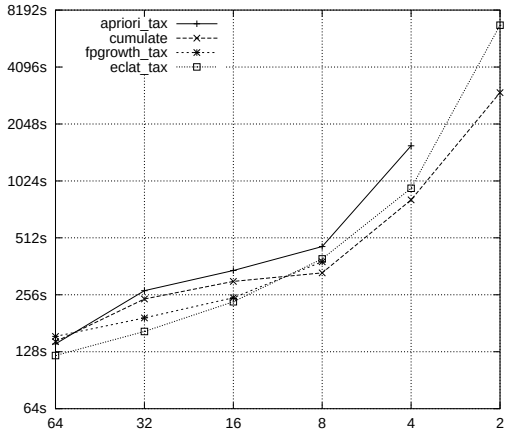


(a) Laufzeit in Sekunden

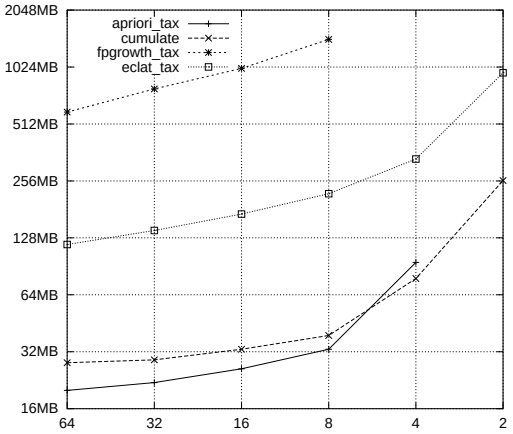


(b) Speicherbedarf in Megabyte

Abbildung A.28: Effizienz erweiterter etablierter Verfahren bei variierender Anzahl Ereignisse auf der zweiten Ebene der Taxonomie

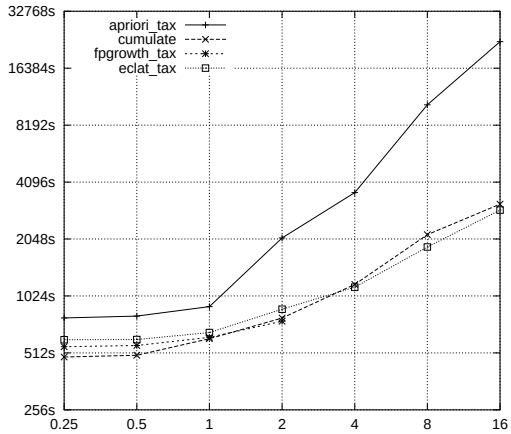


(a) Laufzeit in Sekunden

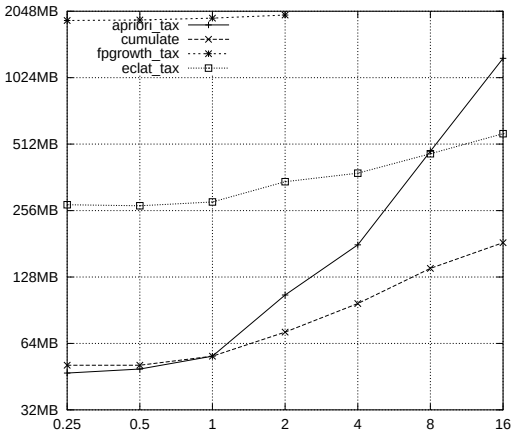


(b) Speicherbedarf in Megabyte

Abbildung A.29: Effizienz taxonomer Verfahren bei variierender mittlerer Anzahl von Söhnen pro Ereignis

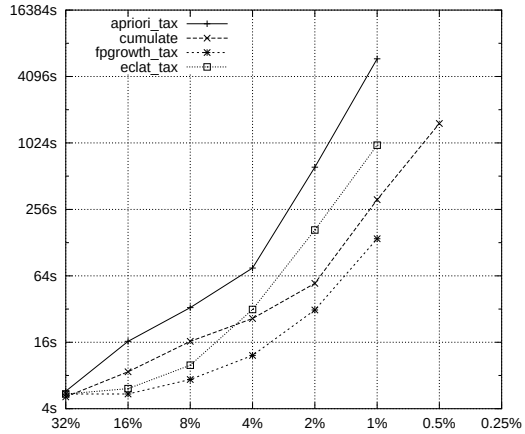


(a) Laufzeit in Sekunden

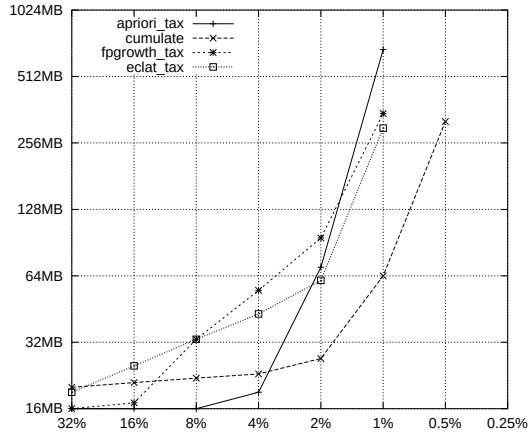


(b) Speicherbedarf in Megabyte

Abbildung A.30: Effizienz erweiterter etablierter Verfahren bei variierendem Wert für das Tiefenverhältnis



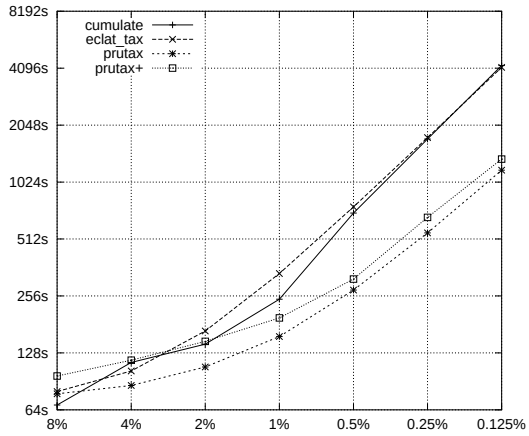
(a) Laufzeit in Sekunden



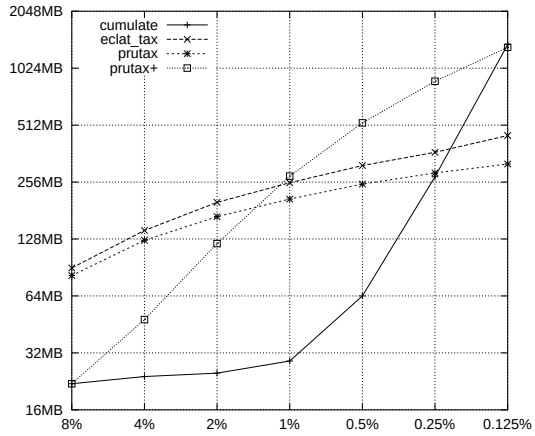
(b) Speicherbedarf in Megabyte

Abbildung A.31: Effizienz erweiterter etablierter Verfahren auf EINZELHANDEL bei variierendem minsupp

A.3.2 Neu entwickelte und etablierte Verfahren

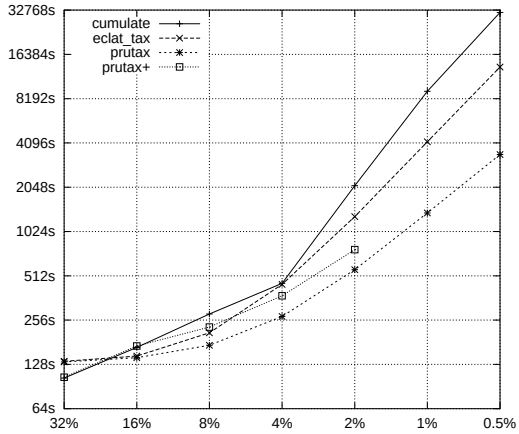


(a) Laufzeit in Sekunden

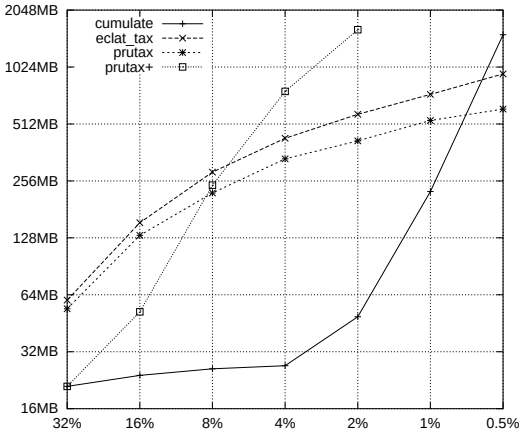


(b) Speicherbedarf in Megabyte

Abbildung A.32: Effizienz ausgewählter taxonomer Verfahren auf T10I2D1000K bei variierendem minsupp

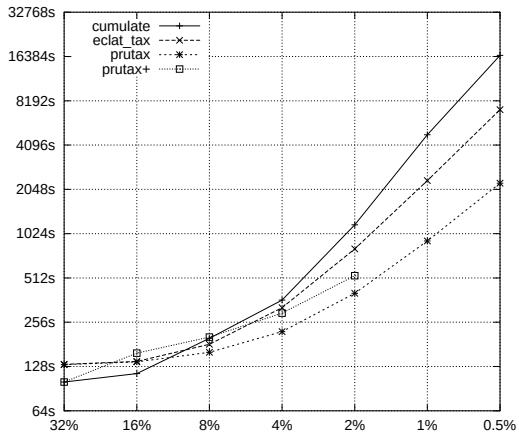


(a) Laufzeit in Sekunden

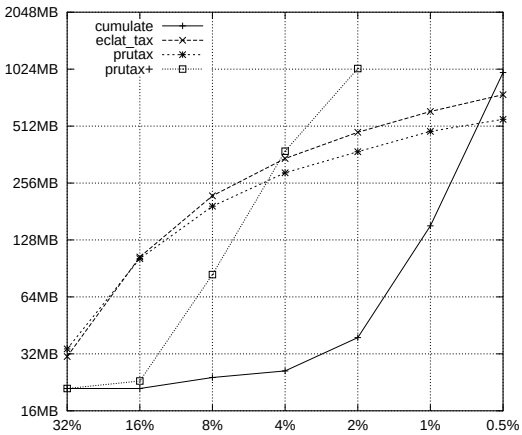


(b) Speicherbedarf in Megabyte

Abbildung A.33: Effizienz ausgewählter taxonomer Verfahren auf T20I2D1000K bei variierendem minsupp



(a) Laufzeit in Sekunden



(b) Speicherbedarf in Megabyte

Abbildung A.34: Effizienz ausgewählter taxonomer Verfahren auf T20I4D1000K bei variierendem minsupp

Literaturverzeichnis

[Adriaans und Zantinge 1996] Adriaans, P. ; Zantinge, D.: *Data Mining*. Harlow, United Kingdom : Addison-Wesley, 1996

[Agarwal et al. 2001] Agarwal, Ramesh C. ; Aggarwal, Charu C. ; Prasad, V. V. V.: A Tree Projection Algorithm For Generation of Frequent Itemsets. In: *Journal of Parallel and Distributed Computing (Special Issue on High Performance Data Mining)* 61 (2001), März, Nr. 3, S. 350–371

[Aggarwal et al. 1998] Aggarwal, Charu C. ; Sun, Zheng ; Yu, Philip Shi-Lung: Online Algorithms for Finding Profile Association Rules. In: *Proceedings of the 1998 ACM CIKM International Conference on Information and Knowledge Management*. Bethesda, Maryland, USA, November 1998, S. 86–95

[Aggarwal und Yu 1997] Aggarwal, Charu C. ; Yu, Philip Shi-Lung: Online Generation of Association Rules / IBM Research Division, T.J. Watson Research Center. Yorktown Heights, New York, USA, Juni 1997 (RC 20899 (92609)). – Forschungsbericht

[Aggarwal und Yu 1998] Aggarwal, Charu C. ; Yu, Philip Shi-Lung: Mining Large Itemsets for Association Rules. In: *Data Engineering Bulletin* 21 (1998), März, Nr. 1, S. 23–31

[Agrawal und Shim 1995] Agrawal, R. ; Shim, K.: Developing Tightly-Coupled Applications on IBM DB2/CS Relational Database System: Methodology and Experience / IBM Almaden Research Center. San Jose, California, USA, 1995 (RJ 10005 (89094)). – Forschungsbericht

[Agrawal und Shim 1996] Agrawal, R. ; Shim, K.: Developing Tightly-Coupled Data Mining Applications on a Relational Database System. In: *Proceedings of the 2nd International Conference on Knowledge Discovery in Databases and Data Mining (KDD '96)*. Portland, Oregon, USA, August 1996, S. 287–290

[Agrawal et al. 1993] Agrawal, Rakesh ; Imielinski, Tomasz ; Swami, Arun: Mining Association Rules between Sets of Items in Large Databases. In: *Procee-*

dings of the ACM SIGMOD International Conference on Management of Data (ACM SIGMOD '93). Washington, USA, Mai 1993, S. 207–216

[Agrawal et al. 1996] Agrawal, Rakesh ; Mannila, Heikki ; Srikant, Ramakrishnan ; Toivonen, Hannu ; Verkamo, Inkeri: Fast Discovery of Association Rules. In: Fayyad, U. M. (Hrsg.) ; Piatetsky-Shapiro, G. (Hrsg.) ; Smyth, P. (Hrsg.) ; Uthurusamy, R. (Hrsg.): *Advances in Knowledge Discovery and Data Mining*. AAAI/MIT Press, 1996, S. 307–328

[Agrawal und Shafer 1996] Agrawal, Rakesh ; Shafer, John C.: Parallel Mining of Association Rules: Design, Implementation and Experience. In: *IEEE Transactions on Knowledge and Data Engineering (TKDE)* 8 (1996), Nr. 6, S. 962–969

[Agrawal und Srikant 1994a] Agrawal, Rakesh ; Srikant, Ramakrishnan: Fast Algorithms for Mining Association Rules. In: *Proceedings of the 20th International Conference on Very Large Databases (VLDB '94)*. Santiago, Chile, Juni 1994, S. 487–499

[Agrawal und Srikant 1994b] Agrawal, Rakesh ; Srikant, Ramakrishnan: Fast Algorithms for Mining Association Rules / IBM Almaden Research Center. San Jose, California, USA, Juni 1994 (RJ9839). – Forschungsbericht

[Agrawal und Srikant 1995] Agrawal, Rakesh ; Srikant, Ramakrishnan: Mining Sequential Patterns. In: *Proceedings of the 11th International Conference on Data Engineering (ICDE)*. Taipei, Taiwan, März 1995, S. 3–14

[Arning et al. 1996] Arning, Andreas ; Agrawal, Rakesh ; Raghavan, Prabhakar: A Linear Method for Deviation Detection in Large Databases. In: *Proceedings of the 2nd International Conference on Knowledge Discovery in Databases and Data Mining (KDD '96)*. Portland, Oregon, USA, August 1996

[Barth 1998] Barth, Tilmann: Guidelines for the Data Mining Process / University of Stuttgart. Stuttgart, Germany, 1998 (ESPRIT Project Number 22700). – Forschungsbericht

[Bartlmae 2002] Bartlmae, Kai: *Die KDD-Experience Factory: Ein Unterstützungsansatz für die Wissensentdeckung in Datenbanken*. Jena, Germany, Wirtschaftswissenschaftliche Fakultät der Friedrich-Schiller-Universität, Dissertation, Februar 2002

[Bauer 1991] Bauer, Heinz: *Wahrscheinlichkeitstheorie*. 4. Aufl. Berlin, New York : de Gryter, 1991

[Bayardo 1997] Bayardo, Roberto J.: Brute-Force Mining of High-Confidence Classification Rules. In: *Proceedings of the 3rd International Conference on KDD and Data Mining (KDD '97)*. Newport Beach, California, USA, August 1997, S. 123–126

[Bayardo 1998] Bayardo, Roberto J.: Efficiently Mining Long Patterns from Databases. In: *Proceedings of the 1998 ACM SIGMOD Conference on Management of Data*. Seattle, Washington, USA, Juni 1998, S. 85–93

[Bayardo 2002] Bayardo, Roberto J.: Special Issue on Constraints in Data Mining: Editorial. In: *SIGKDD Explorations* 4 (2002), Juni, Nr. 1

[Bayardo und Agrawal 1999] Bayardo, Roberto J. ; Agrawal, Rakesh: Mining the Most Interesting Rules. In: *Proceedings of the 5th International Conference on Knowledge Discovery and Data Mining (KDD '99)*. San Diego, California, USA, August 1999, S. 145–154

[Bayardo et al. 1999] Bayardo, Roberto J. ; Agrawal, Rakesh ; Gunopulos, Dimitrios: Constraint-Based Rule Mining in Large, Dense Databases. In: *Proceedings of the 15th International Conference on Data Engineering*. Sydney, Australia, März 1999, S. 188–197

[Berendt et al. 2002] Berendt, B. ; Mobasher, B. ; Nakagawa, M. ; Spiliopoulou, M.: The Impact of Site Structure and User Environment on Session Reconstruction in Web Usage Analysis. In: *Workshop Notes of the 4th WEBKDD: Web Mining for Usage Patterns & User Profiles*. Edmonton, Alberta, Canada, Juli 2002, S. 115–129

[Berendt et al. 2001] Berendt, B. ; Mobasher, B. ; Spiliopoulou, M. ; Wiltshire, J.: Measuring the Accuracy of Sessionizers for Web Usage Analysis. In: *Proceedings of the Workshop on Web Mining at the First SIAM Data Mining Conference (SDM 2001)*. Chicago, Illinois, USA, April 2001, S. 7–14

[Berry und Linoff 1997] Berry, Michael J. A. ; Linoff, Gordon: *Data Mining Techniques: For Marketing, Sales, and Customer Support*. New York, USA : Wiley Computer Publishing, 1997

[Borgelt und Kruse 2002] Borgelt, Christian ; Kruse, Rudolf: Induction of Association Rules: Apriori Implementation. In: *Proceedings of the 15th Conference on Computational Statistics (Compstat 2002)*. Berlin, Germany, August 2002

[Boulicaut 1999] Boulicaut, J. F.: Query Languages for Knowledge Discovery in Databases. In: *Proceedings of Principles of Data Mining and Knowledge*

Discovery, 3rd European Conference, PKDD '99. Prague, Czech Republic, September 1999, S. 582–583

[Boulicaut et al. 1999] Boulicaut, J. F. ; Marcel, P. ; Rigotti, C.: Query Driven Knowledge Discovery in Multidimensional Data. In: *Proceedings of the ACM 2nd International Workshop on Data Warehousing and OLAP DOLAP'99.* Kansas City, USA, November 1999, S. 87–93

[Brachman und Anand 1996] Brachman, Ronald J. ; Anand, Tej: The Process of Knowledge Discovery in Databases: A Human Centered Approach. In: Fayyad, U. M. (Hrsg.) ; Piatetsky-Shapiro, G. (Hrsg.) ; Smyth, P. (Hrsg.) ; Uthurusamy, R. (Hrsg.): *Advances in Knowledge Discovery and Data Mining.* AAAI/MIT Press, 1996, S. 37–57

[Brin et al. 1997a] Brin, Sergey ; Motwani, Rajeev ; Silverstein, Craig: Beyond Market Baskets: Generalizing Association Rules to Correlations. In: *Proceedings of the ACM SIGMOD International Conference on Management of Data (ACM SIGMOD '97)*, 1997, S. 265–276

[Brin et al. 1997b] Brin, Sergey ; Motwani, Rajeev ; Ullman, Jeffrey D. ; Tsur, Shalom: Dynamic Itemset Counting and Implication Rules for Market Basket Data. In: *Proceedings of the ACM SIGMOD International Conference on Management of Data (ACM SIGMOD '97)*, 1997, S. 265–276

[Büchter und Wirth 1998] Büchter, Oliver ; Wirth, Rüdiger: Discovery of Association Rules over Ordinal Data: A New and Faster Algorithm and its Application to Basket Analysis. In: *Proceedings of the 2nd Pacific-Asia Conference on Knowledge Discovery and Data Mining, (PAKDD '98).* Melbourne, Australia, April 1998, S. 36–47

[Bykowski und Rigotti 2001] Bykowski, Artur ; Rigotti, Christophe: A Condensed Representation to Find Frequent Patterns. In: *Proceedings of the 20th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems on Principles of Database Systems.* Santa Barbara, California, USA, Mai 2001, S. 267–273

[Chamberlin 1998] Chamberlin, Donald D.: *A Complete Guide to DB2 Universal Database.* San Francisco : Morgan Kaufmann Publishers, 1998

[Chan und Au 1997] Chan, Keith C. C. ; Au, Wai-Ho: Mining Fuzzy Association Rules. In: *Proceedings of the 6th International Conference on Information and Knowledge Management (CIKM'97).* Las Vegas, Nevada, USA, November 1997, S. 209–215

[Chapman et al. 2000] Chapman, Pete ; Clinton, Julian ; Kerber, Randy ; Khabaza, Thomas ; Reinartz, Thomas ; Shearer, Colin ; Wirth, Rüdiger: *CRISP-DM 1.0*. http://www.crisp-dm.org/ (Zugriff: 14.08.2003). 2000

[Chen et al. 1996] Chen, Ming-Syan ; Han, Jiawei ; Yu, Philip S.: Data Mining: An Overview from Database Perspective. In: *IEEE Transactions On Knowledge And Data Engineering (TKDE)* 6 (1996), Nr. 8, S. 866–883

[Cheung et al. 1997] Cheung, D. ; Lee, S. ; Kao, B.: A General Incremental Technique for Maintaining Discovered Association Rules. In: *Proceedings of the 5th International Conference on Database Systems for Advanced Applications*. Melbourne, Australia, März 1997, S. 185–194

[Cheung et al. 1996] Cheung, David W. ; Han, Jiawei ; Ng, Vincent T. ; Wong, C. Y.: Maintenance of Discovered Association Rules in Large Databases: An Incremental Updating Technique. In: *Proceedings of the 1996 International Conference on Data Engineering (ICDE'96)*. New Orleans, Louisiana, USA, Februar 1996, S. 106–114

[Clementini et al. 2000] Clementini, Eliseo ; Felice, Paolino D. ; Koperski, Krzysztof: Mining multiple-level spatial association rules for objects with a broad boundary. In: *Data & Knowledge Engineering* 34 (2000), Nr. 3, S. 251–270

[Cooley 2000] Cooley, R.: *Web Usage Mining: Discovery and Application of Interesting Patterns from Web Data*. USA, University of Minnesota, PhD Thesis, 2000

[Cristofor et al. 2000] Cristofor, Dana ; Cristofor, Laurentiu ; Simovici, Dan A.: Galois Connections and Data Mining. In: *Journal of Universal Computer Science* 6 (2000), Nr. 1, S. 60–73

[Dalkilic et al. 1995] Dalkilic, Mehmet M. ; Robertson, Edward L. ; Gucht, Dirk V.: CE: The Classifier-Estimator Framework for Data Mining / Computer Science, Indiana University. Bloomington, IN 47405, USA, 1995 (No. 480). – Forschungsbericht

[Date 2000] Date, C. J.: *An Introduction to Database Systems*. 7. Aufl. Addison Wesley Longman, 2000

[Domingo et al. 1998] Domingo, Carlos ; Gavald, Ricard ; Watanabe, Osamu: On-line Sampling Methods for Discovering Association Rules / Tokyo Institute of Technology. Meguro-ku Ookayama, Tokyo 152-8552, Japan, Dezember 1998 (TR-C126). – Forschungsbericht

[Fayyad 1998] Fayyad, Usama: Diving into Databases. In: *Database Programming and Design* 3 (1998), März, S. 24–31

[Fayyad et al. 1996a] Fayyad, Usama ; Piatetsky-Shapiro, Gregory ; Smyth, Padhraic: From Data Mining to Knowledge Discovery: An Overview. In: Fayyad, U. M. (Hrsg.) ; Piatetsky-Shapiro, G. (Hrsg.) ; Smyth, P. (Hrsg.) ; Uthurusamy, R. (Hrsg.): *Advances in Knowledge Discovery and Data Mining*. AAAI/MIT Press, 1996, S. 1–29

[Fayyad et al. 1996b] Fayyad, Usama ; Piatetsky-Shapiro, Gregory ; Smyth, Padhraic: From Data Mining to Knowledge Discovery in Databases. In: *AI Magazine* 17 (1996), Nr. 3, S. 37–54

[Fayyad et al. 1996c] Fayyad, Usama ; Piatetsky-Shapiro, Gregory ; Smyth, Padhraic: The KDD Process for Extracting Useful Knowledge from Volumes of Data. In: *Communications of the ACM* 39 (1996), November, Nr. 11, S. 27–34

[Fayyad et al. 1996d] Fayyad, Usama ; Piatetsky-Shapiro, Gregory ; Smyth, Padhraic: Knowledge Discovery and Data Mining: Towards a Unifying Framework. In: *Proceedings of the 2nd International Conference on Knowledge Discovery in Databases and Data Mining (KDD '96)*. Portland, Oregon, USA, August 1996, S. 82–88

[Feldman et al. 1997] Feldman, Ronen ; Amir, Amihood ; Auman, Yonatan ; Zilberstien, Amir: Incremental Algorithms for Association Generation. In: *Proceedings of the 1st Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD '97)*. Singapore, Februar 1997, S. 227–239

[Feuerstein und Pribyl 2002] Feuerstein, Steven ; Pribyl, Bill: *Oracle PL/SQL Programming*. 3. Aufl. O'Reilly & Associates, September 2002

[Freitas 2000] Freitas, Alex A.: Understanding the Crucial Differences Between Classification and Discovery of Association Rules – A Position Paper. In: *SIGKDD Explorations* 2 (2000), Juli, Nr. 1, S. 65–69

[Fu 1996] Fu, Yongjian: *Discovery of Multiple-Level Rules from Large Databases*. Burnaby, Canada, Simon Fraser University, PhD Thesis, Juli 1996

[Fu und Han 1995] Fu, Yongjian ; Han, Jiawei: Meta-Rule-Guided Mining of Association Rules in Relational Databases. In: *Proceedings of the 1995 International Workshop on Knowledge Discovery and Deductive and Object-Oriented Databases (KDOOD'95)*. Singapore, Dezember 1995, S. 39–46

[Fujiwara et al. 2000] Fujiwara, Shinji ; Ullman, Jeffrey D. ; Motwani, Rajeev: Dynamic Miss-Counting Algorithms: Finding Implication and Similarity Rules with Confidence Pruning. In: *Proceedings of the 16th International Conference on Data Engineering (ICDE 2000)*. San Diego, California, Februar 2000, S. 501–511

[Fukuda et al. 1996] Fukuda, Takeshi ; Morimoto, Yasuhiko ; Morishita, Shinichi ; Tokuyama, Takeshi: Mining Optimized Association Rules for Numeric Attributes. In: *Proceedings of the 15th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS '96)*. Montreal, Quebec, Canada, Juni 1996, S. 182–191

[Ganter und Wille 1999] Ganter, Bernhard ; Wille, Rudolf: *Formal Concept Analysis: Mathematical Foundations,*. Springer Verlag, 1999

[Garofalakis et al. 1999] Garofalakis, Minos ; Rastogi, Rajeev ; Seshadri, S. ; Shim, Kyuseok: Data Mining and the Web: Past, Present and Future. In: *Proceedings of the ACM CIKM'99 2nd Workshop on Web Information and Data Management (WIDM'99)*. Kansas City, Missouri, USA, November 1999, S. 43–47

[Goethals und Bussche 2000] Goethals, B. ; Bussche, Jan Van den: On Supporting Interactive Association Rule Mining. In: *Proceedings of the 2nd International Conference on Data Warehousing and Knowledge Discovery (DaWaK 2000)*. London, United Kingdom, September 2000, S. 307–316

[Goethals und Bussche 1999] Goethals, Bart ; Bussche, Jan Van den: A Priori Versus a Posteriori Filtering of Association Rules. In: *Proceedings of the 1999 ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery (DMKD '99)*. Philadelphia, USA, Mai 1999

[Griffith 2002] Griffith, Arthur: *GCC: The Complete Reference*. McGraw-Hill, 2002

[Gyenesei 2001] Gyenesei, Attila: Interestingness Measures for Fuzzy Association Rules. In: *Proceedings of the 5th European Conference on Principles of Data Mining and Knowledge Discovery*. Freiburg, Germany, September 2001, S. 152–164

[Hafez et al. 1999] Hafez, A. ; Deogun, J. S. ; Raghavan, V. V.: The Item-Set Tree: A Data Structure for Data Mining. In: *Proceedings of the 1st International Conference on Data Warehousing and Knowledge Discovery - (DaWaK'99)*. Florence, Italy, August 1999, S. 183–192

[Han et al. 1997a] Han, E. ; Karypis, G. ; Kumar, V. ; Mobasher, B.: Clustering Based on Association Rule Hypergraphs. In: *Proceedings of the Workshop on Research Issues on Data Mining and Knowledge Discovery*. Tucson, Arizona, USA, 1997, S. 9–13

[Han et al. 1997b] Han, Eui-Hong ; Karypis, George ; Kumar, Vipin: Scalable Parallel Data Mining for Association Rules. In: *Proceedings of the ACM SIGMOD International Conference on Management of Data (ACM SIGMOD '97)*. Tucson, Arizona, USA, Mai 1997, S. 277–288

[Han et al. 1996a] Han, J. ; Fu, Y. ; Wang, W. ; Chiang, J. ; Gong, W. ; Koperski, K. ; Li, D. ; Lu, Y. ; Rajan, A. ; Stefanovic, N. ; Xia, B. ; Zaiane, O.: DBMiner: A System for Mining Knowledge in Large Relational Databases. In: *Proceedings of the 2nd International Conference on Knowledge Discovery in Databases and Data Mining (KDD '96)*. Portland, Oregon, USA, August 1996, S. 250–255

[Han et al. 2000] Han, J. ; Pei, J. ; Yin, Y.: Mining Frequent Patterns without Candidate Generation. In: *Proceedings of the 2000 ACM-SIGMOD International Conference on Management of Data*. Dallas, Texas, USA, Mai 2000, S. 1–12

[Han und Fu 1995] Han, Jiawei ; Fu, Yongjian: Discovery of Multiple-Level Association Rules from Large Databases. In: *Proceedings of the 21st Conference on Very Large Databases (VLDB '95)*. Zürich, Switzerland, September 1995, S. 420–431

[Han et al. 1996b] Han, Jiawei ; Fu, Yongjian ; Wang, Wei ; Koperski, Krysztof ; Zaïane, Osmar R.: DMQL: A Data Mining Query Language for Relational Databases. In: *Proceedings of the 1996 SIGMOD'96 Workshop on Research Issues on Data Mining and Knowledge Discovery (DMKD '96)*. Montreal, Canada, Juni 1996

[Han und Kamber 2001] Han, Jiawei ; Kamber, Micheline: *Data Mining: Concepts and Techniques*. San Francisco, California, USA : Morgan Kaufmann Publishers, 2001

[Han und Pei 2000] Han, Jiawei ; Pei, Jian: Mining Frequent Patterns by Pattern-Growth: Methodology and Implications. In: *SIGKDD Explorations* 2 (2000), Dezember, Nr. 2, S. 30–36

[Hand 1997] Hand, David J.: *Construction and Assessment of Classification Rules*. Sussex, United Kingdom : John Wiley & Sons Ltd., 1997

[Hand et al. 2001] Hand, David J. ; Mannila, Heikki ; Smyth, Padhraic: *Principles of Data Mining*. Cambride, Massachusetts, London, United Kingdom : MIT Press, 2001

[Hätönen et al. 1996] Hätönen, K. ; Klemettinen, M. ; Mannila, H. ; Ronkainen, P. ; Toivonen, H.: Knowledge Discovery from Telecommunication Network Alarm Databases. In: *Proceedings of the 12th International Conference on Data Engineering (ICDE'96)*. New Orleans, Louisiana, USA, Februar 1996, S. 115–122

[Hidber 1998] Hidber, Christian: Online Association Rule Mining / Department of Electrical Engineering and Computer Science, University of California at Berkeley, USA. 1998 (UCB/CSD-98-1004). – Forschungsbericht

[Hidber 1999] Hidber, Christian: Online Association Rule Mining. In: *Proceedings of the 1999 ACM SIGMOD International Conference on Management of Data*. Philadelphia, Pennsylvania, USA, Juni 1999, S. 145–156

[Hilderman und Hamilton 2002] Hilderman, Robert J. ; Hamilton, Howard J.: *Knowledge Discovery and Measures of Interest*. Boston, USA : Kluwer Academic, 2002 (The Kluwer International Series in Engineering and Computer Science)

[Hipp und Güntzer 2002] Hipp, Jochen ; Güntzer, Ulrich: Is Pushing Constraints Deeply into the Mining Algorithms Really What We Want? – An Alternative Approach for Association Rule Mining. In: *SIGKDD Explorations* 4 (2002), Juni, Nr. 1, S. 50–55

[Hipp et al. 2001a] Hipp, Jochen ; Güntzer, Ulrich ; Grimmer, Udo: Data Quality Mining - Making a Virtue of Necessity. In: *Proceedings of the 6th ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery (DMKD 2001)*. Santa Barbara, California, USA, Mai 2001, S. 52–57

[Hipp et al. 2001b] Hipp, Jochen ; Güntzer, Ulrich ; Grimmer, Udo: Integrating Association Rule Mining Algorithms with Relational Database Systems. In: *Proceedings of the 3rd International Conference on Enterprise Information Systems (ICEIS 2001)*. Setúbal, Portugal, Juli 2001, S. 130–137

[Hipp et al. 2000a] Hipp, Jochen ; Güntzer, Ulrich ; Nakhaeizadeh, Gholamreza: Algorithms for Association Rule Mining – A General Survey and Comparison. In: *SIGKDD Explorations* 2 (2000), Juli, Nr. 1, S. 58–64

[Hipp et al. 2000b] Hipp, Jochen ; Güntzer, Ulrich ; Nakhaeizadeh, Gholamreza: Mining Association Rules: Deriving a Superior Algorithm by Analysing

Today's Approaches. In: *Proceedings of the 4th European Symposium on Principles of Data Mining and Knowledge Discovery (PKDD '00)*. Lyon, France, September 2000, S. 159–168

[Hipp et al. 2002a] Hipp, Jochen ; Güntzer, Ulrich ; Nakhaeizadeh, Gholamreza: Data Mining of Association Rules and the Process of Knowledge Discovery in Databases. In: *Data Mining in E-Commerce, Medicine, and Knowledge Management*. Springer, 2002, S. 15–36

[Hipp und Lindner 1999] Hipp, Jochen ; Lindner, Guido: Analysing Warranty Claims of Automobiles. An Application Description Following the CRISP-DM Data Mining Process. In: *Proceedings of 5th International Computer Science Conference (ICSC '99)*. Hong Kong, China, Dezember 1999, S. 31–40

[Hipp et al. 2002b] Hipp, Jochen ; Mangold, Christoph ; Güntzer, Ulrich ; Nakhaeizadeh, Gholamreza: Efficient Rule Retrieval and Postponed Restrict Operations for Association Rule Mining. In: *Proceedings of the 6th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD'02)*. Taipei, Taiwan, Mai 2002, S. 52–65

[Hipp et al. 1998] Hipp, Jochen ; Myka, Andreas ; Wirth, Rüdiger ; Güntzer, Ulrich: A New Algorithm for Faster Mining of Generalized Association Rules. In: *Proceedings of the 2nd European Symposium on Principles of Data Mining and Knowledge Discovery (PKDD '98)*. Nantes, France, September 1998, S. 74–82

[Holsheimer et al. 1995] Holsheimer, Marcel ; Kersten, Martin ; Mannila, Heikki ; Toivonen, Hannu: A Perspective on Databases and Data Mining. In: *Proceedings of the 1st International Conference on Knowlegde Discovery and Data Mining (KDD '95)*. Montreal, Canada, August 1995, S. 150–155

[Holsheimer und Siebes 1994] Holsheimer, Marcel ; Siebes, Arno: Data Mining: The Search for Knowledge in Databases / Centrum voor Wiskunde en Informatika. CWI, P.O. Box 94079, 1090 GB Amsterdam, The Netherlands, 1994 (CS-R9406 1994). – Forschungsbericht

[Hong et al. 1999] Hong, Tzung-Pei ; Kuo, Chan-Sheng ; Chi, Sheng-Chai: Mining Association Rules from Quantitative Data. In: *Intelligent Data Analysis* 3 (1999), November, Nr. 5, S. 363–376

[Hotz et al. 1999] Hotz, E. ; Nakhaeizadeh, G. ; Petzsche, B. ; Spiegelberger, H.: WAPS, A Data Mining Support Environment for the Planning of Warranty and Goodwill Costs in the Automobile Industry. In: *Proceedings of the 5th International Conference on Knowledge Discovery and Data Mining (KDD '99)*. San Diego, California, USA, August 1999, S. 417–419

[Houtsma und Swami 1993] Houtsma, Maurice ; Swami, Arun: Set-Oriented Mining for Association Rules in Relational Databases / IBM Almaden Research Center. San Jose, California, USA, Oktober 1993 (RJ 9567). – Forschungsbericht

[Houtsma und Swami 1995] Houtsma, Maurice ; Swami, Arun: Set-Oriented Mining for Association Rules in Relational Databases. In: *Proceedings of the 11th International Conference on Data Engineering*. Taipei, Taiwan, März 1995, S. 25–33

[Imielinski und Mannila 1996] Imielinski, Tomasz ; Mannila, Heikki: A Database Perspective on Knowledge Discovery. In: *Communications of the ACM* 39 (1996), November, Nr. 11, S. 58–64

[Imielinski und Virmani 1999] Imielinski, Tomasz ; Virmani, Aashu: MSQL: A Query Language for Database Mining. In: *Data Mining and Knowledge Discovery* 3 (1999), Dezember, Nr. 4, S. 373–408

[Imielinski et al. 1999] Imielinski, Tomasz ; Virmani, Aashu ; Abdulghani, Amin: DMajor - Application Programming Interface for Database Mining. In: *Data Mining and Knowledge Discovery* 3 (1999), Dezember, Nr. 4, S. 347–372

[John 1997] John, Georg H.: *Enhancements to the Data Mining Process*. Stanford, USA, Stanford University, PhD Thesis, März 1997

[Johnson und Dasu 1998] Johnson, T. ; Dasu, T.: Comparing Massive High Dimensional Data Sets. In: *Proceedings of the 5th International Conference on Knowledge Discovery in Databases and Data Mining (KDD '99)*. San Diego, California, USA, August 1998, S. 229–233

[Kamber et al. 1997a] Kamber, Micheline ; Han, Jiawei ; Chiang, Jenny Y.: Metarule-Guided Mining of Multi-Dimensional Association Rules Using Data Cubes. In: *Proceedings of the 3rd International Conference on KDD and Data Mining (KDD '97)*. Newport Beach, California, USA, August 1997, S. 207–210

[Kamber et al. 1997b] Kamber, Micheline ; Han, Jiawei ; Chiang, Jenny Y.: Using Data Cubes for Metarule-Guided Mining of Multi-Dimensional Association Rules / Database Systems Research Laboratory, Simon Fraser University. Burnaby, BC, Canada V5A 1S6, Mai 1997 (CMPT-TR-97-10). – Forschungsbericht

[Kischka 1998] Kischka, Peter: *Kausale Interprepation von Graphen*. S. 152–182, Physica-Verlag, 1998

[Kivinen und Mannila 1993] Kivinen, Jyrki ; Mannila, Heikki: The Power of Sampling in Knowledge Discovery / University of Helsinki, Department of Computer Science. P.O. Box 26 (Teollisuuskatu 23), FIN-00014, Finland, Dezember 1993 (C-1993-66). – Technical Report

[Kivinen und Mannila 1994] Kivinen, Jyrki ; Mannila, Heikki: The Power of Sampling in Knowledge Discovery. In: *Proceedings of the 13th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Theory (PODS '94)*. Minneapolis, Minnesota, USA, Mai 1994, S. 77–85

[Klemettinen et al. 1994] Klemettinen, Mika ; Mannila, Heikki ; Ronkainen, Pirjo ; Toivonen, Hannu ; Verkamo, A. I.: Finding Interesting Rules from Large Sets of Discovered Association Rules. In: *Proceedings of the 3rd International Conference on Information and Knowledge Management*. Gaithersburg, Maryland, USA, 29. Nov - 2. Dez 1994, S. 401–408

[Klemettinen et al. 1996] Klemettinen, Mika ; Mannila, Heikki ; Toivonen, Hannu: Interactive Exploration of Discovered Knowledge: A Methodology for Interaction, and Usability Studies / University of Helsinki, Department of Computer Science. P.O. Box 26 (Teollisuuskatu 23), FIN-00014, Finland, 1996 (C-1996-3). – Forschungsbericht

[Klösgen und Zytkow 2002] Klösgen, Willi (Hrsg.) ; Zytkow, Jan M. (Hrsg.): *Handbook of Data Mining and Knowledge Discovery*. Oxford University Press, 2002

[Koperski und Han 1995] Koperski, Krzysztof ; Han, Jiawei: Discovery of Spatial Association Rules in Geographic Information Databases. In: *Proceedings of the 4th International Symposium on Advances in Spatial Databases (SSD '95)*. Portland, Maine, USA, August 1995, S. 47–66

[Krahl et al. 1998] Krahl, Daniela ; Windheuser, Ulricht ; Zick, Friedrich-Karl: *Data Mining: Einsatz in der Praxis*. Bonn : Addison Wesley, 1998

[Kuok et al. 1998] Kuok, Chan M. ; Fu, Ada Wai-Chee ; Wong, Man H.: Mining Fuzzy Association Rules in Databases. In: *SIGMOD Record(ACM Special Interest Group on Management of Data)* 27 (1998), Nr. 1, S. 41–46

[Lakshmanan et al. 1998] Lakshmanan, L. V. S. ; Ng, R. ; Han, J. ; Pang, A.: Optimization of Constrained Frequent Set Queries: 2-Var Constraints. In: *Proceedings of the 3rd SIGMOD '98 Workshop on Research Issues in Data Mining and Knowledge Discovery (DMKD '98)*. Seattle, Wahington, USA, Juni 1998, S. 157–168

[Lee et al. 1998] Lee, Sau D. ; Cheung, David Wai-Lok ; Kao, Ben: Is Sampling Useful in Data Mining? A Case in the Maintenance of Discovered Association Rules. In: *Data Mining and Knowledge Discovery* 2 (1998), Nr. 3, S. 233–262

[Lent et al. 1997] Lent, B. ; Swami, A. ; Widom, J.: Clustering Association Rules. In: *Proceedings of the 1997 International Conference on Data Engineering (ICDE'97)*. Birmingham, United Kingdom, April 1997, S. 220–231

[Leung et al. 2002] Leung, C. K.-S. ; Lakshmanan, L. V. S. ; Ng, R. T.: Exploiting Succinct Constraints Using FP-trees. In: *SIGKDD Explorations* 4 (2002), Juni, Nr. 1, S. 40–49

[Li et al. 2003a] Li, Dan ; Deogun, Jitender S. ; Harms, Sherri K.: Interpolation Techniques for Geo-spatial Association Rule Mining. In: *Proceedings of the 9th International Conference on Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing (RSFDGrC)*. Chongqing, China, Mai 2003, S. 573–580

[Li et al. 2003b] Li, Tao ; Ogihara, Mitsuniori ; Zhu, Shenghuo: Association-based Similarity Testing and Its Applications. In: *IDA Intelligent Data Analysis* 7 (2003), S. 209–323

[Li et al. 2001] Li, W. ; Han, J. ; Pei, J.: CMAR: Accurate and Efficient Classification Based on Multiple Class-Association Rules. In: *Proceedings of the 2001 International Conference on Data Mining (ICDM'01)*. San Jose, California, USA, November 2001, S. 369–376

[Lin 1998] Lin, Dao-I: *Fast Algorithms for Discovering the Maximum Frequent Set*. USA, New York University, PhD Thesis, September 1998

[Lin und Kedem 1998] Lin, Dao-I ; Kedem, Z.: Pincer-Search: A New Algorithm for Discovering the Maximum Frequent Sets. In: *Proceedings of the 6th International Conference on Extending Database Technology (EDBT)*. Valencia, Spain, 1998, S. 105–119

[Lin und Dunham 1998] Lin, J-L. ; Dunham, M. H.: Mining Association Rules: Anti-skew Algorithms. In: *Proceedings of the 14th International Conference on Data Engineering*. Orlando, Florida, USA, Februar 1998, S. 486–493

[Liu et al. 1998] Liu, B. ; Hsu, W. ; Ma, Y.: Integrating Classification and Association Rule Mining. In: *Proceedings of the 1998 International Conference on KDD and Data Mining (KDD '98)*. New York City, USA, August 1998, S. 80–86

[Liu et al. 1999] Liu, Bing ; Hsu, Wynne ; Ma, Yiming: Mining Association Rules with Multiple Minimum Supports. In: *Proceedings of the 5th International Conference on Knowledge Discovery in Databases and Data Mining (KDD '99).* San Diego, California, USA, August 1999, S. 337–341

[Liu et al. 2000] Liu, Bing ; Hu, Minqing ; Hsu, Wynne: Multi-Level Organisation and Summarization of the Discovered Rules. In: *Proceedings of the 6th ACM-SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '00).* Boston, Massachusetts, USA, August 2000, S. 208–217

[Lu et al. 1998] Lu, H. ; Han, J. ; Feng, L.: Stock Price Movement Prediction and N-Dimensional Inter-Transaction Association Rules. In: *Procedings of the ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery (DMKD '98).* Seattle, Washington, USA, Juli 1998

[Mangold 2000] Mangold, Christoph: *Effizientes Retrieval von Assoziationsregeln im Data Mining-Prozeß.* Deutschland, Wilhelm-Schickard-Institut, Universität Tübingen, Diplomarbeit, 2000

[Mannila 1996] Mannila, Heikki: Data Mining: Machine Learning, Statistics, and Databases. In: *Proceedings of the 8th International Conference on Scientific and Statistical Database Management.* Stockholm, Sweden, Juni 1996, S. 2–9

[Mannila und Toivonen 1996] Mannila, Heikki ; Toivonen, Hannu: Multiple Uses of Frequent Sets and Condensed Representations. In: *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining.* Portland, Oregon, USA, August 1996, S. 189–194

[Mannila et al. 1994a] Mannila, Heikki ; Toivonen, Hannu ; Verkamo, Inkeri: Efficient Algorithms for Discovering Association Rules. In: *AAAI Workshop on Knowledge Discovery in Databases.* Seattle, Washington, USA, Juli 1994, S. 181–192

[Mannila et al. 1994b] Mannila, Heikki ; Toivonen, Hannu ; Verkamo, Inkeri: Improved Methods for Finding Association Rules / University of Helsinki, Department of Computer Science. P.O. 26 (Teollisuuskatu 23), FIN-00014, Finland, Februar 1994 (C-1993-65). – Forschungsbericht

[Mannila et al. 1995] Mannila, Heikki ; Toivonen, Hannu ; Verkamo, Inkeri: Discovering Frequent Episodes in Sequences. In: *Proceedings of the 1st International Conference on Knowlegde Discovery and Data Mining (KDD '95).* Montreal, Canada, August 1995, S. 210–215

[Mannila et al. 1997] Mannila, Heikki ; Toivonen, Hannu ; Verkamo, Inkeri: Discovery of Frequent Episodes in Event Sequences. In: *Data Mining and Knowledge Discovery* 1 (1997), November, Nr. 3, S. 259–289

[Matheus et al. 1993] Matheus, Christopher ; Chan, P. ; Piatetsky-Shapiro, Gregory: Systems for Knowledge Discovery in Databases. In: *IEEE Transactions on Knowledge and Data Engineering* 5 (1993), Juni, S. 903–913

[Mauro und McDougall 2000] Mauro, Jim ; McDougall, Richard: *Solaris Internals: Core Kernel Architecture*. Prentice Hall, 2000

[Megiddo und Srikant 1998] Megiddo, Nimrod ; Srikant, Ramakrishnan: Discovering Predictive Association Rules. In: *Proceedings of the 4th International Conference on Knowledge Discovery in Databases and Data Mining (KDD '98)*. New York, USA, August 1998, S. 274–278

[Meo et al. 1998] Meo, Rosa ; Psaila, Giuseppe ; Ceri, Stefano: A Tightly-Coupled Architecture for Data Mining. In: *Proceedings of the 14th International Conference on Data Engineering*. Orlando, Florida, USA, 1998, S. 316–323

[Meo et al. 1996] Meo, Rosa ; Psaila, Guiseppe ; Ceri, Stefano: A New SQL-like Operator for Mining Association Rules. In: *Proceedings of the 22nd International Conference on Very Large Databases (VLDB '96)*. Mumbai (Bombay), India, September 1996, S. 122–133

[Meretakis und Wüthrich 1999] Meretakis, Dimitris ; Wüthrich, Beat: Extending Naive Bayes Classifiers Using Long Itemsets. In: *Proceedings of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. San Diego, California, USA, August 1999, S. 165–174

[Miller und Yang 1997] Miller, R. J. ; Yang, Y.: Association Rules Over Interval Data. In: *Proceedings of the 1997 ACM SIGMOD International Conference on Management of Data (SIGMOD '97)*. Tucson, Arizona, USA, Mai 1997, S. 452–461

[Mobasher et al. 1996] Mobasher, B. ; Jain, N. ; Han, E. ; Srivastava, J.: Web Mining: Pattern Discovery from World Wide Web Transactions / Department of Computer Science, University of Minnesota. Minneapolis, USA, 1996 (TR-96050). – Forschungsbericht

[Moore et al. 1997] Moore, J. ; Han, E. ; Boley, D. ; Gini, M. ; Gross, R. ; Hastings, K. ; Karypis, G. ; Kumar, V. ; Mobasher, B.: Web Page Categorization and Feature Selection Using Association Rule and Principal Component

Clustering. In: *Proceedings of 7th Workshop on Information Technologies and Systems (WITS'97).* Atlanta, Georgia, USA, Dezember 1997

[Motwani et al. 2000] Motwani, Rajeev ; Cohen, Edith ; Datar, Mayur ; Fujiware, Shinji ; Gionis, Aristides ; Indyk, Piotr ; Ullman, Jeffrey D. ; Yang, Cheng: Finding Interesting Associations without Support Pruning. In: *Proceedings of the 16th International Conference on Data Engineering (ICDE).* San Diego, California, USA, 2000, S. 489–499

[Mueller 1995] Mueller, Andreas: Fast Sequential and Parallel Algorithms for Association Rule Mining: A Comparison / Department of Computer Science, University of Maryland-College Park. College Park, MD 20742, USA, 1995 (CS-TR-3515). – Forschungsbericht

[Nag et al. 1999] Nag, Biswadeep ; Deshpande, Prasad M. ; DeWitt, David J.: Using a Knowledge Cache for Interactive Discovery of Association Rules. In: *Proceedings of the 5th International Conference on Knowledge Discovery and Data Mining (KDD '99).* San Diego, California, USA, August 1999, S. 244–253

[Nakhaeizadeh et al. 1998] Nakhaeizadeh, Gholamreza ; Reinartz, Thomas ; Wirth, Rüdiger: *Wissensentdeckung in Datenbanken und Data Mining: Ein Überblick.* S. 1–33, Physica-Verlag, 1998

[Nestorov und Tsur 1999] Nestorov, Svetlozar ; Tsur, Dick: Integrating Data Mining with Relational DBMS: A Tightly-Coupled Approach. In: *Proceedings of the 4th Workshop on Next Generation Information Technologies and Systems NGITS '99,.* Zikhron-Yaakov, Israel, Juli 1999, S. 295–311

[Ng et al. 1999] Ng, R. ; Lakshmanan, L. V. S. ; Han, J. ; Mah, Teresa: Exploratory Mining via Constrained Frequent Set Queries. In: *Proceedings of the 1999 ACM-SIGMOD International Conference on Management of Data (SIGMOD '99).* Philadelphia, USA, Juni 1999, S. 556–558

[Ng et al. 1998] Ng, R. ; Lakshmanan, L. V. S. ; Han, J. ; Pang, A.: Exploratory Mining and Pruning Optimizations of Constrained Associations Rules. In: *Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data (SIGMOD '98).* Seattle, Washington, USA, Juni 1998, S. 13–24

[Ordonez und Omiecinski 1999] Ordonez, Carlos ; Omiecinski, Edward: Discovering Association Rules Based on Image Content. In: *Proceedings of the IEEE Forum on Research and Technology Advances in Digital Libraries.* Baltimore, Maryland, USA, Mai 1999, S. 38–49

[Park et al. 1995a] Park, Jong S. ; Chen, Ming-Syan ; Yu, Philip S.: An Effective Hash-Based Algorithm for Mining Association Rules. In: *Proceedings of the 1995 ACM SIGMOD International Conference on Management of Data.* San Jose, California, USA, Mai 1995, S. 175–186

[Park et al. 1995b] Park, Jong S. ; Chen, Ming-Syan ; Yu, Philip S.: Efficient Parallel Data Mining for Association Rules. In: *Proccedings of the 1995 International Conference on Information and Knowledge Management (CIKM '95).* Baltimore, Maryland, USA, Dezember 1995, S. 31–36

[Park et al. 1997] Park, Jong S. ; Chen, Ming-Syan ; Yu, Phillip S.: Using a Hash-Based Method with Transaction Trimming and Database Scan Reduction for Mining Association Rules. In: *IEEE Transactions on Knowledge and Data Engineering* 9 (1997), September, Nr. 5, S. 813–825

[Parthasarathy et al. 1998] Parthasarathy, S. ; Zaki, M. ; Li, W.: Memory Placement Techniques for Parallel Association Mining. In: *Proceedings of the 1998 International Conference on KDD and Data Mining (KDD '98).* New York City, USA, August 1998, S. 304–308

[Pasquier 2000] Pasquier, Nicolas: *Data Mining: Algorithmes d'Extraction et de Réduction des Règles d'Association dans les Bases de Données.* Clermont-Ferrand, France, Université Clermont-Ferrand, Phd Thesis, Januar 2000

[Pasquier et al. 1999] Pasquier, Nicolas ; Bastide, Yves ; Taouil, Rafik ; Lakhal, Lotfi: Discovering Frequent Closed Itemsets for Association Rules. In: *Proceedings of the 7th International Conference on Database Theory (ICDT '99).* Jerusalem, Israel, Januar 1999, S. 398–416

[Pasquier et al. 2001] Pasquier, Nicolas ; Bastide, Yves ; Taouil, Rafik ; Lakhal, Lotfi: Closed Set Based Discovery of Small Covers. In: *Networking and Information Systems* 3 (2001), Juni, Nr. 2, S. 349–377

[Pei und Han 2002] Pei, J. ; Han, J.: Constrained Frequent Pattern Mining: A Pattern-Growth View. In: *SIGKDD Explorations* 4 (2002), Juni, Nr. 1, S. 31–39

[Pei et al. 2000] Pei, J. ; Han, J. ; Mao, R.: An Efficient Algorithm for Mining Frequent Closed Itemsets. In: *Proceedings of the 2000 ACM-SIGMOD International Conference on Management of Data.* Dallas, Texas, USA, Mai 2000, S. 21–30

[Piatetsky-Shapiro 1991] Piatetsky-Shapiro, Gregory: Discovery, Analysis, and Presentation of Strong Rules. In: Piatetsky-Shapiro, G. (Hrsg.) ; Frawley, W. J. (Hrsg.): *Knowledge Discovery in Databases.* AAAI/MIT Press, 1991, S. 229–238

[Piatetsky-Shapiro 2000] Piatetsky-Shapiro, Gregory: *KDnuggets News 00:12*. http://www.kdnuggets.com (Zugriff: 14.08.2003). Juni 2000

[Piatetsky-Shapiro 2002a] Piatetsky-Shapiro, Gregory: *KDnuggets News 02:15*. http://www.kdnuggets.com (Zugriff: 14.08.2003). August 2002

[Piatetsky-Shapiro 2002b] Piatetsky-Shapiro, Gregory: *KDnuggets News 02:19*. http://www.kdnuggets.com (Zugriff: 14.08.2003). Oktober 2002

[Quinlan 1993] Quinlan, Ross: *C4.5 Programs for Machine Learning*. San Mateo, USA : Morgan Kaufmann, 1993

[Quinlan 2000] Quinlan, Ross: KDD-99 Panel on Last 10 and Next 10 Years. In: *SIGKDD Explorations* 1 (2000), Januar, Nr. 2, S. 62

[Rajamani und Cox 1999] Rajamani, Karthick ; Cox, Alan: Efficient Mining for Association Rules with Relational Database Systems. In: *Proceedings of the 1999 International Database Engineering and Applications Symposium (IDEAS '99)*. Montreal, Canada, August 1999, S. 148–155

[Ramaswamy et al. 1998] Ramaswamy, Sidhar ; Mahajan, Sameer ; Silberschatz, Avi: On the Discovery of Interesting Patterns in Association Rules. In: *Proceedings of the 24th International Conference on Very Large Data Bases (VLDB '98)*. New York, USA, August 1998, S. 368–379

[Ramkumar und Swami 1998] Ramkumar, G. ; Swami, A.: Clustering Data Without Distance Functions. In: *IEEE Data Engineering Bulletin* 21 (1998), März, Nr. 1, S. 9–14

[Rushing et al. 2002] Rushing, John A. ; Ranganath, Heggere ; Hinke, Thomas H. ; Graves, Sara J.: Image Segmentation Using Association Rule Features. In: *IEEE Transactions on Image Processing* 11 (2002), Mai, Nr. 5, S. 558–567

[Sahar 1999] Sahar, Sigal: Interestingness via What is not Interesting. In: *Proceedings of the 5th International Conference on Knowledge Discovery in Databases and Data Mining (KDD '99)*. San Diego, California, USA, August 15-18 1999, S. 332–336

[Sarawagi et al. 1998a] Sarawagi, Sunita ; Thomas, Shiby ; Agrawal, Rakesh: Integrating Association Rule Mining with Relational Database Systems: Alternatives and Implications. In: *SIGMOD Record (ACM Special Interest Group on Management of Data)* 27 (1998), Nr. 2, S. 343–355

[Sarawagi et al. 1998b] Sarawagi, Sunita ; Thomas, Shiby ; Agrawal, Rakesh: Integrating Association Rule Mining with Relational Database Systems: Alternatives and Implications / IBM Almaden Research Center. San Jose, California, USA, März 1998 (RJ 10107 (91923)). – Forschungsbericht

[Saunders 1999] Saunders, Rebecca: *Business the Amazon.Com Way: Secrets of the World's Most Astonishing Web Business (Business Way)*. Capstone Publishing, 1999

[Savasere et al. 1995a] Savasere, Ashok ; Omiecinski, Edward ; Navathe, Shamkat: An Efficient Algorithm for Mining Association Rules in Large Databases. In: *Proceedings of the 21st Conference on Very Large Databases (VLDB '95)*. Zürich, Switzerland, September 1995, S. 432–444

[Savasere et al. 1995b] Savasere, Ashok ; Omiecinski, Edward ; Navathe, Shamkat: An Efficient Algorithm for Mining Association Rules in Large Databases / College of Computing, Georgia Institute of Technology. Atlanta, USA, 1995 (GIT-CC-95-04). – Forschungsbericht

[Savasere et al. 1998] Savasere, Ashok ; Omiecinski, Edward ; Navathe, Shamkat: Mining for Strong Negative Associations in a Large Database of Customer Transactions. In: *Proceedings of the International Conference on Data Engineering (ICDE '98)*, Februar 1998, S. 494–502

[Shah et al. 1999] Shah, Devavrat ; Lakshmanan, L. V. S. ; Ramamritham, Krithi ; Sudarshan, S.: Interestingness and Pruning of Mined Patterns. In: *Proceedings of the 1999 ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery (DMKD '99)*. Philadelphia, USA, 1999

[Shintani und Kitsuregawa 1998] Shintani, Takahiko ; Kitsuregawa, Masaru: Parallel Mining Algorithms for Generalized Association Rules with Classification Hierarchy. In: *SIGMOD Record (ACM Special Interest Group on Management of Data)* 27 (1998), Nr. 2, S. 25–36

[Skillicorn 1998] Skillicorn, D.B.: Strategies for Parallelizing Data Mining. In: *Proceedings of the 1st Workshop on High Performance Data Mining (in Conjunction with IPPS'98)*. Orlando, Florida, USA, März 1998

[Smyth 2000] Smyth, Padhraic: Data Mining: Data Analysis on a Grand Scale? / Information and Computer Science, University of California, Irvine. California, USA, Juli 2000 (UCI-ICS 00-20). – Forschungsbericht

[Sousa et al. 1998] Sousa, M.S. ; Mattoso, M.L.Q. ; Ebecken, N.F.F.: Data Mining: A Tightly-Coupled Implementation on a Parallel Database Server. In:

Proceedings of the 9th International Conference and Workshop on Database and Expert Systems Applications (DEXA'98). Vienna, Austria : IEEE CS Press, August 1998, S. 711–716

[Spiliopoulou et al. 2003] Spiliopoulou, M. ; Mobasher, B. ; Berendt, B. ; Nakagawa, M.: A Framework for the Evaluation of Session Reconstruction Heuristics in Web Usage Analysis. In: *INFORMS Journal on Computing* 15 (2003), Nr. 2

[Spiliopoulou 1999] Spiliopoulou, Myra: The Laborious Way from Data Mining to Web Mining. In: *International Journal of Computer Systems, Sciences & Engineering, Special Issue on Semantics of the Web* 14 (1999), Mar., S. 113–126

[Srikant 1996] Srikant, Ramakrishnan: *Fast Algorithms for Mining Association Rules and Sequential Patterns*. USA, University of Wisconsin, Madison, Phd Thesis, 1996

[Srikant und Agrawal 1995] Srikant, Ramakrishnan ; Agrawal, Rakesh: Mining Generalized Association Rules. In: *Proceedings of the 21st Conference on Very Large Databases (VLDB '95)*. Zürich, Switzerland, September 1995, S. 407–419

[Srikant und Agrawal 1996a] Srikant, Ramakrishnan ; Agrawal, Rakesh: Mining Quantitative Association Rules in Large Relational Tables. In: *Proceedings of the 1996 ACM SIGMOD Conference on Management of Data*. Montreal, Canada, Juni 1996, S. 1–12

[Srikant und Agrawal 1996b] Srikant, Ramakrishnan ; Agrawal, Rakesh: Mining Sequential Patterns: Generalizations and Performance Improvements. In: *Proceedings of the 5th International Conference on Extending Database Technology (EDBT '96)*. Avignon, France, März 1996, S. 3–17

[Srikant et al. 1997] Srikant, Ramakrishnan ; Vu, Quoc ; Agrawal, Rakesh: Mining Association Rules with Item Constraints. In: *Proceedings of the 3rd International Conference on KDD and Data Mining (KDD '97)*. Newport Beach, California, USA, August 1997, S. 67–73

[Stumme et al. 2001] Stumme, Gerd ; Taouil, Rafik ; Bastide, Yves ; Pasquier, Nicolas ; Lakhal, Lotfi: Intelligent Structuring and Reducing of Association Rules with Formal Concept Analysis. In: *Proceedings of Joint German/Austrian Conference on AI*. Vienna, Austria, Juni 2001, S. 349–377

[Tan und Kumar 2000] Tan, Pang-Ning ; Kumar, Vipin: Interestingness Measures for Association Patterns: A Perspective / Department of Computer Science, University of Minnestota, USA. 2000 (TR00-036). – Forschungsbericht

[Thomas et al. 1997] Thomas, Shiby ; Bodagala, Sreenath ; Alsabti, Khaled ; Ranka, Sanja: An Efficient Algorithm for the Incremental Updation of Association Rules in Large Databases. In: *Proceedings of the 3rd International Conference on KDD and Data Mining (KDD '97)*. Newport Beach, California, USA, August 1997, S. 263–266

[Toivonen 1996a] Toivonen, Hannu: *Discovery of Frequent Patterns in Large Data Collections*. P.O. 26 (Teollisuuskatu 23), FIN-00014, Finland, University of Helsinki, Department of Computer Science, PhD Thesis, November 1996

[Toivonen 1996b] Toivonen, Hannu: Sampling Large Databases for Association Rules. In: *Proceedings of the 22nd International Conference on Very Large Databases (VLDB '96)*. Mumbai (Bombay), India, September 1996, S. 134–145

[Tsur et al. 1998] Tsur, Dick ; Ullman, Jeffrey D. ; Abitboul, Serge ; Clifton, Chris ; Motwani, Rajeev ; Nestorov, Svetlozar ; Rosenthal, Arnie: Query Flocks: A Generalization of Association Rule Mining. In: *Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data*. Seattle, Washington, USA, Juni 1998, S. 1–12

[Tung et al. 1999] Tung, Anthony K. ; Lu, Hongjun ; Han, Jiawei ; Feng, Ling: Breaking the Barrier of Transactions: Mining Inter-Transaction Association Rules. In: *Proceedings of the 5th International Conference on Knowledge Discovery in Databases and Data Mining (KDD '99)*. San Diego, California, USA, August 1999, S. 297–301

[Urman 1997] Urman, Scott: *ORACLE 8 - PL/SQL Programming*. 2. Aufl. McGraw-Hill, 1997

[Wang et al. 2000] Wang, Ke ; He, Yu ; Han, Jiawei: Mining Frequent Itemsets Using Support Constraints. In: *Proceedings of the 26th International Conference on Very Large Data Bases (VLDB '00)*. Cairo, Egypt, September 2000, S. 43–52

[Wang et al. 2002] Wang, Ke ; Tang, Liu ; Han, Jiawei ; Liu, Junqiang: Top Down FP-Growth for Association Rule Mining. In: *Proceedings of the 6th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD'02)*. Taipei, Taiwan, Mai 2002, S. 334–340

[Webb 2000] Webb, Geoffrey I.: Efficient Search For Association Rules. In: *Proceedings of the 6th ACM-SIGKDD International Conference on Knowledge*

Discovery and Data Mining (KDD '00). Boston, Massachusetts, USA, August 2000, S. 99–107

[Weber 1998] Weber, Irene: On Pruning Strategies for Discovery of Generalized and Quantitative Association Rules. In: *Proceedings of Knowledge Discovery and Data Mining Workshop on the 4th International Conference on Intelligent Systems*. Singapore, 1998

[Welte 1999] Welte, Oliver: *Evaluierung eines Prozeßmodells für Data Mining Projekte*. Universität Stuttgart, Germany, Institut für Informatik, Diplomarbeit, Februar 1999

[Wille 1992] Wille, Rudolf: Concept Lattices and Conceptual Knowledge Systems. In: *Computers and Mathematics with Applications* 23 (1992), S. 493–515

[Williams und Huang 1996] Williams, Graham J. ; Huang, Zhexue: Modelling the KDD Process / CSIRO Division of Information Technology. GPO Box 664 Canberra ACT 2601 Australia, Februar 1996 (TR DM 96013). – Forschungsbericht

[Wirth et al. 2001] Wirth, Rüdiger ; Borth, Michael ; Hipp, Jochen: When Distribution is Part of the Semantics: A New Problem Class for Distributed Knowledge Discovery. In: *Proceedings of the PKDD 2001 Workshop on Ubiquitous Data Mining for Mobile and Distributed Environments*. Freiburg, Germany, September 2001, S. 56–64

[Wirth und Hipp 2000] Wirth, Rüdiger ; Hipp, Jochen: CRISP-DM: Towards a Standard Process Modell for Data Mining. In: *Proceedings of the 4th International Conference on the Practical Applications of Knowledge Discovery and Data Mining*. Manchester, United Kingdom, April 2000, S. 29–39

[Wittmann 1999] Wittmann, Thomas: *Wissensentdeckung in Datenbanken mit adaptiven Regelsystemen*. Jena, Germany, Wirtschaftswissenschaftliche Fakultät der Friedrich-Schiller-Universität, Dissertation, Juli 1999

[Wrobel et al. 1996] Wrobel, Stefan ; Wettschereck, Dietrich ; Verkamo, A. I. ; Siebes, Arno ; Mannila, Heikki ; Kwakkel, Fred ; Klösgen, Willi: User Interactivity in Very Large Scale Data Mining. In: Dilger, W. (Hrsg.) ; Schlosser, M. (Hrsg.) ; Zeidler, J. (Hrsg.) ; Ittner, A. (Hrsg.): *Proceedings FGML-96 (Annual Meeting of the GI Special Interest Group Machine Learning)*, TU Chemnitz-Zwickau, Germany, 1996, S. 125–130. – URL ftp://ftp.gmd.de/ml-archive/GMD/papers/ML74.ps.gz

[Yang und Singhal 1999] Yang, Yuping ; Singhal, Mukesh: Fuzzy Functional Dependencies and Fuzzy Association Rules. In: *Proceedings of the 1st International Conference on Data Warehousing and Knowledge Discovery (DaWaK '99).* Florence, Italy, August 1999, S. 229–240

[Zaïane 2003] Zaïane, Osmar R.: *Discovering Patterns With and Within Images.* S. 27–42. Norwell, Massachusetts, USA : Kluwer Academic Publishers, 2003

[Zaki et al. 1997a] Zaki, M. J. ; Parthasarathy, S. ; Ogihara, M. ; Li, W.: New Algorithms for Fast Discovery of Association Rules / Computer Science Department, University of Rochester. Rochester, New York, USA, Juli 1997 (651). – Forschungsbericht

[Zaki et al. 1997b] Zaki, M. J. ; Parthasarathy, S. ; Li, W.: A Localized Algorithm for Parallel Association Mining. In: *9th Annual ACM Symposium on Parallel Algorithms and Architectures (SPAA).* Newport, Rhode Island, USA, August 1997, S. 321–330

[Zaki et al. 1997c] Zaki, M. J. ; Parthasarathy, S. ; Ogihara, M. ; Li, W.: New Algorithms for Fast Discovery of Association Rules. In: *Proceedings of the 3rd International Conference on KDD and Data Mining (KDD '97).* Newport Beach, California, USA, August 1997, S. 283–286

[Zaki 1999] Zaki, Mohammed J.: Parallel and Distributed Association Mining: A Survey. In: *IEEE Concurrency, Special Issue on Parallel Mechanisms for Data Mining* 7 (1999), Dezember, Nr. 4, S. 14–25

[Zaki 2000] Zaki, Mohammed J.: Scalable Algorithms for Association Mining. In: *IEEE Transactions on Knowledge and Data Engineering* 12 (2000), Mai, Nr. 3, S. 372–390

[Zaki und Gouda 2001] Zaki, Mohammed J. ; Gouda, Karam: Fast Vertical Mining Using Diffsets / Rensselaer Polytechnic Institute. Troy, New York, USA, Januar 2001 (01-1). – Technical Report

[Zaki et al. 1997d] Zaki, Mohammed J. ; Parthasarathy, Srinivasan ; Ogihara, Mitsunori ; Li, Wei: Parallel Algorithms for Fast Discovery of Association Rules. In: *Data Mining and Knowledge Discovery: An International Journal, Special Issue on Scalable High-Performance Computing for KDD* 1 (1997), Dezember, Nr. 4, S. 343–373

[Zhang et al. 1997] Zhang, Zhaohui ; Lu, Yuchang ; Zhang, Bo: An Effective Partitioning-Combining Algorithm for Discovering Quantitative Association Rules. In: *Proceedings of the 1st Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD '97)*. Singapore, Februar 1997, S. 241–251

[Zheng et al. 2001] Zheng, Zijian ; Kohavi, Ron ; Mason, Llew: Real World Performance of Association Rule Algorithms. In: *Proceedings of the 7th ACM-SIGKDD International Conference on Knowledge Discovery and Data Mining*. San Francisco, California, USA, August 2001, S. 401–406