



Universität Karlsruhe
Fakultät für Informatik
Institut für Programmstrukturen
und Datenorganisation (IPD)



Konzeption und Umsetzung einer Schemaerweiterung durch Kontexte unter Berücksichtigung von Aggregierungsanfragen

STUDIENARBEIT

Andreas Merkel

3. Juni 2004

Verantwortlicher Betreuer: Prof. Dr.-Ing. Dr. h.c. Peter C. Lockemann
Betreuender Mitarbeiter: Dipl.-Inform. Heiko Schepperle

Kurzfassung

Ein Konzept, relationale Schemata so zu erweitern, dass imperfekte Daten abgebildet werden können, ist das Kontextmodell. In dieser Studienarbeit wird das Kontextmodell auf konkrete Messdaten aus dem Verkehr angewendet. Da diese Daten häufig aggregiert werden, muss die Erweiterung so durchgeführt werden, dass Aggregierungsanfragen weiterhin sinnvoll gestellt werden können. Damit dies gewährleistet ist, müssen die Kontexte bestimmte Bedingungen erfüllen.

Die vorliegenden Messdaten – zum einen so genannte Floating Car Daten, die in Fahrzeugen erfasst werden, weiter Simulationsdaten für Verkehrsmessstationen und schließlich Daten einer Statistik über den Güterverkehr in Deutschland – werden auf Imperfektion untersucht. Um diese Imperfektion repräsentieren zu können, wird ein Konzept zur Erweiterung der relationalen Schemata durch geeignete Kontexte ausgearbeitet. Dieses Konzept wird schließlich für die untersuchten Messdaten umgesetzt.

Inhaltsverzeichnis

1	Einleitung	1
1.1	Motivation und Zielsetzung	1
1.2	Umfeld	1
1.3	Verwandte Arbeiten	2
1.4	Vorgehensweise	2
2	Theoretische Überlegungen	3
2.1	Das Kontextmodell	3
2.1.1	Motivation	3
2.2	Scharfe Kontexte	3
2.2.1	Kontexte und Klassen	4
2.2.2	Selektion und Gruppierung	4
2.2.3	Beziehung zum multidimensionalen Datenmodell	4
2.2.4	Kontexte und Hierarchien	5
2.2.5	Hierarchien von Klassen	7
2.3	Unschärfe Kontexte	7
2.3.1	Linguistische Variablen	8
2.3.2	Unschärfe Klassen	8
2.3.3	Selektion und Gruppierung	10
2.3.4	Unschärfe Hierarchien	10
2.4	Aggregation	13
2.4.1	Aggregationsbedingungen	13
2.4.2	Aggregation über unscharfe Klassen	14
2.4.3	Bedingungen für unscharfe Klassen	17
2.4.4	Anforderungen an die Klassenbildung	17
2.4.5	Anforderungen an linguistische Variablen	18
2.4.6	Klassifikation von Objekten, die durch mehrere Tupel beschrieben werden	19
2.4.7	Wiederverwendbarkeit aggregierter Werte	19
2.5	Zusammenfassung	21
3	Analyse der Daten	23
3.1	Imperfektion in Verkehrsdaten	23
3.2	Floating Car Daten	24
3.2.1	Beschreibung des Messverfahrens	24
3.2.2	Erfasste Daten	24
3.2.3	Untersuchung auf Imperfektion	24
3.3	Simulationsdaten für Verkehrsmessstationen	26

3.3.1	Das Programm VISSIM	26
3.3.2	Simulierte Daten	26
3.3.3	Untersuchung auf Imperfektion	27
3.3.4	Vergleich Simulationsdaten – Floating Car Daten	27
3.4	Güterverkehrsdaten	28
3.4.1	Verfahren zur Erhebung der Daten	28
3.4.2	Erhobene Daten	29
3.4.3	Untersuchung auf Imperfektion	29
3.5	Zusammenfassung	30
4	Konzeption	33
4.1	Floating Car Daten	33
4.1.1	Ermitteln des Verkehrszustandes	33
4.1.2	Kontexte	34
4.1.3	Klassenbildung	38
4.1.4	Aggregation	40
4.1.5	Trenngeraden	41
4.2	Simulationsdaten für Verkehrsmessstationen	43
4.2.1	Ermitteln des Verkehrszustandes	43
4.2.2	Kontexte	44
4.2.3	Klassenbildung	45
4.2.4	Aggregation	45
4.3	Güterverkehrsdaten	45
4.3.1	Kontexte	45
4.3.2	Klassenbildung	47
4.3.3	Aggregation	48
4.4	Zusammenfassung	48
5	Umsetzung	49
5.1	Repräsentation von Kontexten	49
5.1.1	Durch zusätzliche Attribute in der Relation	49
5.1.2	Durch eine zusätzliche Relation	50
5.1.3	Durch Angabe von Intervallgrenzen	50
5.1.4	Durch Funktionen	50
5.2	Bestimmung der Klassenzugehörigkeit	51
5.3	Berechnung von Aggregaten für unscharfe Klassen	51
5.4	Floating Car Daten	52
5.4.1	Umsetzung der Kontexte	52
5.4.2	Klassenbildung	55
5.4.3	Anfragen unter Verwendung der Kontexte	56
5.4.4	Trenngeraden	59
5.5	Simulationsdaten für Verkehrsmessstationen	62
5.5.1	Umsetzung der Kontexte	62
5.5.2	Klassenbildung	63
5.5.3	Anfragen unter Verwendung der Kontexte	63
5.6	Güterverkehrsdaten	64
5.6.1	Umsetzung der Kontexte	64

5.6.2	Anfragen unter Verwendung der Kontexte	65
5.7	Zusammenfassung	66
6	Bewertung und Ausblick	67
6.1	Kontexte und Aggregation	67
6.2	Umsetzung von Kontexten	67
6.3	Anwendung des Kontextmodells auf die Verkehrsdaten	68
A	Anhang	69
A.1	Übersicht über die umgesetzten Kontexte	69
A.1.1	Floating Car Daten	69
A.1.2	Simulationsdaten für Verkehrsmessstationen	69
A.1.3	Güterverkehrsdaten	70
A.2	Typen der Attribute	71
A.2.1	Floating Car Daten	71
A.2.2	Simulationsdaten für Verkehrsmessstationen	71
A.2.3	Güterverkehrsdaten	71

1 Einleitung

1.1 Motivation und Zielsetzung

Das relationale Datenmodell ist an sich wenig dafür geeignet, imperfekte Daten abzubilden. Jedoch gibt es Konzepte, relationale Schemata dahingehend zu erweitern, dass imperfekte Information repräsentiert werden kann. Eines dieser Erweiterungskonzepte ist das Kontextmodell. Gegenstand dieser Studienarbeit ist die Anwendung des Kontextmodells auf Daten aus dem Verkehr. Gerade im Verkehrsbereich sind Daten oftmals mit Imperfektion behaftet.

Da Verkehrsdaten häufig aggregiert werden, muss die Schemaerweiterung so erfolgen, dass Aggregierungsanfragen sinnvoll möglich sind. Aus diesem Grund muss geprüft werden, inwieweit Kontexte und Aggregation miteinander vereinbar sind.

Eine große Rolle spielen Aggregierungsoperationen auch im OLAP-Bereich. Daher ist es interessant festzustellen, in welcher Beziehung das Kontextmodell mit dem dort verwendeten multidimensionalen Datenmodell steht. So lassen sich Erkenntnisse über Aggregierbarkeit aus diesem Bereich auf Kontexte übertragen.

Für die konkrete Anwendung des Kontextmodells auf die Verkehrsdaten müssen die Daten zunächst auf Imperfektion untersucht werden. Aufbauend darauf können dann entsprechende Kontexte definiert werden, welche der Imperfektion Rechnung tragen.

Die Schemaerweiterungen sollen schließlich in einem relationalen Datenbanksystem umgesetzt werden. Daher müssen Möglichkeiten gefunden werden, Kontexte im relationalen System zu repräsentieren.

1.2 Umfeld

Die in dieser Arbeit untersuchten Verkehrsmessdaten wurden im Rahmen des Projekts OVID (Stärkung der Selbstorganisationsfähigkeit im Verkehr durch I+K-gestützte Dienste) erworben. Ziel dieses vom Bundesministerium für Bildung und Forschung geförderten Verbundprojekts der Universität Karlsruhe, des Fraunhofer Instituts für Informations- und Datenverarbeitung (Fraunhofer-IITB), der PTV Planung Transport Verkehr AG und der Locom Consulting GmbH ist der Aufbau und die Nutzung einer Plattform zur Modellierung und Bewertung von verkehrsinfrastrukturellen, verkehrstelematischen und logistischen Maßnahmen im Verkehrs- und sozio-ökonomischen System.

Das Projekt ist in mehrere Teilprojekte gegliedert. Im Teilprojekt B1 „Verlässliche Datenbanken für die Informationsbereitstellung im Verkehr“ wird die informationstechnische Infrastruktur in Form erweiterter Datenbanksysteme entwickelt. Das Teilprojekt beschäftigt sich also mit der Speicherung und Verarbeitung von Verkehrsdaten. Hierbei müssen neben den Standardeigenschaften von Datenbanken besonders Verfahren zur Informationsverdichtung in einer verteilten mobilen Umgebung und zum Umgang mit unvollkommenen Daten bereitgestellt werden.

1.3 Verwandte Arbeiten

[FD03] beschreibt die Verwendung linguistischer Variablen im Data Warehouse. Die linguistischen Variablen werden dort auf die quantifizierende Information im multidimensionalen Datenmodell angewendet und so die Kenngrößen durch semantische Beschreibung erweitert. Im Gegensatz dazu werden in dieser Studienarbeit die linguistische Variablen auf die qualifizierende Information angewandt um zu einer unscharfen Strukturierung zu gelangen.

Ähnlich wie in dieser Arbeit werden in [RB91] Aggregatfunktionen dahingehend erweitert, dass sie auch bei unscharfen Einteilungen angewendet werden können. [RB91] verwendet hierbei so genannte α -Partitionen zur Gruppierung, die auf dem Konzept der α -Schnitte aufbauen, so dass nur Tupel ab einer gewissen Mindestzugehörigkeit bei der Aggregation berücksichtigt werden.

[DPJ03] beschäftigt sich mit imperfekter Information in multidimensionalen Datenbanken. Dabei wird auch das Problem nicht überlappungsfreier und nicht vollständiger Einteilungen im Zusammenhang mit Aggregationsoperationen behandelt, allerdings nur für scharfe Einteilungen, bei denen ein Element entweder vollständig oder überhaupt nicht zu einer Kategorie gehört.

1.4 Vorgehensweise

Zunächst werden die theoretischen Grundlagen erarbeitet. Dabei wird das Kontextmodell kurz vorgestellt. Dann wird der Zusammenhang zwischen Kontexten und den Klassifikationshierarchien des multidimensionalen Datenmodells beschrieben und die Vereinbarkeit von Kontexten und Aggregierungsanfragen untersucht. Daran schließt sich eine Beschreibung der vorliegenden Verkehrsdaten sowie die Analyse der Daten auf Imperfektion an. Im nächsten Schritt werden Schemaerweiterungen durch Kontexte konzipiert, welche der in den Daten enthaltenen Imperfektion Rechnung tragen. Außerdem wird untersucht, inwiefern Aggregationen möglich sind. Schließlich werden die für die Verkehrsdaten konzipierten Kontexte in einem relationalen Datenbanksystem umgesetzt. Zuletzt folgt eine Bewertung, in der erörtert wird, inwieweit die Anwendung des Kontextmodells auf die Verkehrsdaten und die Umsetzung der für die Daten konzipierten Kontexte erfolgreich waren.

2 Theoretische Überlegungen

In diesem Kapitel werden zunächst die Grundlagen des Kontextmodells vorgestellt. Anschließend wird erarbeitet, in welcher Beziehung Kontexte zu den im multidimensionalen Datenmodell beheimateten Klassifikationshierarchien stehen. Außerdem beschäftigt sich das Kapitel mit der Frage, inwieweit Aggregation und Kontexte miteinander vereinbar sind. Es werden zunächst scharfe, später unscharfe Kontexte betrachtet.

2.1 Das Kontextmodell

Kontexte sind ein Konzept zur Klassifikation von Daten. Das Kontextmodell stellt eine Erweiterung des relationalen Datenmodells dar, welche es erlaubt, die in einer Relation abgespeicherten scharfen Daten nach unscharfen Kriterien einzuteilen. Vorgestellt wurde dieser Ansatz in [Sch98], eine kürzere Beschreibung findet sich auch in [MMWS03] und [MSSV01].

2.1.1 Motivation

Umfangreiche Datenbestände werden schnell unübersichtlich. Es liegt daher nahe, eine Strukturierung vorzunehmen und Daten, die sich ähnlich sind, zu Klassen zusammenzufassen. Dazu muss jedoch zunächst festgelegt werden, welche Kriterien zwei Datensätze erfüllen müssen, damit sie als ähnlich betrachtet werden. Eine Möglichkeit, solche Kriterien festzulegen, ist die Definition von Kontexten.

Eine Strukturierung kommt dem Benutzer entgegen, der sich oft gar nicht für genaue Werte, sondern nur für den ungefähren Bereich interessiert, in dem diese Werte liegen. Häufig sind die Daten in der Detailliertheit, in der sie vorliegen, für den Benutzer nicht interessant; er möchte vielmehr eine Art Zusammenfassung der Daten haben. Für die Analyse der Daten ist daher eine Klassifizierung und Aggregation über die Klassen notwendig.

Durch Kontexte kann Imperfektion ausgedrückt werden, mit der die Daten eventuell behaftet sind. Oftmals sind nämlich die Werte, die durch ein relationales Schema repräsentiert werden, gar nicht mit der Genauigkeit bekannt, mit der sie abgespeichert werden. Diese Imperfektion wird durch Kontexte modelliert, indem nicht mehr mit dem exakten Wert, sondern mit dem Bereich gearbeitet wird, in dem der Wert liegt. Eine Anfrage, die mit Kontexten arbeitet, erhebt also gar nicht mehr den Anspruch, dass die zugrunde liegenden Werte exakt sind, obwohl sie exakt abgespeichert sind.

2.2 Scharfe Kontexte

Zunächst wird der spezielle Fall der scharfen Kontexte betrachtet. Im Folgenden wird der Begriff „Kontext“ daher immer im Sinne eines scharfen Kontexts gebraucht.

2.2.1 Kontexte und Klassen

Ein Kontext ist definiert als die Einteilung des Wertebereichs eines Attributs in einem relationalen Schema in Äquivalenzklassen. Nimmt man diese Einteilung für jedes Attribut des Schemas vor und sieht zwei Tupel als ähnlich an, falls die Werte sämtlicher Attribute jeweils in der selben Äquivalenzklasse liegen, so ergibt sich daraus eine Einteilung der Relation in Klassen¹. Mengentheoretisch entsprechen diese Klassen den Kreuzprodukten der auf den Attributen definierten Äquivalenzklassen.

Die so definierten Klassen sind scharfe Klassen; das bedeutet, ein Tupel gehört entweder zu einer Klasse oder es gehört nicht dazu. Weiter gehört jedes Tupel zu genau einer dieser Klassen, nämlich zu derjenigen, die das Kreuzprodukt der Äquivalenzklassen ist, in denen die einzelnen Attribute des Tupels liegen.

2.2.2 Selektion und Gruppierung

Die durch die Festlegung der Kontexte entstehenden Klassen kann man verwenden, um gezielt Tupel zu selektieren, die in einer bestimmten Klasse liegen. Möchte man für ein bestimmtes Attribut keine Selektionsbedingung festlegen, so muss der Kontext für das betreffende Attribut aus genau einer Äquivalenzklasse bestehen – dem gesamten Wertebereich des Attributs. So trägt dieses Attribut dann nicht zur Einteilung der Relation bei, das heißt, der Wert des Attributs ist nicht ausschlaggebend dafür, zu welcher Klasse ein Tupel gehört.

Zur Vereinfachung wird daher für Attribute, die nicht in der Selektionsbedingung auftauchen, implizit immer ein Kontext angenommen, der aus dem gesamten Wertebereich des Attributs besteht. Analoges gilt für die im Folgenden beschriebene Verwendung von Kontexten zur Gruppierung.

Eine weitere Möglichkeit, die durch Kontexte definierten Klassen zu verwenden, besteht darin, sie zur Gruppierung bei Aggregationen zu nutzen. So können Aggregate aus allen Tupeln berechnet werden, die in einer Klasse enthalten sind. Welche Bedingungen die Klassen erfüllen müssen, damit Aggregation sinnvoll möglich ist, wird in Abschnitt 2.4.1 behandelt.

2.2.3 Beziehung zum multidimensionalen Datenmodell

Im Gegensatz zum relationalen Datenmodell existieren im multidimensionalen Modell zwei unterschiedliche Arten von Attributen: Die einen repräsentieren quantifizierende Information und werden Summenattribute genannt. Sie stellen Kennzahlen dar, an deren Auswertung man interessiert ist. Die anderen repräsentieren qualifizierende Information und werden als Kategorieattribute bezeichnet. Sie geben die Struktur vor, die der Auswertung zugrunde liegt. Genauere Informationen zum multidimensionalen Datenmodell finden sich in Fachbüchern zum Thema „Data Warehousing“, z.B. in [Leh03] oder [BG01].

Die Parallele, die zwischen dem multidimensionalen Datenmodell und dem Kontextmodell besteht, fällt sofort ins Auge: Sowohl die Kategorieattribute als auch die Kontexte werden zur Klassifikation verwendet. Der Unterschied besteht darin, dass beim multidimensionalen Modell die strukturierende Information in den funktionalen Abhängigkeiten

¹Natürlich sind auch diese Klassen Äquivalenzklassen. Um jedoch diese die Relation unterteilenden Äquivalenzklassen zu unterscheiden von den Äquivalenzklassen, die die Wertebereiche der einzelnen Attribute unterteilen, wird hier stets der Begriff „Klasse“ verwendet, wenn es um die Unterteilung der Relation geht, und der Begriff „Äquivalenzklasse“, wenn die Unterteilung eines Wertebereichs gemeint ist.

Einkommen	Alter	Altersgruppe
1000	21	jung
2000	25	jung
6000	44	mittel
1500	73	alt
1500	89	alt

Tabelle 2.1: Beispiel für Klassifikation im multidimensionalen Datenmodell

der Kategorieattribute enthalten ist, während sie beim Kontextmodell durch die Kontexte repräsentiert wird.

Beispiel: Das Durchschnittseinkommen verschiedener Altersgruppen könnte im multidimensionalen Datenmodell wie in Tabelle 2.1 dargestellt werden². *Einkommen* ist hier das Summenattribut, *Alter* und *Altersgruppe* sind Kategorieattribute. Die Zuordnung, welches Alter zu welcher Altersgruppe gehört, ist durch eine funktionale Abhängigkeit des Attributs *Altersgruppe* vom Attribut *Alter* gegeben.

Soll die selbe Klassifikation im Kontextmodell vorgenommen werden, so hat die entsprechende Relation lediglich die Attribute *Einkommen* und *Alter*. Anstatt eines Attributs *Altersgruppe*, das funktional von *Alter* abhängt, definiert man Kontexte für das Attribut *Alter*, etwa

$$K(\textit{Alter}) = \{\{20, 21, \dots, 30\}, \{31, 32, \dots, 50\}, \{51, 52, \dots, 100\}\}$$

Weist man den einzelnen Äquivalenzklassen des Kontexts Bezeichnungen zu, z.B.

$$\begin{aligned} \{20, 21, \dots, 30\} &= \textit{jung} \\ \{31, 32, \dots, 50\} &= \textit{mittel} \\ \{51, 52, \dots, 100\} &= \textit{alt} \end{aligned}$$

und führt man ein zusätzliches Attribut *Altersgruppe* ein, das als Wert die Bezeichnung der Äquivalenzklasse erhält, so gelangt man zu einer Repräsentation wie im multidimensionalen Datenmodell.

Natürlich existiert im Kontextmodell, welches ja eine Erweiterung des relationalen Modells ist, keine Unterscheidung zwischen Summen- und Kategorieattributen. Als Summenattribute können jedoch die Attribute angesehen werden, die man nicht zur Klassifizierung verwendet, deren Kontext also aus nur einer Äquivalenzklasse besteht (siehe auch Abschnitt 2.2.2), wie hier das Attribut *Einkommen*.

2.2.4 Kontexte und Hierarchien

Die Kategorieattribute im multidimensionalen Datenmodell sind verschiedenen Dimensionen wie z.B. Zeit, Ort oder Produkttyp zugeordnet. Für jede Dimension kann eine ganze Reihe von Kategorieattributen unterschiedlicher Granularität vorhanden sein, in der Zeitdimension z.B. Tage, Monate und Jahre; man spricht hier auch von Klassifikationsstufen.

²Hier im Beispiel existiert der Einfachheit halber nur eine einzige Dimension, es ließe sich aber auf beliebig viele Dimensionen erweitern.

2 Theoretische Überlegungen

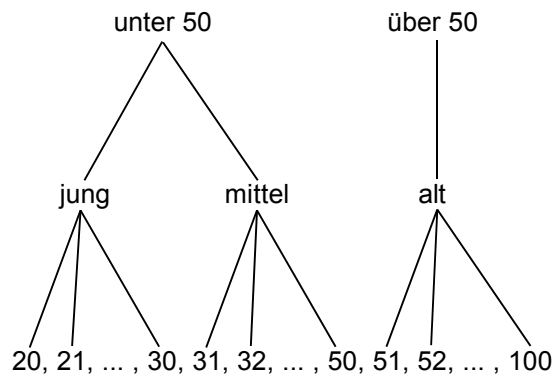


Abbildung 2.1: Durch Kontexte definierte Hierarchie

Aus diesen Klassifikationsstufen ergibt sich die so genannte Klassifikationshierarchie: Ein Jahr besteht beispielsweise aus Monaten, welche wiederum aus Tagen bestehen. Hier sind auch parallele Hierarchien möglich, für die Zeitdimension z.B. Tage, Monate, Jahre und parallel dazu Tage und Wochen.

Formal lässt sich eine Klassifikationshierarchie definieren als eine Menge von Kategorieattributen, die bezüglich der funktionalen Abhängigkeit total geordnet ist. Für die hier als Beispiel angeführte Zeitdimension gilt z.B. $Tag \rightarrow Monat \rightarrow Jahr$ und $Tag \rightarrow Woche$, nicht aber $Woche \rightarrow Monat$, da eine Woche zu zwei Monaten gehören kann. Wochen und Monate sind daher Klassifikationsstufen zweier paralleler Hierarchien.

Es ist nun möglich, auch mit Kontexten Hierarchien zu konstruieren. Dies geschieht, indem man Kontexte von Kontexten definiert, das Prinzip der Kontextbildung also rekursiv anwendet. Ein Kontext auf der Stufe i legt dann fest, welche der Äquivalenzklassen der darunter liegenden Stufe $i - 1$ auf Stufe i als äquivalent anzusehen sind.

Für das obige Beispiel könnte man eine weitere Stufe folgendermaßen konstruieren:

$$K'(K(\text{Alter})) = \{\{20, 21, \dots, 30\}, \{31, 32, \dots, 50\}\}, \{\{51, 52, \dots, 100\}\}$$

oder etwas übersichtlicher unter Verwendung der definierten Bezeichnungen:

$$K'(K(\text{Alter})) = \{\{jung, mittel\}, \{alt\}\}$$

Werte des Attributs *Alter*, die auf der ersten Stufe in die Klassen *jung* und *mittel* fallen, gelten also auf der zweiten Stufe als äquivalent, so dass dort nur noch zwei Äquivalenzklassen vorhanden sind.

Diese Konstruktion ist auch mit der Definition der Klassifikationshierarchie über funktionale Abhängigkeiten vereinbar: Die Äquivalenzklasse, zu der ein Wert auf Stufe $i - 1$ gehört, bestimmt eindeutig die Äquivalenzklasse auf Stufe i . Auch parallele Hierarchien lassen sich konstruieren, indem man auf einer Stufe mehrere unterschiedliche Einteilungen vornimmt, z.B. neben der obigen noch

$$K^*(K(\text{Alter})) = \{\{jung\}, \{mittel, alt\}\}$$

Parallele Hierarchien definieren eine Halbordnung der Kontexte eines Attributs, für die das Symbol \succeq verwendet werden soll. Für zwei Kontexte A und B gilt also $A \succeq B$, falls die Äquivalenzklassen von A sich in einer Hierarchie oberhalb der Äquivalenzklassen von B befinden oder A mit B identisch ist.

2.2.5 Hierarchien von Klassen

Aus den Hierarchien der Äquivalenzklassen für die einzelnen Attribute ergibt sich auch eine Hierarchie der Klassen, die die Relation unterteilen. Eine Klasseneinteilung entsteht, indem für jedes Attribut ein Kontext ausgewählt wird. Die Klassen einer Klasseneinteilung U befinden sich folglich in der Hierarchie oberhalb der Klassen einer anderen Klasseneinteilung V , falls für jedes Attribut die Äquivalenzklassen des für U verwendeten Kontexts entweder in der Hierarchie oberhalb derer des für V verwendeten Kontexts liegen oder der Kontext für beide Klasseneinteilungen identisch ist.

Aus der Halbordnung der Kontexte der einzelnen Attribute lässt sich also eine Halbordnung der Klasseneinteilungen ableiten. Entsteht die Klasseneinteilung U aus den Kontexten A_i und die Klasseneinteilung V aus den Kontexten B_i , so ist die Halbordnung der Klasseneinteilungen definiert durch

$$U \succeq V \Leftrightarrow \bigwedge_{i=1}^n A_i \succeq B_i \quad (2.1)$$

Eine Klasseneinteilung, deren Klassen in der Hierarchie oberhalb derer einer anderen liegen, stellt eine gröbere Einteilung dar als die, deren Klassen in der Hierarchie weiter unten liegen. Eine Einteilung nach Monaten und Bundesländern ist gröber als eine nach Tagen und Landkreisen und auch gröber als eine Einteilung nach Monaten und Landkreisen oder Tagen und Bundesländern.

Es ergeben sich auch für die Klassen parallele Hierarchien; Einteilungen, deren Klassen zu zwei parallelen Hierarchien gehören, sind hinsichtlich ihrer Granularität nicht vergleichbar. Parallele Hierarchien in der Klasseneinteilung entstehen zum einen, falls die Äquivalenzklassen der verwendeten Kontexte aus parallelen Hierarchien stammen. Die Klassen einer Einteilung nach Wochen und Landkreisen befindet sich nicht in der selben Hierarchie wie die einer Einteilung nach Monaten und Landkreisen.

Zum anderen ergeben sich parallele Hierarchien auf der Klassenebene dadurch, dass die Äquivalenzklassen bei einem Attribut in der Hierarchie auf einer höheren Ebene angesiedelt sein können als bei einer anderen Klasseneinteilung, die eines anderen Attributs aber auf einer niedrigeren: Die Klassen einer Einteilung nach Tagen und Bundesländern und die einer Einteilung nach Monaten und Landkreisen gehören zu zwei parallelen Hierarchien.

2.3 Unscharfe Kontexte

Während bisher nur scharfe Einteilungen betrachtet wurden, bei denen eindeutig festgelegt ist, ob ein Tupel zu einer Klasse gehört oder nicht, und bei denen jedes Tupel zu genau einer Klasse gehört, geht es nun um unscharfe Einteilungen.

Oftmals lässt sich zwischen zwei Klassen keine scharfe Grenze ziehen; es gibt Objekte, die weder so richtig zur einen noch zur anderen Klasse gehören, sondern irgendwo dazwischen liegen. Deswegen soll es nun möglich sein, Tupel mehreren Klassen zuzuordnen. Die Klassen sollen daher nicht mehr scharf gegeneinander abgegrenzt sein, sondern ineinander übergehen. Hierzu werden unscharfe Einteilungen der Wertebereiche von Attributen benötigt, die als unscharfe Kontexte bezeichnet werden sollen.

2 Theoretische Überlegungen

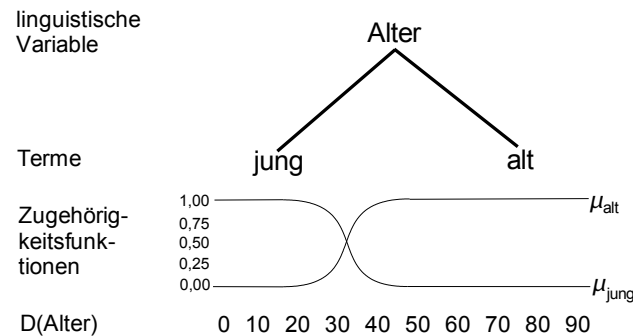


Abbildung 2.2: Linguistische Variable

2.3.1 Linguistische Variablen

Linguistische Variablen sind ein Konzept zur semantischen Beschreibung unscharfer Klassen, das aus dem Bereich der Fuzzy-Logik stammt ([Zim92]). Eine linguistische Variable hat als Werte sprachliche Konstrukte, so genannte Terme. Sie sind vergleichbar mit den Bezeichnungen, die im Falle scharfer Kontexte den Äquivalenzklassen zugewiesen wurden.

Die Bedeutung eines Terms wird durch eine so genannte Zugehörigkeitsfunktion festgelegt. Die Zugehörigkeitsfunktion gibt für jedes Element aus dem Wertebereich des Attributs, zu dem die linguistische Variable gehört, an, wie sehr der Term auf das Element zutrifft. Sie kann Werte zwischen null und eins annehmen. Trifft der Term voll zu, so hat die Zugehörigkeitsfunktion den Wert eins, trifft er absolut nicht zu, hat sie den Wert null. Die Zugehörigkeitsfunktion eines Terms T wird gewöhnlich mit μ_T bezeichnet.

Man kann scharfe Kontexte als einen Spezialfall linguistischer Variablen sehen: Die Terme entsprechen den Bezeichnungen der Äquivalenzklassen und die Zugehörigkeitsfunktion beträgt eins, falls der Wert in der Äquivalenzklasse liegt, zu der der Bezeichner gehört, andernfalls beträgt sie null.

Umgekehrt sind linguistische Variablen eine Verallgemeinerung des Konzepts der scharfen Kontexte. Sie stellen eine Aufteilung des Wertebereichs eines Attributs in unscharfe Teilmengen dar. Eine linguistische Variable definiert daher unscharfe Kontexte. Eine Gegenüberstellung der Begriffe für scharfe und unscharfe Einteilungen findet sich in Tabelle 2.3.2.

2.3.2 Unscharfe Klassen

Definiert man für jedes Attribut einer Relation eine linguistische Variable, so lassen sich unscharfe Klassen bilden, die durch die Werte der linguistischen Variablen bestimmt sind. Hat man z.B. die linguistische Variable *Alter* mit den Termen *jung* und *alt* sowie die linguistische Variable *Größe* mit den Werten *groß* und *klein*, so ergeben sich die Klassen (*jung, groß*), (*jung, klein*), (*alt, groß*) und (*alt, klein*).

Die Zugehörigkeit zu diesen unscharfen Klassen kann aus den Zugehörigkeitsfunktionen der Terme berechnet werden, aus denen die Klasse besteht. Der Operator, der die Zugehörigkeit zur Klasse berechnet, muss einer konjunktiven Verknüpfung der einzelnen Zugehörigkeitswerte der Terme entsprechen. Hierfür wurden verschiedene Operatoren vorgeschlagen ([Zim92]), von denen nun drei betrachtet werden sollen.

scharf	unscharf
scharfer Kontext	unscharfer Kontext, linguistische Variable
Äquivalenzklasse	Term
scharfe Klasse	unscharfe Klasse
Enthaltensein	Zugehörigkeit

Tabelle 2.2: Gegenüberstellung der korrespondierenden Begriffe für scharfe und unscharfe Einteilungen

Minimumoperator

Dem Minimumoperator liegt die Vorstellung zugrunde, dass die Kombination mehrerer Terme auf ein bestimmtes Tupel genau so stark zutrifft wie der am wenigsten zutreffende Term der Kombination. Die Zugehörigkeit zur Klasse ergibt sich also als das Minimum der Zugehörigkeitsfunktionen der Terme, die die Klasse festlegen.

$$\mu^{min} = \min \{ \mu_1, \mu_2, \dots, \mu_n \} \quad (2.2)$$

Logisches Und

Wird das logische Und als Operator verwendet, gehört ein Tupel nur so stark zu einer Klasse, wie die Gesamtheit aller Terme, die die Klasse festlegen, auf das Tupel zutrifft. Dem logischen Und entspricht eine multiplikative Verknüpfung der einzelnen Zugehörigkeitsfunktionen.

$$\mu^{log} = \prod_{i=1}^n \mu_i \quad (2.3)$$

Kompensatorisches Und

Das kompensatorische Und entspricht am ehesten dem menschlichen Entscheidungsverhalten. Trifft ein Term stark auf ein Objekt zu und ein anderer weniger stark, so würde ein Mensch die Zugehörigkeit zur Kombination beider Terme wohl irgendwo dazwischen einordnen. Die starke Zugehörigkeit zu einem Term kompensiert die schwache Zugehörigkeit zu einem anderen. Dies berücksichtigt das so genannte kompensatorische Und (auch als Gammaoperator bezeichnet), wie es in [Zim92] beschrieben wird:

$$\mu^{komp} = \left(\prod_{i=1}^n \mu_i \right)^{1-\gamma} \left(1 - \prod_{i=1}^n (1 - \mu_i) \right)^{\gamma} \quad (2.4)$$

Der Operator ist eine Kombination von logischem Und und einem logischen Oder. Durch den Parameter γ lässt sich der Grad der Kompensation einstellen. Für $\gamma = 0$ ist der Operator mit dem logischen Und identisch. Für $\gamma = 1$ ergibt sich das logische Oder.

Welchen dieser Operatoren man verwendet, hängt davon ab, wie man die Klassenbildung interpretiert und welche Anforderungen man an diese stellt. Der Minimumoperator beispielsweise hat von den drei Operatoren den geringsten Rechenaufwand, während das logische

2 Theoretische Überlegungen

	(jung, klein)	(jung, groß)	(alt, klein)	(alt, groß)
μ^{min}	0,20	0,40	0,20	0,60
μ^{log}	0,08	0,32	0,12	0,48
$\mu^{komp}, \gamma = 0,5$	0,20	0,53	0,29	0,66

$$\mu_{jung}(45) = 0,4 \quad \mu_{alt}(45) = 0,6 \quad \mu_{klein}(1,82) = 0,2 \quad \mu_{gro\beta}(1,82) = 0,8$$

Tabelle 2.3: Beispiel für die Berechnung der Zugehörigkeit zu unscharfen Klassen
(für eine 45-jährige, 1,82m große Person)

Und die Wiederverwendung bereits berechneter Aggregate erlaubt (siehe auch Abschnitt 2.4.4).

2.3.3 Selektion und Gruppierung

Die Zugehörigkeit zu einer unscharfen Klasse lässt sich wie im scharfen Fall als Selektionskriterium verwenden. Anders als dort, wo ein Tupel entweder zu einer Klasse gehört oder nicht, können hier verschiedene Tupel unterschiedlich stark zu einer Klasse zugehörig sein. Eine sinnvolle Möglichkeit der Selektion bestünde daher darin, die Tupel nach ihrer Zugehörigkeit zu ordnen und dann eine bestimmte Anzahl an Tupeln zu selektieren; dies sind dann die Tupel, auf die das Selektionskriterium am besten zutrifft. So kann man z.B. die zehn besten Kunden oder die hundert meistbefahrenen Strecken ermitteln.

Eine andere Möglichkeit besteht darin, aus der unscharfen Klasse eine scharfe Klasse zu machen. In der Theorie der Fuzzy-Mengen wurde dafür der so genannte α -Schnitt entwickelt ([Zim92]). Der α -Schnitt ist definiert als die Menge aller Elemente einer unscharfen Menge, deren Zugehörigkeit größer oder gleich einem Wert α ist:

$$M_\alpha = \{x \in M : \mu_M(x) \geq \alpha\} \quad (2.5)$$

Ein Selektionskriterium wäre daher das Enthaltensein im α -Schnitt. So würden alle Tupel selektiert, deren Zugehörigkeit zur Klasse größer oder gleich einem bestimmten Mindestwert α ist.

Nicht ganz so einfach sieht es mit der Gruppierung aus: Die unscharfen Klassen geben keine feste Einteilung vor, die entscheidet, aus welchen Tupeln ein Aggregat gebildet werden soll. Auch hier besteht natürlich wie bei der Selektion die Möglichkeit, durch Angabe von Grenzen oder auch Intervallen für die Zugehörigkeit zu einer scharfen Einteilung zu gelangen. Es stellt sich aber die Frage, für was überhaupt eine unscharfe Klassifikation vorgenommen wird, wenn danach durch die Angabe von willkürlichen Grenzen doch wieder eine scharfe Einteilung entsteht. Mit der Alternative, zu aggregieren ohne die Daten scharf zu gruppieren, befasst sich Abschnitt 2.4.2.

2.3.4 Unscharfe Hierarchien

Im Folgenden wird nun ein Vorschlag gemacht, wie sich Hierarchien von Termen konstruieren lassen. Dies soll wie bei den Hierarchien der Äquivalenzklassen im Fall der scharfen Kontexte durch rekursive Anwendung des Konzepts der Kontexte geschehen, also indem

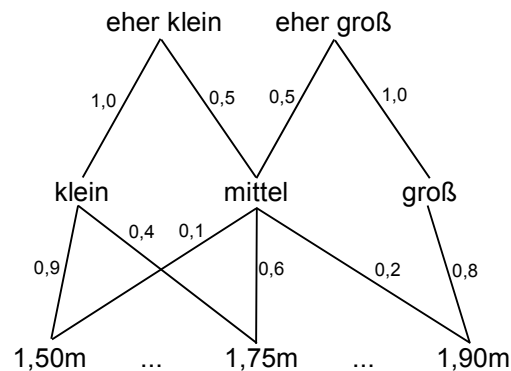


Abbildung 2.3: Durch linguistische Variablen definierte Hierarchie

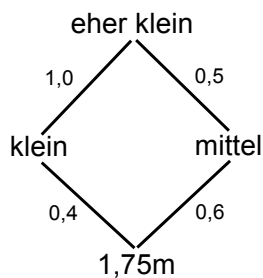
linguistische Variablen auf unterschiedlichen Ebenen definiert werden. Die linguistische Variable auf Ebene i legt über ihre Zugehörigkeitsfunktionen fest, wie sehr ihre Terme auf die Terme der linguistischen Variable auf Ebene $i - 1$ zutreffen.

Beispiel: Existiert für das Attribut *Körpergröße* eine linguistische Variable mit den Termen *klein*, *mittel* und *groß*, kann man eine Ebene höher eine linguistische Variable mit den Termen *eher klein* und *eher groß* definieren. Der Wert der Zugehörigkeitsfunktion $\mu_{\text{eher klein}}(\text{mittel}) = 0,5$ besagt dann, dass der Term *eher klein* zum Grad 0,5 auf den Term *mittel* zutrifft.

Natürlich ist nicht nur von Interesse, wie sehr ein Term einer Hierarchieebene auf einen Term der nächstunteren Ebene, sondern auch, wie sehr ein Term einer höheren Hierarchieebene auf einen bestimmten Attributwert zutrifft. Hier lässt sich eine direkte Zugehörigkeitsfunktion aus den jeweils in Bezug auf die nächstniedrigere Ebene definierten Zugehörigkeitsfunktionen berechnen. Sicher gibt es auch hier mehrere Möglichkeiten, wie diese Berechnung erfolgen kann, abhängig davon, wie man die Zugehörigkeit eines Terms niedrigerer Ebene zu einem Term höherer Ebene interpretiert. Der hier vorgestellten Art der Berechnung liegt die Vorstellung zugrunde, dass ein Term höherer Ebene sich aus mehreren Termen der nächstunteren Ebene zusammensetzt. Die Zugehörigkeitsfunktionen geben an, wie viel von einem Term der niedrigeren Ebene in den Term der höheren Ebene einfließt.

Beispiel: Wie die Berechnung erfolgt, sieht man am besten an einem Beispiel. Wir wollen die Zugehörigkeit einer Körpergröße von $1,75m$ zum Term *eher klein* berechnen. Der Wert $1,75m$ hat eine gewisse Zugehörigkeit zum Term *klein*, für welche man wiederum die Zugehörigkeit zum Term *eher klein* kennt. Um die Zugehörigkeit von $1,75m$ zu *eher klein* zu berechnen, werden beide Zugehörigkeiten multipliziert. Jedoch hat $1,75m$ auch eine Zugehörigkeit zum Term *mittel*, welcher wiederum ebenfalls eine Zugehörigkeit zu *eher klein* hat. Auch dieser Anteil muss berücksichtigt und deswegen addiert werden. Man berechnet also zunächst die Zugehörigkeit zu den Termen auf der ersten Hierarchieebene und multipliziert diese Zugehörigkeiten dann mit den jeweiligen Zugehörigkeiten dieser Terme zum Term *eher klein*. Die sich ergebenden Werte werden anschließend addiert.

2 Theoretische Überlegungen



$$\mu_{\text{eher klein}}(1,75\text{m}) = 0,4 \cdot 1,0 + 0,6 \cdot 0,5 = 0,7$$

Abbildung 2.4: Beispiel für die Berechnung der Zugehörigkeit zu einem Term höherer Ebene

Kennt man für die Terme der Ebene $i - 1$ die Zugehörigkeitsfunktionen in Bezug auf die Elementardaten, lassen sich diese Zugehörigkeitsfunktionen für die Terme der Ebene i also berechnen, indem man die Zugehörigkeitsfunktionen der Terme von Ebene $i - 1$ in Bezug auf die Elementardaten mit den Zugehörigkeitsfunktionen der Terme von Ebene i in Bezug auf die Terme der Ebene $i - 1$ gewichtet addiert.

Wenn T_k^i der k -te Term der i -ten Ebene ist, n die Anzahl der Terme auf Ebene $i - 1$ und x der Elementarwert, dessen Zugehörigkeit man berechnen will, lautet die entsprechende Formel:

$$\mu_{T_k^i}(x) = \sum_{j=1}^n \left(\mu_{T_j^{i-1}}(x) \cdot \mu_{T_k^i}(T_j^{i-1}) \right) \quad (2.6)$$

Rekursiv lassen sich so die Zugehörigkeiten der Elementardaten für alle Ebenen berechnen. Das Ergebnis dieser Berechnung muss natürlich eine Zugehörigkeitsfunktion sein. Daher müssen Einschränkungen erlassen werden, die dafür Sorge tragen, dass sich durch die Addition kein größerer Wert als eins ergibt. Auf diese Einschränkungen wird in Abschnitt 2.4.5 eingegangen.

Da die Zugehörigkeiten der Elementardaten zu den Termen einer linguistischen Variable auf Ebene i berechnet werden können aus den Zugehörigkeiten der Elementardaten zu den Termen auf Ebene $i - 1$ und den Zugehörigkeiten der Terme auf Ebene $i - 1$ zu denen auf Ebene i , bestimmen die Zugehörigkeiten der Elementardaten zu den Termen auf Ebene $i - 1$ die Zugehörigkeiten der Elementardaten zu den Termen auf Ebene i . Es besteht also eine funktionale Abhängigkeit

$$\mu_{T_1^{i-1}}(x), \mu_{T_2^{i-1}}(x), \dots, \mu_{T_n^{i-1}}(x) \rightarrow \mu_{T_1^i}(x), \mu_{T_2^i}(x), \dots, \mu_{T_m^i}(x)$$

Dies ist vergleichbar mit den funktionalen Abhängigkeiten in Hierarchien, die sich durch scharfe Kontexte ergeben. Definiert man auf einer Ebene mehrere linguistische Variablen, so entstehen wie bei scharfen Kontexten parallele Hierarchien, bei deren Termen die Zugehörigkeiten funktional nicht voneinander abhängen. Ebenso wie bei scharfen Kontexten ergeben sich durch die Hierarchien der Terme Hierarchien der aus diesen Termen gebildeten Klassen.

2.4 Aggregation

Thema dieses Abschnitts ist die Aggregation, zunächst über scharfe, später über unscharfe Klassen. In beiden Fällen gibt es Bedingungen, die erfüllt sein müssen, damit überhaupt vernünftig aggregiert werden kann. Im unscharfen Fall muss zusätzlich definiert werden, wie eine Aggregation über unscharfe Klassen auszusehen hat.

Bei einer hierarchischen Anordnung von Klassen ist es zusätzlich interessant, ob – und wenn ja wie – Aggregate grober Granularität aus bereits auf feinerer Granularität aggregierten Werten gewonnen werden können, ohne dass man auf die Elementardaten zurückgreifen muss.

2.4.1 Aggregationsbedingungen

In [LS97] werden drei notwendige Bedingungen aufgeführt, die ein multidimensionales Schema erfüllen muss, damit eine Aggregation sinnvolle Ergebnisse liefert. Weiterhin wird plausibel gemacht, dass diese Bedingungen auch hinreichend sind. Im Folgenden werden die drei Bedingungen – Überlappungsfreiheit, Vollständigkeit und Typverträglichkeit – beschrieben.

Überlappungsfreiheit

Überlappungsfreiheit bedeutet, dass die Schnittmenge zweier Klassen, die sich auf der selben Ebene der Hierarchie befinden, leer ist. Es darf also jedes Tupel zu höchstens einer Klasse gehören. Betrachtet man die Konstruktion der Klassen durch Kontexte, ist dies gleichbedeutend damit, dass jeder Attributwert in höchstens eine Äquivalenzklasse fallen darf. Diese Eigenschaft ist aber für Äquivalenzklassen immer erfüllt. Durch scharfe Kontexte definierte Klassen sind also immer überlappungsfrei.

Eine andere Art der Überlappungsfreiheit, die ebenfalls gegeben sein muss, ist von den Kontexten unabhängig und steckt in den Daten selbst. Beinhalten die Daten an sich schon aggregierte Werte, so muss die Einteilung, die dieser Aggregation zugrunde lag, ebenfalls überlappungsfrei gewesen sein. Eine Relation beispielsweise, die für jede Automarke die Anzahl an Fahrzeugen beinhaltet, kann problemlos weiter aggregiert werden, da jedes Fahrzeug von nur einer Marke sein kann. Beinhalten die Relation aber die Anzahl an Fahrzeugen mit weiblichen und die Anzahl an Fahrzeugen mit männlichen Insassen, so würde eine weitere Aggregation zu fehlerhaften Ergebnissen führen, da es Fahrzeuge gibt, in denen sowohl Frauen als auch Männer sitzen, und diese dann doppelt berücksichtigt würden.

Vollständigkeit

Vollständigkeit bedeutet, dass die Vereinigung aller Klassen einer Ebene der Hierarchie sämtliche Elemente der nächstunteren Ebene – und somit, falls dies für alle Ebenen gegeben ist, auch sämtliche Elementardaten – umfasst. Jedes Tupel muss also in mindestens einer Klasse enthalten sein. Wie die Überlappungsfreiheit ist auch die Vollständigkeit bei der Bildung der Klassen aus Äquivalenzklassen scharfer Kontexte immer gewährleistet.

Auch hier muss bei schon aggregiert vorliegenden Daten ebenfalls die Vollständigkeit gegeben sein. Es genügt nicht, die Anzahl der Fahrzeuge aller Automarken anzugeben, falls Fahrzeuge existieren, die keiner Marke zugeordnet werden können. Behelfen kann man sich hier durch Einführung einer Kategorie „Sonstiges“.

Typverträglichkeit

Die dritte Bedingung schließlich besteht darin, dass der Typ des zu aggregierenden Wertes mit der zu berechnenden Aggregatfunktion kompatibel sein muss. Hierbei werden drei Typen von Werten unterschieden:

1. Werte, die Ereignisse beschreiben, wie z.B. Verkäufe, vorüberfahrende Autos oder Geburten. Sie beziehen sich immer auf einen bestimmten Zeitraum. Dieser Typ von Werten wird als „flow“ bezeichnet. Werte dieses Typs können problemlos in allen Dimensionen aggregiert werden.
2. Werte, die Zustände beschreiben, z.B. Lagerbestand, Anzahl der Fahrzeuge auf einem Streckenabschnitt oder Bevölkerungszahl. Sie beziehen sich immer auf einen bestimmten Zeitpunkt. Dieser Typ von Werten heißt „stock“. Werte dieses Typs dürfen in der Zeitdimension nicht summiert werden. Der Lagerbestand am Jahresende ist eben nicht die Summe der Lagerbestände aller Monate. Andere Aggregatfunktionen wie z.B. Minimum, Maximum, Standardabweichung oder Durchschnitt sind aber möglich. In allen nicht zeitlichen Dimensionen können Werte dieses Typs problemlos summiert werden.
3. Werte, die sich auf eine Einheit beziehen, beispielsweise der Preis pro Stück einer bestimmten Ware, die Geschwindigkeit pro Fahrzeug oder die Bevölkerungsdichte, die ja Bevölkerungszahl pro Flächeneinheit ist. Diesen Typ bezeichnet man als „value per unit“. Summation von Werten dieses Typs ist in keiner Dimension sinnvoll, Minimum, Maximum, Durchschnitt oder Standardabweichung können jedoch gebildet werden.

Die ersten beiden Bedingungen gewährleisten, dass die aggregierten Werte alle Elementardaten repräsentieren. Berechnet man z.B. die Aggregatfunktion „Anzahl“ für jede Klasse, so ist die Summe aller Anzahlen gleich der Gesamtanzahl der klassifizierten Elemente. Keines wurde vergessen und keines mehrmals berücksichtigt; jedes geht in das Aggregat genau einer Klasse ein.

Dies ermöglicht es, bereits auf feiner Granularität aggregierte Werte wiederzuverwenden, wenn man Aggregate gröberer Granularität berechnen möchte. Kennt man z.B. die Anzahl der Verkehrsunfälle für jeden Tag, so kann man daraus die Anzahl für jeden Monat berechnen. Aus diesen Werten kann man wiederum die Anzahl pro Jahr berechnen, ohne auf die Zahlen für die einzelnen Tage zurückgreifen zu müssen. Mit der Wiederverwendbarkeit aggregierter Werte beschäftigt sich Abschnitt 2.4.7.

2.4.2 Aggregation über unscharfe Klassen

Ziel einer Aggregation ist es, die Attributwerte mehrerer Tupel, die zur selben Klasse gehören, zu einem einzigen Wert zusammenzufassen, der etwas Sinnvolles über die Klasse aussagt. Bei unscharfen Klassen kann man jedoch nicht eindeutig sagen, welche Tupel zur Klasse gehören und welche nicht. Eine Möglichkeit, dem Abhilfe zu schaffen, wäre die Verwendung von α -Schnitten der unscharfen Klassen. Für diese α -Schnitte lassen sich Aggregate wie für scharfe Klassen berechnen, da es sich dabei um scharfe Mengen handelt.

Problematisch ist, dass alle im α -Schnitt enthaltenen Tupel in gleichem Maße in die Berechnung des Aggregats eingehen. Für $\alpha = 0,2$ würde z.B. ein Tupel mit der Zugehörigkeit

0,2 genauso stark berücksichtigt wie eines mit der Zugehörigkeit 1,0. Tupel mit einer Zugehörigkeit, die kleiner ist als α , werden dagegen gar nicht berücksichtigt, obwohl auch sie zu einem gewissen Grad zur unscharfen Klasse gehören.

Daher liegt es nahe, die Tupel genau so stark zu berücksichtigen, wie es ihrer Zugehörigkeit zur Klasse entspricht. Die Zugehörigkeitsfunktion erhält dadurch die Rolle eines Gewichtungsfaktors. Im Folgenden werden verschiedene Aggregatfunktionen für unscharfe Klassen vorgeschlagen.

Anzahl

Bei scharfen Klassen gibt die Aggregatfunktion „Anzahl“ die Zahl der Tupel an, die zur Klasse gehören. Da man bei unscharfen Klassen in vielen Fällen (nämlich immer dann, wenn die Zugehörigkeit weder null noch eins ist) nicht sagen kann, ob ein Tupel zu einer Klasse gehört oder nicht, existiert im strengeren Sinn keine Zahl der zur Klasse gehörenden Tupel.

Man kann aber dennoch ein Maß dafür angeben, ob das, was die Klasse ausmacht, also die sie definierenden Terme, stark und auf viele Tupel oder weniger stark und auf wenige Tupel zutrifft, indem man die Zugehörigkeitsfunktionen sämtlicher Tupel addiert. Der sich so ergebende Wert macht eine Aussage über die Mächtigkeit der unscharfen Klasse. Er entspricht der für unscharfe Mengen definierten Kardinalität ([Zim92]).

Im Spezialfall scharfer Klassen ergibt sich durch Addition der Zugehörigkeiten, wie man das erwarten würde, genau die Anzahl der Tupel, die zur Klasse gehören. Dennoch muss man sich stets vor Augen halten, dass dieser Wert im unscharfen Fall keine Anzahl im eigentlichen Sinn ist, was auch daran deutlich wird, dass sich keine ganze Zahl ergeben muss. Der Wert gibt einerseits an, wie viele, andererseits aber auch, wie stark Tupel zur Klasse gehören, ist also eine Art Kompromiss zwischen der Anzahl der zur Klasse mehr oder weniger zugehörigen Tupel und dem Grad ihrer Zugehörigkeit. Der Einfachheit halber soll diese Aggregatfunktion aber auch im unscharfen Fall „Anzahl“ heißen.

Summe

Ähnlich wie bei der Berechnung der Anzahl liegt es bei der Summe nahe, die Werte um so stärker zu gewichten, je mehr die entsprechenden Tupel zur Klasse gehören. Es ergibt sich also eine gewichtete Summe; die Gewichtungsfaktoren entsprechen den Werten der Zugehörigkeitsfunktion.

Wie bei der Anzahl kann es passieren, dass aus lauter ganzzahligen Werten durch die Aggregation nicht mehr ganzzahlige Werte entstehen. Auch die gewichtete Summe entspricht bei scharfen Klasseneinteilungen dem Wert, welchen die für scharfe Einteilungen definierte Aggregatfunktion „Summe“ berechnet, da die Gewichtungsfaktoren für zu einer scharfen Klasse gehörende Tupel eins, für nicht dazu gehörende Tupel null betragen.

Durchschnitt

Bei scharfen Klassen ist der Durchschnitt definiert als der Quotient aus Summe und Anzahl. Auch bei unscharfen Klassen lässt sich der Quotient aus gewichteter Summe und gewichteter Anzahl bilden. Dieser Wert ist ein Maß für den mittleren Wert, den ein Attribut bei Tupeln hat, die zu einer Klasse gehören, wobei wieder Tupel um so stärker berücksichtigt werden, je mehr sie zur Klasse gehören. Dies kommt dem intuitiven Verständnis von „Durchschnitt“

2 Theoretische Überlegungen

recht nahe und soll deswegen auch im unscharfen Fall so bezeichnet werden. Auch hier gilt natürlich, dass der erweiterte Durchschnitt für scharfe Einteilungen den selben Wert liefert wie die übliche Durchschnittsfunktion, da dies schon für gewichtete Summe und gewichtete Anzahl der Fall war, aus denen der gewichtete Durchschnitt berechnet wird.

Standardabweichung

Die Standardabweichung gibt an, wie sehr die Werte innerhalb einer Klasse um den Mittelwert streuen. Bei scharfen Klassen wird eine Schätzung hierfür gewöhnlich berechnet, indem man die Quadrate der Abstände vom Mittelwert addiert, das Ergebnis durch die Anzahl der Werte weniger eins teilt und anschließend die Wurzel zieht.

Analog dazu kann bei unscharfen Klassen ein Maß für die Streuung um den im vorigen Abschnitt definierten Durchschnitt errechnet werden, indem die Abstandsquadrate zu diesem Durchschnitt mit der Zugehörigkeitsfunktion der Werte gewichtet und dann addiert werden. Anschließend teilt man durch die gewichtete Anzahl weniger eins und zieht die Wurzel.

Der so berechnete Wert trifft wie die Standardabweichung bei scharfen Klassen eine Aussage darüber, ob die Werte innerhalb der Klasse alle nahe am Durchschnitt liegen (in diesem Fall ist der Wert klein) oder ob sie eher weiter um ihn verteilt sind (in diesem Fall ist der Wert groß). Dabei tragen Werte, die zwar weit vom Durchschnitt entfernt sind, aber deren Tupel nur eine schwache Zugehörigkeit zur Klasse besitzen, nicht so stark zu einer Erhöhung des Wertes bei wie Werte aus Tupeln mit hoher Zugehörigkeit. Wie die schon bei den vorher definierten Aggregatfunktionen soll hier auch im unscharfen Fall die selbe Bezeichnung (Standardabweichung) verwendet werden wie bei der entsprechenden Funktion für scharfe Klassen. Auch hier stimmt die für unscharfe Klassen erweiterte Aggregatfunktion bei scharfen Einteilungen mit der Originalfunktion überein.

Nun kann es aber bei unscharfen Klassen durchaus vorkommen, dass die gewichtete Anzahl kleiner als eins ist. In diesem Fall kann eine Schätzung für die Standardabweichung nicht berechnet werden. Es macht aber ohnehin, auch im Falle scharfer Klassen, wenig Sinn, die Standardabweichung anhand einer Stichprobe abzuschätzen, die nur wenige Werte enthält.

Minimum und Maximum

Bei scharfen Klassen sind Minimum und Maximum der größte bzw. kleinste in einer Klasse enthaltene Wert. Wie aber soll solch ein Wert bei unscharfen Klassen bestimmt werden? Was, wenn die Zugehörigkeit des größten Wertes nur sehr klein ist? Solch ein Wert kann schwerlich als das Maximum der Klasse bezeichnet werden, gehört er doch kaum zur Klasse. Für unscharfe Klassen lässt sich also der größte oder kleinste in der Klasse enthaltene Wert nicht angeben, solange man nicht genau festlegt, was „in der Klasse enthalten“ bedeutet. Dies kann aber wiederum nur durch Angabe einer Mindestzugehörigkeit geschehen; man kann dann das Minimum oder Maximum der Werte angeben, deren Zugehörigkeit zur Klasse diesen Mindestwert überschreitet, wobei man wieder beim α -Schnitt wäre. Ein Wert, den man als das Minimum oder Maximum einer unscharfen Klasse bezeichnen könnte, existiert jedoch im eigentlichen Sinn nicht.

2.4.3 Bedingungen für unscharfe Klassen

Für scharfe Klassen wurden in Abschnitt 2.4.1 drei Bedingungen festgelegt, die gelten müssen, damit Aggregationen sinnvolle Ergebnisse liefern: Überlappungsfreiheit, Vollständigkeit und Typverträglichkeit. Die letzte dieser Bedingungen, die Forderung nach Vereinbarkeit des Typs der zu aggregierenden Werte mit der Aggregatfunktion, ist unabhängig von der Klasseneinteilung und kann daher unverändert für unscharfe Klassen übernommen werden. Die anderen beiden Bedingungen müssen neu formuliert werden. Es folgt nun ein Vorschlag, wie eine solche Neuformulierung für unscharfe Klassen aussehen kann.

Überlappungsfreiheit in dem Sinn, dass ein Element zu höchstens einer Klasse gehören darf, kann bei unscharfen Klassen nicht gewährleistet werden, da gerade der Wunsch nach kontinuierlichen Übergängen die Motivation für unscharfe Klassen ist. Betrachten wir nochmal den scharfen Fall: Der Grund, warum Überlappungsfreiheit für eine korrekte Aggregation notwendig ist, besteht darin, dass Tupel nicht doppelt berücksichtigt werden dürfen. Umgekehrt sorgt die Vollständigkeit dafür, dass kein Tupel überhaupt nicht berücksichtigt wird. Damit die Aggregation ein korrektes Ergebnis liefert, muss also jedes Tupel genau einmal berücksichtigt werden, was bei scharfen Klassen gleichbedeutend damit ist, dass jedes Tupel zu genau einer Klasse gehört.

Bei Aggregation über unscharfe Klassen fließt ein Wert aber nicht vollständig in den aggregierten Wert für eine Klasse ein, sondern nur in dem Maß, das der Zugehörigkeit der Tupels, aus dem er stammt, zur Klasse entspricht – so sind gerade die unscharfen Aggregatfunktionen definiert. Das Tupel gehört dann zwar zu mehreren Klassen, wird jedoch für das Aggregat jeder dieser Klassen nur zu einem gewissen Anteil berücksichtigt. Insgesamt wird es also genau einmal berücksichtigt, wenn sich diese Anteile zu eins ergänzen. Ein Tupel wird daher über alle Klassen einer Hierarchieebene betrachtet genau einmal berücksichtigt, wenn die Zugehörigkeiten des Tupels über alle Klassen summiert genau eins ergeben. Die Forderung nach Überlappungsfreiheit und Vollständigkeit muss für unscharfe Einteilungen also durch die Forderung ersetzt werden, dass die Klassenzugehörigkeiten jedes Tupels sich zu eins summieren.

Ist dies gewährleistet, liefert die Aggregation korrekte Ergebnisse: Berechnet man für jede Klasse die Aggregatfunktion „Anzahl“ und addiert sämtliche sich ergebenden Werte, erhält man die Gesamtzahl der Tupel. Bildlich gesprochen teilt sich jedes Tupel auf alle Klassen, zu denen es gehört, auf und wird durch die anschließende Summation wieder zusammengeführt, so dass sich vor und nach der Aggregation die selbe Gesamtzahl ergibt. Analoges gilt für die Aggregatfunktion „Summe“. Auf den Durchschnitt lässt sich dies nicht so einfach übertragen, da sich, wie auch bei scharfen Klassen, der Gesamtdurchschnitt nicht aus den einzelnen Durchschnitten der Klassen berechnen lässt. Weil der Durchschnitt aber als Quotient aus Summe und Anzahl gebildet wird und diese korrekt berechnet werden, gilt dies auch für den Durchschnitt. Ähnlich ist es mit der Standardabweichung, die sich durch Anzahl, Summe und Summe der Quadrate ausdrücken lässt.

2.4.4 Anforderungen an die Klassenbildung

In Abschnitt 2.3.2 wurden verschiedene Operatoren vorgestellt, mit Hilfe derer die Zugehörigkeit eines Tupels zu einer unscharfen Klasse berechnet werden kann. Jedoch genügen diese Operatoren im Allgemeinen nicht der im vorigen Abschnitt geforderten Bedingung; die durch sie berechneten Zugehörigkeiten ergeben über alle Klassen summiert nicht den Wert eins.

2 Theoretische Überlegungen

	(jung, klein)	(jung, groß)	(alt, klein)	(alt, groß)	Σ
μ^{min}	0,20	0,40	0,20	0,60	1,40
$\mu^{min,norm}$	0,14	0,29	0,14	0,43	1,00
μ^{log}	0,08	0,32	0,12	0,48	1,00
$\mu^{log,norm}$	0,08	0,32	0,12	0,48	1,00
$\mu^{komp}, \gamma = 0,5$	0,20	0,53	0,29	0,66	1,68
$\mu^{komp,norm}, \gamma = 0,5$	0,12	0,32	0,17	0,39	1,00

Tabelle 2.4: Beispiel für Normierung der Zugehörigkeit zu unscharfen Klassen

Aus diesem Grund ist eine Normierung der Klassenzugehörigkeiten erforderlich, wie sie in [Sch98] beschrieben wird. Dazu wird die berechnete Klassenzugehörigkeit eines Tupels durch die über alle Klassen aufsummierten Zugehörigkeiten des Tupels dividiert.

$$\mu_{K_i}^{norm} = \mu_{K_i} / \sum_{j=1}^n \mu_{K_j} \quad (2.7)$$

Man erhält so aus der absoluten Klassenzugehörigkeit die relative Zugehörigkeit in Bezug auf die Zugehörigkeiten zu den anderen Klassen. Beispielhaft ist diese Normierung in Tabelle 2.4 für die Werte aus Tabelle 2.3.2 durchgeführt.

2.4.5 Anforderungen an linguistische Variablen

Bisher wurden an die Zugehörigkeitsfunktionen der Terme einer linguistischen Variable keine besonderen Anforderungen gestellt. An sich ist das auch nicht notwendig, da durch die Normierung auf Ebene der Klassen die Aggregationsbedingungen erfüllt sind, egal wie die Zugehörigkeiten zu den Termen aussehen.

Trotzdem ist es sinnvoll, auch die Terme so zu definieren, dass die Summe ihrer Zugehörigkeitsfunktionen für jeden Wert eins ergibt. Dies soll anhand eines kleinen Beispiels motiviert werden: Nehmen wir an, für das Attribut *Alter* sei eine linguistische Variable mit den Termen *jung* und *sehr jung* definiert. Für den Wert 80 sei die Zugehörigkeit $\mu_{jung}(80) = 0,1$ und die Zugehörigkeit $\mu_{sehr\ jung}(80) = 0,05$. Benutzt man zur Klassifikation lediglich das Alter, so ergeben sich nur zwei Klassen K_1 und K_2 , die durch die Terme *jung* bzw. *sehr jung* gekennzeichnet sind. Durch die Normierung ergeben sich Zugehörigkeiten $\mu_{K_1}^{norm}(80) = 0,1/(0,1 + 0,05) = 0,67$ und $\mu_{K_2}^{norm}(80) = 0,05/(0,1 + 0,05) = 0,33$. Obwohl der Term *jung* nur zum Grad 0,1 auf einen Achtzigjährigen zutrifft, gehört dieser dennoch zum Grad 0,67 zur Klasse K_1 , die durch den Term *jung* beschrieben wird.

Dies erklärt sich dadurch, dass die relative Zugehörigkeit in Bezug auf zwei Klassen berechnet wurde, zu denen die absolute Zugehörigkeit jeweils sehr gering ist und es keine Klasse gibt, zu der der Achtzigjährige eine starke Zugehörigkeit besitzt. Die Unstimmigkeit lässt sich dadurch vermeiden, dass die Terme einer linguistischen Variable stets so definiert werden, dass die Summe der Zugehörigkeitsfunktionen eins ergibt. Im Beispiel könnte dies durch Hinzunahme eines Terms *alt* mit der Zugehörigkeit $\mu_{alt}(80) = 0,85$ erreicht werden.

Die Hinzunahme eines komplementären Terms löst das Problem, dass die Zugehörigkeitsfunktionen in Summe weniger als eins ergeben. Zugehörigkeitsfunktionen, deren Summe größer als eins ist, können dadurch vermieden werden, dass man ihre Terme in parallele

Hierarchien einordnet. Es ist ohnehin nicht sinnvoll, Terme zur Klassifikation heranzuziehen, die für ein Tupel alle sehr zutreffend sind. Durch die Bildung paralleler Hierarchien kann man entweder den einen oder den anderen gut zutreffenden Term zur Klassifikation nutzen.

Es soll noch einmal erwähnt werden, dass die Normierung bei der Klassenbildung im Allgemeinen auch erforderlich ist, wenn die Summe der Zugehörigkeitsfunktionen einer jeden linguistischen Variable eins ist. Lediglich das logische Und als Operator überträgt diese Eigenschaft auf die Klassen, so dass die Normierung in diesem Fall entfallen kann. Dies sieht man beispielhaft an Tabelle 2.4, wo $\mu^{log,norm}$ mit μ^{log} identisch ist.

Wenn alle linguistischen Variablen so definiert sind, dass die Summe ihrer Zugehörigkeitsfunktionen eins ist, besteht auch bei der Definition von Hierarchien unscharfer Kontexte nicht die Gefahr, dass sich für die Zugehörigkeitsfunktionen in Bezug auf die Elementarwerte größere Werte als eins ergeben. Ist nämlich für die Terme jeder Ebene die Summe der Zugehörigkeitsfunktionen bezogen auf die Terme der nächstunteren Ebene eins, so überträgt sich dies auch auf die Zugehörigkeitsfunktionen der Terme einer jeden Ebene bezogen auf die Elementarwerte. Ist aber deren Summe eins, so kann der Wert einer einzelnen Zugehörigkeitsfunktion nicht größer als eins sein.

2.4.6 Klassifikation von Objekten, die durch mehrere Tupel beschrieben werden

Bisher wurden lediglich einzelne Tupel klassifiziert. Oft möchte man jedoch auch Objekte klassifizieren, die durch mehrere Tupel beschrieben werden, z.B. können für eine Strecke mehrere Geschwindigkeitsmessungen vorgenommen worden sein, anhand derer man die Strecke klassifizieren möchte. Im Falle scharfer Klassen ist eine Klassifikation der Strecke nicht möglich, die einzelnen Messungen können ja in verschiedene Klassen fallen und welcher Klasse sollte man die Strecke dann zuordnen?

Bei der Verwendung unscharfer Klassen hingegen ist es kein Problem, wenn die verschiedenen ein Objekt beschreibenden Tupel dieses in unterschiedliche Klassen einordnen, da das Objekt ja zu mehreren Klassen gehören kann. Die Zugehörigkeit des Gesamtobjekts zu den Klassen ergibt sich nach [Sch98] aus den Zugehörigkeiten der einzelnen Tupel. Da die Tupel alle zusammen das Objekt beschreiben, werden Zugehörigkeiten für jede Klasse aufsummiert. Anschließend wird wie bei der unscharfen Klassifizierung einzelner Tupel normiert.

2.4.7 Wiederverwendbarkeit aggregierter Werte

Will man Aggregate einer bestimmten Granularität berechnen, kann der Rechenaufwand reduziert werden, falls man bereits auf feinerer Granularität aggregierte Werte zur Berechnung verwendet, anstatt auf die Elementardaten zurückzugreifen (beschrieben z.B. in [Leh03]). Dies ist jedoch nur unter bestimmten Bedingungen möglich. Zunächst wird wieder nur der Fall scharfer Klassen betrachtet.

Bei der Wiederverwendung aggregierter Werte soll aus mehreren auf feiner Granularität aggregierten Werten ein Aggregat grober Granularität berechnet werden. Voraussetzung hierfür ist, dass eine Klasse der groben Granularität aus mehreren Klassen der feinen Granularität zusammengesetzt werden kann. Wegen der Eigenschaften Vollständigkeit und Überlappungsfreiheit, die die Klassen ohnehin erfüllen müssen, damit sinnvoll aggregiert werden

2 Theoretische Überlegungen

kann, ist dies dann der Fall, wenn die Klassen der Klasseneinteilung grober Granularität U in der Klassenhierarchie oberhalb der Klassen der Klasseneinteilung feiner Granularität V liegen, wenn also $U \succeq V$ gilt.

Aus Aggregaten für Tage und Landkreise lassen sich z.B. Aggregate für Monate und Bundesländer berechnen. Hingegen kann man aus Aggregaten für Wochen und Landkreise keine Aggregate für Monate und Bundesländer berechnen, da es sich hier um Klassen zweier paralleler Hierarchien handelt (die Wochen lassen sich nicht eindeutig Monaten zuordnen).

Nicht alle Aggregatfunktionen erlauben es, aus voraggregierten Werten weitere Aggregate zu berechnen. Die verwendete Aggregatfunktion muss hierzu die Eigenschaft haben, dass das aus einer bestimmten Menge gebildete Aggregat gleich dem Aggregat der Aggregate von Partitionen dieser Menge ist. Für eine Aggregatfunktion f , die die Werte des Attributs A aggregiert, muss daher gelten:

$$\forall X_1, X_2 \subset D(A) : f(X_1 \cup X_2) = f(\{f(X_1), f(X_2)\}) \quad (2.8)$$

Eine solche Aggregatfunktion heißt semiadditiv. Summe, Anzahl³ Minimum und Maximum sind semiadditive Aggregatfunktionen. Weiter existierten Aggregatfunktionen, die sich mit Hilfe semiadditiver Funktionen berechnen lassen. Der Durchschnitt beispielsweise ist der Quotient aus Summe und Anzahl. Er kann nicht in jedem Fall aus bereits vorberechneten Durchschnitten feinerer Granularität berechnet werden, da er selbst keine semiadditive Funktion ist, wohl aber aus vorberechneten Summen und Anzahlen.

Problematisch ist die Verwendung vorberechneter Aggregate bei unscharfen Klassen. Zwar lässt sich für jede Dimension angeben, wie hoch die Zugehörigkeit eines Terms einer niedrigeren Ebene zu den Termen einer höheren Ebene ist; für Klassen, die aus mehreren Termen unterschiedlicher Dimensionen gebildet werden, lassen sich solche Zugehörigkeiten jedoch im Allgemeinen nicht mehr angeben. Grund hierfür ist die Normierung bei der Berechnung der Zugehörigkeit zu einer unscharfen Klasse. Da der Normierungsfaktor für jedes Tupel ein anderer ist, lässt sich keine Zugehörigkeit von Klassen zu anderen Klassen angeben. Folglich können für eine Klasse berechnete Aggregate nicht zur Berechnung von Aggregaten anderer Klassen herangezogen werden.

Lediglich wenn das logische Und als Verknüpfung gewählt wird, ist dies möglich. Wenn nämlich zusätzlich für jede linguistische Variable die Summe der Zugehörigkeitsfunktionen aller Terme eins ergibt und einzelne Tupel klassifiziert werden, ist bei der Berechnung der Klassenzugehörigkeit keine Normierung erforderlich. In diesem Fall kann die Zugehörigkeit einer Klasse feinerer Granularität zu einer Klasse grober Granularität als Produkt der Zugehörigkeiten der Terme berechnet werden, die die Klassen jeweils kennzeichnen.

Beispiel: Der Term *mittel* habe zum eine Ebene höher definierten Term *ehrer klein* die Zugehörigkeit 0,5. Für ein anderes Attribut sei der Term *fast fehlerfrei* definiert, eine Ebene höher der Term *gut*, die Zugehörigkeit von *fast fehlerfrei* zu *gut* sei 0,8. Die aus den Termen *mittel* und *fast fehlerfrei* gebildete Klasse hat nun zu der Klasse, die aus den Termen *ehrer klein* und *gut* gebildet wird, die Zugehörigkeit $0,5 \cdot 0,8 = 0,4$.

Wird für ein Attribut in beiden Klassen der selbe Term verwendet, muss dieser natürlich bei der Multiplikation nicht berücksichtigt werden. Man kann auch sagen, jeder Term hat zu sich selbst die Zugehörigkeit eins. Die Klasse aus den Termen *mittel* und *gut* hat daher zur Klasse aus den Termen *ehrer klein* und *gut* die Zugehörigkeit $0,5 \cdot 1 = 0,5$.

³Die Aggregatfunktion „Anzahl“ muss als Summe über ein spezielles Attribut mit dem Wert eins definiert werden, damit Formel 2.8 gilt.

Mit Hilfe dieser Zugehörigkeiten lassen sich nun aus Aggregaten für in der Hierarchie weiter unten liegende Klassen Aggregate für Klassen berechnen, die in der Hierarchie weiter oben liegen. Voraussetzung hierfür sind wie bei scharfen Klassen semiadditive Aggregatfunktionen.

2.5 Zusammenfassung

In diesem Kapitel wurden Kontexte als ein Mittel zur Strukturierung von Daten eingeführt. Über Kontexte lassen sich Klassen definieren, in denen ähnliche Daten zusammengefasst sind. Durch Kontexte lässt sich Imperfektion ausdrücken, da Anfragen, die mit Kontexten arbeiten, nicht den Anspruch erheben, dass die zugrunde liegenden Daten exakt sind.

Es wurde festgestellt, dass ein enger Zusammenhang zwischen Kontexten und dem multidimensionalen Datenmodell besteht: Beides sind Konzepte zur Strukturierung von Daten. Kontexte sind vergleichbar mit den Kategorieattributen im multidimensionalen Datenmodell. Auch Hierarchien lassen sich mit Hilfe von Kontexten aufbauen.

Über linguistische Variablen lassen sich unscharfe Kontexte definieren. Die sich daraus ergebenden unscharfen Klassen sind nicht mehr scharf gegeneinander abgegrenzt, sondern besitzen kontinuierliche Übergänge. Es wurde gezeigt, wie auch mit unscharfen Kontexten Hierarchien aufgebaut werden können.

Sowohl über scharfe als auch über unscharfe Klassen kann aggregiert werden. Für die Aggregation über unscharfe Klassen muss jedoch zunächst die Definition der Aggregatfunktionen erweitert werden, die ja in ihrer ursprünglichen Form eine scharfe Trennung zwischen den Klassen voraussetzen, weswegen eine entsprechende Erweiterung der Aggregatfunktionen vorgeschlagen wurde. Damit die Aggregation sinnvolle Ergebnisse liefert, müssen für scharfe Einteilungen die Bedingungen Überlappungsfreiheit, Vollständigkeit und Typverträglichkeit erfüllt sein. Bei Einteilungen, die durch Anwendung scharfer Kontexte entstehen, ist dies definitionsgemäß der Fall. Für unscharfe Einteilungen wurde eine Neuformulierung der ersten beiden Bedingungen vorgeschlagen.

Lässt sich eine Klasse aus mehreren Klassen einer niedrigeren Hierarchieebene zusammensetzen, so können semiadditive Aggregatfunktionen für die grobgranulare Klasse aus den Aggregaten der Klassen feinerer Granularität berechnet werden, aus denen sie besteht.

2 Theoretische Überlegungen

3 Analyse der Daten

In diesem Kapitel wird untersucht, inwiefern Imperfektion in den vorliegenden Verkehrsdaten enthalten ist. Zunächst wird kurz auf die unterschiedlichen Arten von in Verkehrsdaten enthaltener Imperfektion eingegangen. Anschließend werden die Daten beschrieben. Es handelt sich dabei zum einen um so genannte Floating Car Daten (FCD), die durch Sensoren im Fahrzeug erfasst werden. Weiter werden Simulationsdaten für Verkehrsmessstationen untersucht. Als letztes werden Daten einer Statistik über den Güterverkehr in Deutschland behandelt. Auf die Beschreibung der Daten folgt jeweils die eigentliche Untersuchung auf Imperfektion.

3.1 Imperfektion in Verkehrsdaten

In [Koo04] wird beschrieben, welche Arten von Imperfektion in Verkehrsdaten enthalten sein können. Dabei wird zwischen unsicherem, unscharfem und ungenauem Wissen unterschieden.

Unsicheres Wissen ist Wissen, bei dem nicht sicher ist, ob es wahr oder falsch ist. Unscharfes Wissen entsteht bei der Anwendung von Kategorien ohne scharf definierte Grenzen, wie z.B. den im vorigen Kapitel beschriebenen unscharfen Klassen. Bei ungenauem Wissen schließlich sind keine exakten Werte bekannt, sondern lediglich Intervalle oder ähnliche grobkörnige Einheiten. Es entsteht z.B. bei physikalischen Messungen, die immer mit einer gewissen Ungenauigkeit behaftet sind.

Welche Art von Imperfektion in Verkehrsdaten enthalten ist, hängt stark ab von der Art der Daten und vor allem von der Weise, in der diese Daten gewonnen werden. Messdaten, die von Sensoren an der Strecke, wie z.B. Induktionsschleifen, aufgezeichnet werden, sind ungenaue Daten. Messdaten, die aus Sensoren im Fahrzeug stammen, sind ebenfalls ungenau. Werden diese Sensordaten benutzt, um weitere Informationen abzuleiten, die nicht direkt gemessen werden können, sind diese zusätzlich unscharf, wenn sie durch Anwendung von Kategorien entstehen und unsicher, wenn sie vom persönlichen Fahrverhalten abhängen, da auch Fahrer mit atypischem Fahrverhalten existieren.

Daten, die von menschlichen Beobachtern aufgenommen werden, wie z.B. Staumeldungen, sind unsicher, da immer die Gefahr einer Falschmeldung besteht, unscharf, da der Mensch immer eine subjektive Einteilung in Kategorien vornimmt und ungenau, da ein menschlicher Beobachter noch viel weniger als ein technisches System im Stande ist, genaue Messungen vorzunehmen.

Infrastrukturdaten, wie z.B. Straßenkarten, enthalten an sich wenig Imperfektion. Informationen über die Länge und den Verlauf von Strecken sind jedoch in gewissem Maße ungenau, besonders wenn sich durch Baumaßnahmen Änderungen ergeben.

Daten über Störungen, wie z.B. Baustellen, sind mit Unsicherheit behaftet, was die Dauer der Störung angeht. Ein weiterer Störfaktor ist das Wetter; Wetterprognosen sind zum einen ungenau und zum anderen unsicher.

3.2 Floating Car Daten

Als Floating Car Daten werden Daten bezeichnet, die von Sensoren im Fahrzeug erfasst und zur weiteren Verarbeitung an eine Zentrale übermittelt werden. Ziel ist es, aus den von einer repräsentativen Stichprobenflotte übermittelten Daten Rückschlüsse auf den aktuellen Verkehrszustand zu ziehen und diese Informationen dann allen Verkehrsteilnehmern zugänglich zu machen. Sie können dann als Grundlage für Navigation und Routenplanung dienen.

Die hier behandelten Daten wurden von der Firma gedas GmbH im Rahmen des Projekts WAYflow in Frankfurt bzw. Hessen gesammelt ([SW03]).

3.2.1 Beschreibung des Messverfahrens

Das zum Sammeln der Daten verwendete Verfahren beruht darauf, dass die im Fahrzeug gemessenen Daten nicht ständig, sondern nur an bestimmten Knoten übermittelt werden. Dazu wird das Verkehrsnetz in so genannte Links eingeteilt; ein Link ist die Verbindungsstrecke zwischen zwei Knoten und besitzt einen Start- und einen Endknoten. Ein Fahrzeug beginnt seine Messung bei Erreichen des Startknotens und übermittelt die gemessenen Daten bei Erreichen des Endknotens. Die Daten beziehen sich dann auf den jeweiligen Link zwischen Start- und Endknoten.

3.2.2 Erfasste Daten

Pro Link werden folgende Daten erfasst und übermittelt:

- die durchschnittliche Geschwindigkeit
- die Standardabweichung von diesem Durchschnitt
- der prozentuale Stillstandsanteil
- die minimale und die maximale Geschwindigkeit
- die durchschnittliche positive und negative Beschleunigung

Weiterhin wird der Typ des Fahrzeugs übermittelt. Zusätzlich wird noch der Zeitpunkt der Erfassung durch den Endknoten sowie die Straßenkategorie (Autobahn, Landstraße oder Stadtstraße) festgehalten.

Da sich die übermittelten Daten auf den gesamten Link beziehen, handelt es sich um bereits aggregierte Werte: Aus vielen während des Durchfahrens des Links gesammelten Messwerten werden die oben genannten Werte gebildet, die dann bei Erreichen des Endknotens an diesen übermittelt werden.

3.2.3 Untersuchung auf Imperfektion

Nun wird untersucht, in welcher Hinsicht Imperfektion in den einzelnen Komponenten der Messdaten vorliegt.

Erfassungszeitpunkt

An sich ist der Erfassungszeitpunkt ein recht präziser Wert, der mit hoher Genauigkeit gemessen werden kann. Allerdings muss man bedenken, dass die Daten, die das Fahrzeug verschickt, wenn es vom Endknoten erfasst wird, nicht zum Erfassungszeitpunkt, sondern während der vorangegangenen Befahrung des Links gesammelt wurden. Die Zeitangabe, die eigentlich von Interesse ist, nämlich die Zeit, zu der die Werte gemessen wurden, ist ungenaues Wissen, da es sich hierbei um ein Intervall handelt. Hinzu kommt, dass von diesem Intervall nur die obere Grenze bekannt ist, die untere ließe sich jedoch mit Hilfe der Durchschnittsgeschwindigkeit und der Linklänge berechnen oder, falls vorhanden, durch den Zeitpunkt der Messung davor.

Jedoch ist eine scharfe Zeitangabe hier sowieso nicht von Nöten. Um den Verkehrszustand eines Links zuverlässig einzuschätzen, ist es sinnvoll, mehrere Fahrzeuge zu betrachten, die den Link in etwa zu dem Zeitpunkt befahren haben, für den der Zustand festgestellt werden soll. Will man den aktuellen Verkehrszustand feststellen, hat man gar keine andere Wahl, als Werte von Befahrungen aus der nahen Vergangenheit zu betrachten.

Fahrzeugtyp

Der Fahrzeugtyp ist für jedes Fahrzeug der Testflotte eindeutig festgelegt. Daher ist das entsprechende Attribut nicht mit Imperfektion behaftet.

Geschwindigkeit, Beschleunigung, Stillstandsanteil

Die Geschwindigkeit ist ein physikalischer Messwert und fällt als solcher in die Kategorie „ungenaueres Wissen“. Die Beschleunigung und der Stillstandsanteil werden aus der Geschwindigkeit und der Zeit berechnet und sind deswegen ebenfalls ungenau. Die Werte sind in Intervallen von 1km/h bei der Geschwindigkeit, $0,1\text{km/h pro } 10\text{s}$ bei der Beschleunigung und 1% beim Stillstandsanteil bekannt.

Straßenkategorie

Die Straßenkategorie ergibt sich aus einer scharfen Einteilung der Strecken und enthält daher keinerlei Imperfektion. Es ist jedoch die Frage, ob eine solche scharfe Einteilung wirklich sinnvoll ist. Zwar existiert z.B. eine genaue Festlegung, welche Strecken Autobahnen sind, aber eine mehrspurig ausgebaute Bundesstraße ohne Ampeln hat wahrscheinlich eher den Charakter einer Autobahn als den einer Landstraße. Auch lassen sich viele Strecken nicht eindeutig einer Kategorie zuordnen. Eine unscharfe Einteilung wäre deswegen eventuell angebrachter.

Verkehrszustand

Der berechnete Verkehrszustand stellt unsicheres Wissen dar, da nicht gewährleistet ist, dass der aus den Stichproben berechnete Zustand dem wirklichen Zustand entspricht. Hinzu kommt, dass die Verkehrszustände recht subjektive Begriffe sind; was der eine Autofahrer als zähfließenden Verkehr betrachtet, ist für den anderen vielleicht schon Stau. Definiert man die Verkehrszustände als Konsequenz dessen durch unscharfe Kategorien, wird das Wissen zusätzlich unscharf.

Nichtvorhandensein von Daten

Auch das Nichtvorhandensein von Daten beinhaltet Imperfektion. Wenn für einen Link innerhalb eines längeren Zeitraums keine Daten anfallen, kann eigentlich keine Aussage über den Verkehrszustand des Links getroffen werden.

Ein Link, für den keine Daten vorhanden sind, wurde in letzter Zeit nicht von einem FCD-Fahrzeug befahren. Nimmt man an, dass der Anteil von FCD-Fahrzeugen an der Gesamtzahl der Fahrzeuge ausreichend hoch und statistisch über die unterschiedlichen Arten von Verkehrsteilnehmern (also z.B. Berufspendler, Wochenendausflügler, Berufskraftfahrer, etc.) verteilt ist, so kann man annehmen, dass die Strecke von überhaupt keinem Fahrzeug befahren wird – der Verkehrszustand ist also „freie Fahrt“.

Andererseits könnte der Link sehr wohl von Fahrzeugen befahren sein, unter denen sich nur zufällig kein Fahrzeug der Stichprobenflotte befindet. Möglicherweise ist auf der Strecke Stau, in dem nur gerade kein Fahrzeug der Stichprobenflotte steht. Auch die Abwesenheit von Daten ist also unsicheres Wissen. Die Unsicherheit dieses Wissens lässt sich durch Vergrößerung der Stichprobenflotte verringern.

3.3 Simulationsdaten für Verkehrsmessstationen

Im Gegensatz zu den im vorigen Abschnitt behandelten Floating Car Daten, die aus Messungen im wirklich existierenden Verkehrsgeschehen entstanden sind, werden nun Daten betrachtet, welche aus einem simulierten Verkehrsgeschehen stammen. Die Daten wurden unter Verwendung des Programms VISSIM der Firma PTV AG simuliert.

3.3.1 Das Programm VISSIM

Das Programm simuliert Verkehrsflüsse und speichert Daten, die an simulierten Messstationen gesammelt werden. In der Realität würden diese Messstationen unter der Fahrbahn verlegten Messschleifen, Lichtschranken, o.ä. entsprechen. Für die Simulation lassen sich Szenarien mit einzelnen Fahrspuren, Ampeln, Messstationen und verschiedenen Fahrzeugtypen, jeweils mit bestimmten Eigenschaften, definieren. So kann der Verkehr einer Kreuzung oder eines ganzen Stadtteils simuliert werden.

3.3.2 Simulierte Daten

Die hier betrachteten Daten werden von simulierten Messstationen erzeugt. Diese lassen sich an bestimmten Stellen einer Fahrspur anlegen und erfassen von den vorüber fahrenden Fahrzeugen

- die eindeutige Fahrzeugnummer des simulierten Fahrzeugs
- den Typ des Fahrzeugs
- den Zeitpunkt des Einfahrens in die und des Ausfahrens aus der Messstation
- die Geschwindigkeit und die Beschleunigung des Fahrzeugs
- die Anzahl der Personen, die sich in dem Fahrzeug befinden
- die Zeit, die das Fahrzeug während der Simulation schon gestanden hat

- die Länge des Fahrzeugs

[San04] beschäftigt sich mit der Frage, wie die Simulationsdaten des Programms VISSIM in einem Data Warehouse gespeichert werden können. Im Rahmen dessen wird ein relationales Schema für die Daten konzipiert und umgesetzt. Dieses relationale Schema soll hier die Grundlage für eine Erweiterung durch Kontexte sein.

Das Simulationsprogramm speichert die oben aufgeführten Werte pro Durchfahrung einer Messstation zweimal ab: einmal für den Zeitpunkt des Einfahrens in die Station und einmal für den Zeitpunkt des Ausfahrens aus der Station. Da Ein- und Ausfahren zeitlich sehr nahe beieinander liegen, und daher auch die Werte für beide Zeitpunkte sehr ähnlich sind, wird in [San04] die Vereinfachung vorgenommen, dass nur die Werte für den Zeitpunkt des Einfahrens in das Schema übernommen werden.

3.3.3 Untersuchung auf Imperfektion

Da es sich bei den betrachteten Daten um simulierte Daten handelt, sind die Werte natürlich mit der Genauigkeit bekannt, mit der sie abgespeichert werden. Es handelt sich also um Daten, die nicht mit Imperfektion behaftet sind. In zweierlei Hinsicht lässt sich die Imperfektion dennoch ins Spiel bringen:

Zum einen soll durch das Szenario, das der Simulation zugrunde liegt, in den meisten Fällen eine tatsächlich existierende Situation, also beispielsweise eine bestimmte Kreuzung in einer bestimmten Stadt, modelliert werden. Die simulierten Daten erheben also den Anspruch, den Daten ähnlich zu sein, die gesammelt würden, wenn tatsächlich in der Realität Messungen an der betreffenden Strecke vorgenommen würden. Betrachtet man die Simulationsdaten aus diesem Blickwinkel, so handelt es sich um hochgradig imperfekte Daten. Die für ein simuliertes Fahrzeug an einer simulierten Messstation gesammelten Daten sind nicht nur unsicher, sondern garantiert falsch, da in Wirklichkeit zu der betreffenden Zeit bestimmt kein Fahrzeug mit den selben Parametern die betreffende Stelle passiert hat. Betrachtet man jedoch die Daten nicht für ein einzelnes Fahrzeug, sondern auf einer gröberen Ebene aggregiert, z.B. für eine ganze Stunde oder einen ganzen Tag, so können die simulierten Werte z.B. für die Anzahl der vorüber fahrenden Fahrzeuge oder deren Durchschnittsgeschwindigkeit, eine gute Simulation vorausgesetzt, der Realität recht nahe kommen. Es handelt sich hierbei also um ungenaue Werte.

Zum anderen kann man die simulierten Daten natürlich so betrachten, als ob es reale Messwerte wären und so tun, als ob sie mit der gleichen Imperfektion behaftet wären wie entsprechende in der Realität gemessene Werte. Dann wären z.B. die Geschwindigkeit und die Beschleunigung ungenaue Werte, ebenso wie die Länge des Fahrzeugs. Andere Werte könnten bei einer realen Messung gar nicht erfasst werden, es gäbe z.B. keine Fahrzeugnummer. Die Anzahl der Personen im Fahrzeug ließe sich durch Messstationen unter der Fahrbahn oder am Fahrbahnrand ebenso wenig bestimmen wie die Zeit, die das Fahrzeug schon gestanden hat. Der Typ des Fahrzeugs ließe sich allenfalls anhand anderer gemessener Werte wie z.B. des Gewichts und der Länge bestimmen und wäre dann auf jeden Fall unsicher.

3.3.4 Vergleich Simulationsdaten – Floating Car Daten

Die Simulationsdaten und die in Abschnitt 3.2 behandelten Floating Car Daten sind einander ähnlich, da die selben Größen simuliert bzw. gemessen werden: die Geschwindigkeit

3 Analyse der Daten

und Beschleunigung eines Fahrzeugs, sowie die Zeit, in der das Fahrzeug gestanden hat.

Der Unterschied zwischen den beiden Arten von Daten besteht jedoch darin, dass die Simulationsdaten sich auf einen bestimmten Punkt, nämlich den Ort der Messstation und einen bestimmten Zeitpunkt, nämlich den, zu dem das Fahrzeug die Station passiert, beziehen, die Floating Car Daten jedoch über einen ganzen Streckenabschnitt und damit auch über einen ganzen Zeitraum hinweg gesammelt werden; es handelt sich um über einen gesamten Link aggregierte Werte.

Für die Simulationsdaten lassen sich solche aggregierten Werte für eine Strecke nur berechnen, falls genügend Messstationen entlang der betreffenden Strecke vorhanden sind. In der Simulation ließe sich eine ausreichend hohe Dichte eventuell noch erreichen, in der Realität wohl kaum. Eine Bestimmung der durchschnittlichen Beschleunigung und der Standardabweichung der Geschwindigkeit ist deswegen durch Messstationen nicht möglich. Für die Bestimmung der Durchschnittsgeschwindigkeit reichen je eine Messstation am Anfang und am Ende der Strecke, vorausgesetzt, man kennt deren Länge. In diesem Fall lässt sich die Durchschnittsgeschwindigkeit aus den Zeiten, zu denen das Fahrzeug die Messstationen passiert, und der Länge der Strecke berechnen. Allerdings setzt dies voraus, dass Fahrzeuge eindeutig identifiziert werden können.

Auch der relative Stillstandsanteil lässt sich in der Simulation anhand je einer Messstation am Anfang und am Ende der Strecke bestimmen. Bei jeder Station wird die Gesamtstandzeit des Fahrzeugs erfasst. Zieht man also die an der Station am Ende der Strecke erfasste Gesamtstandzeit von der an der Station am Anfang erfassten ab, erhält man die Zeit, in der das Fahrzeug auf der Strecke gestanden hat. Um den relativen Stillstandsanteil zu erhalten, muss diese Zeit noch durch die Zeit dividiert werden, die das Fahrzeug zum Zurücklegen der Strecke benötigt hat.

Daten, die an Messstationen gesammelt werden, lassen sich also nur zum Teil in eine über Strecken aggregierte Form umwandeln. Umgekehrt lassen sich aus den aggregierten Floating Car Daten keine Werte für einen bestimmten Punkt auf der Strecke zurückrechnen.

3.4 Güterverkehrsdaten

Die in diesem Abschnitt untersuchten Daten beschreiben alle Güterverkehrstransporte, die in den Berichtsjahren 1999, 2001 und 2002 von in Deutschland gemeldeten Fahrzeugen durchgeführt wurden. Dies umfasst den Verkehr dieser Fahrzeuge sowohl bei Fahrten durch Deutschland als auch bei Fahrten im Ausland. Die Daten wurden vom Kraftfahrt-Bundesamt und vom Bundesamt für Güterverkehr erhoben.

3.4.1 Verfahren zur Erhebung der Daten

Die Daten wurden auf Grundlage von Stichproben erhoben. Um Aussagen für das gesamte Transportaufkommen treffen zu können, wurden die Ergebnisse der Stichproben aggregiert und anschließend mit Hochrechnungsfaktoren multipliziert. Die Aggregation sichert die Anonymität der Daten, da aus den aggregierten Werten die tatsächlichen Werte nicht mehr zurückgerechnet werden können. Weitere Informationen hierzu finden sich in [KBB01].

3.4.2 Erhobene Daten

Die erhobenen Daten können unterteilt werden in solche, die bei der Aggregation zur Gruppierung verwendet wurden und solche, die aggregiert wurden.

Zur Gruppierung wurden verwendet:

- das betrachtete Jahr
- die Art des Fahrzeugs und des Anhängers
- der Be- und Entladeort
- der genutzte Rauminhalt
- die Art der transportierten Güter
- die Form der Ladung
- die Größe der Container
- die Art der Fahrt
- das Verkehrsmittel, auf das die Fracht anschließend verladen wurde

Aggregiert wurden:

- die zurückgelegte Entfernung
- die Anzahl der zurückgelegten Einzelstrecken
- das transportierte Gewicht
- das zulässige Gesamtgewicht
- die Nutzlast

3.4.3 Untersuchung auf Imperfektion

Aggregierte Werte

Bei sämtlichen aggregierten Werten handelt es sich aufgrund der Hochrechnung nicht um exakte Werte. Selbst wenn diese Werte für eine einzelne Fahrt genau bekannt sind, wie das zulässige Gesamtgewicht oder die Anzahl der Einzelstrecken, wird dieses Wissen durch die Hochrechnung auf das Gesamtverkehrsaufkommen doch ungenau. Andere Werte, wie das transportierte Gewicht und die zurückgelegte Entfernung, sind schon für die einzelne Fahrt ungenau. Alle aggregierten Werte stellen also ungenaues Wissen dar.

Jahr, Containergröße, Art des Fahrzeugs

Das Jahr ist eine exakte und (sieht man von dem seltenen Fall von Transporten ab, die über die Jahreswende hinweg stattfinden) eindeutig bekannte Größe, die keinerlei Imperfektion enthält. Bei der Containergröße wurde eine Einteilung mit scharfen Grenzen vorgenommen, deswegen liegt auch hier keine Imperfektion vor.

Auch die Art des Fahrzeugs, bei der zwischen LKW und Sattelzugmaschinen, sowie nach der Art des Aufbaus und des Anhängers unterschieden wird, stellt eine scharfe Einteilung dar.

Be- und Entladeort

Zur Anonymisierung der Daten wird nicht der genaue Be- und Entladeort eines Fahrzeugs, sondern lediglich das Bundesland, bei Be- oder Entladung im Ausland nur das Land, bekanntgegeben. Die Orte sind also nur auf einer sehr grobkörnigen Ebene bekannt. Es handelt sich hier also um ungenaues Wissen.

Art der transportierten Güter, Form der Ladung

Ähnlich wie bei der Straßenkategorie der Floating Car Daten liegen auch hier scharfe Einteilungen vor, wobei es sicherlich Daten gibt, die sich nicht eindeutig zuordnen lassen. Beispielsweise könnten auf einer Fahrt mehrere Arten von Gütern transportiert werden oder Güter, die in keine vorhandene Kategorie passen.

Eine Fahrt wird immer der Güterart zugeordnet, deren Gewicht den höchsten Anteil am Gesamtgewicht aller transportierten Güter hat. Werden z.B. 6000kg Getreide und 4000kg Kartoffeln transportiert, so wird das gesamte Gewicht von 10000kg in der Kategorie „Getreide“ aufgeführt. Eine unscharfe Zuordnung, z.B. mit dem Grad 0,6 zur Kategorie „Getreide“ und mit dem Grad 0,4 zur Kategorie „Kartoffeln“, würde zwar zu einer höheren Genauigkeit führen, jedoch wäre auch ein beträchtlich höherer Aufwand bei der Erfassung der Daten die Folge, der nicht gerechtfertigt ist; das transportierte Gewicht ist als hochgerechneter Wert ohnehin ungenau. Die Ungenauigkeit der aggregierten Werte wird durch die Zuordnung der Fahrten zu scharfen Kategorien jedoch weiter erhöht.

Genutzter Rauminhalt

Der von der Fracht in Anspruch genommene Rauminhalt lässt sich in vielen Fällen nur grob abschätzen. Daher handelt es sich hier um einen hochgradig ungenauen Wert.

3.5 Zusammenfassung

Verkehrsdaten können mit unterschiedlichen Arten von Imperfektion behaftet sein. Man unterscheidet zwischen unsicherem, unscharfem und ungenauem Wissen.

Die Floating Car Daten, die im Fahrzeug gemessen werden, stellen wegen der begrenzten Genauigkeit der Messung ungenaues Wissen dar. Da die Messwerte nur stichprobenartig von einem gewissen Anteil der am Verkehr beteiligten Fahrzeuge gemessen werden, ergibt sich aus dem Nichtvorhandensein von Messwerten unsicheres Wissen.

Die Simulationsdaten des Programms VISSIM sind als solche nicht mit Imperfektion behaftet. Betrachtet man sie jedoch mit dem Anspruch, dass sie das wirkliche Verkehrsgeschehen beschreiben sollen, kann man sie, aggregiert über einen gewissen Zeitraum, als ungenaue Daten betrachten. Eine andere Betrachtungsweise ergibt sich, wenn man die Simulationsdaten so behandelt, als ob sie mit Imperfektion behaftete Messwerte wären. Die simulierten Daten und die gemessenen Floating Car Daten beschreiben die selben Größen, jedoch beziehen sich erstere auf einen festen Ort und Zeitpunkt, letztere auf einen ganzen Streckenabschnitt und einen Zeitraum.

Bei den Güterverkehrsdaten handelt es sich um bereits aggregierte und hochgerechnete Werte. Durch das Hochrechnen sind sie mit einer gewissen Ungenauigkeit behaftet und fallen in die Kategorie „unsicheres Wissen“. Der Aggregation wurde eine scharfe Einteilung zugrunde gelegt. Die Art oder Form der transportierten Güter lässt sich in manchen Fällen

aber nicht exakt festlegen. Die künstlich scharfe Einteilung trägt mit zur Ungenauigkeit der aggregierten Werte bei.

3 *Analyse der Daten*

4 Konzeption

In diesem Kapitel werden Schemaerweiterungen durch Kontexte für die vorliegenden Verkehrsdaten konzipiert. Ziel ist es zum einen, die Imperfektion in den im vorigen Kapitel analysierten Daten zu repräsentieren, zum anderen die Daten zu strukturieren, damit Anfragen durch den Benutzer einfacher gestaltet werden können.

Im Rahmen der Konzeption wird untersucht, für welche Attribute es Sinn macht, Kontexte zu definieren. Weiter soll festgestellt werden, wo scharfe und wo unscharfe Einteilungen angebracht sind. An Stellen, an denen es sinnvoll ist, Einteilungen unterschiedlicher Granularitäten zu definieren, werden Hierarchien von Kontexten verwendet.

Die Erweiterungen sollen den in Kapitel 2 definierten Anforderungen für Aggregierbarkeit genügen. Daher wird auch untersucht, welche Attribute mit welchen Aggregatfunktionen aggregiert werden können und welche Attribute oder Kontexte sich zur Gruppierung eignen. Nacheinander werden wieder Floating Car Daten, Simulationsdaten und Güterverkehrsdaten behandelt.

4.1 Floating Car Daten

Bei den Floating Car Daten dienen Kontexte zur Definition von verschiedenen Verkehrszuständen, die auf einer Strecke herrschen können. Dazu werden Kontexte für die Attribute benötigt, welche die Geschwindigkeit, Beschleunigung und den Stillstandsanteil der Fahrzeuge beschreiben. Zur Strukturierung der Daten sind Kontexte für den Fahrzeugtyp und die Zeit der Messung sinnvoll.

4.1.1 Ermitteln des Verkehrszustandes

Der Verkehrszustand einer bestimmten Strecke soll anhand von Geschwindigkeit, Beschleunigung und Stillstandsanteil der Fahrzeuge, die auf der Strecke unterwegs sind, festgestellt werden. Nach [SW03] sind die Attribute, die hierfür betrachtet werden müssen, je nach Straßenkategorie andere. Für Autobahnen werden beispielsweise die Durchschnittsgeschwindigkeit und die Standardabweichung der Geschwindigkeit zur Bestimmung des Verkehrszustandes benutzt. Eine hohe Durchschnittsgeschwindigkeit, von der es nur geringe Abweichungen gibt, deutet auf frei fließenden Verkehr hin, während eine niedrigere Durchschnittsgeschwindigkeit mit einer hohen Standardabweichung für lebhaften Verkehr spricht.

[SW03] bestimmt den Verkehrszustand anhand von Trenngeraden¹ bzw. aus mehreren Geradenabschnitten zusammengesetzten Streckenzügen, die in einem Koordinatensystem definiert werden, bei dem die für die jeweilige Straßenkategorie relevanten Attribute an den Achsen abgetragen werden. Vereinfacht ist dies in Abbildung 4.2 dargestellt. Eine solche

¹Falls mehr als zwei Attribute betrachtet werden, handelt es sich dabei natürlich nicht mehr um Geraden, sondern um Ebenen oder Hyperebenen.

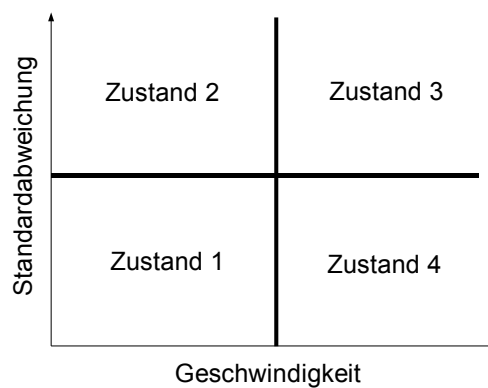


Abbildung 4.1: Einteilung in Klassen durch Kontexte

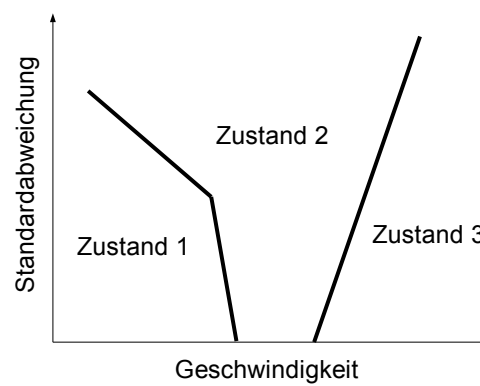


Abbildung 4.2: Einteilung in Klassen durch Trenngeraden

Einteilung kann mit dem Kontextmodell jedoch nicht ohne Weiteres bewerkstelligt werden. Da die Einteilung in Kontexte für jedes Attribut getrennt erfolgt, ergibt sich eine Klasseneinteilung, die einer Trennung durch horizontale und vertikale Geraden entspricht, wie sie in Abbildung 4.1 dargestellt ist. Daher wird zunächst eine solche einfachere Einteilung konzipiert, indem Kontexte für die einzelnen Attribute definiert werden.

Anschließend wird in Abschnitt 4.1.5 eine Alternative betrachtet, die darin besteht, dass zusätzliche Attribute eingeführt werden, welche als Werte den räumlichen Abstand des durch das Tupel im Koordinatensystem definierten Punktes zu den schräg verlaufenden Trenngeraden besitzen. Indem man Kontexte für diese zusätzlichen Attribute definiert, lässt sich schließlich eine Einteilung wie in Abbildung 4.2 erreichen.

4.1.2 Kontexte

Im Folgenden werden für die einzelnen Attribute sinnvolle Erweiterungen durch Kontexte vorgeschlagen. Hier wie auch für die in späteren Abschnitten behandelten Simulations- und Güterverkehrsdaten sollen die trivialen Kontexte, die sich für jedes Attribut definieren lassen, nicht jedesmal extra aufgeführt werden: Dies ist zum einen der Kontext, der nur aus einer einzigen Äquivalenzklasse besteht, welche sämtliche Attributwerte umfasst. Dieser Kontext wird immer dann verwendet, wenn das betreffende Attribut nicht ausschlaggebend für die Klassenzugehörigkeit sein soll. Zum anderen handelt es sich um den Kontext, der

sich ergibt, wenn man für jeden Attributwert eine eigene Äquivalenzklasse vorsieht. Wird dieser Kontext zur Klassenbildung verwendet, so wird für das entsprechende Attribut die feinstmögliche Einteilung verwendet.

Durchschnittsgeschwindigkeit

Eine Durchschnittsgeschwindigkeit von 50 km/h bedeutet auf einer Stadtstraße freie Fahrt – auf einer Autobahn hingegen deutet sie wohl eher auf stockenden Verkehr hin. Aus diesem Grund müssen für die Durchschnittsgeschwindigkeit verschiedene Kontexte definiert werden, die je nach Straßenkategorie Anwendung finden.

In Abschnitt 3.2.3 wurde motiviert, warum Verkehrszustände als unscharfe Kategorien modelliert werden sollten. Hinzu kommt, dass eine bestimmte Geschwindigkeit unterschiedlich interpretiert werden muss, je nachdem, ob sich ein Stau gerade bildet oder dabei ist, sich aufzulösen. Daher ist es angebracht, für die Durchschnittsgeschwindigkeit unscharfe Kontexte zu verwenden.

Es müssen also drei linguistische Variablen definiert werden – eine für Autobahnen (*Geschwindigkeit Autobahn*), eine für Landstraßen (*Geschwindigkeit Land*) und eine für Stadtstraßen (*Geschwindigkeit Stadt*). Nun stellt sich noch die Frage, wie viele Terme diese Variablen besitzen sollen. Da für den Endanwender, der seine Route planen möchte, letztlich nur interessant ist, ob er auf einer Strecke problemlos und schnell vorankommt oder nicht, reicht eine sehr grobe Einteilung aus. Die linguistischen Variablen erhalten daher jeweils zwei Terme – einen für niedrige und einen für hohe Geschwindigkeit. Die linguistische Variable *Geschwindigkeit Autobahn* beispielsweise besitzt also die Terme *Geschwindigkeit Autobahn niedrig* und *Geschwindigkeit Autobahn hoch*. Der Einfachheit halber sollen die Terme an Stellen, an denen klar ist, zu welcher linguistischen Variable sie gehören oder an denen Aussagen über alle linguistischen Variablen gemacht werden, mit *niedrig* und *hoch* abgekürzt werden.

Nun müssen noch die Zugehörigkeitsfunktionen festgelegt werden. Für jede der drei Straßenkategorien gibt es eine Geschwindigkeit, ab der man davon sprechen kann, dass die Geschwindigkeit für die betreffende Straßenkategorie hoch ist.² Ebenso gibt es eine Geschwindigkeit, unterhalb derer man definitiv von einer für die Straßenkategorie niedrigen Geschwindigkeit sprechen kann. Für jede der drei Straßenkategorien soll es daher eine untere Grenzgeschwindigkeit v_{min} geben, unterhalb derer Geschwindigkeiten ausschließlich zum Term *niedrig* gehören und eine bestimmte obere Grenzgeschwindigkeit v_{max} , oberhalb derer sie ausschließlich zum Term *hoch* gehören. Es gilt also (v steht hier für den Wert des Attributs *Geschwindigkeit*):

$$\begin{aligned} \mu_{niedrig}(v) = 1 \quad \text{und} \quad \mu_{hoch}(v) = 0 \quad \text{für} \quad v \leq v_{min} \\ \mu_{niedrig}(v) = 0 \quad \text{und} \quad \mu_{hoch}(v) = 1 \quad \text{für} \quad v \geq v_{max} \end{aligned}$$

Geschwindigkeiten zwischen v_{min} und v_{max} sollen sowohl zum Term *niedrig* als auch zum Term *hoch* gehören. Die einfachste Möglichkeit für den Verlauf der Zugehörigkeitsfunktionen zwischen v_{min} und v_{max} ist ein linearer, also

$$\begin{aligned} \mu_{niedrig}(v) &= \frac{v_{max}-v}{v_{max}-v_{min}} \quad \text{und} \\ \mu_{hoch}(v) &= \frac{v-v_{min}}{v_{max}-v_{min}} \quad \text{für} \quad v_{min} < v < v_{max} \end{aligned}$$

²„hoch“ wird hier nicht im Sinne von überhöhter Geschwindigkeit, sondern im Sinne einer für die jeweilige Straßenkategorie normalen Geschwindigkeit verstanden, die gefahren wird, falls es auf der Strecke keine Behinderungen gibt.

4 Konzeption

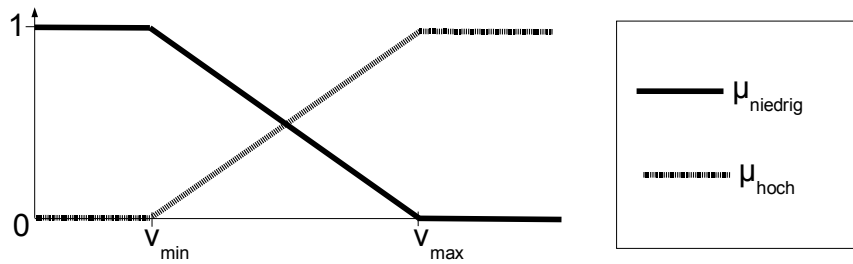


Abbildung 4.3: Zugehörigkeitsfunktionen der linguistischen Variable *Geschwindigkeit*

Graphisch dargestellt ergibt dies für $\mu_{niedrig}$ eine Strecke vom Punkt $(v_{min}, 1)$ zum Punkt $(v_{max}, 0)$ und für μ_{hoch} eine Strecke vom Punkt $(v_{min}, 0)$ zum Punkt $(v_{max}, 1)$. Eine Geschwindigkeit wird in diesem Zwischenbereich also um so mehr dem Term *niedrig* zugeordnet, je näher sie an v_{min} liegt und um so mehr dem Term *hoch*, je näher sie an v_{max} liegt. Außerdem ist somit gegeben, dass die Summe vom v_{min} und v_{max} eins ergibt.

Standardabweichung der Geschwindigkeit

Nach [SW03] ist die Standardabweichung der Geschwindigkeit nur für den Verkehrszustand auf Autobahnen und Landstraßen aussagekräftig. Auf Stadtstraßen kommt es aufgrund von Kreuzungen, Ampeln, auf der Fahrbahn parkenden Fahrzeugen, etc. immer wieder zu Veränderungen der Geschwindigkeit, so dass deren Standardabweichung dort immer recht hoch ist.

Es werden also für die Standardabweichung nur zwei linguistische Variablen *Standardabweichung Autobahn* und *Standardabweichung Land* definiert. Wie die linguistischen Variablen für die Durchschnittsgeschwindigkeit erhalten auch sie je zwei Terme *niedrig* und *hoch*. Die Zugehörigkeitsfunktionen werden analog zu denen für die Durchschnittsgeschwindigkeit festgelegt; auch hier soll es jeweils eine untere Grenze σ_{min} und eine obere Grenze σ_{max} geben, unter- bzw. oberhalb derer die Werte lediglich zum Term *niedrig* bzw. *hoch* gehören. Der Bereich dazwischen wird wieder jeweils durch eine Strecke beschrieben.

Relativer Stillstandsanteil

An Stelle der Standardabweichung wird für Stadtstraßen der relative Stillstandsanteil zur Bestimmung des Verkehrszustandes herangezogen. Es ist hier also nur eine linguistische Variable *Stillstandsanteil Stadt* nötig. Auch für diese soll es wieder zwei Terme *niedrig* und *hoch* geben. Der Term *niedrig* soll Stillstandsanteile beschreiben, die durch Standzeiten an Ampeln, Zebrastreifen, Verkehrshindernissen, etc. verursacht werden, während der Term *hoch* Stillstandsanteile beschreibt, die auf Stau hindeuten. Die Zugehörigkeitsfunktionen werden wieder nach dem bekannten Muster anhand zweier Grenzwerte s_{min} und s_{max} gebildet.

Zeit

Für das Attribut *Zeit* sind mehrere parallele Hierarchien sinnvoll. Zum einen eine scharfe Hierarchie mit einer Einteilung in Stunden, Tage, Monate und Jahre. Auf der untersten Ebene gehören zwei Tupel zur selben Äquivalenzklasse, falls sie im selben Jahr im selben

Monat am selben Tag in der selben Stunde erfasst wurden. Die nächsthöhere Ebene fasst alle Stunden eines Tages zu Äquivalenzklassen zusammen, so dass alle am selben Tag erfassten Tupel in einer Klasse liegen. Die wieder nächsthöhere unterscheidet nur noch nach Monaten und auf der höchsten Ebene gehören alle im selben Jahr erfassten Tupel zu einer Äquivalenzklasse. Diese Einteilung ist für Auswertungen und Statistiken geeignet, man könnte z.B. die Entwicklung der Anzahl an Fahrzeugen, die eine bestimmte Strecke befahren, über die Jahre hinweg verfolgen, oder untersuchen, welchen Einfluss der Monat (Wintermonate ggü. Sommermonaten) auf die gefahrene Durchschnittsgeschwindigkeit hat.

Parallel dazu ist eine weitere Hierarchie sinnvoll. Man kann feststellen, dass sich der Verkehr an Tagen, die auf den selben Wochentag fallen, ähnlich verhält. Daher ist es angebracht, einen Kontext zu definieren, dessen Äquivalenzklassen aus allen Tupeln bestehen, die am selben Wochentag und in der selben Stunde (des jeweiligen Tages) erfasst wurden. Eine Ebene höher in dieser Hierarchie lässt sich ein Kontext ansiedeln, dessen Äquivalenzklassen nicht mehr nach Stunden, sondern nur noch nach Wochentagen unterscheiden. Als unterste Ebene dieser Hierarchie lässt sich die gleiche Stundeneinteilung verwenden wie bei der gerade eben definierten Hierarchie (siehe Abbildung 4.4).

Die feinste Einteilung, die in beiden Hierarchien gleich ist, – also die auf Stundenebene – kann auch benutzt werden, um den Verkehrszustand festzustellen. Dieser würde dann aus allen in einer bestimmten Stunde angefallenen Tupeln berechnet werden. Eine gröbere Einteilung ist für die Bestimmung des Verkehrszustandes nicht sinnvoll, da dieser sich relativ schnell ändern kann. Eine feinere Einteilung dagegen würde für die Bestimmung des Verkehrszustandes schon Sinn machen – vorausgesetzt es sind genügend Daten vorhanden. Wird eine Strecke pro Stunde nur von drei FCD-Fahrzeugen befahren, macht es keinen Sinn, den Verkehrszustand minutengenau feststellen zu wollen, da für die meisten Minuten keine Daten vorhanden sind. Je mehr FCD-Fahrzeuge hingegen Daten für die Bestimmung des Verkehrszustandes beisteuern, desto verlässlicher kann dieser bestimmt werden, da so eventuelle Ausreißer ausgeglichen werden, die von Fahrzeugen mit atypischem Fahrverhalten stammen.

Leider existieren für die vorliegenden Messdaten zahlreiche Links, die nur an wenigen Tagen überhaupt von einem, an den meisten Tagen jedoch von keinem FCD-Fahrzeug befahren wurden. Hier macht eine Einteilung für die Bestimmung des Verkehrszustandes keinen Sinn – der Zustand kann eben nur für den bestimmten Tag und die bestimmte Zeit, zu der das Fahrzeug den Link befahren hat, aus diesem einen von dem Fahrzeug übermittelten Tupel bestimmt werden. Einige Links wurden jedoch an bestimmten Tagen über mehrere Stunden hinweg von zahlreichen FCD-Fahrzeugen befahren. Hier sind genügend Daten vorhanden, um den Verkehrszustand auf der Basis einer Einteilung nach Stunden zu bestimmen. Wären die Daten zeitlich noch dichter gesät, wäre auch eine noch feinere zeitliche Einteilung denkbar, etwa in Viertelstunden- oder Fünfminutenintervalle. Eine solche wird in Abschnitt 4.2.2 für die Simulationsdaten vorgenommen, die pro Zeitintervall in größeren Mengen anfallen als die real gemessenen Floating Car Daten.

Weiter ist eine Einteilung zur Bestimmung des momentanen Verkehrszustandes angebracht. Hierzu wird der Wertebereich des Attributs *Zeit* eingeteilt in eine Äquivalenzklasse *jetzt*, die die Zeitspanne von einem gewissen Zeitpunkt in der nahen Vergangenheit bis zur aktuellen Zeit umfasst und in eine Äquivalenzklasse *Rest*, die den restlichen Wertebereich umfasst. So kann der momentane Verkehrszustand anhand der Daten bestimmt werden, die im Zeitintervall *jetzt* gesammelt wurden. Auch hier stellt sich wieder die Frage, wie groß dieses Zeitintervall sein soll. Aus den selben Gründen wie oben ist ein Intervall von einer

4 Konzeption

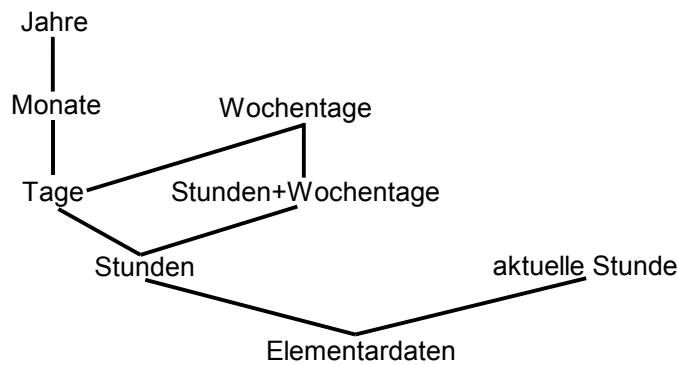


Abbildung 4.4: Anordnung der Kontexte des Attributs *Zeit* in parallelen Hierarchien

Stunde sinnvoll. Um 23:42h würde die Äquivalenzklasse *jetzt* beispielsweise die Zeitspanne von 22:43h bis 23:42h umfassen.

Fahrzeugtyp

Es wird zwischen fünf verschiedenen Fahrzeugtypen unterschieden: LKW, Taxi, PKW Testfahrt, PKW allgemein und unbekannter Fahrzeugtyp. Durch scharfe Kontexte kann man hier eine etwas gröbere Einteilung festlegen, in der nur noch zwischen PKW, LKW und unbekanntem Fahrzeugtyp unterschieden wird. Taxi, PKW Testfahrt und PKW allgemein gelten dann also nach dieser Einteilung als äquivalent.

4.1.3 Klassenbildung

Nun wird die Bildung von Klassen mit Hilfe der Kontexte betrachtet. Bei Attributen, für die nicht explizit angegeben ist, welcher Kontext zur Klassenbildung verwendet wird, soll hier wie auch in den Abschnitten über Klassenbildung für die Simulations- und Güterverkehrsdaten immer der aus nur einer Äquivalenzklasse bestehende Kontext angenommen werden; der Wert des betreffenden Attributs ist in diesem Fall irrelevant für die Klassenzugehörigkeit.

Klassen für Verkehrszustände

Aus den im vorigen Abschnitt definierten Kontexten und linguistischen Variablen können Klassen gebildet werden, mit Hilfe derer sich der Verkehrszustand bestimmen lässt.

Für Autobahnen bildet man Klassen aus den Termen der linguistischen Variablen *Geschwindigkeit Autobahn* und *Standardabweichung Autobahn*. Da jede der Variablen zwei Terme besitzt, entstehen so vier Klassen. Eine hohe Durchschnittsgeschwindigkeit, von der wenig abgewichen wird, bedeutet freien Verkehr. Die entsprechende Klasse aus den Termen *Geschwindigkeit Autobahn hoch* und *Standardabweichung Autobahn niedrig* erhält daher die Bezeichnung *Autobahn freier Verkehr*. Eine hohe Durchschnittsgeschwindigkeit bei gleichzeitig hoher Standardabweichung bedeutet lebhafteren Verkehr, die entsprechende Klasse heißt daher *Autobahn lebhafter Verkehr*. Sind Durchschnittsgeschwindigkeit und Standardabweichung niedrig, so deutet das auf Stau hin (Klasse *Autobahn Stau*). Niedrige

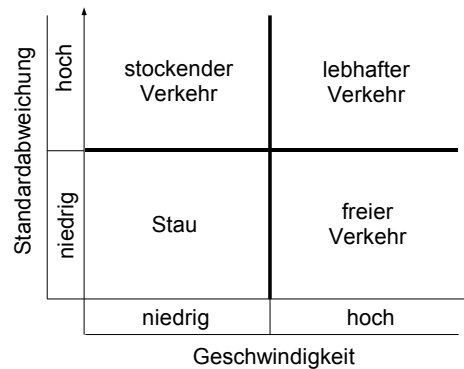


Abbildung 4.5: Verkehrszustände auf Autobahnen, definiert durch Kontexte der Attribute *Durchschnittsgeschwindigkeit* und *Standardabweichung*

Durchschnittsgeschwindigkeit und hohe Standardabweichung bedeuten stockenden Verkehr (Klasse *Autobahn stockender Verkehr*).

Mit einem der in Abschnitt 2.3.2 definierten Operatoren lässt sich aus den Zugehörigkeitsfunktionen der einzelnen Terme die Zugehörigkeit zu den Klassen berechnen. Da die Verkehrszustände eine Einteilung darstellen sollen, wie sie auch ein menschlicher Verkehrsteilnehmer vornehmen würde, wird das kompensatorische Und als Operator verwendet.

Für Landstraßen erfolgt die Klassenbildung ebenso wie bei den Autobahnen: Aus den Termen der entsprechenden linguistischen Variablen für die Attribute *Durchschnittsgeschwindigkeit* und *Standardabweichung* werden vier Klassen für freien, lebhaften und stockenden Verkehr sowie Stau gebildet.

Für Stadtstraßen werden die Klassen aus den Termen der linguistischen Variablen *Geschwindigkeit Stadt* und *Stillstandsanteil Stadt* gebildet. Hohe Durchschnittsgeschwindigkeit und geringer Stillstandsanteil bedeuten freien Verkehr (*Stadt freier Verkehr*), niedrige Durchschnittsgeschwindigkeit und hoher Stillstandsanteil bedeuten Stau (*Stadt Stau*). Niedrige Geschwindigkeit und niedriger Stillstandsanteil deuten auf zähfließenden Verkehr hin (*Stadt zähfließender Verkehr*). Hohe Durchschnittsgeschwindigkeit in Kombination mit hohem Stillstandsanteil kommt so gut wie nicht vor; die entsprechende Klasse existiert nur der Vollständigkeit halber.

Bestimmen des Verkehrszustandes aus mehreren Tupeln

Durch Klassifikation nach den im vorigen Abschnitt definierten Klassen kann man den Verkehrszustand anhand eines einzelnen Tupels bestimmen. Verlässlicher lässt sich der Verkehrszustand bestimmen, wenn man mehrere Tupel als Grundlage nimmt. Man wählt hierzu die Tupel aus, die für einen Link in einem bestimmten Zeitintervall angefallen sind.

Diese Auswahl lässt sich ebenfalls durch Klassenbildung beschreiben: Man benutzt hierfür die Äquivalenzklassen des Attributs *Zeit* auf Stundenebene. Zusätzlich definiert man einen Kontext für das Attribut *LinkId*, welches einen Link eindeutig identifiziert. Die Äquivalenzklassen dieses Kontexts bestehen jeweils nur aus einem einzigen Wert. Durch Kreuzproduktbildung erhält man aus den Äquivalenzklassen dieser beiden Kontexte Klassen, die jeweils diejenigen Tupel enthalten, welche für den selben Link in der selben Stunde erfasst wurden. Veranschaulicht wird dies in Abbildung 4.6. Aus den Tupeln einer dieser Klassen

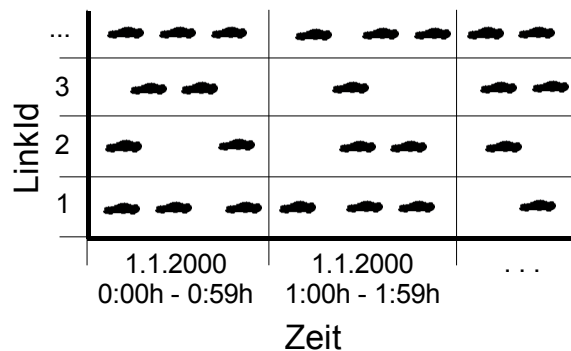


Abbildung 4.6: Einteilung der Linkbefahrungen in Klassen nach Zeit und LinkId

lässt sich nun wie in Abschnitt 2.4.6 beschrieben der Verkehrszustand des jeweiligen Links für die jeweilige Stunde bestimmen.

Will man den momentanen Verkehrszustand bestimmen, verwendet man für das Attribut *Zeit* den Kontext mit den Äquivalenzklassen *jetzt* und *Rest*. Man betrachtet jedoch nur die Kreuzprodukte aus der Äquivalenzklasse *jetzt* und den Äquivalenzklassen des Attributs *LinkId* – aus den in diesen Klassen enthaltenen Tupeln kann der momentane Verkehrszustand für die einzelnen Links bestimmt werden.

Weitere Klasseneinteilungen

Mit den in Abschnitt 4.1.2 definierten Kontexten lassen sich weitere Klasseneinteilungen vornehmen, die zur Auswertung der Floating Car Daten verwendet werden können. Z.B. könnte man aus Kontexten für *LinkId*, *Zeit* und *Fahrzeugtyp* Klassen bilden und für diese die Zahl der Linkbefahrungen ermitteln. Welche Aggregatfunktionen noch möglich sind, ist Thema des nächsten Abschnitts.

4.1.4 Aggregation

Nun soll noch geklärt werden, für welche Attribute über welche Klassen welche Aggregatfunktionen verwendet werden können. Zunächst sei noch einmal erwähnt, dass alle Klasseneinteilungen, die durch Kontexte oder linguistische Variablen definiert werden, die Bedingungen „Vollständigkeit“ und „Überlappungsfreiheit“ erfüllen. Da außerdem keine Attribute vom Typ „stock“ vorhanden sind, die in der Zeitdimension nicht summiert werden dürften, schränkt die Auswahl der Klasseneinteilung die Möglichkeiten der Aggregation nicht ein.³

Die Attribute für Durchschnittsgeschwindigkeit, minimale und maximale Geschwindigkeit, Standardabweichung der Geschwindigkeit sowie durchschnittliche Beschleunigung sind vom Typ „value per unit“ und bereits aus einzelnen Messungen für einen Link aggregiert.⁴ Da Durchschnitt und Standardabweichung keine semiadditiven Funktionen sind, können die entsprechenden Attribute nicht mit den selben Funktionen weiter aggregiert werden. Es

³Abgesehen davon, dass für unscharfe Einteilungen keine Minima und Maxima berechnet werden können.

⁴Die Durchschnittsgeschwindigkeit wird wohl am einfachsten aus der zurückgelegten Strecke und der benötigten Zeit berechnet, doch auch hier kann man sich eine Durchschnittsbildung aus vielen Einzelmessungen vorstellen, die zum selben Ergebnis führen würde.

kann z.B. nicht die Durchschnittsgeschwindigkeit für eine aus mehreren Links bestehende Gesamtstrecke berechnet werden.

Jedoch kann aus den Durchschnittswerten mehrerer Befahrungen ein und des selben Links durch unterschiedliche Fahrzeuge der Gesamtdurchschnitt für den Link gebildet werden. In diesem Fall führt die zweifache Durchschnittsbildung zu aussagekräftigen Ergebnissen, weil die Durchschnittswerte sich alle auf den selben Link beziehen. Dies ist kein Widerspruch dazu, dass der Durchschnitt keine semiadditive Funktion ist. Der durch die doppelte Durchschnittsbildung berechnete Wert ist nicht unbedingt gleich dem Durchschnitt aller Einzelmessungen. Im hier vorliegenden Fall ist aber der Durchschnitt aller Einzelmessungen gar nicht aussagekräftig; man stelle sich nur ein Fahrzeug vor, dessen Durchschnittsgeschwindigkeit anhand von zehn und ein anderes, dessen Durchschnittsgeschwindigkeit anhand von zwanzig Einzelmessungen ermittelt wurde.⁵

Auch eine weitere Aggregation mit anderen Aggregatfunktionen ist möglich. Beispielsweise kann das Maximum der Durchschnittsgeschwindigkeit oder der Durchschnitt der Standardabweichung für einen Link gebildet werden. Die Bildung der Summe führt jedoch für Attribute vom Typ „value per unit“ zu keinem sinnvollen Ergebnis. Maximum und Minimum sind semiadditiv, daher können aus den Attributen für die minimale und maximale Geschwindigkeit weitere Maxima bzw. Minima für gröbere Einteilungen berechnet werden.

Der relative Stillstandsanteil ist ebenfalls vom Typ „value per unit“ und darf daher nicht summiert werden. Für alle anderen Attribute (LinkId, Zeit, etc.) ist lediglich die Aggregatfunktion „Anzahl“ sinnvoll.

Eine Übersicht über die Typen der einzelnen Attribute findet sich, auch für die später behandelten Messstations- und Güterverkehrsdaten, in Anhang A.2.

4.1.5 Trenngeraden

Zuletzt soll nun noch eine Einteilung betrachtet werden, die nicht nur durch horizontale und vertikale, sondern durch beliebig verlaufende Geraden entsteht. Diese wird hier lediglich für die Straßenkategorie „Autobahn“ konzipiert, wäre aber auch für die anderen Straßenkategorien möglich.

[SW03] verwendet für die Trennung zweier Verkehrszustände jeweils zwei Geraden. Welche der beiden betrachtet wird, hängt vom aktuellen Verkehrszustand des jeweiligen Links ab. So ergibt sich ein Hystereseverhalten; der Bereich zwischen den Geraden wird als Hysteresebereich bezeichnet. Tupel, die beispielsweise im Hysteresebereich zwischen den Zuständen *Stau* und *zähfließender Verkehr* liegen, werden als *Stau* klassifiziert, falls der aktuelle Zustand der Strecke *Stau* ist und als *zähfließender Verkehr*, falls der aktuelle Zustand der Strecke *zähfließender Verkehr* ist. So soll den Vorgängen der Staubildung und Stauauflösung Rechnung getragen werden.

Ein solches Hystereseverhalten, bei dem die Grenze zwischen zwei Klassen vom aktuellen Verkehrszustand des betrachteten Links abhängt, kann durch Kontexte nur schwer nachgebildet werden, da hierzu abhängig vom aktuellen Verkehrszustand jeweils andere Kontexte zur Klassifikation ausgewählt und natürlich vorher auch für jeden Verkehrszustand unterschiedlich definiert werden müssten. Deswegen werden anstatt der Hysteresebereiche unscharfe Übergänge zwischen den Verkehrszuständen verwendet, die leicht über unscharfe Kontexte definiert werden können. Ein Tupel, das im Übergangsbereich liegt, wird also

⁵ Abgesehen davon kann der Durchschnitt der Einzelmessungen mehrerer Fahrzeuge nicht berechnet werden, da die Fahrzeuge bereits aggregierte Werte an die Messstationen senden.

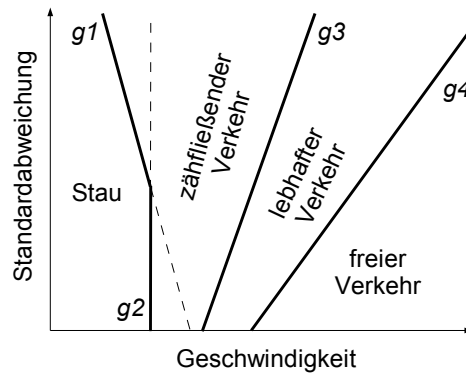


Abbildung 4.7: Verkehrszustände auf Autobahnen, definiert durch Trenngeraden

nicht abhängig vom aktuellen Verkehrszustand dem einen oder dem anderen Verkehrszustand zugeordnet, sondern unabhängig vom aktuellen Verkehrszustand jeweils zu einem gewissen Grad beiden Verkehrszuständen.

Angelehnt an die in [SW03] vorgenommene Klasseneinteilung erfolgt die Trennung zwischen den unterschiedlichen Verkehrszuständen durch vier Geraden g_1 bis g_4 (Abbildung 4.7). Anders als dort, wo wegen der Hysteresebereiche je zwei Geraden zur Trennung zweier Zustände verwendet werden, werden die Zustände hier durch nur eine Gerade getrennt, die die Mitte des Übergangsbereichs kennzeichnen soll. Dadurch ergeben sich die vier Verkehrszustände *freier Verkehr*, *lebhafter Verkehr*, *Stau* und *zähfließender Verkehr*.⁶

Damit die Tupel klassifiziert werden können, erhalten sie für jede Trenngerade g_i ein zusätzliches Attribut *Abstand_* g_i , das den Abstand zur Trenngeraden angibt. Dieser lässt sich aus den Koordinaten des Tupels (das sind Geschwindigkeit und Standardabweichung) und der Geradengleichung berechnen. Da wesentlich ist, auf welcher Seite der Geraden sich ein Tupel befindet, werden die Abstände für Tupel auf der rechten Seite mit positivem, für Tupel auf der linken Seite mit negativem Vorzeichen versehen.⁷

Für jedes dieser zusätzlichen Attribute benötigt man nun eine linguistische Variable *Seite*, jeweils mit den Termen *links* und *rechts*. Ist der Abstand zur Geraden g_i größer als ein Mindestabstand d_i , so soll die Zugehörigkeit zu den Termen null bzw. eins sein. Für den Übergangsbereich dazwischen, der die Breite $2d$ hat, sollen die Zugehörigkeitsfunktionen wie schon diejenigen in Abschnitt 4.1.2 linearen Verlauf haben. Es gilt also (das Attribut *Abstand_* g_i ist hier durch a abgekürzt):

$$\begin{aligned} \mu_{links}(a) &= 1 & \text{und} & & \mu_{rechts}(a) &= 0 & \text{für} & & a \leq -d_i \\ \mu_{links}(a) &= 0 & \text{und} & & \mu_{rechts}(a) &= 1 & \text{für} & & a \geq d_i \\ \mu_{links}(a) &= \frac{d_i - a}{2d_i} & \text{und} & & \mu_{rechts}(a) &= \frac{d_i + a}{2d_i} & \text{für} & & -d_i < a < d_i \end{aligned}$$

Mit Hilfe dieser linguistischen Variablen lassen sich nun Klassen definieren, die die einzelnen Verkehrszustände beschreiben. Für den Verkehrszustand *freier Verkehr* müssen alle

⁶Die Bezeichnungen der Verkehrszustände sind hier zum Teil identisch mit denjenigen für die in Abschnitt 4.1.3 definierten Verkehrszustände. Natürlich handelt es sich hier jedoch um zwei verschiedene Einteilungen.

⁷Für eher flach verlaufende oder gar horizontale Geraden erfolgt die Unterscheidung zweckmäßigerweise nicht zwischen linker und rechter, sondern zwischen oberer und unterer Seite. Solche Geraden kommen jedoch hier nicht vor.

linguistischen Variablen den Wert *rechts* haben, für den Verkehrszustand *Stau* müssen alle den Wert *links* haben. Für den Zustand *lebhafter Verkehr* muss die zu g_4 gehörende Variable den Wert *links* besitzen, während alle anderen Variablen den Wert *rechts* haben.

Der Zustand *zähfließender Verkehr* wird durch mehrere Klassen beschrieben. Er beschreibt alle Tupel, die links von g_3 und g_4 und rechts von g_1 **oder** g_2 liegen. Da Klassen jedoch nur durch konjunktive Verknüpfung von Termen gebildet werden können, existiert keine Klasse, die alle diese Tupel umfasst. Der Zustand muss daher durch drei Klassen beschrieben werden: Durch die der Tupel links von g_3 und g_4 und rechts von g_1 und g_2 , durch die der Tupel links von g_2 , g_3 und g_4 und rechts von g_1 , sowie durch die der Tupel links von g_1 , g_3 und g_4 und rechts von g_2 . Zudem ergeben sich noch zahlreiche Klassen, in die keine Tupel fallen.⁸ Es existieren beispielsweise keine Tupel, die links von g_3 und gleichzeitig rechts von g_4 liegen. Die Klassenzugehörigkeit wird wie im vorigen Abschnitt auch durch Verknüpfung der Zugehörigkeiten zu den einzelnen Termen mit dem kompensatorischen Und berechnet.

4.2 Simulationsdaten für Verkehrsmessstationen

Ähnlich wie bei den Floating Car Daten sollen auch für die nun betrachteten Simulationsdaten für Verkehrsmessstationen Kontexte zur Bestimmung des Verkehrszustandes definiert werden. Daneben werden auch hier wieder Kontexte zur strukturierten Auswertung der Daten konzipiert.

4.2.1 Ermitteln des Verkehrszustandes

Wie in Abschnitt 3.3.4 festgestellt wurde, bestehen gewisse Ähnlichkeiten zwischen den Floating Car Daten und den Simulationsdaten für Verkehrsmessstationen. Es werden teilweise die selben Größen gemessen, jedoch beziehen sich diese bei den Floating Car Daten auf Strecken, bei den Daten der Verkehrsmessstationen auf Punkte.

Für den idealisierten Fall, dass am Anfang und am Ende jeder Strecke für jede Fahrspur eine Messstation existiert, lassen sich die auf Punkte bezogenen Daten in auf Strecken bezogene Daten umrechnen. In diesem Fall können die Durchschnittsgeschwindigkeit und der Stillstandsanteil eines Fahrzeugs für eine Strecke berechnet werden. Dies sind genau die Werte, die bei den Floating Car Daten zur Bestimmung des Verkehrszustandes auf Stadtstraßen verwendet wurden. Für die umgerechneten Messstationsdaten könnten also die selben Kontexte, wie für die Floating Car Daten verwendet und so Verkehrszustände auf Stadtstraßen bestimmt werden. Für Landstraßen und Autobahnen funktioniert diese Vorgehensweise nicht, da die hier benötigte Standardabweichung der Geschwindigkeit nicht anhand von Messstationen am Anfang und Ende der Strecke bestimmt werden kann.

Im allgemeinen Fall sind Messstationen nicht an Anfang und Ende jeder Strecke vorhanden. Daher kann der Verkehrszustand hier nicht für Strecken, sondern nur für die einzelnen Punkte bestimmt werden, an denen die Stationen positioniert sind. Einziger Anhaltspunkt ist die Geschwindigkeit der vorüber fahrenden Fahrzeuge; Stillstandsanteil bzw. Standardabweichung der Geschwindigkeit lassen sich für ein Fahrzeug auf einen einzelnen Punkt bezogen nicht angeben.

⁸Zumindest nicht, wenn die Übergangsbereiche nicht zu groß gewählt werden.

4.2.2 Kontexte

Geschwindigkeit, Stillstandsanteil

Ist für Stadtstraßen der ideale Fall mit Messstationen am Anfang und Ende jeder Strecke gegeben, können die entsprechenden Kontexte der Attribute *Geschwindigkeit* und *Stillstandsanteil* direkt von den Floating Car Daten übernommen werden.

Für den allgemeinen Fall können lediglich Kontexte für das Attribut *Geschwindigkeit* definiert werden. Auch diese sollen prinzipiell von den Floating Car Daten übernommen werden. Jedoch muss beachtet werden, dass die Messstationen immer auf einzelnen Fahrspuren positioniert sind. Während sich auf Stadtstraßen die Geschwindigkeiten, die bei mehrspurigen Straßen auf den verschiedenen Fahrspuren gefahren werden, im Allgemeinen nicht wesentlich unterscheiden, werden auf Autobahnen und mehrspurigen Landstraßen auf weiter links liegenden Spuren meistens höhere Geschwindigkeiten gefahren als auf rechts liegenden Spuren. Daher wird für diese Straßenkategorien eine eigene linguistische Variable für jede Spur mit jeweils charakteristischen Werten für v_{min} und v_{max} vorgesehen.

Zeit

Die Kontexte für die Zeit können zum Teil ebenfalls von den Floating Car Daten übernommen werden. Allerdings werden bei den Simulationsdaten pro Zeiteinheit viel mehr Daten erfasst als bei den Floating Car Daten, wo längst nicht jedes vorüber fahrende Fahrzeug ein FCD-Fahrzeug ist. Daher können der ersten in Abschnitt 4.1.2 für das Attribut *Zeit* definierten Hierarchie weitere, feinere Stufen hinzugefügt werden. Es sind genug Daten vorhanden, dass eine Einteilung in Viertelstunden und eine Ebene tiefer in Fünfminutenintervalle sinnvoll ist.

Werden diese kürzeren Intervalle zur Bestimmung des Verkehrszustandes benutzt, so lassen sich Veränderungen des selbigen zeitlich genauer zuordnen. Dafür wird der ermittelte Verkehrszustand aber auch unsicherer, da er anhand einer geringeren Anzahl von Tupeln bestimmt wird als bei grobkörnigeren Zeitintervallen. Allerdings stehen selbst bei der Verwendung von Fünfminutenintervallen mehr Tupel zur Verfügung, als für die meisten FCD-Links innerhalb einer Stunde anfallen.

Auch das Intervall, das durch die Äquivalenzklasse *jetzt* beschrieben wird, kann für die Simulationsdaten kürzer gewählt werden, falls der ermittelte momentane Verkehrszustand aktueller sein soll. So können z.B. neue Staus schneller erkannt werden, da nur noch Daten betrachtet werden, die wenige Minuten alt sind.

Allerdings ist das Zeitintervall, für welches das Verkehrsgeschehen typischerweise simuliert wird, recht kurz im Vergleich zu dem Zeitraum, innerhalb dessen die vorliegenden Floating Car Daten gesammelt wurden. Deswegen werden für die simulierten Daten keine Kontexte auf Tages-, Monats- und Jahresebene benötigt. Auch auf die Kontexte, welche Äquivalenzklassen für am selben Wochentag gesammelte Daten besitzen, kann verzichtet werden.

Fahrzeugtyp

Der Fahrzeugtyp ist sehr detailliert angegeben. Es werden z.B. mehrere Arten von Bussen und Straßenbahnen unterschieden. Für Analysezwecke ist daher eine Einteilung in gröbere Fahrzeuggruppen angebracht, bei der nur noch zwischen PKW, LKW, Bus, Zweirad, Bahn und Fußgänger unterschieden wird.

Nicht alle Fahrzeugtypen sind für die Bestimmung des Verkehrszustandes relevant. Eine Ebene höher kann daher eine noch gröbere Einteilung in relevante und nicht relevante Typen angesiedelt werden. Dabei werden PKW, LKW, Bus und Zweirad zur Äquivalenzklasse der relevanten Typen, Bahn und Fußgänger zur Äquivalenzklasse der nicht relevanten Typen zusammengefasst. Beide Einteilungen sind natürlich scharf.

4.2.3 Klassenbildung

Die Klassen für die Verkehrszustände werden bei den auf Strecken umgerechneten Daten gebildet wie bei den Floating Car Daten. Für die auf Punkte bezogenen Daten ergeben sich für jede Straßenkategorie und jede Spur lediglich zwei Verkehrszustände. Sie sollen mit *hohe Geschwindigkeit* und *niedrige Geschwindigkeit* bezeichnet werden. Da jeweils nur eine linguistische Variable betrachtet wird, entsprechen die Klassenzugehörigkeiten eines Tupels hier den Zugehörigkeiten des Geschwindigkeitswerts zu den Termen *hoch* und *niedrig*. Die Klassen für sonstige Auswertungen können wieder wie bei den Floating Car Daten gebildet werden.

4.2.4 Aggregation

Wie schon bei den Floating Car Daten soll nun noch untersucht werden, für welche Attribute mit welchen Aggregatfunktionen Aggregate gebildet werden können.

Die Geschwindigkeit und die Beschleunigung sowie die Länge eines Fahrzeugs sind vom Typ „value per unit“. Außer der Summe dürfen hier alle Aggregatfunktionen verwendet werden. Die Anzahl der Personen im Fahrzeug ist ein Attribut vom Typ „stock“ und darf daher nicht über die Zeit summiert werden.

Bei der Zeit, die ein Fahrzeug schon im Stau gestanden hat, handelt es sich, wenn man ein bestimmtes Fahrzeug betrachtet, um eine laufende Summe. Man kann diesen Wert als Aggregat aus den einzelnen Standzeiten des Fahrzeugs vom Beginn der Simulation bis zur aktuellen Messung betrachten. Allerdings erfolgt die Aggregation nicht überlappungsfrei: eine einmal aufgetretene Standzeit wird für alle nachfolgenden Messungen jedes Mal wieder mitsummiert. Aus den Werten dieses Attributs können daher keine weiteren Aggregate berechnet werden.

4.3 Güterverkehrsdaten

Bei den Güterverkehrsdaten handelt es sich um Daten, die bereits aus detaillierteren Daten aggregiert wurden. Durch die Definition von Kontexten lassen sich grobkörnigere Einteilungen vorgeben, für welche die Daten weiter aggregiert werden können. Wie schon bei den Floating Car Daten muss auch hier bei der Aggregation berücksichtigt werden, welche Aggregatfunktion auf die Daten bereits angewendet wurde.

4.3.1 Kontexte

Kontexte müssen für diejenigen Attribute definiert werden, welche bei der Aggregation der Güterverkehrsdaten zur Gruppierung verwendet wurden.

4 Konzeption

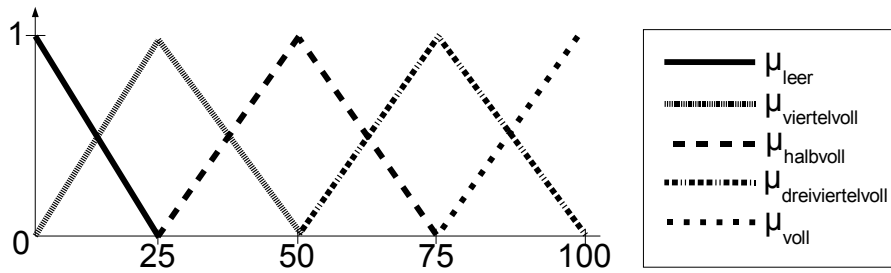


Abbildung 4.8: Zugehörigkeitsfunktionen für den genutzten Rauminhalt, feinere Einteilung (Der Übersichtlichkeit halber sind die Funktionen nur in den Bereichen dargestellt, in denen sie von null verschiedene Werte annehmen.)

Art des Fahrzeugs und des Anhängers

Bei der Art des Fahrzeugs wird zwischen normalen LKW und Sattelzugmaschinen unterschieden. Da nur zwei Attributwerte möglich sind, macht es wenig Sinn, hier Kontexte zu definieren. Sowohl für das Fahrzeug als auch für den Anhänger werden verschiedene Aufbauarten unterschieden. Hier gibt es aber keine sinnvolle gröbere Strukturierung, die durch Kontexte vorgenommen werden könnte. Einzig eine Unterteilung in Fahrzeuge mit und Fahrzeuge ohne Anhänger ist hier angebracht.

Be- und Entladeort

Be- und Entladeort sind ohnehin nur sehr grob angegeben. Für Orte in Deutschland ist nur das Bundesland, für Orte im Ausland nur das Land bekannt. Durch Kontexte kann man eine noch gröbere Einteilung in Orte im Inland und Orte im Ausland vornehmen.

Genutzter Rauminhalt

Der genutzte Rauminhalt ist angegeben als ein Wert zwischen 0 (leer) und 100 (voll). Da es sich um einen sehr ungenauen Wert handelt, ist eine ziemlich grobe, unscharfe Einteilung angebracht. Hierzu wird eine linguistische Variable mit den Termen *leer*, *viertelvoll*, *halbvoll*, *dreiviertelvoll* und *voll* eingeführt. Da die Terme jeweils nur für einen einzigen Wert voll zutreffen (*leer* bei 0, *viertelvoll* bei 25, *halbvoll* bei 50, usw.), haben die Zugehörigkeitsfunktionen Dreiecksform, wie in Abbildung 4.8 dargestellt.

Eine noch gröbere Einteilung soll durch eine linguistische Variable mit den zwei Termen *eher leer* und *eher voll* vorgegeben werden. Diese linguistische Variable soll sich in der Hierarchie eine Stufe über der eben definierten befinden. Die Zugehörigkeitsfunktionen sollen wie folgt aussehen:

$$\begin{aligned}
 \mu_{eher\ leer}(leer) &= \mu_{eher\ leer}(viertelvoll) &= 1 \\
 \mu_{eher\ leer}(halbvoll) &= 0,5 \\
 \mu_{eher\ leer}(voll) &= \mu_{eher\ leer}(dreiviertelvoll) &= 0 \\
 \mu_{eher\ voll}(dreiviertelvoll) &= \mu_{eher\ voll}(voll) &= 1 \\
 \mu_{eher\ voll}(halbvoll) &= 0,5 \\
 \mu_{eher\ voll}(viertelvoll) &= \mu_{eher\ voll}(leer) &= 0
 \end{aligned}$$

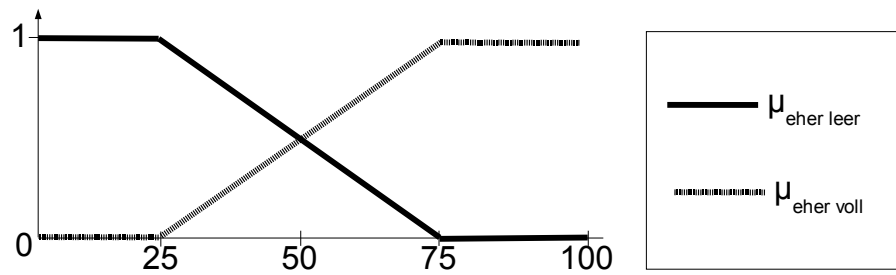


Abbildung 4.9: Zugehörigkeitsfunktionen für den genutzten Rauminhalt, gröbere Einteilung

Wie in Abschnitt 2.3.4 beschrieben, lassen sich aus diesen rekursiv definierten Zugehörigkeitsfunktionen die Zugehörigkeitsfunktionen in Bezug auf die Elementardaten berechnen. Diese sind in Abbildung 4.9 dargestellt.

Art der transportierten Güter

Für die Art der transportierten Güter sind durch die NST/R-Klassifikation, die bei der Erhebung der Daten verwendet wird, bereits zwei Grobeinteilungen unterschiedlicher Granularität vorgegeben. Die Güter, beschrieben durch eine Identifikationsnummer, gehören jeweils zu einer Güterhauptgruppe, die wiederum zu einer Güterabteilung gehört. Diese Einteilungen stellen eine Hierarchie scharfer Äquivalenzklassen dar und sind daher nichts anderes als Kontexte.

Containergröße

Bei der Containergröße werden vier verschiedene Größen unterschieden: Container à 20 Fuß, Container zwischen 20 und 30 Fuß, Container ab 30 bis unter 40 Fuß und Container von 40 Fuß und mehr. Außerdem gibt es Fahrzeuge, die nur einen und solche, die mehrere Container transportieren.

Hier lässt sich mit Hilfe eines Kontexts eine gröbere scharfe Einteilung in Fahrzeuge mit einem Container unter 30 Fuß, Fahrzeuge mit einem Container ab 30 Fuß und Fahrzeuge mit mehreren Containern vornehmen. Weiter wird hier eine Kategorie für Fahrzeuge benötigt, über deren Containergröße keine Angabe vorhanden ist.

4.3.2 Klassenbildung

Mit Hilfe der im vorigen Abschnitt definierten Kontexte lassen sich auch hier wieder Klassen bilden. Dabei ist es auch möglich, scharfe Einteilungen wie die für die Containergröße mit unscharfen Einteilungen wie der für den genutzten Rauminhalt zu kombinieren, da scharfe Kontexte als ein Spezialfall unscharfer Kontexte betrachtet werden können. Falls nur einer der zur Klassenbildung verwendeten Kontexte unscharf ist und alle anderen scharf sind, spielt es zudem keine Rolle, welcher Operator für die Berechnung der Klassenzugehörigkeit verwendet wird, da in diesem Fall alle Operatoren das selbe Ergebnis liefern.

4.3.3 Aggregation

Nun muss noch geklärt werden, welche Werte wie weiter aggregiert werden dürfen. Sämtliche zur Aggregation verwendeten Attribute (Entfernung, Anzahl Einzelstrecken, transportiertes Gewicht, zulässiges Gesamtgewicht und Nutzlast) sind vom Typ „flow“ und wurden mit der Aggregatfunktion „Summe“ aggregiert, welche eine semiadditive Aggregatfunktion ist. Sie dürfen also mit beliebigen Aggregatfunktionen in jeder beliebigen Dimension weiter aggregiert werden.

4.4 Zusammenfassung

In diesem Kapitel wurden Schemaerweiterungen durch Kontexte konzipiert. Für die Floating Car Daten und auch für die Simulationsdaten lassen sich diese Kontexte verwenden, um den Verkehrszustand zu bestimmen. Generell können Kontexte immer zur Auswertung der Daten verwendet werden. Hierzu werden Aggregate für die aus den Kontexten entstehenden Klassen berechnet. Es wurde untersucht, für welche Attribute sich Einschränkungen dafür ergeben, welche Aggregatfunktionen angewendet werden können. Auch Attribute, die bereits aggregierte Werte enthalten, wurden auf sich je nach angewendeter Aggregatfunktion ergebende Einschränkungen untersucht.

5 Umsetzung

Die im vorigen Kapitel konzipierten Schemaerweiterungen werden nun für die vorliegenden Daten umgesetzt. Dazu werden zunächst verschiedene Möglichkeiten vorgestellt, wie scharfe sowie unscharfe Kontexte repräsentiert werden können. In einem nächsten Schritt werden dann die für die jeweiligen Kontexte passenden Methoden ausgewählt und die konkrete Umsetzung wird erläutert. Weiter werden die Klassenbildung sowie die Berechnung von Aggregaten beschrieben.

Die in dieser Arbeit behandelten Daten liegen als Relationen in einem Oracle-System vor. Die Umsetzung wird daher für dieses System beschrieben. Hierbei werden die Anfragesprache SQL sowie deren Oracle-spezifische prozedurale Erweiterung PL/SQL verwendet. Die Umsetzung ließe sich leicht auf andere relationale Datenbanksysteme übertragen, sofern diese ähnliche Mechanismen zur Verfügung stellen.

Eine Übersicht über die für die einzelnen Attribute umgesetzten Kontexte findet sich in Anhang A.1.

5.1 Repräsentation von Kontexten

5.1.1 Durch zusätzliche Attribute in der Relation

Die wohl einfachste Möglichkeit, den Kontext eines Attributs A zu repräsentieren, besteht darin, der Relation einfach zusätzliche Attribute zuzuschlagen, deren Werte beschreiben, in welcher Äquivalenzklasse der jeweilige Wert von A liegt.

Um einen scharfen Kontext zu repräsentieren, ist lediglich ein zusätzliches Attribut erforderlich. Es erhält einen Wert, der die Äquivalenzklasse eindeutig identifiziert. Dieses Attribut kann dann zur Selektion oder zur Gruppierung benutzt werden.

Zur Repräsentation eines unscharfen Kontextes ist für jeden Term der zugehörigen linguistischen Variable ein zusätzliches Attribut erforderlich, welches als Wert die Zugehörigkeit des Wertes von A zu dem Term enthält.

Es wird ein Mechanismus benötigt, der die zusätzlichen, den Kontext beschreibenden Attribute abhängig vom Wert, den A hat, mit den jeweils richtigen Werten belegt. Dies muss beim Hinzufügen neuer Tupel sowie beim Ändern vorhandener Tupel geschehen. Die eigentliche Information, wie genau der Kontext aussieht, muss also in diesem Mechanismus enthalten sein. Umsetzen ließe sich ein solcher Mechanismus beispielsweise durch einen Trigger.

Der Nachteil einer Umsetzung von Kontexten durch zusätzliche Attribute besteht darin, dass sehr viel Platz benötigt wird, um die Information über die Zugehörigkeit zu den Äquivalenzklassen abzuspeichern: Allein wenn für jedes Attribut nur ein einziger scharfer Kontext definiert wird, verdoppelt sich die Anzahl der Attribute in der Relation. Werden gar mehrere Kontexte für ein Attribut oder unscharfe Kontexte mit vielen Termen defi-

5 Umsetzung

niert, braucht man weit mehr Attribute für die Information über die Zugehörigkeit als für die eigentlichen Daten.

Der Vorteil bei der Umsetzung von Kontexten durch zusätzliche Attribute besteht darin, dass auf die Informationen, die den Kontext beschreiben, sehr schnell zugegriffen werden kann, da diese direkt in der Relation abgespeichert sind.

5.1.2 Durch eine zusätzliche Relation

Eine weitere Möglichkeit besteht darin, den Kontext durch eine eigene Relation zu repräsentieren. Bei scharfen Kontexten enthält diese Relation für jeden Wert aus dem Wertebereich des Attributs, für das der Kontext definiert werden soll (hier und auch in den folgenden Abschnitten wieder mit A bezeichnet), ein Tupel, welches angibt, in welcher Äquivalenzklasse dieser Wert liegt. Für unscharfe Kontexte wird wieder für jeden Term ein Attribut benötigt, welches die Zugehörigkeit zu dem Term enthält.

Diese zusätzliche Relation enthält also genau die Attribute, die im eben vorgestellten Vorschlag der Relation, welche die eigentlichen Daten enthält, hinzugefügt worden wären, plus ein Attribut, welches alle Werte aus dem Wertebereich von A annimmt.

Durch eine Verbundoperation zwischen dieser Relation und der Relation, die die Daten enthält, gelangt man zu einer Form wie im vorigen Abschnitt und kann ebenso selektieren und gruppieren. Ein Vorteil gegenüber der im vorigen Abschnitt vorgestellten Methode ist jedoch, dass in der Relation, die den Kontext repräsentiert, für jeden möglichen Wert aus dem Wertebereich von A die Zugehörigkeit zu den Äquivalenzklassen bzw. Termen abgespeichert und der Kontext so wirklich definiert ist, während aus der erweiterten Relation im vorigen Abschnitt nur die Zugehörigkeit der tatsächlich in der Relation vorkommenden Werte von A hervorging und ein zusätzlicher Mechanismus benötigt wurde, falls neue Tupel eingefügt oder vorhandene geändert werden sollten. Ein weiterer Vorteil besteht darin, dass die Zugehörigkeit für jeden Wert von A nur einmal abgespeichert ist, während sie beim vorigen Vorschlag für mehrfach vorkommende Werte von A auch mehrfach abgespeichert wurde.

Allerdings empfiehlt sich die Methode nur für Attribute mit kleinem Wertebereich. Außerdem sind Verbundoperationen immer mit einem gewissen Aufwand verbunden.

5.1.3 Durch Angabe von Intervallgrenzen

Bei scharfen Kontexten, die eine Einteilung des Wertebereichs in Intervalle vorgeben, ist der Kontext durch die Angabe der Intervallgrenzen definiert. Diese Intervallgrenzen können zur Selektion mittels Vergleichen benutzt werden. In SQL werden hierzu `where`-Klauseln mit größer-gleich- und kleiner-gleich-Operatoren verwendet. Gruppieren kann jedoch direkt anhand der Intervallgrenzen nicht.

5.1.4 Durch Funktionen

Eine weitere Möglichkeit zur Repräsentation eines Kontextes ist durch die Definition von Funktionen gegeben, die als Argument einen Wert aus dem Wertebereich von A nehmen und als Ergebnis im unscharfen Fall die Zugehörigkeit des Wertes zu einem bestimmten Term zurückgeben. Man benötigt also für jeden Term eine Funktion.

Auch scharfe Kontexte können durch Funktionen repräsentiert werden. Entweder man definiert für jede Äquivalenzklasse eine Funktion, die je nach Enthaltensein des Wertes ent-

weder null oder eins zurückliefert; da ein Wert jedoch immer in genau einer Äquivalenzklasse liegt, ist es auch möglich für den gesamten Kontext nur eine Funktion zu definieren, welche die Äquivalenzklasse (oder besser einen Wert, der diese eindeutig identifiziert) zurückliefert, in der das Argument liegt.

Unter Oracle besteht die Möglichkeit, mit Hilfe der Programmiersprache PL/SQL so genannte „stored functions“ zu programmieren und diese dann in SQL-Anfragen zu verwenden. So kann dann z.B. nach den Funktionsergebnissen selektiert und gruppiert werden.

Funktionen bieten sich immer dann an, wenn es eine Regel gibt, mit Hilfe derer die Zugehörigkeit bestimmt bzw. berechnet werden kann. Sie stellen, was den Speicherplatz angeht, eine sehr effiziente Umsetzung von Kontexten dar, jedoch ist eine Verwendung der so realisierten Kontexte immer mit Rechenaufwand verbunden, da jedes Mal die entsprechende Funktion aufgerufen werden muss.

5.2 Bestimmung der Klassenzugehörigkeit

Für scharfe Kontexte erfolgt die Klassenbildung recht einfach: Sollen die Klassen zur Selektion benutzt werden, verknüpft man die Selektionsbedingungen konjunktiv, die sich durch die Wahl der Äquivalenzklasse für die einzelnen Attribute ergeben. In SQL erfolgt dies durch die Verknüpfung mehrerer **where**-Klauseln mit **and**. Sollen scharfe Klassen zur Gruppierung verwendet werden, werden die Attribute oder Funktionen, welche die Kontexte beschreiben, in einer **group by**-Klausel kombiniert.

Für unscharfe Klassen muss die Klassenzugehörigkeit mit einem Operator berechnet werden, wie sie in Abschnitt 2.3.2 vorgestellt wurden. Dieser Operator kann als Funktion umgesetzt werden, die als Argumente die Zugehörigkeiten zu den einzelnen Termen nimmt und als Ergebnis die Zugehörigkeit zur unscharfen Klasse liefert. Durch einen Vergleich in einer **where**-Klausel kann man nun alle Tupel selektieren, deren Klassenzugehörigkeit größer als ein Mindestwert ist. Weiter kann es sinnvoll sein, die Tupel nach ihrer Klassenzugehörigkeit zu sortieren (**order by**-Klausel), um die am stärksten zur Klasse gehörenden Tupel zu finden.

5.3 Berechnung von Aggregaten für unscharfe Klassen

Die in Abschnitt 2.4.2 vorgeschlagenen Aggregatfunktionen können leicht unter Verwendung der bereits für scharfe Einteilungen existierenden Aggregatfunktionen umgesetzt werden.

Anzahl

Die unscharfe Anzahl wurde definiert als die Summe der Zugehörigkeiten aller Tupel zu einer bestimmten unscharfen Klasse. Es kann also einfach die Aggregatfunktion „Summe“ auf das Attribut bzw. den Funktionswert angewendet werden, der die Zugehörigkeit zu der unscharfen Klasse beschreibt. Da die Zugehörigkeiten aller Tupel in der Relation addiert werden müssen, braucht nicht gruppiert zu werden.

Summe

Ähnlich wie die Anzahl wird auch die unscharfe Summe berechnet. Hier wird für jedes Tupel der Relation der Wert des Attributs, dessen Summe berechnet werden soll, mit der

5 Umsetzung

Zugehörigkeit des Tupels zur Klasse multipliziert, für die die Summe berechnet werden soll. Das Ergebnis wird über alle Tupel mit der herkömmlichen Aggregatfunktion „Summe“ summiert.

Durchschnitt, Standardabweichung

Der unscharfe Durchschnitt kann als Quotient aus unscharfer Summe und unscharfer Anzahl berechnet werden, die unscharfe Standardabweichung wie in Abschnitt 2.4.2 beschrieben aus unscharfer Anzahl und der gewichteten Summe der Quadrate. Letztere wird wie die unscharfe Summe berechnet, mit dem Unterschied, dass die Werte quadriert werden, bevor man sie mit der Zugehörigkeitsfunktion gewichtet.

Es besteht ein wesentlicher Unterschied zwischen der Aggregation über scharfe und der über unscharfe Klassen, der sich im benötigten Rechenaufwand niederschlägt. Dies soll am Beispiel der Aggregatfunktion „Summe“ verdeutlicht werden. Summiert man über scharfe Klassen, so werden die Attribute, welche die Äquivalenzklassen beschreiben, zur Gruppierung verwendet. Tupel, bei denen diese Attribute alle den selben Wert haben, gehören zur selben Klasse und der Wert des zu summierenden Attributs fließt dann in die Summe für die entsprechende Klasse ein. Jeder Wert wird also für genau eine Klasse aufsummiert.

Im Gegensatz dazu wird bei unscharfen Klassen der zu summierende Wert in die Summe für jede Klasse eingerechnet – jedesmal gewichtet mit der Zugehörigkeit zu der Klasse. Es wird also für jede Klasse über alle Tupel summiert, was natürlich einen entsprechend höheren Aufwand zur Folge hat, da für jede unscharfe Klasse eine Aggregation mit der herkömmlichen Aggregatfunktion „Summe“ benötigt wird.

5.4 Floating Car Daten

5.4.1 Umsetzung der Kontexte

Durchschnittsgeschwindigkeit, Standardabweichung und Stillstandsanteil

Die Kontexte für Durchschnittsgeschwindigkeit, Standardabweichung und Stillstandsanteil sind einander sehr ähnlich. Es handelt sich immer um linguistische Variablen mit zwei Termen *hoch* und *niedrig*. Für die Bestimmung der Zugehörigkeit zu den Termen *hoch* und *niedrig* lässt sich gemäß den Gleichungen auf Seite 35 eine einfache Regel angeben. Betrachten wir den Term *hoch*: Die Zugehörigkeit eines bestimmten Wertes x zu diesem Term ist null, falls x kleiner als ein unterer Grenzwert ist, und eins, falls x größer als ein oberer Grenzwert ist. Für Werte dazwischen lässt sich die Zugehörigkeit aufgrund des linearen Verlaufs der Zugehörigkeitsfunktion in diesem Bereich nach einer einfachen Formel berechnen.

Hier bietet sich also eine Realisierung der Kontexte durch Funktionen an. Da sich die einzelnen Kontexte lediglich durch die jeweiligen Grenzwerte unterscheiden, können sie alle durch die selbe Funktion umgesetzt werden. Für die linguistische Variable, zu deren Term die Zugehörigkeit berechnet werden soll, werden die zugehörigen Grenzwerte dann jeweils eingesetzt. Hierzu wurden sämtliche Grenzwerte mit dem Namen der zugehörigen linguistischen Variable in der Relation `fcd_k_grenzwerte` abgespeichert, aus der die jeweils

benötigten Werte dann von der Funktion extrahiert werden. Die Grenzwerte wurden recht willkürlich festgesetzt; hier müssen, etwa durch Evaluation, eventuell geeignetere Werte bestimmt werden.

Die „stored function“ `fcd_hoch` berechnet die Zugehörigkeit zum Term *hoch*. Als erster Parameter muss der Name der linguistischen Variable übergeben werden, so wie er in der Relation `fcd_k_grenzwerte` abgespeichert ist. Der zweite Parameter beinhaltet den Wert, dessen Zugehörigkeit berechnet werden soll. Wie oben erläutert werden zunächst mit Hilfe einer `select`-Anweisung die benötigten Grenzwerte extrahiert; zur anschließenden Bestimmung der Zugehörigkeit sind zwei Fallunterscheidungen nötig:

```
FUNCTION fcd_hoch (lv IN VARCHAR2, x IN NUMBER) RETURN NUMBER AS
  xmin number;
  xmax number;
BEGIN
  SELECT xmin, xmax INTO xmin, xmax FROM ovidsa05.fcd_k_grenzwerte
  WHERE lingvar=lv;
  IF x <= xmin THEN
    RETURN 0;
  ELSIF x >= xmax THEN
    RETURN 1;
  ELSE
    RETURN (x-xmin)/(xmax-xmin);
  END IF;
END;
```

Da die Summe der Zugehörigkeiten zu den Termen *hoch* und *niedrig* immer eins ergibt, lässt sich die Zugehörigkeit zum Term *niedrig* berechnen, indem man die Zugehörigkeit zum Term *hoch* von eins subtrahiert. Hier die entsprechende Funktion:

```
FUNCTION fcd_niedrig (lv IN VARCHAR2, x IN NUMBER) RETURN NUMBER AS
  BEGIN
  RETURN 1-fcd_hoch(lv, x);
  END;
```

Die so definierten Kontexte können nun in Anfragen verwendet werden. Hier würden z.B. alle Messungen für Autobahnen (Straßenkategorie 3) selektiert, für welche die Zugehörigkeit der Geschwindigkeit zum Term *hoch* größer als 0,3 ist:

```
SELECT id, geschwindigkeitsdurschn,
fcd_hoch('geschwindigkeit_autobahn', geschwindigkeitsdurschn) as hoch,
fcd_niedrig('geschwindigkeit_autobahn', geschwindigkeitsdurschn) as niedrig
FROM ovidsa05.fcd
WHERE fcd_hoch('geschwindigkeit_autobahn', geschwindigkeitsdurschn) >= 0.3
AND strassenkategorie = 3;
```

Ein Ergebnis, welches diese Anfrage liefern könnte, zeigt Tabelle 5.1.

Anfragen unter Verwendung der Kontexte für Standardabweichung und Stillstandsanteil werden, auch für die anderen Straßenkategorien, analog hierzu formuliert. Es muss lediglich der Name der jeweiligen linguistischen Variable an die Funktionen `fcd_hoch` und `fcd_niedrig` übergeben werden, damit die passenden Grenzwerte ausgewählt werden.

id	geschwindigkeitsdurschn	hoch	niedrig
14085	98	0,56	0,44
14086	88	0,36	0,64
13833	106	0,72	0,28
13834	93	0,46	0,54
13836	138	1,00	0,00

Tabelle 5.1: Beispiel für eine Anfrage unter Verwendung des Kontexts für die Durchschnittsgeschwindigkeit auf Autobahnen

Zeit

Für das Attribut *Zeit* wurden drei verschiedene Hierarchien definiert. Betrachten wir zunächst die Einteilung in Stunden, Tage, Monate und Jahre. Die Kontexte dieser Hierarchie lassen sich sehr gut durch Funktionen repräsentieren. Da es sich um scharfe Kontexte handelt, genügt für jeden Kontext eine Funktion, die einen Identifikator für die jeweilige Äquivalenzklasse zurückliefert.

Die Zeitangaben in den vorliegenden Floating Car Daten sind als Zeichenketten abgespeichert. Für eine Verarbeitung durch Funktionen ist aber eine Repräsentation durch den Oracle-Datentyp DATE geeigneter. Die Zeitangaben wurden daher zunächst mit der im Funktionsumfang von PL/SQL mitgelieferten Funktion TO_DATE konvertiert.

Als Funktion, die die Äquivalenzklasse eines Wertes zurückliefert, lässt sich die ebenfalls im PL/SQL-Funktionsumfang enthaltene Funktion TRUNC verwenden. Diese Funktion schneidet von einem Datumswert einen bestimmten Teil ab. TRUNC(d, 'HH') schneidet vom Datumswert *d* beispielsweise alles ab, was genauer als die Stundenangabe ist. Aus 21.03.2004 14:34:54h würde dann 21.03.2004 14:00:00h. Ruft man die Funktion an Stelle von 'HH' mit den Parametern 'DD', 'MM' bzw. 'YY' auf, wird auf Tages-, Monats-, bzw. Jahresebene abgeschnitten.

Betrachten wir nun die Kontexte der zweiten Hierarchie. Messungen, die in der selben Stunde und am selben Wochentag gemacht wurden, sollen zu einer Äquivalenzklasse gehören. Die Stunde und der Wochentag könnten aus der Zeit der Messung berechnet werden. Jedoch existieren für die vorliegenden Messdaten bereits die Attribute *Stunde* und *Tag*, die für alle Tupel die Stunde bzw. den Wochentag der Messung enthalten. Zusammengefasst¹ repräsentieren diese so wie in Abschnitt 5.1.1 beschrieben den Kontext, der nach Stunden und Wochentagen unterscheidet. Soll eine Ebene höher in der Hierarchie lediglich nach Wochentagen unterschieden werden, wird ausschließlich das Attribut *Tag* betrachtet.

Die dritte parallele Einteilung, die zwischen Messungen, welche innerhalb der letzten Stunde gemacht wurden, und allen übrigen unterscheidet, lässt sich am einfachsten durch Angabe einer Intervallgrenze beschreiben. Die Grenze zwischen den beiden Intervallen ist der Zeitpunkt, der genau eine Stunde vor der aktuellen Zeit liegt. Da sich diese Grenze ständig ändert und sowieso nur eine der beiden Äquivalenzklassen interessant ist, nämlich die der Messungen, die innerhalb der letzten Stunde erfolgten, wird dieser Kontext am einfachsten durch Angabe der Grenze in einer **where**-Klausel umgesetzt. Da es keine Mes-

¹In diesem Fall identifiziert nicht der Wert eines Attributs, sondern die Kombination der Werte zweier Attribute die Äquivalenzklasse.

Fahrzeugtyp	Äquivalenzklasse
0	0
1	1
2	1
3	2
4	1

Tabelle 5.2: Relation für den Kontext des Attributs *Fahrzeugtyp*

Äquivalenzklasse	Beschreibung
0	unbekannt
1	PKW
2	LKW

Tabelle 5.3: Beschreibung der Äquivalenzklassen aus Tabelle 5.2

sungen gibt, die zu einem späteren Zeitpunkt als der aktuellen Zeit gemacht wurden, muss keine Bedingung für die oberen Grenze des Intervalls angegeben werden.

Die untere Intervallgrenze kann mit Hilfe der PL/SQL-Funktion `SYSDATE` bestimmt werden, welche die aktuelle Zeit zurückliefert. Vom Rückgabewert dieser Funktion muss eine Stunde abgezogen werden. Oracle erlaubt die Anwendung arithmetischer Operationen direkt auf Zeitangaben. Um eine Zeitangabe um eine Stunde zurückzusetzen, muss der Wert $\frac{1}{24}$ abgezogen werden, da der Wert eins einem ganzen Tag entspricht. Alle in der letzten Stunde für den Link 14030 angefallenen Tupel erhält man so z.B. durch folgende Anfrage:

```
SELECT * FROM ovidsa05.fcd WHERE zeit>=SYSDATE-1/24 AND linkid=14030;
```

Natürlich sind in den vorliegenden Daten keine so aktuellen Werte enthalten. Das letzte Tupel wurde am 30.10.2002 um 18:24h registriert. Daher kann eine Auswertung für die aktuelle Stunde simuliert werden, indem man diesen Zeitpunkt an Stelle der aktuellen Zeit einsetzt:

```
SELECT * FROM ovidsa05.fcd WHERE
zeit>=TO_DATE('30.10.02 18:24','DD.MM.YY HH24:MI')-1/24 AND linkid=14030;
```

Fahrzeugtyp

Da der Wertebereich des Attributs *Fahrzeugtyp* nur fünf Werte umfasst, wird der zugehörige Kontext als zusätzliche Relation umgesetzt (siehe Tabelle 5.2). Zur Identifikation der drei Äquivalenzklassen werden hier Zahlen von null bis zwei eingesetzt. In einer weiteren Relation werden diese Identifikatoren als Schlüssel verwendet, um den Äquivalenzklassen Bezeichnungen zuzuweisen (Tabelle 5.3).

5.4.2 Klassenbildung

Für die unscharfen Kontexte muss nun noch die Frage der Klassenbildung geklärt werden. Für Autobahnen und Landstraßen sollen die Klassen aus den Kontexten für die Durchschnittsgeschwindigkeit und die Standardabweichung gebildet werden, für Stadtstraßen aus

LinkId	Stunde	lebhafter Verkehr	freier Verkehr	stockender Verkehr	Stau
550	04-Mai-2001 02:00:00 PM	0,10	0,24	0,23	0,43
5400	04-Mai-2001 03:00:00 PM	0,38	0,00	0,62	0,00
820	04-Mai-2001 04:00:00 PM	0,00	0,00	1,00	0,00
840	04-Mai-2001 04:00:00 PM	0,00	0,65	0,00	0,35
410	13-Mai-2001 04:00:00 PM	0,11	0,25	0,22	0,42

Tabelle 5.4: Beispiel für die Klassifikation von Strecken

den Kontexten für die Durchschnittsgeschwindigkeit und den Stillstandsanteil. Hierzu wurde die Funktion `gamma2` implementiert, welche die Klassenzugehörigkeit aus zwei Einzelzugehörigkeiten gemäß Formel 2.4 berechnet:

```
FUNCTION gamma2 (m1 IN NUMBER, m2 IN NUMBER, gamma IN NUMBER)
RETURN NUMBER AS
BEGIN
RETURN ((m1*m2)**(1-gamma))*((1-(1-m1)*(1-m2))**gamma);
END;
```

Die ersten beiden Parameter sind die Zugehörigkeiten zu den Termen, aus denen die Klasse gebildet wird, zu der die Zugehörigkeit berechnet werden soll. Der dritte Parameter bestimmt den Wert, der für γ verwendet wird, und gibt daher den Grad der Kompensation an.

5.4.3 Anfragen unter Verwendung der Kontexte

Nun wird anhand einiger Beispiele gezeigt, wie die Kontexte in Anfragen verwendet werden können.

Bestimmen des Verkehrszustandes für Strecken

Dieses etwas umfangreiche Beispiel zeigt, wie mit Hilfe der Kontexte, die in Abschnitt 5.4.1 umgesetzt wurden, eine SQL-Anfrage zur Bestimmung des Verkehrszustandes einzelner Strecken formuliert werden kann. Das Ergebnis der Anfrage ordnet jedem Link und jeder Stunde Zugehörigkeiten zu den vier Verkehrszuständen zu (Tabelle 5.4). Damit der Code übersichtlich bleibt, wurden mehrere geschachtelte Anfragen verwendet, die von unten nach oben zu lesen sind: Im fünften Block werden die Zugehörigkeiten der Werte für die Durchschnittsgeschwindigkeit und die Standardabweichung zu den jeweiligen Termen *hoch* und *niedrig* für alle Tupel berechnet, die Autobahnen beschreiben. Gleichzeitig wird aus dem Attribut *Zeit* die Stunde der Messung bestimmt. Im vierten Block werden mit Hilfe des Gammaoperators die Zugehörigkeiten zu den Klassen berechnet. Im dritten Block werden schließlich die Zugehörigkeiten aller Tupel summiert, die eine bestimmte Strecke in einer bestimmten Stunde beschreiben. Dies geschieht durch eine Aggregation mit Gruppierung nach *LinkId* und *Stunde*. Im zweiten Block werden die Klassenzugehörigkeiten normiert, indem jede Zugehörigkeit durch die Summe aller Zugehörigkeiten dividiert wird. Im ersten Block werden die Ergebnisse noch gerundet.

```

/* Block 1: Runden der Zugehoerigkeiten auf zwei Nachkommastellen */
SELECT linkid, stunde,
ROUND(lebhafter_verkehr, 2) AS lebhafter_verkehr,
ROUND(freier_verkehr, 2) AS freier_verkehr,
ROUND(stockender_verkehr, 2) AS stockender_verkehr,
ROUND(stau, 2) AS stau
FROM(

/* Block 2: Normieren der Klassenzugehoerigkeiten */
SELECT stunde, linkid,
lebhafter_verkehr_s/(lebhafter_verkehr_s+freier_verkehr_s
+stockender_verkehr_s+stau_s) AS lebhafter_verkehr,
freier_verkehr_s/(lebhafter_verkehr_s+freier_verkehr_s
+stockender_verkehr_s+stau_s) AS freier_verkehr,
stockender_verkehr_s/(lebhafter_verkehr_s
+freier_verkehr_s+stockender_verkehr_s+stau_s) AS stockender_verkehr,
stau_s/(lebhafter_verkehr_s+freier_verkehr_s
+stockender_verkehr_s+stau_s) AS stau
FROM(

/* Block 3: Summieren der Zugehoerigkeiten der einzelnen Tupel */
SELECT stunde, linkid,
SUM(lebhafter_verkehr_e) AS lebhafter_verkehr_s,
SUM(freier_verkehr_e) AS freier_verkehr_s,
SUM(stockender_verkehr_e) AS stockender_verkehr_s,
SUM(stau_e) AS stau_s
FROM(

/* Block 4: Berechnen der Klassenzugehoerigkeiten */
SELECT id, stunde, linkid,
gamma2(geschw_hoch, stdabw_hoch, 0.5) AS lebhafter_verkehr_e,
gamma2(geschw_hoch, stdabw_niedrig, 0.5) AS freier_verkehr_e,
gamma2(geschw_niedrig, stdabw_hoch, 0.5) AS stockender_verkehr_e,
gamma2(geschw_niedrig, stdabw_niedrig, 0.5) AS stau_e
FROM(

/* Block 5: Berechnen der Zugehoerigkeit zu den Termen */
SELECT id, TRUNC(zeit, 'HH') AS stunde, linkid,
fcd_hoch('geschwindigkeit_autobahn', geschwindigkeitsdurschn)
AS geschw_hoch,
fcd_niedrig('geschwindigkeit_autobahn', geschwindigkeitsdurschn)
AS geschw_niedrig,
fcd_hoch('standardabweichung_autobahn', geschwindigkeitstandardabw)
AS stdabw_hoch,
fcd_niedrig('standardabweichung_autobahn', geschwindigkeitstandardabw)
AS stdabw_niedrig
FROM ovidsa05.fcd WHERE strassenkategorie=3))

```

5 Umsetzung

```
/* Die GROUP BY-Klausel gehoert eigentlich zu Block 3, muss aber hier am
   Ende stehen */
GROUP BY stunde, linkid));
```

Aggregation

Nun folgen noch Beispiele dafür, wie die Kontexte für sonstige Auswertungen verwendet werden können. Das erste Beispiel zeigt, wie mit Hilfe der Einteilung, die nach Wochentagen und Stunden unterscheidet, die an den einzelnen Wochentagen in jeder Stunde gefahrene Durchschnittsgeschwindigkeit berechnet werden kann. Da der entsprechende Kontext durch zusätzliche Attribute in der Relation umgesetzt ist (Abschnitt 5.4.1), geschieht dies, indem diese Attribute zur Gruppierung verwendet werden:

```
SELECT linkid, tag, stunde, AVG(geschwindigkeitsdurschn) FROM ovidsa05.fcd
GROUP BY linkid, tag, stunde ORDER BY linkid, tag, stunde;
```

Das nächste Beispiel zeigt die Verwendung des Kontexts für den Fahrzeugtyp: Hier wird für jede Äquivalenzklasse die maximale Geschwindigkeit bestimmt, die von einem in dieser Klasse enthaltenen Fahrzeug auf jedem Link gefahren wurde.

Da der entsprechende Kontext als eigene Relation umgesetzt ist, muss diese zunächst über einen join-Operator mit der Relation verbunden werden, welche die zu klassifizierenden Daten enthält. Um das Ergebnis der Anfrage benutzerfreundlicher zu gestalten, wird zusätzlich eine Verbundoperation mit der Relation durchgeführt, die die Bezeichnungen der Äquivalenzklassen enthält.

```
SELECT linkid, aeq, beschreibung, MAX(geschwindigkeitsdurschn) FROM(

SELECT linkid, fcd.fahrzeugtyp, fcd_k_fahrzeugtyp.aeq AS aeq,
beschreibung, geschwindigkeitsdurschn
FROM ovidsa05.fcd, ovidsa05.fcd_k_fahrzeugtyp, fcd_k_fahrzeugtyp_b
WHERE fcd.fahrzeugtyp=fcd_k_fahrzeugtyp.fahrzeugtyp
AND fcd_k_fahrzeugtyp.aeq=fcd_k_fahrzeugtyp_b.aeq)

GROUP BY aeq, linkid, beschreibung ORDER BY linkid, aeq;
```

Schließlich soll nun noch gezeigt werden, wie bereits berechnete Aggregate zur Berechnung weiterer Aggregate verwendet werden können. Die Anfrage

```
CREATE TABLE fcd_max_geschw_link_und_stunde AS
SELECT linkid, TRUNC(zeit, 'HH') AS stunde,
MAX(geschwindigkeitsdurschn) AS max_geschw
FROM ovidsa05.fcd GROUP BY linkid, TRUNC(zeit, 'HH');
```

berechnet für jeden Link und jede Stunde die maximale Geschwindigkeit der FCD-Fahrzeuge, die den Link befahren haben, und speichert das Ergebnis in einer Relation. Will man nun die maximale Geschwindigkeit für jeden Tag erfahren, kann die Berechnung zum einen direkt aus den Elementardaten erfolgen:

```
SELECT linkid, TRUNC(zeit, 'DD') AS tag, MAX(geschwindigkeitsdurschn)
FROM ovidsa05.fcd GROUP BY linkid, TRUNC(zeit, 'DD');
```

Effizienter ist es, die bereits auf Stundenebene berechneten Aggregate zu verwenden:

```
SELECT linkid, TRUNC(stunde, 'DD') AS tag, MAX(max_geschw)
FROM ovidsa05.fcd_max_geschw_link_und_stunde
GROUP BY linkid, TRUNC(stunde, 'DD');
```

5.4.4 Trenngeraden

Nun soll noch die Umsetzung der alternativen Klasseneinteilung beschrieben werden, die durch beliebig verlaufende Geraden entsteht. Beispielhaft geschieht dies wieder für Autobahnen.

Bestimmung des Abstandes zu den Geraden

Im zweidimensionalen Raum lässt sich der Abstand eines Punktes zu einer Geraden aus den Koordinaten des Punktes sowie aus den Koordinaten eines Stützpunktes der Geraden und deren Normalenvektor berechnen nach der Formel (d ist der gesuchte Abstand, \vec{p} ist der Koordinatenvektor des Punktes, \vec{s} der des Stützpunktes der Geraden, \vec{n} ist der Normalenvektor der Geraden und $*$ steht für das innere Produkt):

$$d = (\vec{p} - \vec{s}) * \frac{\vec{n}}{|\vec{n}|} \quad (5.1)$$

Dabei ist d positiv, falls der Punkt auf der Seite der Geraden liegt, nach der der Normalenvektor zeigt.

Wie in Abschnitt 4.1.5 beschrieben, sollen die Verkehrszustände durch vier Geraden g_1 bis g_4 festgelegt werden. Die Geradenparameter (also Stützpunkt und Normalenvektor) wurden angelehnt an Abbildung 1-3 in [SW03] festgelegt. Um die Berechnungen zu vereinfachen, wurde der Stützpunkt jeweils auf die x -Achse gelegt und ein Normaleneinheitsvektor verwendet. Dies erspart die Subtraktion der y -Koordinate des Stützpunktes sowie die Division durch die Länge des Normalenvektors. Damit die Abstände für Punkte links einer Geraden ein negatives Vorzeichen erhalten, wurde der Normalenvektor stets nach rechts zeigend gewählt.

Für die vier Geraden wurden jeweils die x - und die y -Komponente des Normaleneinheitsvektors sowie die x -Koordinate des Stützpunktes, außerdem die Breite des durch die Gerade beschriebenen Übergangsbereichs in der Relation `fcd_geraden` abgespeichert. Die Berechnung der Abstände erfolgt nach Gleichung 5.1 durch die PL/SQL-Funktion `fcd_abstand`. Als ersten Parameter benötigt sie die Nummer der Geraden, zu der der Abstand berechnet werden soll. Der zweite und der dritte Parameter geben die Koordinaten des Punktes an, dessen Abstand bestimmt werden soll.

Geraden zur Festlegung von Verkehrszuständen für die anderen Streckentypen ließen sich leicht durch das Einfügen zusätzlicher Tupel in die Relation `fcd_geraden` definieren.

Kontexte

Nun müssen die Kontexte mit den Termen *links* und *rechts*, wie sie in Abschnitt 4.1.5 beschrieben werden, für die vier Geraden umgesetzt werden. Dies geschieht wieder mit Hilfe von PL/SQL-Funktionen. Die Funktion `fcd_linksvon` berechnet aus dem Abstand die

5 Umsetzung

Zugehörigkeit zum Term *links*. Da die Breite des Übergangsbereichs für die einzelnen Geraden verschieden sein kann, muss auch die Nummer der Geraden als Parameter übergeben werden.

```
FUNCTION fcd_linksvon (g IN NUMBER, abstand IN NUMBER) RETURN NUMBER AS
  d NUMBER;
BEGIN
  SELECT breite INTO d FROM ovidsa05.fcd_geraden WHERE gerade=g;
  IF abstand <=-d THEN
    return 1;
  ELSIF abstand >=d THEN
    RETURN 0;
  ELSE
    RETURN (d-abstand)/(2*d);
  END IF;
END;
```

Die Funktion *fcd_rechtsvon* berechnet die Zugehörigkeit zum Term *rechts* durch Subtraktion der Zugehörigkeit zum Term *links* von eins.

Klassen und Verkehrszustände

Um die Klassenzugehörigkeit zu berechnen, wird hier ein Gammaoperator benötigt, der vier Zugehörigkeiten als Parameter hat. Dies leistet die Funktion *gamma4*, die ansonsten wie die Funktion *gamma2* funktioniert.

Von den sich ergebenden Klassen entspricht, wie in Abschnitt 4.1.5 erläutert, jeweils eine den Verkehrszuständen *Stau*, *lebhafter Verkehr* und *freier Verkehr*. Der Verkehrszustand *zähfließender Verkehr* wird durch drei Klassen beschrieben; die Zugehörigkeiten zu jenen Klassen werden addiert, um die Zugehörigkeit zu diesem Verkehrszustand zu erhalten.

Beispielanfrage

Nun soll noch gezeigt werden, wie mittels der durch Geraden festgelegten Einteilung der Verkehrszustand einer Strecke bestimmt werden kann. Die Anfrage ist ähnlich aufgebaut wie diejenige zur Bestimmung des Verkehrszustandes anhand der Kontexte für die Geschwindigkeit und die Standardabweichung (Abschnitt 5.4.3). Hier ist jedoch noch ein sechster Block hinzugefügt worden, in dem zunächst die Abstände zu den vier Geraden berechnet werden. Für diese Abstandswerte werden dann in Block 5 die Zugehörigkeiten zu den jeweiligen Termen *links* und *rechts* berechnet. Die Berechnung der Klassenzugehörigkeit, die Summierung, die Normierung und das Runden geschehen wie bei der anderen Anfrage auch.

Hier ist lediglich noch zu beachten, dass nur die Zugehörigkeiten zu den sechs tatsächlich vorkommenden Klassen berechnet werden (siehe Abschnitt 4.1.5). Die Addition der Zugehörigkeiten zu den Klassen, die den Verkehrszustand *zähfließender Verkehr* beschreiben, erfolgt in Block 1.

```
/* Block 1: Runden der Zugehoerigkeiten auf zwei Nachkommastellen */
SELECT linkid, stunde,
ROUND(rrrr, 2) AS freier_verkehr,
ROUND(rrr1, 2) AS lebhafter_verkehr,
```

```

ROUND(rrll+rlll+lrll, 2) AS zaehfliessender_verkehr,
ROUND(llll, 2) AS stau
FROM(

/* Block 2: Normieren der Klassenzugehoerigkeiten */
SELECT linkid, stunde,
rrrr_s/(rrrr_s+rrrl_s+rrll_s+rlll_s+lrll_s+llll_s) AS rrrr,
rrrl_s/(rrrr_s+rrrl_s+rrll_s+rlll_s+lrll_s+llll_s) AS rrrl,
rrll_s/(rrrr_s+rrrl_s+rrll_s+rlll_s+lrll_s+llll_s) AS rrll,
rlll_s/(rrrr_s+rrrl_s+rrll_s+rlll_s+lrll_s+llll_s) AS rlll,
lrll_s/(rrrr_s+rrrl_s+rrll_s+rlll_s+lrll_s+llll_s) AS lrll,
llll_s/(rrrr_s+rrrl_s+rrll_s+rlll_s+lrll_s+llll_s) AS llll
FROM(

/* Block 3: Summieren der Zugehoerigkeiten der einzelnen Tupel */
SELECT linkid, stunde,
SUM(rrrr_e) AS rrrr_s,
SUM(rrrl_e) AS rrrl_s,
SUM(rrll_e) AS rrll_s,
SUM(rlll_e) AS rlll_s,
SUM(lrll_e) AS lrll_s,
SUM(llll_e) AS llll_s
FROM(

/* Block 4: Berechnen der Klassenzugehoerigkeiten */
SELECT id, linkid, stunde,
gamma4(g1_rechts, g2_rechts, g3_rechts, g4_rechts, 0.5) AS rrrr_e,
gamma4(g1_rechts, g2_rechts, g3_rechts, g4_links, 0.5) AS rrrl_e,
gamma4(g1_rechts, g2_rechts, g3_links, g4_links, 0.5) AS rrll_e,
gamma4(g1_rechts, g2_links, g3_links, g4_links, 0.5) AS rlll_e,
gamma4(g1_links, g2_rechts, g3_links, g4_links, 0.5) AS lrll_e,
gamma4(g1_links, g2_links, g3_links, g4_links, 0.5) AS llll_e
FROM(

/* Block 5: Berechnen der Zugehoerigkeiten zu den Termen links bzw.
rechts */
SELECT id, linkid, stunde,
fcd_linksvon(1, abstand_g1) AS g1_links,
fcd_rechtsvon(1, abstand_g1) AS g1_rechts,
fcd_linksvon(2, abstand_g2) AS g2_links,
fcd_rechtsvon(2, abstand_g2) AS g2_rechts,
fcd_linksvon(3, abstand_g3) AS g3_links,
fcd_rechtsvon(3, abstand_g3) AS g3_rechts,
fcd_linksvon(4, abstand_g4) AS g4_links,
fcd_rechtsvon(4, abstand_g4) AS g4_rechts
FROM(

```

5 Umsetzung

```
/* Block 6: Berechnen der Abstände zu den Geraden */
SELECT id, linkid, TRUNC(zeit, 'HH') AS stunde,
fcd_abstand(1, geschwindigkeitsdurschn, geschwindigkeitstandardabw)
AS abstand_g1,
fcd_abstand(2, geschwindigkeitsdurschn, geschwindigkeitstandardabw)
AS abstand_g2,
fcd_abstand(3, geschwindigkeitsdurschn, geschwindigkeitstandardabw)
AS abstand_g3,
fcd_abstand(4, geschwindigkeitsdurschn, geschwindigkeitstandardabw)
AS abstand_g4
FROM ovidsa05.fcd WHERE strassenkategorie=3)))

/* GROUP BY-Klausel gehoert zu Block 3 */
GROUP BY linkid, stunde));
```

5.5 Simulationsdaten für Verkehrsmessstationen

5.5.1 Umsetzung der Kontexte

Geschwindigkeit und Stillstandsanteil

Die Kontexte für Geschwindigkeit und Stillstandsanteil sollen wie bei den Floating Car Daten durch Funktionen umgesetzt werden. Dies geschieht auch hier durch zwei Funktionen `messst_hoch` und `messst_niedrig`, die wie `fcd_hoch` bzw. `fcd_niedrig` funktionieren und ihre Grenzwerte aus der Relation `messst_k_grenzwerte` holen.

Zeit

Anders als bei den Floating Car Daten ist der Zeitpunkt der Messung bei den Simulationsdaten für die Messstationen als Anzahl der Sekunden seit Beginn der Simulation abgespeichert. Auch hier kann die PL/SQL-Funktion `TRUNC` verwendet werden, um aus einem Wert dessen Äquivalenzklasse zu berechnen. Übergibt man als einziges Argument einen numerischen Wert an `TRUNC`, so schneidet die Funktion alle Nachkommastellen des Wertes ab. Um nun also zu berechnen, in welchem Fünfminutenintervall sich ein Wert befindet, dividiert man diesen zunächst durch 300 (die Anzahl der Sekunden, die ein Fünfminutenintervall umfasst) und wendet dann die Funktion `TRUNC` auf das Ergebnis an. Man erhält 0, falls der Wert im ersten Fünfminutenintervall liegt, 1, falls er im zweiten liegt, usw. Will man Fünfzehnminutenintervalle verwenden, dividiert man an Stelle von 300 durch 900, für Stundenintervalle durch 3600, usw.

Die Tupel, die im Zeitintervall *jetzt* erfasst wurden, erhält man wieder durch einen Vergleich in einer `where`-Klausel: Man rechnet von der aktuellen Simulationszeit die Anzahl an Sekunden zurück, die der gewünschten Intervallbreite entspricht, und vergleicht mit dem sich ergebenden Wert.

Fahrzeugtyp

Für die feinere Einteilung in die Äquivalenzklassen *PKW*, *LKW*, *Bus*, *Zweirad*, *Bahn* und *Fußgänger* wurde im Rahmen von [San04] bereits in der Relation `fahrzeugtyp` eine Zu-

Kategorie	relevant
PKW	1
LKW	1
BUS	1
Zweirad	1
Bahn	0
Fußgänger	0

Tabelle 5.5: Relation zur Unterscheidung von relevanten und nicht relevanten Fahrzeugen

ordnung der Fahrzeugtypen zu den Äquivalenzklassen² umgesetzt, die auch hier verwendet werden soll.

Die eine Ebene höher angesiedelte gröbere Einteilung wurde im Rahmen dieser Arbeit zusätzlich durch die Relation `messst_k_fahrzeugtyp` umgesetzt, welche die Äquivalenzklassen der feineren Einteilung in für die Bestimmung des Verkehrszustandes relevante bzw. nicht relevante Typen unterteilt (Tabelle 5.5).

5.5.2 Klassenbildung

Für die auf Strecken umgerechneten Daten kann zur Bestimmung der Zugehörigkeit zu den Klassen, welche die Verkehrszustände beschreiben, wie bei den Floating Car Daten die Funktion `gamma2` benutzt werden. Da die Klassen der Verkehrszustände für die auf Punkte bezogenen Daten aus den Termen nur einer linguistischen Variable gebildet werden, kann die Berechnung der Klassenzugehörigkeit hier entfallen; die Klassenzugehörigkeit entspricht direkt der Zugehörigkeit zu den Termen. Falls nur einzelne Tupel klassifiziert werden, kann sogar auf die Normierung der Klassenzugehörigkeit verzichtet werden.

5.5.3 Anfragen unter Verwendung der Kontexte

Es folgt nun noch eine beispielhafte Anfrage, in der die Kontexte Verwendung finden, deren Umsetzung in den vorhergehenden Abschnitten beschrieben wurde. Für die Messstation mit der Identifikationsnummer 1 soll der Verkehrszustand in Fünfminutenintervallen bestimmt werden. Dabei sollen jedoch nur relevante Fahrzeuge berücksichtigt werden.

Im untersten Block wird das Intervall mittels der Funktion `TRUNC` bestimmt. Weiter werden in diesem Block nur Fahrzeuge mit relevantem Fahrzeugtyp selektiert, die an Messstation 1 erfasst wurden. Der Fahrzeugtyp ist nicht direkt in der Relation `messung` abgespeichert, sondern über eine Fremdschlüsselbeziehung in der Relation `fahrzeug`, weswegen eine Verbundoperation mit dieser Relation erforderlich ist. Die Äquivalenzklasse feiner Granularität wird durch eine Verbundoperation mit der Relation `fahrzeugtyp` bestimmt, die Äquivalenzklasse grober Granularität, nach der dann schließlich selektiert wird, durch einen Verbund mit der Relation `messst_k_fahrzeugtyp`. Der Rest der Anfrage läuft, bis auf die hier nicht nötige Berechnung der Klassenzugehörigkeit, nach dem bekannten Schema.

```
SELECT intervall, ROUND(hoch, 2) AS hoch, ROUND(niedrig, 2) AS niedrig
FROM(
```

²Diese werden dort als „Kategorien“ bezeichnet.

5 Umsetzung

```
SELECT intervall, hoch_s/(hoch_s+niedrig_s) AS hoch,  
niedrig_s/(hoch_s+niedrig_s) AS niedrig FROM(  
  
SELECT intervall, SUM(hoch_e) AS hoch_s, SUM(niedrig_e) AS niedrig_s FROM(  
  
SELECT intervall,  
messst_hoch('geschwindigkeit_stadtstrasse', geschwindigkeit) AS hoch_e,  
messst_niedrig('geschwindigkeit_stadtstrasse', geschwindigkeit)  
AS niedrig_e FROM(  
  
SELECT TRUNC(teinfahrt/300) AS intervall, geschwindigkeit  
FROM ovidsa01.messung, ovidsa01.fahrzeug, ovidsa01.fahrzeugtyp,  
ovidsa05.messst_k_fahrzeugtyp  
WHERE ovidsa01.messung.fahrzeug_id = ovidsa01.fahrzeug.fz_nr  
AND ovidsa01.fahrzeug.fz_typ_id = ovidsa01.fahrzeugtyp.fz_typ_id  
AND  
ovidsa01.fahrzeugtyp.kategorie = ovidsa05.messst_k_fahrzeugtyp.kategorie  
AND messstation_id=1 AND relevant=1))  
  
GROUP BY intervall));
```

5.6 Güterverkehrsdaten

5.6.1 Umsetzung der Kontexte

Art des Anhängers

Der sehr einfachen Kontext für die Art des Anhängers kann mit Hilfe eines Vergleichs realisiert werden. Die Klausel `where artanhaenger = 0` selektiert alle Fahrzeuge ohne, die Klausel `where artanhaenger != 0` alle Fahrzeuge mit Anhänger.

Be- und Entladeort

Bei Be- und Entladeort wird für beide Attribute der selbe Kontext verwendet. Er wird durch die Relation `gv_k_ort` repräsentiert, die für jeden Regionscode angibt, ob die entsprechende Region im In- oder im Ausland liegt.

Genutzter Rauminhalt

Die beiden unscharfen Kontexte für den genutzten Rauminhalt werden durch Funktionen umgesetzt. Da die Zugehörigkeitsfunktionen zu den Termen feiner Granularität alle sehr ähnlich aussehen und lediglich entlang der x-Achse gegeneinander verschoben sind, wird die selbe Funktion für alle Terme verwendet. Die Dreiecksfunktion, die den Zugehörigkeitsfunktionen aller Terme zu Grunde liegt, wird entsprechend dem Term verschoben, der als Parameter an die Funktion übergeben wird.

```
FUNCTION gv_rauminhalt_fein (term IN VARCHAR2, x IN NUMBER)
```

```

RETURN NUMBER AS
  x2 NUMBER;
  r NUMBER;
BEGIN

  IF term = 'leer' THEN x2:=x;
  ELSIF term = 'viertelvoll' THEN x2:=x-25;
  ELSIF term = 'halbvoll' THEN x2:=x-50;
  ELSIF term = 'dreiviertelvoll' THEN x2:=x-75;
  ELSIF term = 'voll' THEN x2:=x-100;
  ELSE RETURN 0;
  END IF;

  IF x2 > 0 THEN x2:=25-x2; ELSE x2:=25+x2; END IF;
  r:=x2/25;
  IF r < 0 THEN RETURN 0; ELSE RETURN r; END IF;

END;

```

Kennt man die Zugehörigkeiten der Elementarwerte zu den Termen feiner Granularität, lassen sich anhand der auf Seite 46 angegebenen Zugehörigkeiten gemäß Gleichung 2.6 auf Seite 12 die Zugehörigkeiten zu den Termen grober Granularität berechnen. Da die Zugehörigkeiten der Elementarwerte zu den Termen grober Granularität eine sehr einfache Form haben (siehe Abbildung 4.9), ist hier die direkte Berechnung effizienter. Hierfür wurden die beiden Funktionen `gv_eher_voll` und `gv_eher_leer` implementiert. Die Zugehörigkeitsfunktionen der Terme *eher voll* und *eher leer* haben den gleichen Verlauf wie diejenigen der Terme *hoch* und *niedrig* bei den FCD- und Messstationsdaten. Daher sehen die beiden Funktionen `gv_eher_voll` und `gv_eher_leer` ähnlich aus wie die dort verwendeten Funktionen.

Art der transportierten Güter

Für die durch die NST/R-Klassifikation definierte Einteilung existiert im vorhandenen Schema bereits die Relation `gut`, welche jeder Güteridentifikationsnummer eine Güterhauptgruppe und eine Güterabteilung zuordnet. Dies stellt genau die Umsetzung der beiden Kontexte durch eine zusätzliche Relation dar.

Containergröße

Der Kontext des Attributs *Containergröße* wird durch die Relation `gv_k_containergroesse` umgesetzt. Die verwendeten Identifikatoren für die Äquivalenzklassen werden in der Relation `gv_k_containergroesse_b` beschrieben.

5.6.2 Anfragen unter Verwendung der Kontexte

Anhand der Güterverkehrsdaten soll nun noch gezeigt werden, wie eine Aggregation über unscharfe Klassen aussieht. Es soll die Entfernung festgestellt werden, die im Berichtsjahr 2001 jeweils insgesamt von leeren, viertelvoll, halbvoll, dreiviertelvoll bzw. voll beladenen

5 Umsetzung

Fahrzeugen zurückgelegt wurde. Da die Klassen hier wieder aus jeweils nur einem Term gebildet werden, entspricht die Klassenzugehörigkeit der Zugehörigkeit zum Term. Die von den Fahrzeugen jeder Klasse zurückgelegte Entfernung wird, wie in Abschnitt 2.4.2 beschrieben, als gewichtete Summe durch Multiplikation der Entfernung mit der Zugehörigkeit und anschließende Addition berechnet:

```
SELECT
SUM(entfernungkm*gv_rauminhalt_fein('leer', genrauminhalt))
AS leer,
SUM(entfernungkm*gv_rauminhalt_fein('viertelvoll', genrauminhalt))
AS viertelvoll,
SUM(entfernungkm*gv_rauminhalt_fein('halbvoll', genrauminhalt))
AS halbvoll,
SUM(entfernungkm*gv_rauminhalt_fein('dreiviertelvoll', genrauminhalt))
AS dreiviertelvoll,
SUM(entfernungkm*gv_rauminhalt_fein('voll', genrauminhalt))
AS voll
FROM ovidgv.fahrt_aggregiert WHERE berichtsjaehr = 2001;
```

5.7 Zusammenfassung

Im ersten Teil dieses Kapitels wurde allgemein beschrieben, wie die konkrete Umsetzung von Kontexten aussehen kann. Es existieren mehrere Möglichkeiten, Kontexte in einem relationalen Datenbanksystem zu repräsentieren: durch zusätzliche Attribute in der Relation, durch eine eigene Relation, durch Angabe von Intervallgrenzen und mittels Funktionen. Die Klassen können bei Verwendung von scharfen Kontexten einfach durch Angabe der Attribute bzw. Funktionen, welche die Kontexte beschreiben, in `where` bzw. `group by`-Klauseln gebildet werden. Für unscharfe Kontexte müssen die hierzu benötigten Operatoren durch Funktionen umgesetzt werden.

Der Rest des Kapitels beschrieb die Umsetzung der für die vorliegenden Verkehrsdaten in Kapitel 4 konzipierten Kontexte, sowie die Bildung von Klasseneinteilungen mit Hilfe dieser Kontexte. Anhand von Beispielen wurde gezeigt, wie die Kontexte in Anfragen verwendet werden können.

6 Bewertung und Ausblick

6.1 Kontexte und Aggregation

Im Vorfeld der eigentlichen Anwendung des Kontextmodells auf die Verkehrsdaten wurde geprüft, ob Kontexte mit Aggregierungsoperationen vereinbar sind. Dabei stellte sich heraus, dass die Klassen, die sich aus scharfen Kontexten ergeben, zur Gruppierung für die Aggregatbildung geeignet sind, da sie definitionsgemäß immer überlappungsfreie und vollständige Einteilungen darstellen. Scharfe Kontexte eignen sich also dazu, die Klassen zu definieren, für die Aggregate berechnet werden sollen.

Es wurde gezeigt, wie sich über Kontexte eine Hierarchie von Klassen definieren lässt, so dass Einteilungen unterschiedlicher Granularität verwendet werden können. Aggregiert man mit semiadditiven Aggregatfunktionen, lassen sich im Falle scharfer Einteilungen auf feiner Granularität vorberechnete Aggregate effizient wiederverwenden.

Das eben Gesagte gilt im Prinzip auch für unscharfe Kontexte, wobei hier einige Einschränkungen gemacht werden müssen. Zunächst mussten die ursprünglich nur für scharfe Einteilungen definierten Aggregatfunktionen auf unscharfe Einteilungen erweitert werden. Für die meisten gängigen Aggregatfunktionen gelang diese Erweiterung – die in dieser Arbeit vorgeschlagenen erweiterten Funktionen liefern Ergebnisse, die mit dem intuitiven Verständnis übereinstimmen und aussagekräftig sind. Allerdings stellte es sich heraus, dass auch Aggregatfunktionen existieren, die nicht auf unscharfe Einteilungen erweitert werden können.

Damit die für unscharfe Klassen berechneten Aggregate insgesamt betrachtet Sinn ergeben und schlüssig sind, müssen Einschränkungen für die Zugehörigkeitsfunktionen erlassen werden, die den Bedingungen „Vollständigkeit“ und „Überlappungsfreiheit“ im scharfen Fall entsprechen. Deswegen wurde ein Vorschlag gemacht, wie sich die beiden Bedingungen für unscharfe Einteilungen umformulieren lassen. Es zeigte sich bei der Konzeption, dass für die Verkehrsdaten Kontexte definiert werden konnten, die diesen Einschränkungen genügen. Daher ist hier die Aggregation auch mit unscharfen Kontexten vereinbar.

6.2 Umsetzung von Kontexten

Für die Umsetzung von Kontexten in einem relationalen Datenbanksystem wurden verschiedene Möglichkeiten erarbeitet, wie Kontexte im relationalen System repräsentiert werden können. Je nach der Art des Kontexts, der Größe des zu unterteilenden Wertebereichs und der Anzahl der Äquivalenzklassen bzw. Terme sind diese unterschiedlich gut zur Repräsentation des Kontexts geeignet. Für die einzelnen Kontexte, die für die Verkehrsdaten konzipiert wurden, konnte immer eine jeweils geeignete Methode gefunden werden.

Generell gilt, dass für die Berechnung der Zugehörigkeit zu unscharfen Klassen ein recht hoher Rechenaufwand erforderlich ist. Bei Daten, auf die meistens nur lesend zugegrif-

fen wird, könnte der Aufwand eingeschränkt werden, indem einmal berechnete Klassenzugehörigkeiten abgespeichert werden. Ebenfalls einen recht hohen Aufwand verursacht die Berechnung von Aggregaten für unscharfe Klassen, da hierfür die Zugehörigkeit benötigt wird. Gerade bei unscharfen Klassen können vorberechnete Aggregate jedoch im Allgemeinen nicht wiederverwendet werden, so dass grobgranulare Aggregate auch jedesmal wieder aus den Elementardaten berechnet werden müssen.

Es stellte sich heraus, dass Kontexte recht einfach in SQL-Anfragen verwendet werden können. Jedoch können sich, wie auch die Beispielanfragen zeigen, ziemlich umfangreiche SQL-Statements ergeben. Die Ursache hierfür liegt darin, dass SQL das Konzept der Kontexte nicht kennt und diese daher immer durch ein Attribut, durch Kombinationen von Attributen oder durch Funktionen ausgedrückt werden müssen. Abhilfe würde hier die Verwendung einer Anfragesprache schaffen, die explizit für die Verwendung von Kontexten in Anfragen konzipiert wurde, wie z.B. die an der Universität Fribourg entwickelte Anfragesprache fCQL ([MMWS03]), welche eine Erweiterung von SQL darstellt. Auch die Entwicklung einer Anfragesprache, in welche Aggregatfunktionen für unscharfe Klassen integriert sind, wäre sicher ein lohnendes Ziel.

6.3 Anwendung des Kontextmodells auf die Verkehrsdaten

Für die Floating Car Daten konnten mit Hilfe von Kontexten Verkehrszustände definiert werden. Somit konnte das Ziel erreicht werden, die detaillierten, mit Imperfektion behafteten Elementarwerte vor dem Benutzer zu verbergen und diesem statt dessen intuitiv verständliche Kategorien anzubieten. Gleichzeitig wurden hier aber auch die Grenzen des Kontextmodells deutlich. Eine flexiblere und sicherlich mit dem menschlichen Empfinden von Verkehrszuständen besser übereinstimmende Einteilung konnte nur über den Umweg der Einführung zusätzlicher Attribute und die anschließende Definition von Kontexten auf diesen Attributen erreicht werden. Es wäre daher lohnenswert zu erforschen, ob das Kontextmodell so erweitert werden kann, dass eine Klassenbildung möglich ist, die beliebig im Raum verlaufenden Trenngeraden entspricht.

Für die Simulationsdaten für Verkehrsmessstationen konnten ähnliche Kontexte wie für die Floating Car Daten definiert werden. Bei den Güterverkehrsdaten ließ sich durch Kontexte eine gröbere Einteilung definieren, auf Grundlage derer die bereits aggregierten Daten weiter aggregiert und somit auf gröbergranularer Ebene zusammengefasst werden können. Zusammenfassend lässt sich also sagen, dass das Kontextmodell in allen drei Fällen mit Erfolg auf die vorliegenden Verkehrsdaten angewendet werden konnte.

A Anhang

A.1 Übersicht über die umgesetzten Kontexte

A.1.1 Floating Car Daten

Attribut	Kontext umgesetzt durch	Name
Durchschnittsgeschwindigkeit	Funktion	fcd_hoch, fcd_niedrig
Standardabweichung der Geschwindigkeit	Funktion	fcd_hoch, fcd_niedrig
Stillstandsanteil	Funktion	fcd_hoch, fcd_niedrig
Zeit (Einteilung in Stunden, Tage, ...)	Funktion	trunc
Zeit (Einteilung in Wochentage und Stunden)	Attribute	<i>Tag, Stunde</i>
Zeit (aktuelle Stunde)	Intervallgrenze	—
Fahrzeugtyp	Relation	fcd_k_fahrzeugtyp
Abstand zu den Trenngeraden	Funktion	fcd_linksvon, fcd_rechtsvon

A.1.2 Simulationsdaten für Verkehrsmesstationen

Attribut	Kontext umgesetzt durch	Name
Geschwindigkeit	Funktion	messst_hoch, messst_niedrig
Stillstandsanteil	Funktion	messst_hoch, messst_niedrig
Zeit (Einteilung in Minuten, Stunden, ...)	Funktion	trunc
Zeit (aktuelles Intervall)	Intervallgrenze	—
Fahrzeugtyp	Relation	ovidsa01.fahrzeugtyp
Fahrzeugtyp (relevante Typen)	Relation	messst_k_fahrzeugtyp

A.1.3 Güterverkehrsdaten

Attribut	Kontext umgesetzt durch	Name
Art des Anhängers	where-Klausel	—
Beladeort	Relation	gv_k_ort
Entladeort	Relation	gv_k_ort
genutzter Rauminhalt (fein)	Funktion	gv_rauminhalt_fein
genutzter Rauminhalt (grob)	Funktion	gv_eher_voll, gv_eher_leer
Art der transportierten Güter	Relation	ovidgv.gut
Containergröße	Relation	gv_k_containergroesse

A.2 Typen der Attribute

A.2.1 Floating Car Daten

Attribut	Typ
Durchschnittsgeschwindigkeit	value per unit
Standardabweichung der Geschwindigkeit	value per unit
minimale Geschwindigkeit	value per unit
maximale Geschwindigkeit	value per unit
durchschnittliche positive Beschleunigung	value per unit
durchschnittliche negative Beschleunigung	value per unit
relativer Stillstandsanteil	value per unit

A.2.2 Simulationsdaten für Verkehrsmessstationen

Attribut	Typ
Geschwindigkeit	value per unit
Beschleunigung	value per unit
Anzahl der Personen im Fahrzeug	stock
Länge des Fahrzeugs	value per unit

A.2.3 Güterverkehrsdaten

Attribut	Typ
zurückgelegte Entfernung	flow
Anzahl der Einzelstrecken	flow
transportiertes Gewicht	flow
zulässiges Gesamtgewicht	flow
Nutzlast	flow

A Anhang

Abbildungsverzeichnis

2.1	Durch Kontexte definierte Hierarchie	6
2.2	Linguistische Variable	8
2.3	Durch linguistische Variablen definierte Hierarchie	11
2.4	Beispiel für die Berechnung der Zugehörigkeit zu einem Term höherer Ebene	12
4.1	Einteilung in Klassen durch Kontexte	34
4.2	Einteilung in Klassen durch Trenngeraden	34
4.3	Zugehörigkeitsfunktionen der linguistischen Variable <i>Geschwindigkeit</i> . . .	36
4.4	Anordnung der Kontexte des Attributs <i>Zeit</i> in parallelen Hierarchien	38
4.5	Verkehrszustände auf Autobahnen, definiert durch Kontexte der Attribute <i>Durchschnittsgeschwindigkeit</i> und <i>Standardabweichung</i>	39
4.6	Einteilung der Linkbefahrungen in Klassen nach <i>Zeit</i> und <i>LinkId</i>	40
4.7	Verkehrszustände auf Autobahnen, definiert durch Trenngeraden	42
4.8	Zugehörigkeitsfunktionen für den genutzten Rauminhalt, feinere Einteilung	46
4.9	Zugehörigkeitsfunktionen für den genutzten Rauminhalt, gröbere Einteilung	47

Abbildungsverzeichnis

Tabellenverzeichnis

2.1	Beispiel für Klassifikation im multidimensionalen Datenmodell	5
2.2	Gegenüberstellung der korrespondierenden Begriffe für scharfe und unscharfe Einteilungen	9
2.3	Beispiel für die Berechnung der Zugehörigkeit zu unscharfen Klassen	10
2.4	Beispiel für Normierung der Zugehörigkeit zu unscharfen Klassen	18
5.1	Beispiel für eine Anfrage unter Verwendung des Kontexts für die Durchschnittsgeschwindigkeit auf Autobahnen	54
5.2	Relation für den Kontext des Attributs <i>Fahrzeugtyp</i>	55
5.3	Beschreibung der Äquivalenzklassen aus Tabelle 5.2	55
5.4	Beispiel für die Klassifikation von Strecken	56
5.5	Relation zur Unterscheidung von relevanten und nicht relevanten Fahrzeugen	63

Tabellenverzeichnis

Literaturverzeichnis

- [BG01] ANDREAS BAUER UND HOLGER GÜNZEL (Hrsg.) *Data Warehouse Systeme*. dpunkt-Verlag, 2001.
- [DPJ03] CURTIS E. DYRESON, TORBEN BACH PEDERSEN UND CHRISTIAN S. JENSEN. Incomplete Information in Multidimensional Databases. In: MAURIZIO RAFANELLI (Hrsg.) *Multidimensional Databases: Problems and Solutions*, S. 282–309. Idea Group, 2003.
- [FD03] LING FENG UND THARAM S. DILLON. Using Fuzzy Linguistic Representations to Provide Explanatory Semantics for Data Warehouses. In: *IEEE Transactions on Knowledge and Data Engineering*, Bd. 15(1): S. 86–102, 2003.
- [KBB01] Statistische Mitteilungen Güterkraftverkehr deutscher Lastkraftfahrzeuge. Kraftfahrt-Bundesamt und Bundesamt für Güterverkehr, 2001.
- [Koo04] ERIK ALEXANDER KOOP. *Datenbankunterstützung für imperfekte Daten im Verkehrsumfeld*. Diplomarbeit, Universität Karlsruhe (TH), 2004.
- [Leh03] WOLFGANG LEHNER. *Datenbanktechnologie für Data-Warehouse-Systeme: Konzepte und Methoden*. dpunkt-Verlag, 2003.
- [LS97] HANS-JOACHIM LENZ UND ARIE SHOSHANI. Summarizability in OLAP and Statistical Data Bases. In: *Statistical and Scientific Database Management*, S. 132–143. 1997.
- [MMWS03] ANDREAS MEIER, CHRISTIAN MEZGER, NICOLAS WERRO UND GÜNTER SCHINDLER. Zur unscharfen Klassifikation von Datenbanken mit fCQL. In: *Proceedings of the GI-Workshop LLWA Lernen, Lernen, Wissen, Adaptivität*, S. 151–158. Karlsruhe, Germany, 2003.
- [MSSV01] ANDREAS MEIER, CHRISTIAN SAVARY, GÜNTER SCHINDLER UND YAUHENI VERYHA. *Database Schema with Fuzzy Classification and Classification Query Language*. Tech. Rep. 01-06, Department of Informatics, University of Fribourg, 2001.
- [RB91] ELKE A. RUNDENSTEINER UND LUBOMIR BIC. Evaluating Aggregates in Possibilistic Relational Databases. In: *Data & Knowledge Engineering*, Bd. 7: S. 239–267, 1991.
- [San04] JAN SANDBERGER. *Konzeption und Evaluation eines Data Warehouse zur Analyse von Daten im Verkehrsbereich*. Studienarbeit, Universität Karlsruhe (TH), 2004.

Literaturverzeichnis

- [Sch98] GÜNTER SCHINDLER. *Fuzzy-Datenanalyse durch kontextbasierte Datenbankabfragen*. Dissertation, Rheinisch-Westfälische Technische Hochschule Aachen, 1998.
- [SW03] FRANK STEINERT UND RALF WILLENBROCK. *Dokumentation zum Projekt „Bereitstellung und Consulting Floating-Car-Daten“, gedas Deutschland GmbH, Version 1.0*, 2003.
- [Zim92] HANS J. ZIMMERMANN. *Fuzzy Set Theory – and Its Applications, Sec., rev. edition*. Kluwer, Boston; Dordrecht, London, 1992.