LEUPHANA
UNIVERSITÄT LÜNEBURG

Strukturanalogien in Datenmodellen

Riebesehl, Dieter

Publication date:
2006

Document Version
Verlags-PDF (auch: Version of Record)

Link to publication

Citation for pulished version (APA):
Riebesehl, D. (2006). *Strukturanalogien in Datenmodellen*. (Final; Band 16, Nr. 3). Universität Lüneburg, Fakultät III, Umwelt und Technik.

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

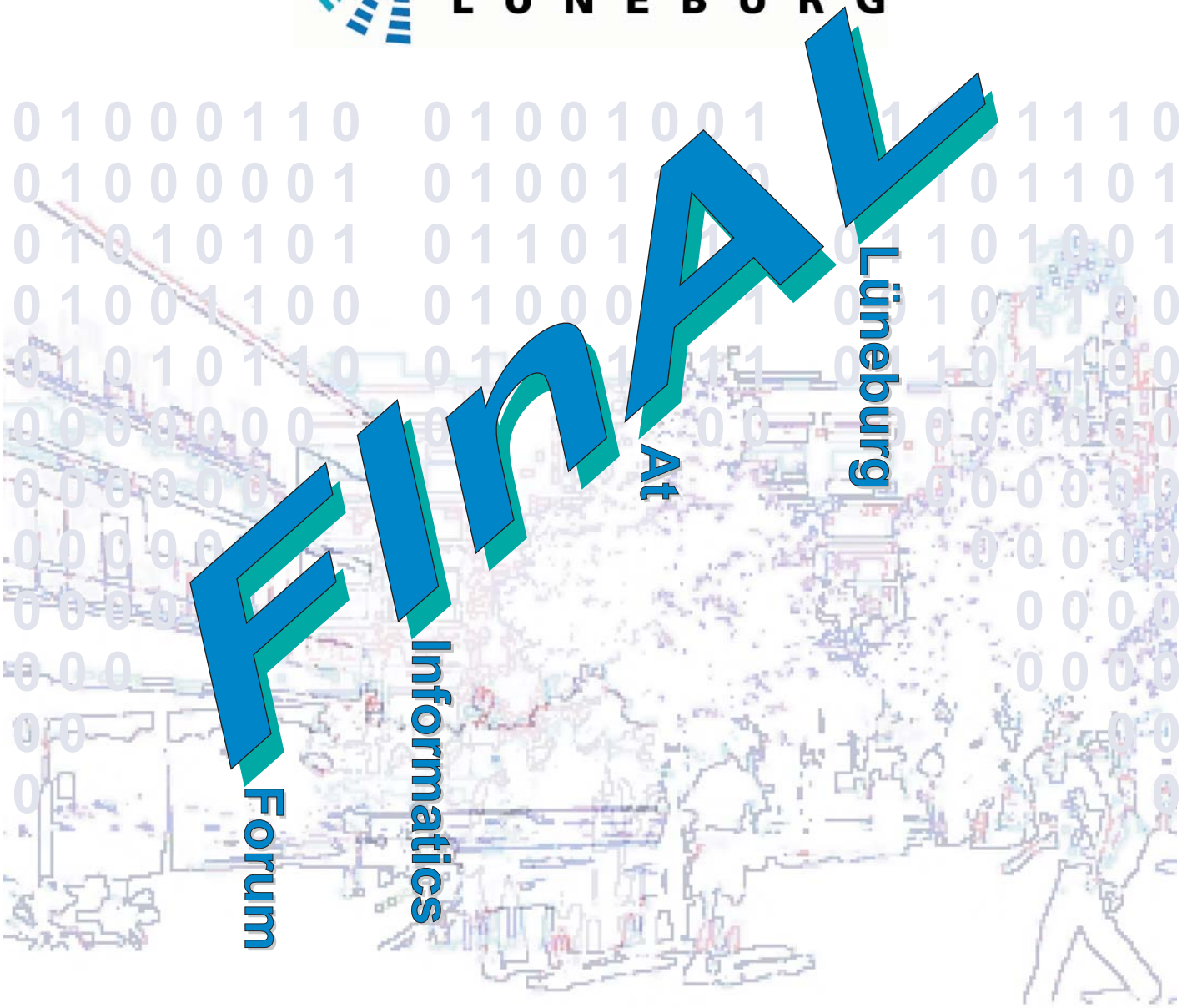
- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal ?

Take down policy
If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

Download date: 19. Apr.. 2024

16. Jahrgang, Heft 3, Dezember 2006, ISSN 0939-8821

Universität Lüneburg



Strukturanalogien in Datenmodellen

Dieter Riebesehl

Information Science: Technical Reports and Working Papers
Hrsg: Hinrich E. G. Bonin
Volgershall 1, D-21339 Lüneburg
Phone: xx49.4131.6775175 Fax: xx49.4131.6775140

In der Reihe FINAL sind bisher erschienen:

1. Jahrgang 1991:

1. Hinrich E. G. Bonin; Softwaretechnik, Heft 1, 1991 (ersetzt durch Heft 2, 1992).
2. Hinrich E. G. Bonin (Herausgeber); Konturen der Verwaltungsinformatik, Heft 2, 1991 (überarbeitet und erschienen im Wissenschaftsverlag, Bibliographisches Institut & F. A. Brockhaus AG, Mannheim 1992, ISBN 3-411-15671-6).

2. Jahrgang 1992:

1. Hinrich E. G. Bonin; Produktionshilfen zur Softwaretechnik --- Computer-Aided Software Engineering --- CASE, Materialien zum Seminar 1992, Heft 1, 1992.
2. Hinrich E. G. Bonin; Arbeitstechniken für die Softwareentwicklung, Heft 2, 1992 (3. überarbeitete Auflage Februar 1994), PDF-Format (Passwort: arbeiten).
3. Hinrich E. G. Bonin; Object-Orientedness --- a New Boxologie, Heft 3, 1992.
4. Hinrich E. G. Bonin; Objekt-orientierte Analyse, Entwurf und Programmierung, Materialien zum Seminar 1992, Heft 4, 1992.
5. Hinrich E. G. Bonin; Kooperative Produktion von Dokumenten, Materialien zum Seminar 1992, Heft 5, 1992.

3. Jahrgang 1993:

1. Hinrich E. G. Bonin; Systems Engineering in Public Administration, Proceedings IFIP TC8/ WG8.5: Governmental and Municipal Information Systems, March 3--5, 1993, Lüneburg, Heft 1, 1993 (überarbeitet und erschienen bei North-Holland, IFIP Transactions A-36, ISSN 0926-5473).
2. Antje Binder, Ralf Linhart, Jürgen Schultz, Frank Sperschneider, Thomas True, Bernd Willenbockel; COTEXT --- ein Prototyp für die kooperative Produktion von Dokumenten, 19. März 1993, Heft 2, 1993.
3. Gareth Harries; An Introduction to Artificial Intelligence, April 1993, Heft 3, 1993.
4. Jens Benecke, Jürgen Grothmann, Mark Hilmer, Manfred Hölzen, Heiko Köster, Peter Mattfeld, Andre Peters, Harald Weiss; ConFusion --- Das Produkt des AWÖ-Projektes 1992/93, 1. August 1993, Heft 4, 1993.
5. Hinrich E. G. Bonin; The Joy of Computer Science --- Skript zur Vorlesung EDV ---, September 1993, Heft 5, 1993 (4. ergänzte Auflage März 1995).
6. Hans-Joachim Blanke; UNIX to UNIX Copy --- Interactive application for installation and configuration of UUCP ---, Oktober 1993, Heft 6, 1993.

4. Jahrgang 1994:

1. Andre Peters, Harald Weiss; COMO 1.0 --- Programmierumgebung für die Sprache COBOL --- Benutzerhandbuch, Februar 1994, Heft 1, 1994.
2. Manfred Hölzen; UNIX-Mail --- Schnelleinstieg und Handbuch ---, März 1994, Heft 2, 1994.
3. Norbert Kröger, Roland Seen; EBrain --- Documentation of the 1994 AWÖ-Project Prototype ---, June 11, 1994, Heft 3, 1994.
4. Dirk Mayer, Rainer Saalfeld; ADLATUS --- Documentation of the 1994 AWÖ-Project Prototype -- -, July 26, 1994, Heft 4, 1994.
5. Ulrich Hoffmann; Datenverarbeitungssystem 1, September 1994, Heft 5, 1994. (2. überarbeitete Auflage Dezember 1994)
6. Karl Goede; EDV-gestützte Kommunikation und Hochschulorganisation, Oktober 1994, Heft 6 (Teil 1), 1994.

7. Ulrich Hoffmann; Zur Situation der Informatik, Oktober 1994, Heft 6 (Teil 2), 1994.

5. Jahrgang 1995:

1. Horst Meyer-Wachsmuth; Systemprogrammierung 1, Januar 1995, Heft 1, 1995.
2. Ulrich Hoffmann; Datenverarbeitungssystem 2, Februar 1995, Heft 2, 1995.
3. Michael Guder / Kersten Kalischefski / Jörg Meier / Ralf Stöver / Cheikh Zeine; OFFICE-LINK --- Das Produkt des AWÖ-Projektes 1994/95, März 1995, Heft 3, 1995.
4. Dieter Riebesehl; Lineare Optimierung und Operations Research, März 1995, Heft 4, 1995.
5. Jürgen Mattern / Mark Hilmer; Sicherheitsrahmen einer UTM-Anwendung, April 1995, Heft 5, 1995.
6. Hinrich E. G. Bonin; Publizieren im World-Wide Web --- HyperText Markup Language und die Kunst der Programmierung ---, Mai 1995, Heft 6, 1995
7. Dieter Riebesehl; Einführung in Grundlagen der theoretischen Informatik, Juli 1995, Heft 7, 1995
8. Jürgen Jacobs; Anwendungsprogrammierung mit Embedded-SQL, August 1995, Heft 8, 1995
9. Ulrich Hoffmann; Systemnahe Programmierung, September 1995, Heft 9, 1995 (ersetzt durch Heft 1, 1999)
10. Klaus Lindner; Neuere statistische Ergebnisse, Dezember 1995, Heft 10, 1995

6. Jahrgang 1996:

1. Jürgen Jacobs / Dieter Riebesehl; Computergestütztes Repetitorium der Elementarmathematik, Februar 1996, Heft 1, 1996
2. Hinrich E. G. Bonin; "Schlanker Staat" & Informatik, März 1996, Heft 2, 1996
3. Jürgen Jacobs; Datenmodellierung mit dem Entity-Relationship-Ansatz, Mai 1996, Heft 3, 1996
4. Ulrich Hoffmann; Systemnahe Programmierung, (2. überarbeitete Auflage von Heft 9, 1995), September 1996, Heft 4, 1996 (ersetzt durch Heft 1, 1999).
5. Dieter Riebesehl; Prolog und relationale Datenbanken als Grundlagen zur Implementierung einer NF2-Datenbank (Sommer 1995), November 1996, Heft 5, 1996

7. Jahrgang 1997:

1. Jan Binge, Hinrich E. G. Bonin, Volker Neumann, Ingo Stadtsholte, Jürgen Utz; Intranet-/Internet- Technologie für die Öffentliche Verwaltung --- Das AWÖ-Projekt im WS96/97 --- (Anwendungen in der Öffentlichen Verwaltung), Februar 1997, Heft 1, 1997
2. Hinrich E. G. Bonin; Auswirkungen des Java-Konzeptes für Verwaltungen, FTVI'97, Oktober 1997, Heft 2, 1997

8. Jahrgang 1998:

1. Hinrich E. G. Bonin; Der Java-Coach, Oktober 1998, Heft 1, 1998 (CD-ROM, PDF-Format; aktuelle Fassung)

2. Hinrich E. G. Bonin (Hrsg.); Anwendungsentwicklung WS 1997/98 --- Programmierbeispiele in COBOL & Java mit Oracle, Dokumentation in HTML und tcl/tk, September 1998, Heft 2, 1998 (CD-ROM)

3. Hinrich E. G. Bonin (Hrsg); Anwendungsentwicklung SS 1998 --- Innovator, SNiFF+, Java, Tools, Oktober 1998, Heft 3, 1998 (CD-ROM)

4. Hinrich E. G. Bonin (Hrsg); Anwendungsentwicklung WS 1998 --- Innovator, SNiFF+, Java, Mail und andere Tools, November 1998, Heft 4, 1998 (CD-ROM)

5. Hinrich E. G. Bonin; Persistente Objekte --- Der Elchtest für ein Java-Programm, Dezember 1998, Heft 5, 1998 (CD-ROM)

9. Jahrgang 1999:

1. Ulrich Hoffmann; Systemnahe Programmierung (3. überarbeitete Auflage von Heft 9, 1995), Juli 1999, Heft 1, 1999 (CD-ROM und Papierform), Postscript-Format, zip-Postscript-Format, PDF-Format und zip-PDF-Format.

10. Jahrgang 2000:

1. Hinrich E. G. Bonin; Citizen Relationship Management, September 2000, Heft 1, 2000 (CD-ROM und Papierform), PDF-Format --- Password: `arbeiten`

2. Hinrich E. G. Bonin; WI>DATA --- Eine Einführung in die Wirtschaftsinformatik auf der Basis der Web_Technologie, September 2000, Heft 2, 2000 (CD-ROM und Papierform), PDF-Format --- Password: `arbeiten`

3. Ulrich Hoffmann; Angewandte Komplexitätstheorie, November 2000, Heft 3, 2000 (CD-ROM und Papierform), PDF-Format

4. Hinrich E. G. Bonin; Der kleine XMLer, Dezember 2000, Heft 4, 2000 (CD-ROM und Papierform), PDF-Format, aktuelle Fassung --- Password: `arbeiten`

11. Jahrgang 2001:

1. Hinrich E. G. Bonin (Hrsg.): 4. SAP-Anwenderforum der FHNON, März 2001, (CD-ROM und Papierform), Downloads & Videos.

2. J. Jacobs / G. Weinrich; Bonitätsklassifikation kleiner Unternehmen mit multivariater linear Diskriminanzanalyse und Neuronalen Netzen; Mai 2001, Heft 2, 2001, (CD-ROM und Papierform), PDF-Format und MS Word DOC-Format --- Password: `arbeiten`

3. K. Lindner; Simultanttestprozedur für globale Nullhypothesen bei beliebiger Abhängigkeitsstruktur der Einzeltests, September 2001, Heft 3, 2001 (CD-ROM und Papierform).

12. Jahrgang 2002:

1. Hinrich E. G. Bonin: Aspect-Oriented Software Development. März 2002, Heft 1, 2002 (CD-ROM und Papierform), PDF-Format --- Password: `arbeiten`.

2. Hinrich E. G. Bonin: WAP & WML --- Das Projekt Jagdzeit ---. April 2002, Heft 2, 2002 (CD-ROM und Papierform), PDF-Format --- Password: `arbeiten`.

3. Ulrich Hoffmann: Ausgewählte Kapitel der Theoretischen Informatik (CD-ROM und Papierform), PDF-Format.

4. Jürgen Jacobs / Dieter Riebesehl; Computergestütztes Repetitorium der Elementarmathematik, September 2002, Heft 4, 2002 (CD-ROM und Papierform), PDF-Format.

5. Verschiedene Referenten; 3. Praxisforum "Systemintegration", 18.10.2002, Oktober 2002, Heft 5, 2002 (CD-ROM und Papierform), Praxisforum.html (Web-Site).

13. Jahrgang 2003:

1. Ulrich Hoffmann; Ausgewählte Kapitel der Theoretischen Informatik; Heft 1, 2003, (CD-ROM und Papierform) PDF-Format.

2. Dieter Riebesehl; Mathematik 1, Heft 2, 2003, (CD-ROM und Papierform) PDF-Format.

3. Ulrich Hoffmann; Mathematik 1, Heft 3, 2003, (CD-ROM und Papierform) PDF-Format und Übungen

4. Verschiedene Autoren; Zukunft von Verwaltung und Informatik, Festschrift für Heinrich Reinermann, Heft 4, 2003, CD-ROM und Papierform) PDF-Format

14. Jahrgang 2004:

1. Jürgen Jacobs; Multilayer Neural Networks; Heft 1, 2004, (CD-ROM und Papierform) PDF-Format.

15. Jahrgang 2005:

1. Ulrich Hoffmann; Mathematik für Wirtschaftsinformatiker; Heft 1, 2005, (CD-ROM und Papierform) PDF-Format.
2. Ulrich Hoffmann; Übungen & Lösungen zur Mathematik für Wirtschaftsinformatiker; Heft 1, 2005, (CD-ROM und Papierform) PDF-Format.

3. Ulrich Hoffmann; Datenstrukturen & Algorithmen; Heft 2, 2005, (CD-ROM und Papierform) PDF-Format.

16. Jahrgang 2006:

1. Hinrich E. G. Bonin: Systemanalyse für Softwaresysteme; Heft 1, August 2006, (CD-ROM und Papierform), PDF-Format.

2. Hinrich E. G. Bonin: Faszination Programmierung; Heft 2, August 2006, (CD-ROM und Papierform), PDF-Format.

3. Dieter Riebesehl; Strukturanalogien in Datenmodellen, Heft 3, Dezember 2006, (CD-ROM und Papierform) PDF-Format.

Herausgeber:
Prof. Dr. Dipl.-Ing. Dipl.-Wirtsch.-Ing. Hinrich E. G. Bonin
Universität Lüneburg, Volgershall 1, D-21339 Lüneburg, Germany
email: bonin@uni-lueneburg.de

Verlag:
Eigenverlag (Fotographische Vervielfältigung), Universität Lüneburg (vormals Fachhochschule Nordostniedersachsen)

Erscheinungsweise:
ca. 4 Hefte pro Jahr.
Für unverlangt eingesendete Manuskripte wird nicht gehaftet. Sie sind aber willkommen.

Copyright:
All rights, including translation into other languages reserved by the authors. No part of this report may be reproduced or used in any form or by any means --- graphic, electronic, or mechanical, including photocopying, recording, taping, or information and retrieval systems --- without written permission from the authors, except for noncommercial, educational use, including classroom teaching purposes.
Copyright Bonin Apr-1995,..., Aug-2006 all rights reserved

Strukturanalogien in Datenmodellen

Dieter Riebesehl

28. November 2006

Kurzfassung

Datenmodelle sind ein zentraler Bestandteil betrieblicher Informationssysteme. In der Praxis eingesetzte Datenmodelle erreichen oft eine erhebliche Komplexität. Eine Möglichkeit zur Reduktion der Komplexität besteht darin, gleiche oder ähnliche Teilstrukturen zu identifizieren. Durch Ausnutzung solcher Strukturanalogien kann dann das Datenmodell vereinheitlicht und vereinfacht werden. Ziel dieses Aufsatzes ist es, die bestehenden Ansätze zur Definition, Identifikation und Ausnutzung von Strukturanalogien in Datenmodellen weiterzuentwickeln.

Inhaltsverzeichnis

1 Einleitung **3**

2 Strukturanalogien **3**

3 Ähnlichkeitswerte **4**

3.1 Definition für verschiedene Modelltypen 5

3.1.1 PERM 5

3.1.2 SERM 8

3.1.3 ERM 9

3.2 Eigenschaften des Ähnlichkeitswertes 14

3.2.1 Rang 14

3.2.2 Generalisierung 15
3.3 Ein alternatives Ähnlichkeitsmaß 16
3.3.1 Probleme bei der Nutzung von Strukturidentitäten . . 16
3.3.2 Ein strukturiertes Ähnlichkeitsmaß 17
3.3.3 Anwendung: intermediäre Relationen 19
3.3.4 Strukturanalogien zwischen entfernten Teilen eines Datenmodells 20
3.4 Anwendung auf Datenmodelle 20
3.4.1 Bedarfsplanung, Zeit- und Kapazitätsplanung 21
3.4.2 CAM 22
3.4.3 Beschaffungslogistik 22
3.4.4 Vertriebsabwicklung 22
3.4.5 Gesamtmodell 27
3.4.6 Anwendung des strukturierten Ähnlichkeitsmaßes . . . 28
3.5 Zusammenfassung 32

4 Clusteranalyse **33**
4.1 Metriken 36
4.2 Ein anderes Ähnlichkeitsmaß 38
4.3 Anwendung auf Datenmodelle 39
4.3.1 Bedarfsplanung, Zeit- und Kapazitätsplanung 39
4.3.2 CAM 43
4.3.3 Beschaffungslogistik 45
4.3.4 Vertriebsabwicklung 47
4.3.5 Gesamtmodell 49
4.4 Bestimmung der optimalen Anzahl von Clustern 53
4.5 Zuordnung der Entitäten zu Clustern und Klassen 58
4.6 Zusammenfassung 60

5 Implementierung **61**

A Dendogramme für Bedarfsplanung etc. **79**

B Zuordnung Teilmodelle - Klassen 86

1 Einleitung

Unter Strukturanalogien zwischen Datenmodellen werden allgemein Ähnlichkeiten zwischen verschiedenen Teilen eines Modells verstanden. Es sind verschiedene Möglichkeiten vorstellbar, solche Analogien zu definieren. Diese werden für unterschiedliche Modellierungsmethoden verschieden ausfallen. Datenmodelle werden im Allgemeinen grafisch dargestellt, deshalb können Analogien in Form von isomorphen oder homomorphen Teilgraphen vorliegen. Man kann auch von den Informationen ausgehen, die über ein Datenmodell in einem *data dictionary* abgelegt werden, dann können Analogien zwischen Objekten des *data dictionary* vorliegen, also typischerweise zwischen Entitäten und/oder Relationen. Ein Vorschlag, Analogien zwischen Relationen durch eine Maßzahl zu charakterisieren, findet sich in [FE05]. Als Wert für die Ähnlichkeit zweier Relationen wird die normierte symmetrische Differenz der Entitätenmengen, auf denen die Relationen definiert sind, genommen.

In dieser Arbeit soll dieser Ansatz weiter verfolgt werden. Es werden verschiedene Modellierungsvarianten und die daraus resultierenden Unterschiede betrachtet. Das Ähnlichkeitsmaß wird jeweils entsprechend umdefiniert, weitere Möglichkeiten, ein Maß zu definieren, werden vorgestellt. Mit der Möglichkeit, die Relationen eines Datenmodells in Clustern zusammenfassen, wird ein Weg aufgezeigt, das Ähnlichkeitsmaß zu interpretieren und für eine verbesserte Modellierung zu nutzen.

Alle vorgestellten Verfahren werden auf mehrere Datenmodelle der betrieblichen Praxis angewendet, welche durchweg aus [SC97] entnommen sind.

2 Strukturanalogien

Grundsätzlich kann die Identifikation von Strukturanalogien auf zwei Wegen erfolgen:

1. Die grafische Darstellung der Datenmodells kann – per Augenschein oder durch graphentheoretische Methoden – auf Analogien hin untersucht werden, oder

2. aus einer abstrakten Representation des Datenmodells – etwa im *data dictionary* – können mittels eines Algorithmus numerische Maßzahlen für das Vorliegen von Analogien gewonnen werden.

Hier soll hauptsächlich der zweite Weg verfolgt werden. Allerdings läßt sich zeigen, dass durch eine leichte Modifikation des verwendeten Algorithmus brauchbare Schritte in Richtung auf die erste Vorgehensweise getan werden können.

3 Ähnlichkeitswerte

Es soll zunächst einmal die Definition des Ähnlichkeitswertes aus [FE05] wiederholt werden. Dazu sei ein Datenmodell mit Entitäten und Relationen gegeben, $\mathcal{R}$ sei die Menge der Relationen, und $\mathcal{E}$ die Menge der Entitäten. Die Behandlung von als Entitäten reinterpretierten Relationen sei zunächst zurückgestellt. Dann gibt es zu einer Relation $R \in \mathcal{R}$ die Menge $\hat{E}(R) \subseteq \mathcal{E}$ der Entitäten, über denen R definiert ist. $\hat{E}(R)$ ist i.A. eine Multimenge, in der Elemente mit Vielfachheit (mehrfach) vorkommen können.

Bezeichnet man für eine Multimenge $\hat{M}$ und ein Element $a \in \hat{M}$ mit $r_{\hat{M}}(a)$ die Vielfachheit von a in $\hat{M}$, dann kann man Durchschnitt und Vereinigung von Multimengen wie folgt definieren:

$$\begin{aligned} r_{\hat{M}\cap\hat{N}}(a) &:= \min(r_{\hat{M}}(a), r_{\hat{N}}(a)), \\ r_{\hat{M}\cup\hat{N}}(a) &:= \max(r_{\hat{M}}(a), r_{\hat{N}}(a)). \end{aligned}$$

Definiert man die Mächtigkeit für eine Multimenge durch

$$|\hat{M}| := \sum_{a\in\hat{M}} r_{\hat{M}}(a),$$

dann ist der Ähnlichkeitswert zweier Relationen einfach gegeben durch[1]

$$a(R_1, R_2) = \frac{|\hat{E}(R_1) \cap \hat{E}(R_2)|}{|\hat{E}(R_1) \cup \hat{E}(R_2)|}.$$

Bei der Definition des Ähnlichkeitswertes ist natürlich die Definition von $\hat{E}(R)$ entscheidend, und diese wiederum hängt vom Typ des verwendeten Datenmodells ab. In den folgenden Abschnitten sollen drei davon näher dargestellt werden:

[1] Abweichend von [FE05] wird der Ähnlichkeitswert mit a statt d bezeichnet, da noch eine auf a basierende Metrik d eingeführt werden wird.

ERM: Das Standard-ERM nach Chen ([CH76]), welches nur Entitäten, dargestellt durch Rechtecke, und Relationen, dargestellt durch Rauten, unterscheidet und insbesondere keine als Entitäten reinterpretierten Relationen kennt.

PERM: Das erweiterte ERM nach Loos ([LO97]), welches reinterpretierte Relationen kennt. Es enthält noch weitere Modellierungsmöglichkeiten, auf die hier aber nicht eingegangen wird.

SERM: Das strukturierte ERM nach Sinz ([SI93]), welches zusätzlich eine hierarchische Anordnung der Relationen vorsieht.

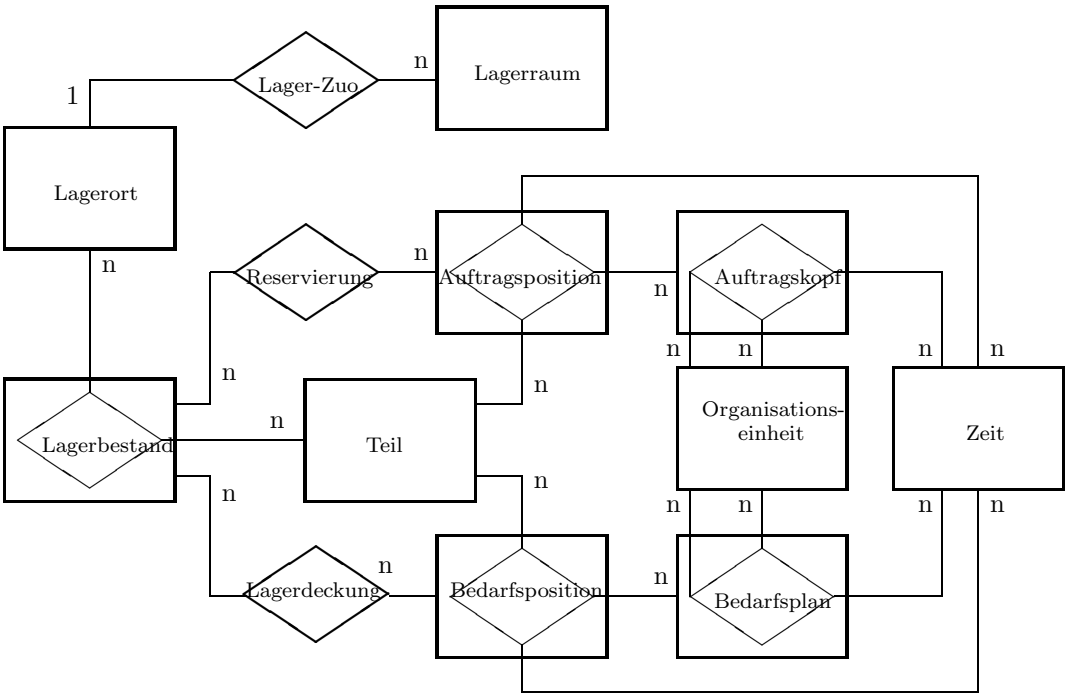
Die Verwendung der Abkürzung ERM für ein Modell ohne reinterpretierte Relationen entspricht nicht unbedingt dem Standard, wird aber von manchen Autoren von in die Modellierung einführender Literatur und auch in mehreren Modellierungswerkzeugen so genutzt, siehe z.B. Jarosch in [JA02].

3.1 Definition für verschiedene Modelltypen

In diesem Abschnitt sollen die Definitionen des Ähnlichkeitswertes für die drei Modellierungsvarianten gegeben und diskutiert werden. Zur Veranschaulichung und zum Vergleich der Auswirkungen wird ein Ausschnitt aus dem Referenzdatenmodell zur Bedarfsplanung nach Scheer ([SC97], S. 176, Abb. B.I.66) in allen Varianten dargestellt. Der Übergang von PERM zu SERM ist schon bei Scheer dargestellt. Die Vorgehensweise bei der Umwandlung eines PER-Modells in ein ER-Modell ist wegen der unterschiedlichen Semantik nicht von vornherein klar, und es sind verschiedene Möglichkeiten denkbar. Da sich die Aufgabe einer solchen Umwandlung in der Praxis kaum stellt, soll hier keine vollständige Übergangsvorschrift gegeben werden. Einge Beispiele werden auf Seite 9 gegeben.

3.1.1 PERM

Da der Ähnlichkeitskoeffizient für das PERM entwickelt wurde, wird hier mit der Darstellung des Modellausschnitts im PERM begonnen.



Definition von $\hat{E}(R)$

Die Mengen $\mathcal{E}$ und $\mathcal{R}$ haben hier einen nichtleeren Durchschnitt bestehend aus den Relationen, die als Entitäten reinterpretiert werden.

Die Bestimmung der Mengen $\hat{E}(R)$ geschieht rekursiv:

- Für $E \in \mathcal{E} \setminus \mathcal{R}$ ist
$$\hat{E}(E) := \{E\}.$$
- Für $R = E_1 \times E_2 \times \cdots \times E_n \in \mathcal{R}$ ist
$$\hat{E}(R) := \bigoplus_{1}^{n} \hat{E}(E_i).$$

Dabei ist die Summe zweier Multimengen definiert durch

$$r_{\hat{M} \oplus \hat{N}}(a) := r_{\hat{M}}(a) + r_{\hat{N}}(a).$$

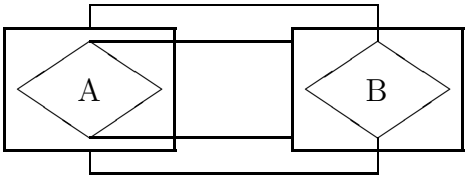
Wegen des Fehlens rekursiver Bezüge muss die Rekursion enden, siehe dazu auch den nächsten Abschnitt.

Aus den damit berechneten Ähnlichkeitskoeffizienten ergeben sich für den Modellausschnitt folgende Gruppen strukturidentischer Relationen, also solcher mit Ähnlichkeitswert 1:

```
Auftragskopf = Bedarfsplan
Auftragsposition = Bedarfsposition
Lagerdeckung = Reservierung
```

Die Modellierung im PERM gestattet es, durch geschickte Anordnung der Objekte im Diagramm diese Analogien direkt über Symmetrien zu erkennen - so wie im obigen Bild.

Unendliche Rekursionen bei PERM Die Modellierungsmethode PERM lässt Datenmodelle zu, bei denen reinterpretierte Relationen sich rekursiv aufeinander beziehen. Ein einfaches Beispiel sieht so aus:



Ein solches Modell kann durchaus implementiert werden, beispielsweise durch folgende SQL-Anweisungen:

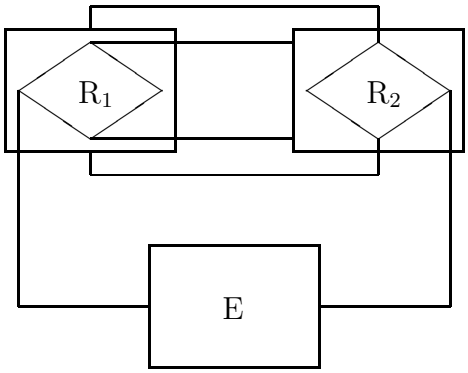
```
create table a
  (id number primary key,
   b1 number foreign key references b(id),
   b2 number foreign key references b(id));

create table b
  (id number primary key,
   a1 number foreign key references a(id),
   a2 number foreign key references a(id));
```

Es kann prinzipiell auch in der Praxis auftreten. So könnten Binärbäume implementiert werden, wenn man jeden Baumknoten `A` als Aggregation der – maximal zwei – von ihm ausgehenden Kanten darstellt, und jede Kante `B` als Aggregation der beiden Endknoten.

Aus Symmetriegründen sind `A` und `B` strukturäquivalent. Der Ähnlichkeitswert ist jedoch nicht definiert – unter anderem deshalb, weil es keine Entitäten gibt, auf denen die Relationen `A` und `B` definiert sind.

Die folgende Variante führt bei der Berechnung der Entitäten, auf denen z.B. $\mathtt{R}_1$ definiert ist, auf einen unendlichen Regress:

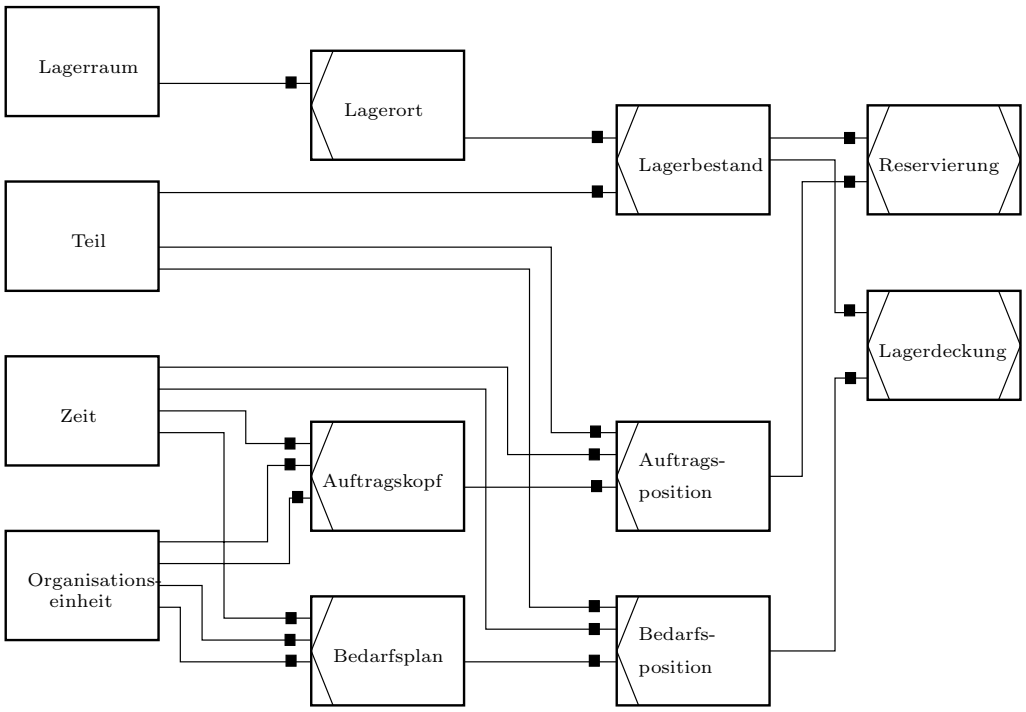


Es ist ja

$$R_1 \subseteq R_2 \times R_2 \times E \subseteq (R_1 \times R_1 \times E) \times (R_1 \times R_1 \times E) \times E \subseteq \ldots$$

Man muss also im PERM solche Konstrukte explizit ausschließen, d.h. rekursive Bezüge von Relationen auf sich selbst, auch indirekt, sind nicht erlaubt. Dann lässt sich PERM aber im Prinzip auf SERM reduzieren. Im betrachteten Modellausschnitt sowie in allen anderen hier betrachteten Modellen kommen solche Bezüge auch tatsächlich nicht vor.

3.1.2 SERM



Im SERM ist die Anordnung der Objekte genauer geregelt als im PERM. Man beachte, dass die Relation `Lager_Zuo` als eigenständiges Objekt verschwunden ist.

Durch die strikt hierarchische Anordnung der Objekte im SERM ist es allerdings so gut wie unmöglich – selbst in diesem sehr günstigen Beispiel – die bestehenden Analogien „mit dem bloßen Auge“ zu erkennen.

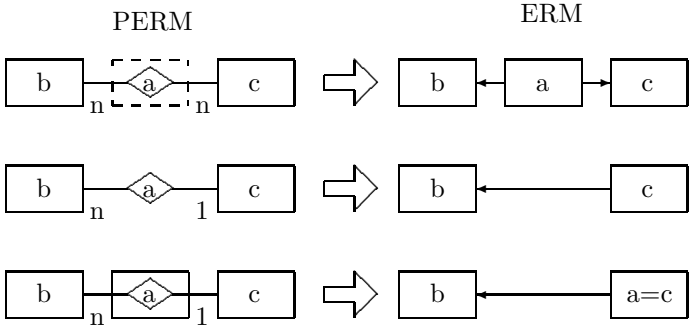
Ähnlichkeitswerte können hier ansonsten genau so berechnet werden wie im PERM. Als einziger Unterschied ergibt sich, dass 1:n-Relationen nicht in die Vergleiche einbezogen werden, da sie wie im vereinfachten ERM (siehe S. 10) nicht als eigene Relationenkästen auftauchen.

Das ist als Vorteil zu sehen, denn bei der Berechnung des Ähnlichkeitswertes im PERM wird nicht zwischen 1:n- und n:m-Beziehungen unterschieden. Sie können sogar als strukturidentisch bewertet werden, was aber wenig Sinn macht. Ein Beispiel dafür findet sich im Datenmodell zur Vertriebsabwicklung, siehe Seite 23.

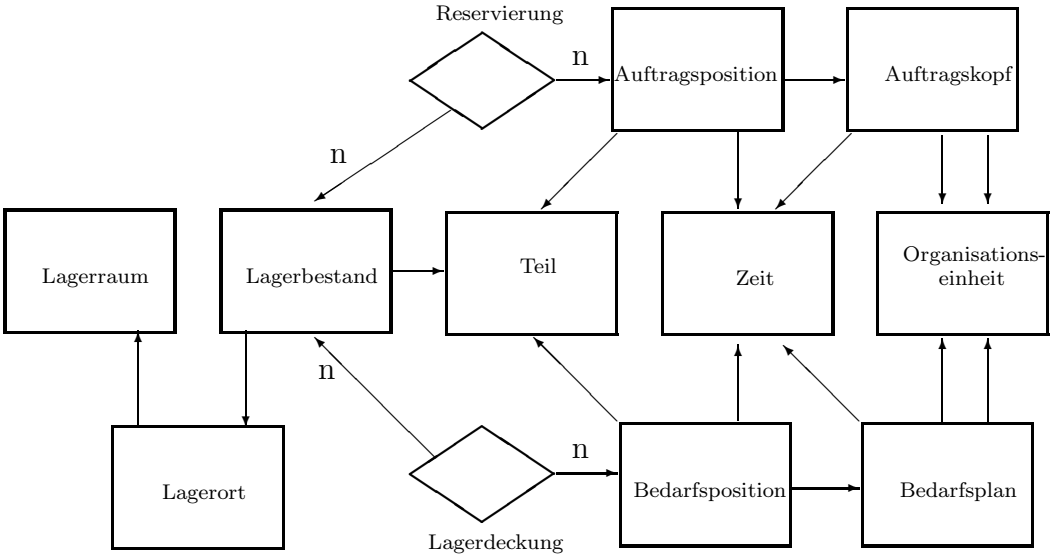
3.1.3 ERM

Da es im ER-Modell keine reinterpretierten Relationen gibt, sind Relationen stets Aggregationen aus Entitäten. Außerdem können zweistellige Relationen der Kardinalität $1 : n$ im Diagramm weggelassen werden – so wie ihnen auch keine eigenen physischen Tabellen entsprechen –, da sie durch Fremdschlüssel in den anderen Tabellen abgebildet werden. In den folgenden Diagrammen sind sie daher durch Pfeile dargestellt. Der Pfeil zeigt dabei von der Entität mit dem Fremdschlüssel auf die referenzierte Entität, also in Richtung auf die 1 in der Kardinalität.

Die Umwandlung eines PERM in ein ERM wird wie in der nächsten Abbildung dargestellt vorgenommen:



Der Modellausschnitt stellt sich im ERM nun so dar:

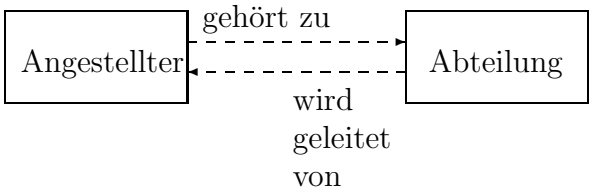


Die beiden gezeigten Relationen der Kardinalität $n : n$ könnten auch noch durch Entitäten ersetzt werden, dabei entstehen je zwei neue 1:n-Beziehungen. Strukturähnlichkeiten lassen sich hier an der Struktur des Graphen erkennen, der symmetrische Teile enthält. Dazu bedarf es jedoch einer geschickten Anordnung der Entitäten, die bei komplexen Modellen nur schwierig zu erreichen ist.

Der in [FE05] definierte Ähnlichkeitswert ist zwar formal auch im ERM definierbar, führt aber zu wenig interessanten Ergebnissen, da er nur feststellen kann, ob zwei Relationen direkt auf denselben Entitäten definiert sind. Weitergehende Ähnlichkeiten können nicht entdeckt werden: im obigen Bild wird die Strukturidentität zwischen `Lagerdeckung` und `Reservierung` nicht erkannt. Es gibt außerdem überhaupt nur diese beiden Relationen, die verglichen werden können.

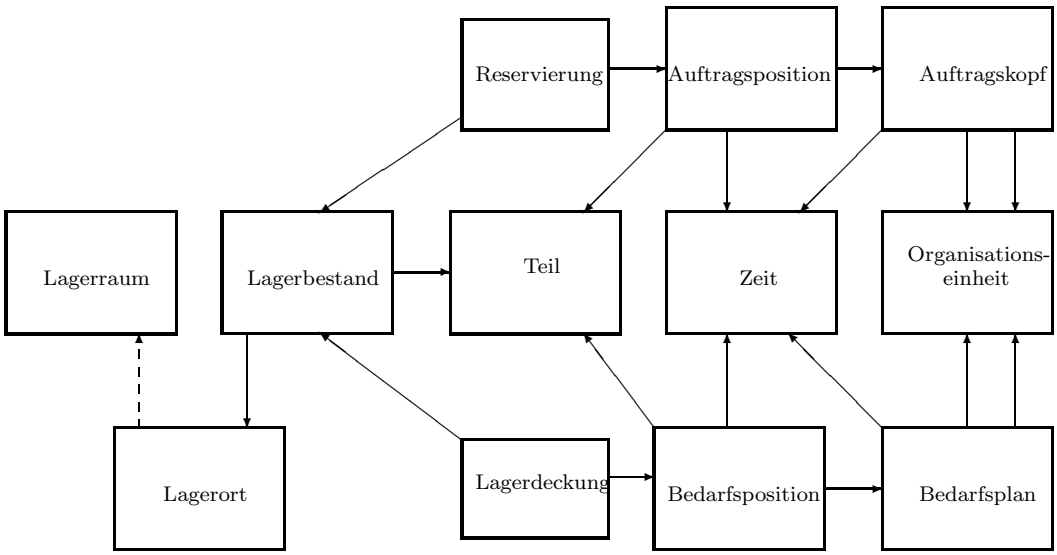
Eine Lösungsmöglichkeit für dieses Problem besteht darin, ganz auf Relationen als Modellierungselement zu verzichten und stattdessen alle Relationen durch Entitätstypen zu ersetzen - die sogenannten Koppelentitäten (vereinfachtes oder physisches ERM). Beziehungen zwischen Entitäten werden durch Fremdschlüssel in den zugehörigen Tabellen dargestellt, die Kardinalität der Beziehungen ist durchgehend 1:n und wird durch einen Pfeil dargestellt. Ähnlichkeitswerte werden nun zwischen Entitätstypen berechnet.

Dazu wird zunächst jedem Entitätstyp E die Multimenge $\hat{E}(E)$ von Entitätstypen zugeordnet, von der er über Beziehungen abhängig ist. Diese Zuordnung geschieht wieder rekursiv, Details siehe weiter unten. Dabei besteht die Gefahr von Zyklen, wie das Beispiel



zeigt. Hier sind `Angestellter` und `Abteilung` zyklisch voneinander abhängig. Solche Zyklen lassen sich vermeiden, wenn man unterscheidet, ob ein Fremdschlüssel Teil des Primärschlüssels ist oder nicht, d.h. ob die zugehörige Beziehung identifizierend ist oder nicht. Diese wichtige Unterscheidung wird von den meisten Autoren für das ER-Modell auch gemacht. Nicht-identifizierende Beziehungen werden dann gestrichelt dargestellt, wie im Bild oben.

Das Bild auf Seite 10 sieht mit diesen Änderungen so aus:



Die Definition von $\hat{E}(E)$ geschieht nun ganz pragmatisch über die Schlüssel – Primär- wie Fremdschlüssel – die in der Tabelle für den Entitätstyp E vorkommen. Dazu werden ein paar vereinfachende Annahmen getroffen, die die Allgemeinheit des Modells nicht einschränken[2]:

1. Entitätstypen E, von denen keine identifizierenden Beziehungen ausgehen, haben als Schlüssel ein einziges Attribut id_E und keine Fremdschlüssel. Es gibt also für diese keine zusammengesetzten Primärschlüssel. Diese Entitätstypen heißen *unabhängig*. Ein solcher Schlüssel heißt *freier Schlüssel*.

[2]Vorsicht: manche Modellierungstools lassen zyklische Abhängigkeiten über identifizierende Beziehungen zu, ohne dabei zu bemerken, dass sie sich einen unendlichen Regress von Fremdschlüsseln einhandeln!

2. Entitätstypen E, von denen identifizierende Beziehungen $E \to E'$ ausgehen, übernehmen den Primärschlüssel $id_{E'}$ von jedem E' als Primärschlüsselteil. Ihr Primärschlüssel ist also im allgemeinen zusammengesetzt. Sie dürfen außerdem noch höchstens ein weiteres freies Primärschlüsselattribut id_E besitzen, welches nicht von einer identifizierenden Beziehung herrührt.

3. Für jede nichtidentifizierende Beziehung $E \to E'$ wird der Primärschlüssel $id_{E'}$ als Fremdschlüssel in E übernommen.

Man beachte, dass die übernommenen Primärschlüssel ihrerseits zusammengesetzt sein können. Im Ergebnis enthält dann die Tabelle für einen Entitätstypen eine Multimenge von freien Primärschlüsseln. $\hat{E}(E)$ ist dann einfach die Multimenge der zugehörigen Entitätstypen.

Wenn man annimmt, dass nur unabhängige Entitätstypen freie Schlüssel haben, dann kann man die Mengen $\hat{E}(E)$ direkt der graphischen Darstellung entnehmen. Alternativ kann man zur Bestimmung von $\hat{E}$ auf die DDL zurückgreifen, die die Tabellen definiert.

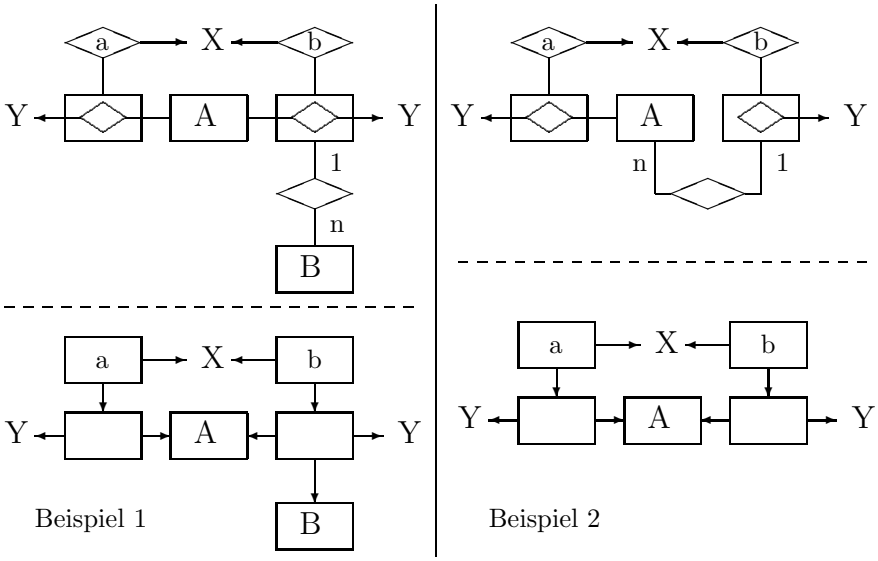
Im Vergleich mit der Darstellung des Modells im PERM ist die Relation `Lager_Zuo` verschwunden. Sie ist als Fremdschlüssel in der Entität `Lagerort` enthalten. Für die Ähnlichkeitswerte mit `Lagerbestand` erhält man im PERM:

$$\begin{aligned} \hat{E}(\text{ Lager\_Zuo }) &= \{\text{ Lagerraum , Lagerort }\}, \\ \hat{E}(\text{ Lagerbestand }) &= \{\text{ Teil , Lagerort }\}, \\ a(\text{ Lager\_Zuo , Lagerbestand }) &= \frac{1}{3}. \end{aligned}$$

Im ERM werden stattdessen `Lagerort` und `Lagerbestand` verglichen:

$$\begin{aligned} \hat{R}(\text{ Lagerort }) &= \{\text{ Lagerraum , Lagerort }\}, \\ \hat{R}(\text{ Lagerbestand }) &= \{\text{ Teil , Lagerort }\}, \\ a(\text{ Lagerort , Lagerbestand }) &= \frac{1}{3}. \end{aligned}$$

Man gewinnt also hier denselben Ähnlichkeitswert. Das ist aber die Ausnahme. Meist ändern sich die Ähnlichkeitswerte. Dazu soll weiter unten auf Seite 24 ein detaillierteres Beispiel vorgestellt werden. Es können sogar Strukturidentitäten verschwinden oder neu entstehen:



In den beiden Beispielen ist ein fiktiver Datenmodellauschnitt oben als PERM, unten als ERM dargestellt. X und Y stehen für nicht näher ausgeführte, aber identische Modellteile.

Beispiel 1: Im PERM sind die Relationen a und b strukturidentisch,

$$\hat{E}_{\mathrm{PERM}}(a) = \hat{E}_{\mathrm{PERM}}(b) = \{A,\ \ldots\}.$$

Die Punkte stehen hier und im Folgenden für die Entitätenmengen, die aus X und Y herkommen.

Im ERM hingegen gewinnt b über die aufgelöste 1:n-Relation einen Schlüssel hinzu:

$$\hat{E}_{\mathrm{ERM}}(a) \neq \hat{E}_{\mathrm{ERM}}(b) = \{A, B,\ \ldots\}.$$

Beispiel 2: Im PERM sind die Relationen a und b *nicht* strukturidentisch,

$$\hat{E}_{\mathrm{PERM}}(a) = \{A,\ \ldots\} \neq \hat{E}_{\mathrm{ERM}}(b).$$

Im ERM hingegen gewinnt b über die aufgelöste 1:n-Relation den Schlüssel von A hinzu:

$$\hat{E}_{\mathrm{ERM}}(a) = \hat{E}_{\mathrm{ERM}}(b) = \{A,\ \ldots\}.$$

Der Ähnlichkeitswert für das ERM ist besonders aussagekräftig, da er unmittelbar ein Maß für die Ähnlichkeit der Schlüssel darstellt. Er unterscheidet auch identifizierende von nicht-identifizierenden Relationen und behandelt 1:n- und n:n-Relationen unterschiedlich. Strukturidentische Entitätstypen können ohne Weiteres in einer Tabelle zusammengefasst werden, da die Schlüssel identisch sind.

Ist der Ähnlichkeitswert kleiner als 1, so ist manchmal eine Deutung der zusammengelegten Entitäten zu einem neuem Entitätstyp als Denormalisierung deutbar, so im obigen Beispiel: fasst man `Lagerort` und `Lagerbestand` in einer Tabelle zusammen, so hat diese als Schlüssel

$$(\underline{\mathrm{id}}_{Teil}, \underline{\mathrm{id}}_{Lagerort}, \mathrm{id}_{Lagerraum}),$$

das ist eine denormalisierte Version der beiden Entitätstypen, bei der die zweite Normalform nicht erfüllt ist. Obwohl der Ähnlichkeitswert gering ist, zeigt dies keine besonders geringe Qualität der Denormalisierung an. Die Brauchbarkeit der Denormalisierung ist im Gegenteil eher schlechter, wenn die Menge der gemeinsamen Schlüssel der beteiligten Entitäten und damit auch der Ähnlichkeitswert groß ist. Beispielsweise haben `Bedarfsposition` und `Lagerdeckung` den Ähnlichkeitswert $\frac{5}{7}$, die Zusammenfassung der beiden in einer Tabelle führt zum Schlüssel

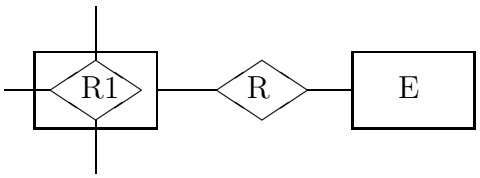
$$(\underline{\mathrm{id}}_{Lo}, \underline{\mathrm{id}}_{T}, \underline{\mathrm{id}}_{T}, \underline{\mathrm{id}}_{Z}, \underline{\mathrm{id}}_{Z}, \underline{\mathrm{id}}_{O}, \underline{\mathrm{id}}_{O}),$$

mit den Abkürzungen Lo, T, Z, O für Lagerort, Teil, Zeit und Organisationseinheit. Die Abhängigkeiten zwischen Nichtschlüsselattributen und Schlüsseln sind nun kaum mehr nachvollziehbar.

3.2 Eigenschaften des Ähnlichkeitswertes

3.2.1 Rang

Man erhält immer besonders große Ähnlichkeitswerte in Situationen wie der folgenden:



Ist R_1 über (mit Vielfachheiten) n Entitäten definiert, so ist R über $n+1$ Entitäten definiert, und man erhält für die Ähnlichkeit

$$a(R, R_1) = 1 - \frac{1}{n+1},$$

einen Wert evtl. sehr nahe bei 1. Es ist aber keineswegs klar, was man mit dieser hohen formalen Ähnlichkeit anfangen soll.

Im SERM lassen sich Relationen aber hinsichtlich ihrer „Entfernung“ von den Entitäten anordnen, da sie mittels ER-Typen hierarchisch aufeinander aufbauen. Die gleiche Möglichkeit besteht im PERM, da zyklische Abhängigkeiten explizit ausgeschlossen werden müssen. Deshalb lässt sich rekursiv für die Relationen ein **Rang** definieren:

- $\text{rang}(R) = 1$, falls R ausschließlich auf Entitäten definiert ist,
- $\text{rang}(R) = \max_i(\text{rang}(R_i)) + 1$, falls R auf Relationen R_i (reinterpretiert) definiert ist.

Es sollten dann nur Ähnlichkeiten zwischen Relationen gleichen Ranges beachtet werden. Man beachte dabei, dass auch Relationen verschiedenen Ranges strukturidentisch sein können.

3.2.2 Generalisierung

Wie schon die Autoren Fettke und Loos in [FE05] feststellen, wird der Ähnlichkeitswert größer, wenn man Entitäts-Untertypen zur selben Generalisierung als gleich betrachtet, anders gesagt, wenn man die Entitätsmengen stets bis zur größtmöglichen Generalisierung fortführt. Dadurch können sogar Strukturidentitäten neu entstehen, die insofern nicht ganz korrekt sind, als die beiden Relationen auf verschiedenen Teilmengen der Superentitäten definiert sind.

Es geht aber darum, mittels des Ähnlichkeitswertes möglichst viele Strukturanalogien aufzudecken. Deshalb ist es günstiger, wenn für den Ähnlichkeitswert Generalisierungen aufgelöst werden.

Gelegentlich sind aber auch Relationen Gegenstand von Generalisierungen, so z.B. zu sehen im Referenzmodell für die Zeit- und Kapazitätsplanung ([SC97] S. 255) für die Relationen `Fertigungsauftragskopf` und `Auftragskopf`, letztere ist die Superentität. Dabei stellt sich heraus, dass bei Auflösung der Generalisierung der Ähnlichkeitswert auch *kleiner* werden kann, wenn nicht sorgfältig gerechnet wird! Der Grund dafür ist, dass der `Auftragskopf` auf anderen Entitäten definiert ist, der `Fertigungsauftragskopf` aber nicht. Dadurch wird die Vereinigungsmenge der beteiligten Entitäten größer, wenn die Generalisierung aufgelöst wird, damit auch der Nenner im Ähnlichkeitswert. I.A. bleibt aber die Durchschnittsmenge, also der Zähler, gleich. Offensichtlich ist der kleinere Ähnlichkeitswert, der sich mit Auflösung der Generalisierung ergibt, der „Richtige“. Der größere Wert ergibt sich nur, wenn man – durch die Abbildung suggeriert – vergisst, dass Subrelationen über exakt

denselben Entitäten definiert sein müssen wie die Superrelation. Andernfalls wäre das Modell fehlerhaft.

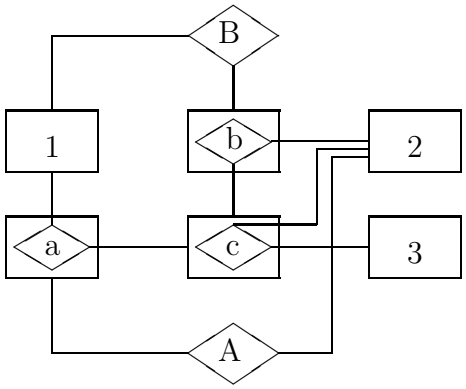
Die Auflösung von Generalisierungen bedarf jedenfalls einer genauen Überlegung. Die Entscheidung kann für verschiedene Teilmodelle eines Datenmodells durchaus verschieden ausfallen!

3.3 Ein alternatives Ähnlichkeitsmaß

In [FE05] wird vorgeschlagen, strukturidentische Entitäten zu einer Entität zu vereinigen. In den bisher betrachteten Beispielen war dies stets einigermaßen sinnvoll und führte zu einer Vereinfachung des Datenmodells. Es ist aber ganz leicht, Beispiele zu konstruieren, in denen eine solche Identifikation problematisch ist.

3.3.1 Probleme bei der Nutzung von Strukturidentitäten

Man betrachte das folgende formale Datenmodell als Beispiel:



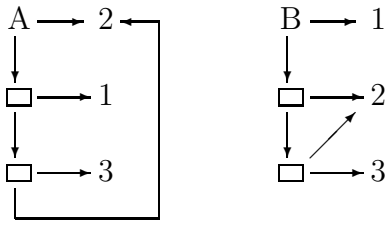
Man rechnet sofort $a(A, B) = 1$ nach. Dennoch lässt sich das Diagramm nicht sinnvoll dadurch vereinfachen, dass man A und B zu einer Relation zusammenlegt. Es gibt Strukturunterschiede, die sich in verschiedener Weise manifestieren:

1. Zwar „erben“ A und B die gleichen Fremdschlüssel in gleicher Anzahl von den drei Entitäten 1,2 und 3, aber sie sind jeweils anders „geschachtelt“:

$$\begin{aligned} A &= A(\mathrm{id}_2, [\mathrm{id}_1, [\mathrm{id}_2, \mathrm{id}_3]]) \\ B &= B(\mathrm{id}_1, [\mathrm{id}_2, [\mathrm{id}_2, \mathrm{id}_3]]) \end{aligned}$$

Mit dieser geschachtelten Liste von Fremdschlüsseln oder Entitäten hat man zugleich einen neuen Kandidaten für ein besseres Ähnlichkeitsmaß.

2. Man kann den Relationen A und B Subgraphen des Datenmodells zuordnen:



Diese sind nicht isomorph, auch nicht als ungerichtete Graphen, da sie u.a. verschieden lange Zyklen enthalten.

Die Isomorphie dieser gerichteten Subgraphen könnte ein neues Kriterium für Strukturidentität sein.

3.3.2 Ein strukturiertes Ähnlichkeitsmaß

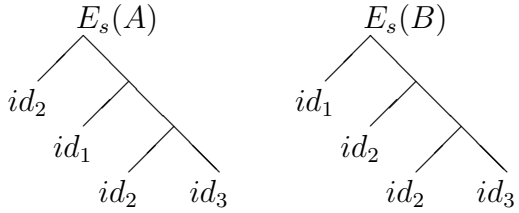
Der oben am Beispiel gezeigte geschachtelte Liste von Entitäten als neues Ähnlichkeitsmaß kann formal leicht definiert werden. Zunächst wird jeder Relation R eine Struktur $E_s(R)$ zugeordnet, die rekursiv definiert wird (für PERM):

1. $E_s(E) = \{E\}$ für eine Entität E, die keine reinterpretierte Relation ist.
2. Für eine Relation R ist die Multimenge

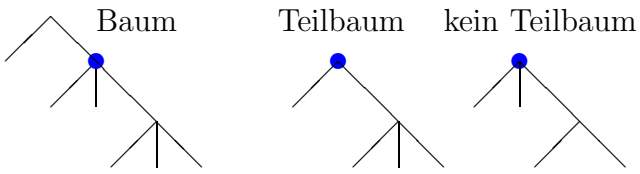
$$E_s(R) = \{E_s(E_i) | R \text{ definiert über } E_i,\ i = 1, \ldots, n\}\,,$$

Wiederholungen von Elementen sind also erlaubt. Wie bereits oben angedeutet, lässt sich diese Definition leicht auf das ERM übertragen und ergibt eine geschachtelte Struktur von Fremdschlüsseln.

Die Definition des Ähnlichkeitsmaßes nutzt aus, dass die $E_s(R)$ einen Baum bilden. Die Bäume zu A und B aus dem Beispiel sind



Das Ähnlichkeitsmaß $A_s(R_1, R_2)$ für zwei Relationen R_1 und R_2 ist dann definiert als der größte Teilbaum von $E_s(R_1)$, der auch Teilbaum von $E_s(R_2)$ ist. Dabei ist ein Teilbaum eines Baumes gegeben durch einen ausgewählten Knoten des Baumes, eine Teilmenge der Nachfolger dieses Knotens, sowie den gesamten Unterbäumen an den ausgewählten Nachfolgern. Die Größe eines Teilbaums meint die Anzahl seiner Blätter. Zur Verdeutlichung (der blaue Knoten ist der ausgewählte):



Die Entscheidung, den zweiten Unterbaum nicht als Teilbaum im Sinne von Strukturähnlichkeit zuzulassen, lässt sich aus der Sicht der Datenmodelle begründen. Es soll nämlich zu einer Relation eine Teilrelation betrachtet werden, die auf weniger Entitäten definiert ist. Sind diese Entitäten reinterpretierte Relationen, dann sollen diese komplett in das Teilmodell aufgenommen werden. Täte man dies nicht, so wäre die Interpretation einer Strukturähnlichkeit sehr erschwert. Außerdem gilt ja, dass der zweite Unterbaum immer noch untersucht werden kann, indem einfach der blaue Knoten eine Stufe tiefer gewählt wird.

Zurück zum Beispiel. Die Teilbäume von $E_s(A)$ außer $E_s(A)$ selbst sind:

$$\{id_1\},\ \{id_2\},\ \{id_3\},\ \{id_2, id_3\},\ \{id_1, \{id_2, id_3\}\},$$

die von $E_s(B)$ hingegen

$$\{id_1\},\ \{id_2\},\ \{id_3\},\ \{id_2, id_3\},\ \{id_2, \{id_2, id_3\}\}.$$

Der größte gemeinsame Teilbaum ist also

$$A_s(A, B) = \{id_2, id_3\}.$$

Daraus lässt sich auch ein numerischer Wert gewinnen, indem zunächst jeder geschachtelten Menge E_s oder A_s eine Multimenge durch Verflachung zugeordnet wird:

$$E_s \mapsto \hat{E}_s := \{E_i | E_i \text{ ist Blatt in } E_s\}.$$

Damit wird definiert

$$a_s(R_1, R_2) := \frac{|\hat{A}_s(R_1, R_2)|}{|\hat{E}_s(R_1) \cup \hat{E}_s(R_2)|}.$$

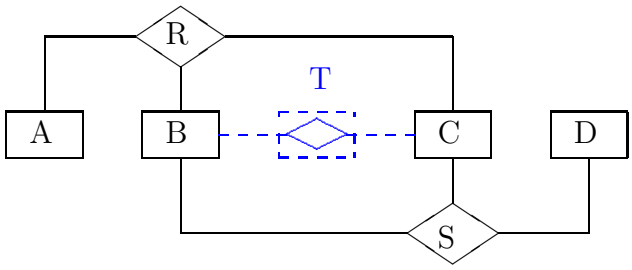
Für das Beispiel wird

$$a_s(A,B) = \frac{|\{id_2, id_3\}|}{|id_1, id_2, id_2, id_3|} = \frac{1}{2}.$$

Die beiden Relationen werden also nicht mehr als strukturidentisch angesehen.

3.3.3 Anwendung: intermediäre Relationen

Das neue Ähnlichkeitsmass kann Relationen entdecken, die im Datenmodell nicht vorhanden sind, aber als zusätzliche Relationen Sinn machen, weil sie als Aggregation in vorhandenen Relationen vorkommen. Dazu ein Beispiel:



Man erhält

$$\begin{aligned} E_s(R) &= \{A, B, C\} \\ E_s(S) &= \{B, C, D\} \\ A_s(R,S) &= \{B, C\} \end{aligned}$$

Die R und S gemeinsame Teilstruktur entspricht einer neuen intermediären-Relation $T \subseteq B \times C$, die definiert werden kann durch

$$T = \{(b,c) \mid \exists a : (a,b,c) \in R \vee \exists d : (b,c,d) \in S\}$$

Es ist gut vorstellbar, dass in konkreten Datenmodellen diese neue Relation fachlich interpretierbar ist und daher tatsächlich im Datenmodell modelliert werden sollte.

3.3.4 Strukturanalogien zwischen entfernten Teilen eines Datenmodells

Bisher wurden Relationen nur dann als strukturähnlich angesehen, wenn sie direkt oder indirekt zumindest teilweise über denselben Entitäten definiert sind. Das in diesem Abschnitt vorgestellte Ähnlichkeitsmass kann aber auch Analogien zwischen Relationen aufdecken, die diese Voraussetzung nicht erfüllen, sondern vielleicht sogar aus zwei völlig verschiedenen Datenmodellen stammen. Zwei Relationen sollen dann strukturähnlich heißen, wenn sie gemeinsame Teilbäume besitzen, die die gleiche topologische Struktur haben. Das bedeutet, das die Etikettierung der Blätter mit Entitätsnamen oder Schlüsseln vernachlässigt wird. Die daraus resultierende Struktur zu einer Relation soll *anonyme Struktur* heißen und mit $E_a(R)$ bezeichnet werden. Man erhält dann daraus das anonyme Ähnlichkeitsmaß $A_a(R_1, R_2)$.

Im Beispiel sieht das dann so aus, dass

$$E_a(A) = E_a(B) = \{\circ, \{\circ, \{\circ, \circ\}\}\}$$

ist – $\circ$ steht dabei für ein Blatt, das sozusagen anonym bleibt. Nun weisen A und B die gleiche topologische Struktur auf.

Diese Vorgehensweise berücksichtigt nicht, dass das Datenmodell in Wirklichkeit kein Baum ist. Im Beispiel etwa sind zwei Blätter identisch. Soll das berücksichtigt werden, d.h. sollen zwei topologische Strukturen nur dann als gleich gelten, wenn identische Blätter in beiden übereinstimmen (bis auf Reihenfolge natürlich), dann ist eine kompliziertere Vorgehensweise nötig.

Ein ausführlicheres Beispiel findet sich in Abschnitt 3.4.6 auf Seite 30.

Will man zwei verschiedene Datenmodelle auf Analogien untersuchen, so kann man folgenden Trick anwenden: Man führt in jedem Datenmodell eine neue Relation `Global` ein, die formal als Kreuzprodukt aller Relationen des Modells definiert ist. Das Ähnlichkeitsmaß $A_a(Global_1, Global_2)$ liefert dann die größte gemeinsame Teilstruktur zurück. Ein Beispiel dazu findet sich ebenfalls auf Seite 30. Dort wird auch diskutiert, welche Auswirkungen es hat, wenn identische Baumblätter berücksichtigt werden, also in beiden Strukturen übereinstimmen sollen.

3.4 Anwendung auf Datenmodelle

Aus der bereits genannten Quelle [SC97] sollen nun vier ausgewählte große Datenmodelle näher betrachtet werden. Die anderen werden deshalb nicht

betrachtet, weil sie entweder zu kleine Ausschnitte darstellen oder weil sie mit bereits ausgewählten Modellen zu viele Überschneidungen aufweisen. Das Modell zur Vertriebslogistik (S. 450f, Abb. B.II.21) wird nicht betrachtet, da es sich bereits durch unmittelbaren Augenschein im Ganzen als strukturidentisch mit der Beschaffungslogistik erweist. Die ausgewählten Modelle weisen dennoch an mehreren Stellen deutliche Überlappungen auf. Dieser Umstand spielt aber erst dann eine Rolle, wenn diese Modelle zu einem Gesamtmodell zusammengefasst werden. Darauf wird in Abschnitt 3.4.5 auf Seite 27 eingegangen.

Da die Datenmodelle in der Quelle als PERM vorliegen, wird auch stets der Ähnlichkeitswert für PERM verwendet, eine Ausnahme davon wird um des Vergleiches willen für ein Teilmodell der Vertriebsabwicklung gemacht.

3.4.1 Bedarfsplanung, Zeit- und Kapazitätsplanung

Quelle des Modells: [SC97], S. 176, Abb. B.I.66, zusammen mit S. 254f, Abb. B.I.133.

Der Ähnlichkeitswert ist 1 für folgende Relationen:

```
(Auftragsposition = Bedarfsposition)
(Auftragskopf     = Bedarfsplan
                  = Primaerbedarfsplan)
(Bedarfsableitung = Bedarfsdeckung)
(Lagerdeckung     = Reservierung)
```

Geht man bei Generalisierungen bis zum Superobjekt (hier handelt es sich dabei immer um Entitäten), so erweitert sich ein Paar:

```
(Bedarfsposition = Auftragsposition = Primaerbedarfsposition)
```

Die nächsthöchsten Ähnlichkeiten ranggleicher Relationen sind:

```
d(Auftragsposition, Fertigungsauftrag) = 0.8,
d(Bedarfsposition, Fertigungsauftrag) = 0.8,
d(Primaerbedarfsposition, Fertigungsauftrag) = 0.8,
d(Position, Vorranggraph) = 0.6,
d(Bedarfsableitung, Lagerdeckung) = 0.545,
d(Bedarfsableitung, Reservierung) = 0.545,
d(Bedarfsdeckung, Lagerdeckung) = 0.545,
d(Bedarfsdeckung, Reservierung) = 0.545.
```

3.4.2 CAM

Quelle des Modells: [SC97], S. 366f, Abb. B.I.221.

Der Ähnlichkeitswert ist 1 für folgende Relationen:

```
Reservierung = Verteilung,
```

Die nächsthöchsten Ähnlichkeiten ranggleicher Relationen sind:

```
d(Standort, Teilladung) =0.75,
d(Auftragsarbeitsgang, Arbeitsgangbereich_Zuo) = 0.714,
d(Maschinenbelegung, Standort) = 0.643.
```

3.4.3 Beschaffungslogistik

Quelle des Modells: [SC97], S. 418f, Abb. B.II.07.

Es gibt keine strukturgleichen Relationen. Die höchsten vorkommenden Ähnlichkeiten sind:

```
d(Zuo, Beschaffungsbelegposition) = 0.833,
d(Zeitreihe-1, Beschaffungsbelegposition) = 0.833,
d(Beschaffungsbeleg, Beschaffungswegzuordnung) = 0.75,
d(Zuo, Zeitreihe-1) = 0.714,
d(Beschaffungsbelegposition, Lieferanteninformationsquelle) = 0.714.
```

Hier sind auch Ähnlichkeiten zwischen Relationen verschiedenen Ranges berücksichtigt.

3.4.4 Vertriebsabwicklung

Quelle des Modells: [SC97], S. 458f, Abb. B.II.25.

Der Ähnlichkeitswert ist 1 für folgende Relationen (ohne Auflösung von Generalisierungen):

```
(Kundenanfrage = Kundenangebot
               = Kundenauftrag
               = AbgesKundenauftrag
               = Lieferschein
```

```
                = Kundenrechnung)
(Kundenanfrageposition = Kundenangebotsposition
                       = Kundenauftragsposition
                       = AbgesKundenauftragsposition
                       = Lieferscheinposition
                       = Kundenrechnungsposition)
(Kundenauftrag_Angebot_Zuo = Folgeauftrag
                           = Teillieferung
                           = Kundenanfrage_Angebot_Zuo
                           = Lieferschein_Rechnung_Zuo)
(Kundenauftragstyp_Zuo = KundenAbgesauftragstyp_Zuo)
(Bedarfsableitung = Bedarfsdeckung)
(Buchung = Sachbuchung)
```

Diese Gruppen enthalten jeweils strukturgleiche Relationen. Allerdings ist `Kundenanfrage_Angebot_Zuo` eine 1:n-Relation, während die andern aus der Gruppe n:m-Relationen sind!

Bei Beachtung von Generalisierungen ergeben sich Änderungen. Eine Gruppe wird aufgeteilt:

```
(Kundenauftrag_Angebot_Zuo = Folgeauftrag
                           = Teillieferung)
(Kundenanfrage_Angebot_Zuo = Lieferschein_Rechnung_Zuo)
```

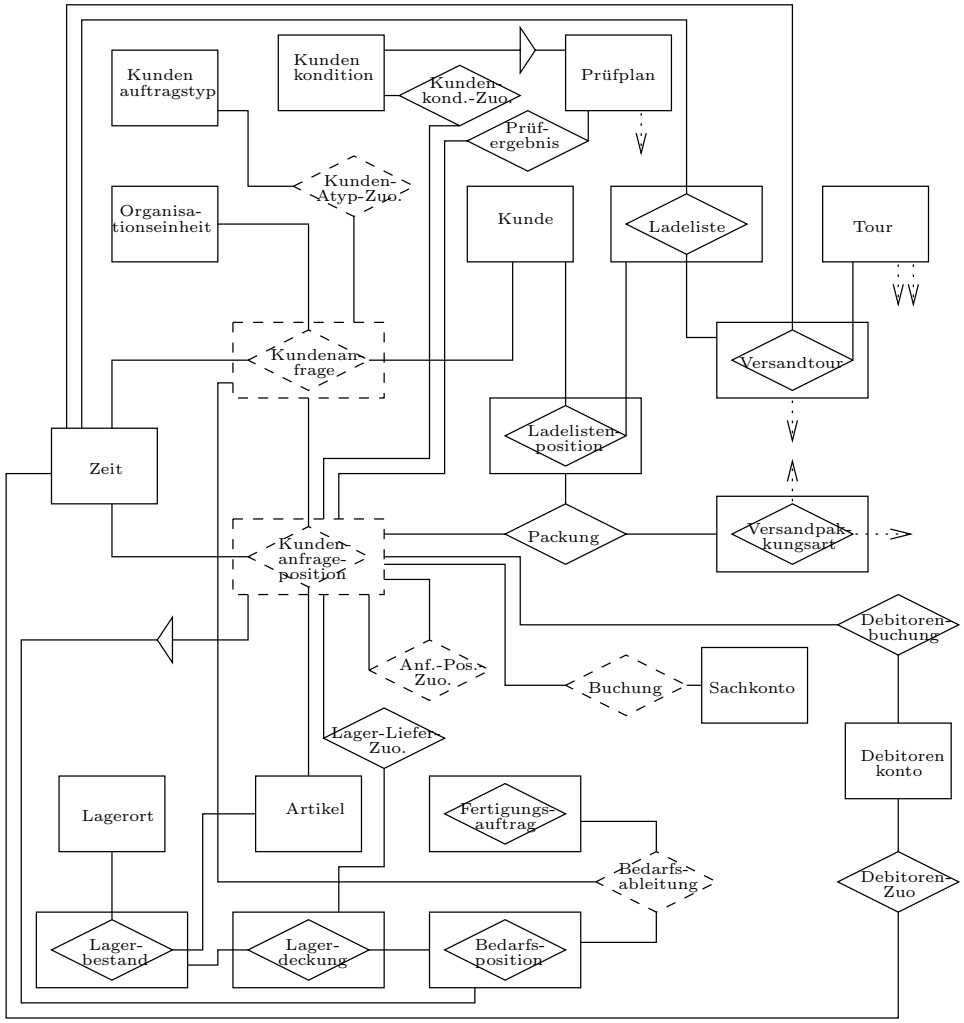
Eine neue Strukturidentität entsteht:

```
(Kundenkonditionen_Zuo = Pruefergebnis)
```

Die nächsthöchsten Ähnlichkeiten sind:

```
d(Kundenauftrag_Angebot_Zuo, Lager_Liefer_Zuo) = 0.75,
d(Folgeauftrag, Lager_Liefer_Zuo) = 0.75,
d(Lager_Liefer_Zuo, Teillieferung) = 0.75.
```

Werden die strukturidentischen Relationen identifiziert, so reduziert sich das PER-Modell ganz erheblich:



Gestrichelte Relationen stehen dabei für Gruppen von strukturidentischen Relationen. Als Name der Gruppe wurde eine darin enthaltene Relation zufällig ausgewählt. Gestrichelte Pfeile weisen auf nicht abgebildete Beziehungen zu weiteren Entitäten und Relationen hin, die weggelassen wurden, da sie nicht von Strukturidentäten betroffen sind. Da der weggelassene Teil aber 1:n-Relationen enthält, ist er Gegenstand des nun folgenden Abschnitts.

Ein Ausschnitt aus der Vertriebsabwicklung als komplexeres Beispiel für Unterschiede zwischen PERM und ERM

Im Datenmodell für die Vertriebsabwicklung kommen einige 1:n-Relationen vor, die auf einer als Entität reinterpretierten Relation definiert sind. Hier sollte sich ein deutlicher Unterschied in den Ähnlichkeitskoeffizienten zeigen. Es soll ein Ausschnitt aus dem Datenmodell sowohl im PERM als

auch im vereinfachten ERM untersucht werden. Der Ausschnitt enthält alle Entitäten und Relationen, die Abhängigkeiten zu `Ladeliste` und/oder `Versandtransporteinheit` haben. Aufgenommen werden alle Relationen, die direkt oder indirekt von den eben genannten Entitäten abhängen, sowie alle Entitäten und Relationen, auf denen sie direkt oder indirekt definiert sind. Man erhält so 12 Relationen:

1	Lieferscheinposition	7	Transport_Zuo
2	Ladelisteposition	8	Transportzusammenstellung
3	Lieferschein	9	Packungs_Zuo
4	Versandtour	10	Ladeliste
5	Versandtransportmittel	11	Versandtransporteinheit
6	Versandpackungsart	12	Packung

Diese Relationen sind im PERM definiert über den Relationen und Entitäten laut folgender Tabelle:

R	über	R	über
1	3,b,c	7	e,f
2	10,b	8	f,h
3	1,a,b,d	9	g,h
4	a,e	10	4*,a
5	8,11	11	4*,7
6	5,9	12	1,2,6

a=Zeit, b=Kunde, c=Artikel
d=Organisationseinheit
e=Tour, f=Transporteinheit
g=Packungsart, h=Transportmittel

Die Sternchen „*“bezeichnen die 1-Enden von 1:n-Relationen: im PER-Modell sind `Ladeliste` und `Versandtransporteinheit` 1:n-Relationen zwischen `Versandtour` und `Zeit` bzw. zwischen `Versandtour` und `Transport_Zuo`, das 1-Ende der Relation zeigt dabei beide Male auf `Versandtour`. Im vereinfachten ERM verschwinden diese beiden Relationen, da sie durch Fremdschlüssel in `Versandtour` dargestellt werden. Andere Relationen, die über `Ladeliste` oder `Versandtransporteinheit` definiert sind, werden dann über `Versandtour` definiert.

Es ist dem PERM nicht zu entnehmen, ob die neuen von `Versandtour` ausgehenden Relationen identifizierend sind oder nicht. Beide Varianten werden dargestellt.

Die Ähnlichkeitsmatrix für den Ausschnitt modelliert mit PERM ist

	1	2	3	4	5	6	7	8	9	10	11	12
1	1.	0.29	0.6	0.17	0.1	0.08	0.	0.	0.	0.14	0.12	0.29
2	0.29	1.	0.4	0.5	0.25	0.2	0.2	0.	0.	0.75	0.33	0.24
3	0.6	0.4	1.	0.25	0.12	0.1	0.	0.	0.	0.2	0.17	0.18
4	0.17	0.5	0.25	1.	0.33	0.25	0.33	0.	0.	0.67	0.5	0.12
5	0.1	0.25	0.12	0.33	1.	0.75	0.33	0.33	0.14	0.29	0.67	0.35
6	0.08	0.2	0.1	0.25	0.75	1.	0.25	0.25	0.25	0.22	0.5	0.47
7	0.	0.2	0.	0.33	0.33	0.25	1.	0.33	0.	0.25	0.5	0.12
8	0.	0.	0.	0.	0.33	0.25	0.33	1.	0.33	0.	0.2	0.12
9	0.	0.	0.	0.	0.14	0.25	0.	0.33	1.	0.	0.	0.12
10	0.14	0.75	0.2	0.67	0.29	0.22	0.25	0.	0.	1.	0.4	0.18
11	0.12	0.33	0.17	0.5	0.67	0.5	0.5	0.2	0.	0.4	1.	0.24
12	0.29	0.24	0.18	0.12	0.35	0.47	0.12	0.12	0.12	0.18	0.24	1.

Die Ähnlichkeitsmatrix für den Ausschnitt modelliert mit dem vereinfachten ERM und identifizierenden neuen Relationen:

	1	2	3	4	5	6	7	8	9	10	11	12
1	1.	0.22	0.6	0.11	0.09	0.08	0.	0.	0.	–	–	0.25
2	0.22	1.	0.29	0.83	0.62	0.5	0.33	0.14	0.	–	–	0.3
3	0.6	0.29	1.	0.14	0.11	0.09	0.	0.	0.	–	–	0.15
4	0.11	0.83	0.14	1.	0.71	0.56	0.4	0.17	0.	–	–	0.25
5	0.09	0.62	0.11	0.71	1.	0.78	0.29	0.29	0.12	–	–	0.35
6	0.08	0.5	0.09	0.56	0.78	1.	0.22	0.22	0.22	–	–	0.45
7	0.	0.33	0.	0.4	0.29	0.22	1.	0.33	0.	–	–	0.1
8	0.	0.14	0.	0.17	0.29	0.22	0.33	1.	0.33	–	–	0.1
9	0.	0.	0.	0.	0.12	0.22	0.	0.33	1.	–	–	0.1
10	–	–	–	–	–	–	–	–	–	–	–	–
11	–	–	–	–	–	–	–	–	–	–	–	–
12	0.25	0.3	0.15	0.25	0.35	0.45	0.1	0.1	0.1	–	–	1.

Die Werte für die nicht mehr vorhandenen Relationen sind durch „–“ ersetzt worden, damit die beiden Tabellen leicht vergleichbar sind. Man erkennt, dass fast alle Ähnlichkeitswerte verändert werden, meist verringern sie sich, oft aber werden sie auch teils erheblich größer (21 mal verringert, 12 mal erhöht).

Modelliert man die neuen Beziehungen als *nicht* identifizierend, so erhält man

	1	2	3	4	5	6	7	8	9	10	11	12
1	1.	0.29	0.6	0.11	0.11	0.09	0.	0.	0.	–	–	0.31
2	0.29	1.	0.4	0.75	0.5	0.38	0.5	0.2	0.	–	–	0.19
3	0.6	0.4	1.	0.14	0.14	0.11	0.	0.	0.	–	–	0.19
4	0.11	0.75	0.14	1.	0.43	0.33	0.4	0.17	0.	–	–	0.5
5	0.11	0.5	0.14	0.43	1.	0.71	0.4	0.4	0.17	–	–	0.71
6	0.09	0.38	0.11	0.33	0.71	1.	0.29	0.29	0.29	–	–	0.44
7	0.	0.5	0.	0.4	0.4	0.29	1.	0.33	0.	–	–	0.13
8	0.	0.2	0.	0.17	0.4	0.29	0.33	1.	0.33	–	–	0.13
9	0.	0.	0.	0.	0.17	0.29	0.	0.33	1.	–	–	0.13
10	–	–	–	–	–	–	–	–	–	–	–	–
11	–	–	–	–	–	–	–	–	–	–	–	–
12	0.31	0.19	0.19	0.5	0.71	0.44	0.13	0.13	0.13	–	–	1.

Auch hier treten Veränderungen in beiden Richtungen auf.

3.4.5 Gesamtmodell

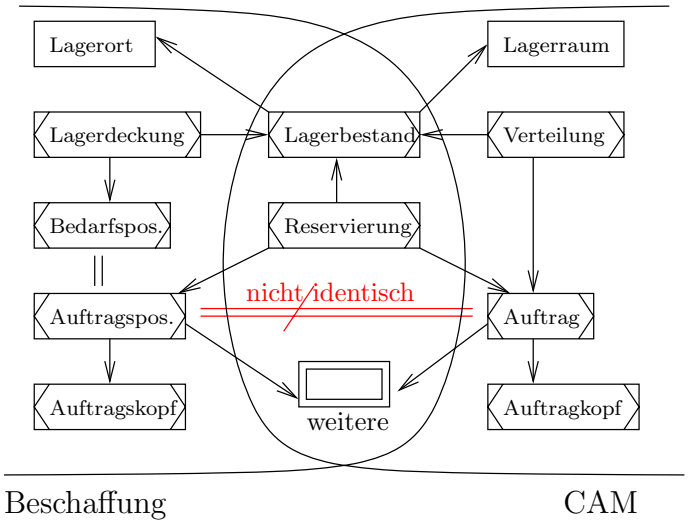
Vereinigt man die bisher betrachteten Datenmodelle zu einem Gesamtmodell, so kommen einige Strukturidentitäten hinzu. Dies ist in Übereinstimmung mit den Anforderungen an Ähnlichkeitswerte: durch die Vereinigung von Teilmodellen werden Relationen aus verschiedenen Teilen vergleichbar und haben Ähnlichkeitswerte, die auch gleich 1 sein können. Bestehende Ähnlichkeitswerte innerhalb eines Teilmodells bleiben aber gleich.

Diese Tatsache kann zur Aufdeckung von Namensinkonsistenzen zwischen Teilmodellen genutzt werden. Hat eine Entität, die in zwei Teilmodellen vorkommt, in beiden verschiedene Namen, so können sich Ähnlichkeitswerte fälschlicherweise verringern. Dieser Fall tritt ein bei der Vereinigung der Modelle *Bedarfsplanung, Zeit- und Kapazitätsplanung* und *CAM*. Sie enthalten folgende Relationen und Entitäten:

Bedarfsplanung etc.: `Reservierung` ist Relation zwischen `Lagerbestand` und **Auftragsposition**,
`Auftragsposition` ist Relation zwischen **Auftragskopf**, `Zeit` und `Teil`

CAM: `Reservierung` ist Relation zwischen `Lagerbestand` und **Auftrag**,
`Auftrag` ist Relation zwischen **Auftragkopf**, `Zeit` und `Teil`

Der Schreibfehler *Auftragkopf/Auftragskopf* führt dazu, dass `Auftragsposition` und `Auftrag` im Gesamtmodell nicht strukturidentisch sind, obwohl sie es eigentlich sein sollten. Im Gesamtmodell ist daher `Reservierung` über vier Entitäten definiert statt über drei in den Einzelmodellen. Daher verringern sich im Gesamtmodell alle Ähnlichkeitswerte von Relationen zu `Reservierung`. In diesem Fall tritt sogar der Verlust von zwei Strukturidentitäten ein:



In den Teilmodellen ist {`Reservierung = Lagerdeckung`} und {`Reservierung = Verteilung`}, im Gesamtmodell nicht mehr.

Da solche Änderungen in den Ähnlichkeitswerten nur durch Namensinkonsistenzen verursacht werden können, hat man einen einfachen Test auf Konsistenz der Teilmodelle. Das Bestehen des Tests ist aber nur eine notwendige, keine hinreichende Bedingung für Konsistenz. In obiger Abbildung finden sich die Entitäten `Lagerort` und `Lagerraum`, die eigentlich dieselbe Entität meinen und auch gleich heißen sollten. In dieser Konstellation hat die Namensungleichheit aber keine Folgen für Strukturidentitäten.

3.4.6 Anwendung des strukturierten Ähnlichkeitsmaßes

Intermediäre Relationen

Eine ausführliche Suche im Gesamtmodell nach intermediären Relationen erbrachte vier Kandidaten:

(Organisationseinheit, Zeit) als gemeinsame Teilstruktur von z.B. `Bedarfsableitung` und `Beschaffungsbeleg` oder von `Bedarfsableitung` und `Beschaffungsbelegposition`, sowie an vielen anderen Stellen;

(Teil, Zeit) als gemeinsame Teilstruktur von z.B. `Maschinenbelegung` mit `Auftragsposition`, `Lagerdeckung` oder `Lager_Liefer_Zuo`;

(Arbeitsplatz, Zeit, Zeit) als gemeinsame Teilstruktur von z.B. `Bedarfsableitung` und `Primaerbedarfsposition`;

(Zeit, (Organisationseinheit, Organisationseinheit, Zeit)) als gemeinsame Teilstruktur von z.B. `Werkzeugbelegung` und `Mitarbeiterbelegung`, `MitarbeiterMaschinenbelegung` oder `Mitarbeiter_Unterbrechung_Zuo`.

Insbesondere die letzte mit ihrer komplexen Struktur scheint einer näheren Betrachtung wert zu sein. Die Einführung einer neuen Relation für diese Struktur könnte sich lohnen und fachlich interpretierbar sein.

Strukturierte Ähnlichkeitswerte

Offensichtlich gilt $a_s \leq a$ für die beiden verschiedenen Ähnlichkeitswerte. Da die Ähnlichkeitswerte für einen Ausschnitt aus dem Modell zur Vertriebsabwicklung bereits vollständig gezeigt wurden, werden für dieses Teilmodell die Änderungen beim Übergang von a zu a_s gezeigt.

Überraschenderweise stellt sich heraus, dass sich nur zwei Ähnlichkeitswerte verringern:

$$a(1,2) = \frac{2}{7} \quad , \quad a_s(1,2) = \frac{1}{7};$$
$$a(2,3) = \frac{2}{5} \quad , \quad a_s(2,3) = \frac{1}{5}.$$

Führt man den Vergleich für das Gesamtmodell aus, so ergibt sich summarisch das folgende Verhalten:

1. Über alle Ähnlichkeitswerte ist a_s im Mittel um 0.00225 kleiner als a.
2. Von $\frac{107 \cdot 106}{2} = 5671$ relevanten Einträgen in der Tabelle (die Diagonale ist nicht relevant, und die Tabelle ist symmetrisch) verkleinern sich nur 574 Einträge.
3. Die Änderung beträgt für die Werte im Mittel 0.111.

4. Die prozentuale Änderung für die geänderten Werte beträgt allerdings im Mittel 45%, die Änderungen liegen zwischen 25% und 90%.

Daraus ist zu entnehmen, dass nur ein kleiner Teil der Ähnlichkeitswerte betroffen ist, und dass außerdem fast immer kleine, also nicht besonders aussagekräftige Werte betroffen sind.

Strukturanalogien

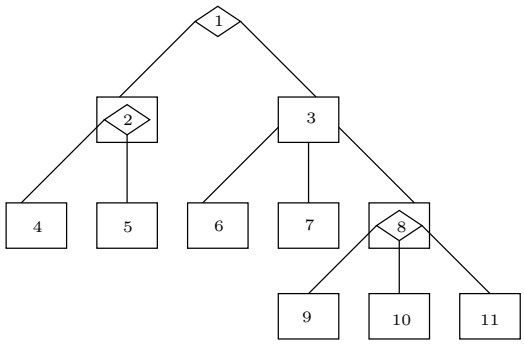
Das Teilmodell zur Vertriebsabwicklung ist besonders reich an strukturidentischen Relationen. Es ist zu erwarten, dass eine Suche nach Strukturanalogien mit dem Ähnlichkeitsmaß ohne Berücksichtigung der Entitätsnamen (reine topologische Ähnlichkeit, anonyme Ähnlichkeit) lohnend ist.

Neben den bereits bestehenden Strukturidentitäten kommen nun hinzu:

- **Versandtour, Lagerbestand, Debitoren_Zuo, Transport_Zuo, Transportzusammenstellung, Packung_Zuo, Teilstrecke** mit der Struktur $(\circ, \circ)$;
- **Ladeliste, Tour_Zuo, Lagerdeckung** mit der Struktur $(\circ, (\circ, \circ))$;
- **Kundenkonditionen_Zuo, Pruefergebnis, Buchung, Sachbuchung, Debitorenbuchung** mit der Struktur $(\circ, (\circ, \circ, (\circ, \circ, \circ)))$. Vorher waren in dieser Gruppe nur *Buchung* und *Sachbuchung* enthalten.

Die Erwartung wird allerdings enttäuscht, denn erstens ergeben sich nur wenige neue Analogien, und zweitens ist keineswegs klar, welchen Nutzen man aus der – sehr einfachen – gemeinsamen Struktur ziehen soll.

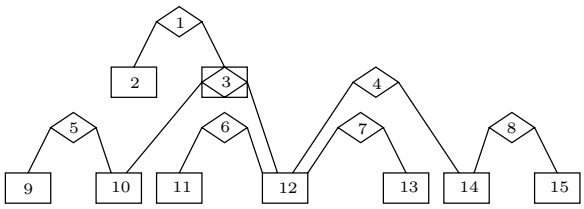
Der erwähnte Trick, um in zwei Datenmodellen das größte gemeinsame topologisch gleiche Teilmodell zu finden, ist aufwändig. Vergleicht man auf diese Weise das Modell zur Bedarfsplanung etc. mit dem Modell zur Beschaffungslogistik, so erhält man als Ergebnis den folgenden Baum:



Dieser Baum gehört zur Relation `Lieferanteninformationsquelle` (Beschaffungslogistik) bzw. zu `Lagerdeckung` und der strukturidentischen `Reservierung` (Bedarfsplanung etc.). Die genaue Zuordnung der Nummern im Bild zu Entitäten oder Relationen ist wie folgt:

lfd. Nr.	Beschaffungslogistik	Bedarfsplanung etc.
1	Lieferanteninformationsquelle	Lagerdeckung
2	Lieferantenmerkmal	Lagerbestand
4	Lieferant	Lagerort
5	Fremdgut	Teil
3	Beschaffungsbelegposition	Auftragsposition
6	Fremdgut	Teil
7	Zeit	Zeit
8	Beschaffungsbeleg	Auftragskopf
9	Lieferant	Organisationseinheit
10	Organisationseinheit	Organisationseinheit
11	Zeit	Zeit

Man findet in beiden Spalten identische Entitäten, aber nicht auf denselben Positionen ($4 \equiv 9$ in der Beschaffungslogistik, aber $9 \equiv 10$ in der Bedarfsplanung)[3]. Will man Teilmodelle nur dann als strukturgleich ansehen, wenn solche Identitäten berücksichtigt werden, dann reicht die hier verwendete einfache Listenverarbeitung nicht aus und es muss auf graphentheoretische Verfahren zurückgegriffen werden. Eine von Hand durchgeführte Suche nach dem größtmöglichen isomorphen Untergraphen in beiden Datenmodellen, der natürlich zu jeder Relation auch alle die Entitäten enthalten muss, auf denen sie definiert ist, führte auf folgendes Teilmodell:



Für die Benennung der Entitäten und Relationen gibt es in beiden Datenmodellen mehrere Möglichkeiten. Ohne zusätzliche Informationen lässt sich

[3]Gleichnamige Entitäten in den beiden Modellen werden hier als zufällig gleichbenannt angesehen, d.h. es wird für den Moment „vergessen“, dass sich die beiden Modelle als Teilmodelle des Gesamtmodells überlappen

diese Strukturanalogie also nicht ausnutzen. Relevant wäre, ob es semantische Parallelen in den beiden Modellen gibt.

Eine Möglichkeit, den Nummern Namen zuzuordnen, ist in der folgenden Tabelle gezeigt:

lfd. Nr.	Beschaffungslogistik	Bedarfsplanung etc.
9	Lieferantenmerkmal	TechnischesVerfahren
5	Lieferantenmerkmal_Zuo	Verfahren_Zuo
2	Periodenraster	Arbeitsplatzgruppe
1	Zeitreihe_2	Arbeitsgang_Zuo
10	Lieferant	Standardarbeitsgang
3	Lieferantenkondition	Arbeitsgang
11	Materialmerkmal	Reihenfolge
6	Materialmerkmal_Zuo	Arbeitsplanreihenfolge
12	Fremdgut	Arbeitsplan
7	Fremdgut_Text_Zuo	Arbeitsplan_Zuo
13	Lieferantentext	Teil
4	Fremdgut_Zuo	Arbeitsplan_Werk_Zuo
14	Einkaufsgruppe	Werk
8	Einkaeufergruppierung	Werksgruppierung
15	Einkaeufer	Werksbereich

Man erkennt deutliche semantische Übereinstimmungen, z.B. bei den Nummern 10, 3, 11, 6, 12, 7, 13 und 4; dabei ist der Abschnitt 12, 7, 13 besonders deutlich. Ebenso bei 14, 8, 15. Hier zeigen sich in gewisser Weise *Patterns* der Datenmodellierung.

3.5 Zusammenfassung

Die Diskussion des Ähnlichkeitswertes für die verschiedenen Modellierungstypen zeigt, dass Ähnlichkeitswerte stark von der gewählten Modellierung abhängen. Sie hängen also keineswegs nur von den fachlichen oder sachlichen Eigenschaften der modellierten Objekte ab. Da die Ähnlichkeitswerte zum ERM am engsten mit den implementierten Tabellen zusammenhängen, sind sie am ehesten geeignet, Kandidaten von Tabellen zu finden, die aggregiert werden können.

Weitere Ergebnisse:

- Der Rang von Relationen ist ein wichtiges Kriterium zur Beurteilung von Ähnlichkeiten. Besonders interessant sind Ähnlichkeiten oder Strukturidentitäten zwischen ranggleichen Relationen.

- Generalisierungen sollten stets aufgelöst werden, um so viele Ähnlichkeiten wie möglich aufzufinden.
- Die Berechnung von Ähnlichkeitswerten bietet den Nebeneffekt, Namensinkonsistenzen bei aufzudecken, die zwischen verschiedenen, sich teilweise überlappenden Datenmodellen bestehen.
- Mit der alternativen Definition von Ähnlichkeit, die die topologische Struktur des Modells mit einbezieht, lassen sich nicht modellierte Relationen finden, die als intermediäre Relationen häufig referenziert werden und die deshalb möglicherweise auch fachlich eine Bedeutung haben.
- Strukturierte Ähnlichkeitswerte hingegen verbessern den Nutzen von Ähnlichkeitswerten nur unwesentlich.
- Die Aufdeckung von Analogien zwischen verschiedenen Datenmodellen oder nicht überlappenden Datenmodellteilen lässt mit den hier vorgestellten Mitteln nicht erreichen, sondern es müssen dafür graphentheoretische Methoden eingesetzt werden.

4 Clusteranalyse

Die Clusteranalyse dient zur Zusammenfassung einer Menge von Objekten in möglichst homogene oder eng zusammenliegende Gruppen. Bei den Objekten handelt es hier natürlich um die Relationen (und evtl. auch Entitäten) eines Datenmodells. Unter Homogenität wird eine weitgehend ähnliche Struktur der Clustermitglieder verstanden. Es ist klar, dass die Strukturähnlichkeit von Relationen mittels des Ähnlichkeitswertes zu messen sein wird. Die Nähe von Objekten zueinander wird durch ein Abstandsmaß, also eine Metrik beschrieben.

In diesem Abschnitt sollen Methoden der Clusteranalyse eingesetzt werden, um die Relationen eines Datenmodells zu Gruppen zusammenzufassen. Ziel dabei ist es, graphische ER-Modelle besser zu strukturieren, übersichtlicher zu gestalten und vereinfachte Sichten zu erstellen.

Gegenüberstellung mit *Entity Tree Clustering*

In [RA92] wird durch O. Rauh und E. Stickel eine andere Methode zur Clusterung von ER-Modellen vorgestellt, die ebenfalls das Ziel hat, ER-Modelle

klarer und übersichtlicher zu machen. Zur Abgrenzung von der Vorgehensweise in dieser Arbeit sollen kurz die Unterschiede dargelegt werden:

1. Rauh und Stickel gehen bei der Clusterung von den *Entitäten* aus, während in dieser Arbeit von den *Relationen* ausgegangen wird.

2. Bei Rauh und Stickel werden im Verlauf der Clusterung Entitätstypen aggregiert, so dass neue Entitätstypen entstehen. Zwischen den aggregierten Entitätstypen entstehen durch Transformation neue Relationstypen aus den alten. Im Gegensatz dazu bleiben in dieser Arbeit Entitäts- und Relationstypen unverändert.

3. Bei Rauh und Stickel liefern Kardinalitäten von Relationen und identifizierende Beziehungen die Kriterien für die Clusterung, in dieser Arbeit wird für die Clusterung der Ähnlichkeitswert herangezogen.

Vorstellung der Verfahren

Für die Clusterbildung müssen vorab zwei Auswahlen getroffen werden:

1. Wahl eines Proximitätsmaßes

   Ein Proximitätsmaß misst die Ähnlichkeit zweier Objekte bzw. deren Nähe zueinander. Da im Laufe des Verfahrens Objekte zu Clustern zusammengefasst werden, muss das Proximitätsmaß auch für zwei Cluster bestimmt werden. Aus dem Ähnlichkeitswert für Objekte muss also ein Ähnlichkeitswert für Mengen von Objekten abgeleitet werden. Gleiches gilt für den Einsatz einer Metrik: aus dem Abstand zwischen Objekten muss ein Abstand zwischen Mengen von Objekten abgeleitet werden.

2. Wahl eines Algorithmus für die Clusterbildung

   Hierbei gibt es zwei grundsätzlich verschiedene Vorgehensweisen:

   Austauschverfahren

   Man geht von der gewünschten Clusterzahl aus und zerlegt die Objektmenge auf eine geeignete heuristische Weise in so viele vorläufige Cluster, wie man am Ende haben möchte. Der Fusionsalgorithmus wechselt dann in mehreren iterativen Schritten jeweils die Clusterzugehörigkeit eines Objektes mit dem Ziel, ein Gütemaß für die Clusterung zu maximieren. Dieser Schritt wird so lange durchgeführt, bis sich entweder die Güte nicht mehr verbessern läßt oder bis eine gewünschte Güte erreicht ist.

Agglomeratives, hierarchisches Verfahren

Man geht von der maximalen Clusterzahl aus, in dem jedes Objekt R zu einem ein-elementigen Cluster $\{R\}$ erklärt wird. In mehreren iterativen Schritten werden dann jeweils die zwei Cluster mit der größten Ähnlichkeit bzw. geringsten Distanz vereinigt. Dieser Schritt wird so lange wiederholt, bis alle Objekte in einem Cluster vereinigt sind.

Das Ergebnis der Clusterung hat die Struktur eines Binärbaumes, genannt *Dendogramm*. Seine Blätter sind die einzelnen Objekte. Jeder Knoten steht für die Zusammenfassung zweier Cluster, die Wurzel ist der Cluster, der alle Objekte enthält.

Daneben gibt es noch weitere Verfahren, die aber in dieser Arbeit nicht betrachtet werden sollen, für eine Übersicht siehe [BA96], S. 281f.

Auswahl aus den Verfahren

Für den in dieser Arbeit betrachteten Fall der Analyse von Ähnlichkeiten zwischen Relationen in einem ER-Modell ist die Anwendung eines Austauschverfahrens nicht besonders interessant, da es voraussetzt, dass die Anzahl der Cluster im voraus bekannt ist. Dies ist aber nicht der Fall, sondern die Clusteranalyse soll im Gegenteil Hinweise auf eine günstige Zusammenfassung von Relationen zu Gruppen liefern.

Für ein agglomeratives Verfahren benötigt man nun ein Proximitätsmaß für Objekte sowie eine Fortsetzung dieses Maßes auf Cluster.

Proximitätsmaße unterscheiden sich je nach der Art der Merkmale der Objekte: handelt es sich um metrische Merkmale, so lassen sich die einem Objekt zugeordneten Merkmale als Vektoren in einem $\mathbb{R}^n$ auffassen, und als Proximitätsmaß kommt eine Metrik auf dem $\mathbb{R}^n$ in Frage, z.B. die Euklidische Metrik. Dieser Fall liegt bei Relationen aber nicht vor, sondern es handelt sich hier um nominale Merkmale mit binärer Merkmalstruktur: Relation R ist / ist nicht über der Entität E definiert. Für solche Nominalskalen werden in [BA86] verschiedene Proximitätsmaße definiert, die natürlich *keine* Metriken sind. Das bisher benutzte Ähnlichkeitsmaß a kann mit dem Tanimoto-Koeffizienten identifiziert werden.

Die Fortsetzung des Proximitätsmaßes auf Cluster kann nach verschiedenen Kriterien erfolgen. Eine Auswahl solcher Fortsetzungen findet sich in Tabelle 6.17, S. 287 in [BA86], wo auch ihre Eigenschaften diskutiert werden. Dabei ist zu beachten, dass von den sieben dort genannten Fortsetzungen drei zwingend eine Metrik zwischen den Objekten voraussetzen.

Im folgenden Abschnitt wird deshalb gezeigt, dass sich aus den verwendeten Proximitätsmaßen eine Metrik herleiten läßt.

4.1 Metriken

Wenn man strukturäquivalente Relationen identifiziert, dann kann man den Ähnlichkeitswert nutzen, um eine Metrik d auf der Menge der Äquivalenzklassen von Relationen zu definieren:

$$d(R_1, R_2) := 1 - a(R_1, R_2)$$

ist nämlich automatisch symmetrisch und erfüllt

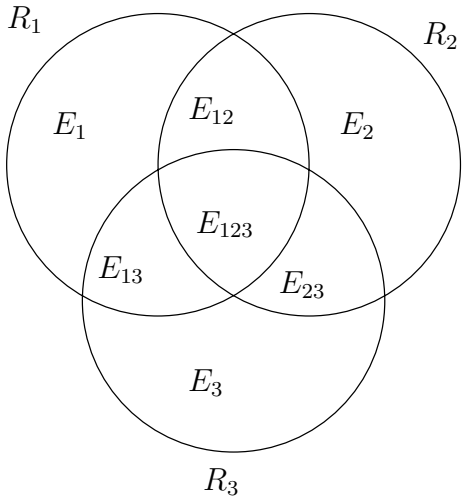
$$d(R_1, R_2) = 0 \iff R_1 \cong R_2.$$

($\cong$ steht für „strukturidentisch“)

Die Dreiecksungleichung

$$d(R_1, R_2) \leq d(R_1, R_3) + d(R_3, R_2)$$

ergibt sich aus dem folgenden Venn-Diagramm für die Entitätenmengen zu den Relationen:



In diesem Venn-Diagramm sind die Multimengen von Entitäten, auf denen die Relationen R_1, R_2 und R_3 definiert sind, in disjunkte Teile zerlegt:

$$E_{123} \quad := \quad \hat{E}(R_1) \cap \hat{E}(R_2) \cap \hat{E}(R_3),$$

$$\begin{aligned} E_{12} &:= \hat{E}(R_1) \cap \hat{E}(R_2) \setminus E_{123}, \\ &\dots \\ E_1 &:= \hat{E}(R_1) \setminus (\hat{E}_2 \cup \hat{E}_3), \\ &\dots \end{aligned}$$

Wenn man die Mächtigkeiten dieser Teilmengen mit $e_{\dots} := |E_{\dots}|$ bezeichnet - wobei die Punkte für einander entsprechende gleiche Indizes stehen, dann ist

$$a_{ij} := a(R_i, R_j) = \frac{e_{123} + e_{ij}}{G - e_k},$$

hier ist $G = |\hat{E}(R_1) \cup \hat{E}(R_2) \cup \hat{E}(R_3)|$, und $\{i, j, k\} = \{1, 2, 3\}$.

Daraus ergibt sich

$$d_{ij} := d(R_i, R_j) = \frac{e_{ik} + e_{jk} + e_i + e_j}{e_{ij} + e_{ik} + e_{jk} + e_i + e_j + e_{123}}.$$

Die Dreiecksungleichung ist erfüllt, wenn stets

$$d_{ij} + d_{jk} - d_{ik} \geq 0$$

ist.

Bringt man die linke Seite auf einen Nenner, so errechnet sich der Zähler zu

$$\begin{aligned} &e_{123}\, e_i\, e_{ij} + {e_i}^2\, e_{ij} + e_i\, {e_{ij}}^2 + 2\, {e_{123}}^2\, e_{ik} + 3\, e_{123}\, e_i\, e_{ik} + {e_i}^2\, e_{ik} + 4\, e_{123}\, e_{ij}\, e_{ik} + \\ &4\, e_i\, e_{ij}\, e_{ik} + 2\, {e_{ij}}^2\, e_{ik} + 4\, e_{123}\, {e_{ik}}^2 + 3\, e_i\, {e_{ik}}^2 + 4\, e_{ij}\, {e_{ik}}^2 + 2\, {e_{ik}}^3 + \\ &2\, {e_{123}}^2\, e_j + 2\, e_{123}\, e_i\, e_j + {e_i}^2\, e_j + 3\, e_{123}\, e_{ij}\, e_j + 2\, e_i\, e_{ij}\, e_j + {e_{ij}}^2\, e_j + \\ &6\, e_{123}\, e_{ik}\, e_j + 4\, e_i\, e_{ik}\, e_j + 5\, e_{ij}\, e_{ik}\, e_j + 4\, {e_{ik}}^2\, e_j + 2\, e_{123}\, {e_j}^2 + e_i\, {e_j}^2 + e_{ij}\, {e_j}^2 + \\ &2\, e_{ik}\, {e_j}^2 + e_i\, e_{ij}\, e_{jk} + 4\, e_{123}\, e_{ik}\, e_{jk} + 3\, e_i\, e_{ik}\, e_{jk} + 4\, e_{ij}\, e_{ik}\, e_{jk} + 4\, {e_{ik}}^2\, e_{jk} + \\ &3\, e_{123}\, e_j\, e_{jk} + 2\, e_i\, e_j\, e_{jk} + 2\, e_{ij}\, e_j\, e_{jk} + 5\, e_{ik}\, e_j\, e_{jk} + {e_j}^2\, e_{jk} + 2\, e_{ik}\, {e_{jk}}^2 + e_j\, {e_{jk}}^2 \\ &+2\, e_{123}\, e_i\, e_k + {e_i}^2\, e_k + 2\, e_i\, e_{ij}\, e_k + 3\, e_{123}\, e_{ik}\, e_k + 4\, e_i\, e_{ik}\, e_k + 3\, e_{ij}\, e_{ik}\, e_k + \\ &3\, {e_{ik}}^2\, e_k + 2\, e_{123}\, e_j\, e_k + 2\, e_i\, e_j\, e_k + 2\, e_{ij}\, e_j\, e_k + 4\, e_{ik}\, e_j\, e_k + {e_j}^2\, e_k + \\ &e_{123}\, e_{jk}\, e_k + 2\, e_i\, e_{jk}\, e_k + e_{ij}\, e_{jk}\, e_k + 4\, e_{ik}\, e_{jk}\, e_k + 2\, e_j\, e_{jk}\, e_k + {e_{jk}}^2\, e_k + e_i\, {e_k}^2 + \\ &e_{ik}\, {e_k}^2 + e_j\, {e_k}^2 + e_{jk}\, {e_k}^2, \end{aligned}$$

und das ist offensichtlich nichtnegativ! Außer in Trivialfällen ist die Dreiecksungleichung also sogar eine echte Ungleichung.

Der Sonderfall $e_2 = e_3 = e_{13} = e_{23} = 0$, $e_{123} = 1$ ergibt

$$d_{12} + d_{23} - d_{13} = \frac{e_1\, e_{12}}{(1 + e_{12})\ (1 + e_1 + e_{12})} < \frac{e_1}{e_1 + e_{12}},$$

welches beliebig klein werden kann. Ebenso kann

$$\frac{d_{13}}{d_{12}+d_{23}} = 1 - \frac{e_1\, e_{12}}{e_1 + e_{12} + 2\, e_1\, e_{12} + {e_{12}}^2} > 1 - \frac{e_1}{e_1 + 2e_{12}}$$

beliebig nahe an 1 liegen. Die Ungleichung läßt sich also weder absolut noch relativ verschärfen.

Für die Ähnlichkeitswerte ergibt sich aus der Dreiecksungleichung die Beziehung

$$a_{12} \geq a_{13} + a_{23} - 1.$$

Dies ist eine intuitiv einleuchtende Eigenschaft von Ähnlichkeitswerten, die, wie es hier der Fall ist, einen relativen Grad von Ähnlichkeit messen. In Worten heißt es nämlich, dass zwei Relationen, die mit einer dritten Relation beide gemeinsame Entitäten haben, und deren relative Anteile in der Summe 100% überschreiten, untereinander auch mindestens eine Entität gemein haben müssen.

4.2 Ein anderes Ähnlichkeitsmaß

Schon in [FE05] wird darauf hingewiesen, dass die Ähnlichkeitswerte auch anders berechnet werden könnten. Interessanterweise führt auch das in [BA86], S. 267 beschriebene *Simple-Matching*-Maß (M-Maß) auf eine Metrik.

Angewendet auf unsere Situation berechnet sich der Ähnlichkeitswert mit dem M-Maß a_M durch

$$a_M(R_1, R_2) = \frac{m - |\hat{E}(R_1) \triangle \hat{E}(R_2)|}{m},$$

dabei ist $\triangle$ die symmetrische Differenz und

$$m = \left| \bigcup_R \hat{E}(R) \right|.$$

Setzt man wieder für die Metrik $d_M = 1 - a_M$ und berücksichtigt, dass m eine feste Konstante ist, kann auch einfach

$$d_M(R_1, R_2) = |\hat{E}(R_1) \triangle \hat{E}(R_2)|$$

als Metrik verwendet werden, was im Folgenden geschehen soll (M-Metrik).

d_M hat die gewünschten Eigenschaften:

1. $d_M = 0$ genau für strukturidentische Relationen,
2. d_M ist symmetrisch,
3. d_M erfüllt die Dreiecksungleichung, wie man sich sofort am Venn-Diagramm für drei Relationen klarmacht:

$$\begin{aligned} d_M(R_1, R_2) &= e_1 + e_{13} + e_2 + e_{23} \\ &\leq e_1 + e_{13} + e_2 + e_{23} + 2e_{12} + 2e_3 \\ &= d_M(R_1, R_3) + d_M(R_2, R_3). \end{aligned}$$

4.3 Anwendung auf Datenmodelle

Im folgenden sollen ausführlich die Cluster von Relationen diskutiert werden, die sich bei verschiedenen Verfahren der Clusteranalyse ergeben. Dabei kommen als Metriken die nach Tanimoto und die M-Metrik zum Einsatz, als Proximitätsmaße für Gruppen alle sieben in [BA86] genannten. Es ergeben sich 14 verschiedene Kombinationen, die mit Abkürzungen bezeichnet werden, siehe die folgende Tabelle.

Metrik / Proximitätsmaß	Tanimoto	M-Metrik
Nearest Neighbour (Single Linkage)	NT	NM
Furthest Neighbour (Complete Linkage)	FT	FM
Average Linkage ungewichtet	AuT	AuM
Average Linkage gewichtet	AgT	AgM
Centroid	CT	CM
Median	MT	MM
Ward	WT	WM

Aus jedem bei der Clusterung entstehenden Dendogramm lässt sich durch Augenschein eine kleine Anzahl von Hauptclustern (im folgenden kurz Cluster genannt) auswählen. Die Anzahl und genaue Zusammensetzung der Cluster unterliegt dabei einer gewissen Willkür.

4.3.1 Bedarfsplanung, Zeit- und Kapazitätsplanung

Zur besseren Übersicht sind die Relationen durchnummeriert worden (fehlende Nummern gehören zu Relationen, die zu anderen strukturidentisch sind):

Struktur	1	Merkmalgruppierung	23
Teil_Merkmaltyp_Zuo	2	Struktur_Merkmal_Zuo	25
Auftragsposition	3	Struktur_Variante_Zuo	26
Auftragskopf	5	Teil_Variante_Zuo	27
Primaerbedarfsposition	7	Arbeitsgang_Komponente_Zuo	28
Lagerbestand	9	Werkzeug_Zuo	29
Werkzeugeinsatz	10	NC_Programm	30
Arbeitsgang	11	Verfahren_Zuo	31
Arbeitsgang_Zuo	12	Arbeitsplan_Zuo	32
Arbeitsplanreihenfolge	13	Arbeitsgruppierung	33
Fertigungsauftragskopf	14	Arbeitsplan_Werk_Zuo	34
Fertigungsauftrag	15	Position	35
Auftragsarbeitsplan	16	Vorranggraph	36
Auftrag_Arbeitsgang_Zuo	17	Mitarbeiter_Zuo	37
Bedarfsableitung	18	Werksgruppierung	38
Dispositionsstufe_Zuo	20	Bereichsgruppierung	39
Lager_Zuo	21	Belastung	40
Lagerdeckung	22		

Die Bilder im Anhang (Seite 79) zeigen die für alle 14 Methoden entstehenden Dendogramme. Die Anordnung entspricht der in der Tabelle der Abkürzungen auf Seite 39, spaltenweise von oben nach unten. Für die Tanimoto-Metrik und für die M-Metrik findet man Cluster, die für alle Proximitätsmaße mehr oder weniger klar wieder auftauchen. Ein typisches für die Tanimoto-Metrik, bestehend aus den Relationen 11, 12, 17, 29, 30, 35, 36 und 40 ist blau hervorgehoben, für die M-Metrik sind die Relationen 1, 5, 10, 14, 21, 23, 27, 33 und 38 rot hervorgehoben. Die Relation 11 gehört eigentlich auch in dieses Cluster für die M-Metrik, wird aber aus drucktechnischen Gründen nur in blau gezeigt.

Es zeigt sich, dass die ausgewählten Cluster über alle Methoden und bei der „richtigen“ Metrik sehr stabil sind, bei der „falschen“ Metrik aber so gut wie gar nicht erkennbar. Daraus kann man vorsichtig schließen, dass das Proximitätsmaß einen vergleichsweise geringen Einfluss auf die grobe Clusterung hat, diese aber stark von der Metrik abhängen kann.

In der Literatur wird beschrieben, dass die verschiedenen Proximitätsmaße Clusterungen mit bestimmten Eigenschaften ergeben:

kontrahierend: es werden zunächst wenige große Cluster gebildet, denen viele kleine gegenüberstehen. Bei metrischen Merkmalen können damit Ausreißer indentifiziert werden.

dilatierend: viele einzelne, etwa gleich große Cluster

kettenbildend: große Cluster durch Aneinanderreihung von Objekten

Eine Inspektion der Clusterbildung für das Datenmodell zur Bedarfsplanung etc. zeigt folgende Eigenschaften:

Nearest Neighbour: lange Ketten, kaum erkennbare Cluster, besonders schlecht mit M-Metrik

Furthest Neighbour: dilatierend, mit M-Metrik etwa gleichgroße Cluster

Average Linkage (mit und ohne Wichtung): Kettenbildung mit M-Metrik, dilatierend mit Tanimoto-Metrik, am Ende werden schlecht zuordbare Relationen erkennbar

Centroid: starke Neigung zur Kettenbildung, wenig brauchbare Cluster, kontrahierend

Median: starke Neigung zur Kettenbildung am Ende, davor Bildung etwa gleichgroßer Cluster (mit M-Metrik zwei deutliche Cluster), kontrahierend

Ward: keine Kettenbildung am Ende, zwei etwa gleichgroße Cluster, die aber bei der M-Metrik nicht gut in Untercluster zerlegt sind

Natürlich lässt sich aus nur einem Beispiel nicht ableiten, dass die Proximitätsmaße allgemein auf diese Eigenschaften führen. Die obige Liste stimmt aber gut mit den Angaben in der Literatur überein ([BA96], S.298, sowie [BE81], S. 96f).

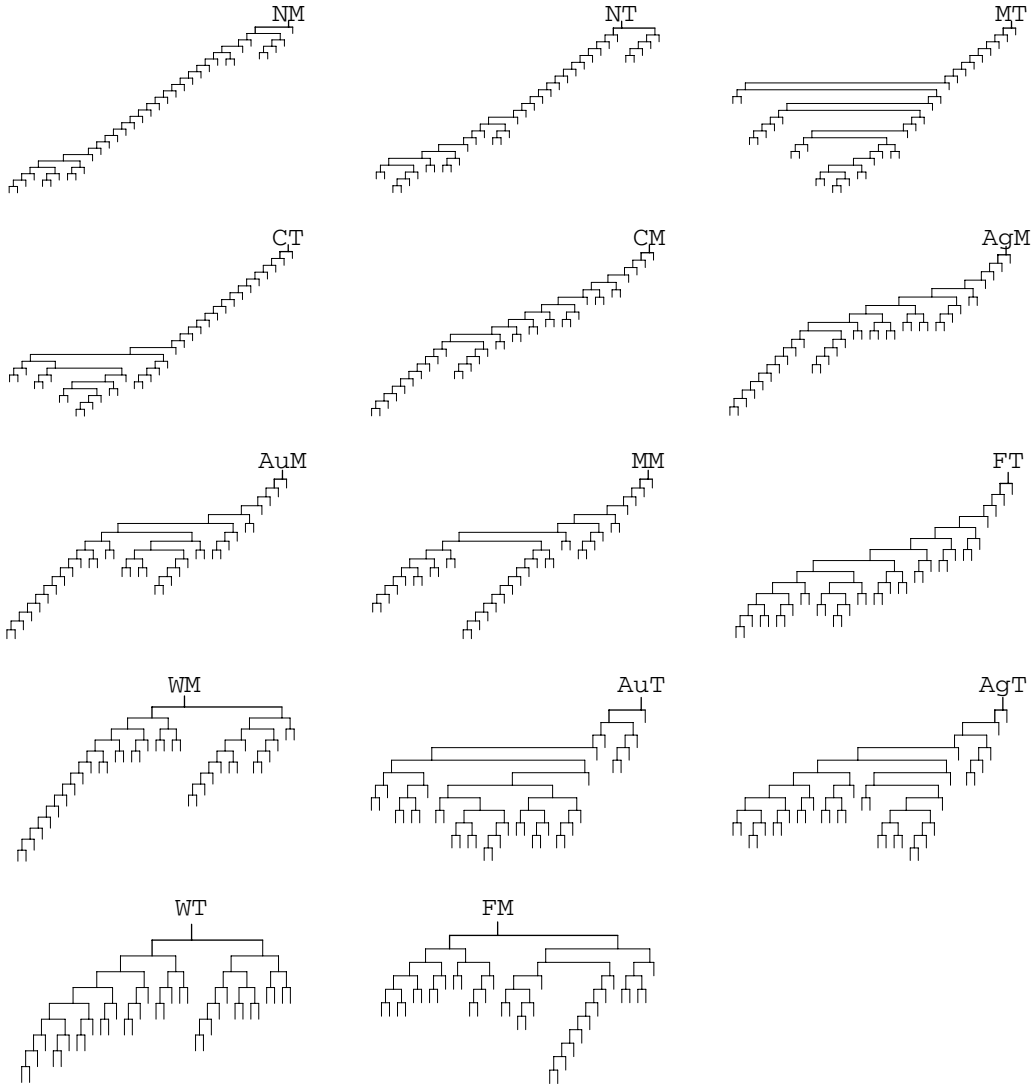
Die Neigung zur Kettenbildung und die Bildung vergleichbar großer Cluster lässt sich durch eine einfache Testgröße erfassen, die für ein Dendogramm bestimmt wird und die wie folgt definiert ist:

$$G(\text{ Dendogramm }) := \sum_{\substack{\text{Vereinigung}\\ \text{zweier}\\ \text{Gruppen}}} |\text{ kleinere Gruppe }|.$$

Diese Größe ist in der folgenden Tabelle noch auf Werte zwischen 0 und 1 normiert worden, dabei entspricht 1 einer linearen Kette ohne Verzweigungen und 0 einem perfekt ausgeglichenen Binärbaum, also mit optimal gleichgroßen Clustern:

Methode	G	Methode	G	Methode	G	Methode	G
NT	0.82	NM	0.86	CT	0.82	CM	0.78
FT	0.59	FM	0.37	MT	0.82	MM	0.66
AuT	0.46	AuM	0.62	WT	0.39	WM	0.53
AgT	0.48	AgM	0.70				

Im folgenden Bild sind die Dendogramme gezeigt, absteigend sortiert nach G, von links nach rechts und von oben nach unten:



Man erkennt eine gute Übereinstimmung des Augenscheins mit der Sortierung nach der Testgröße.

Die Normierung der Testgröße im Detail:

1. Bei einer linearen Kette von n Objekten ergeben sich $n-1$ Vereinigungen von Gruppen, wobei die kleinere Gruppe stets aus einem Objekt besteht. Man erhält

$$G(\text{ lineare } n\text{-Kette }) = n - 1.$$

2. Bei einem perfekten Binärbaum mit $n = 2^m$ Blättern = Objekten errechnet man leicht

$$G(\text{ perfekter Binärbaum der Höhe } m+1\ ) = m \cdot 2^{m-1} = \frac{n}{2} \log n.$$

3. Dieser Wertebereich wird linear auf [0,1] abgebildet.

In den folgenden Abschnitten sollen nur noch drei Proximitätsmaße als repräsentative Auswahl weiterverfolgt werden, nämlich **Nearest Neighbour**, **Furthest Neighbour** und **Ward**. Damit ist ein kettenbildendes, ein eher dilatierendes und ein zu gleichgroßen Gruppen neigendes Verfahren (je nach Metrik) ausgewählt. Ihnen entspricht eine Testgröße im oberen, mittleren und unteren Bereich. Es soll gezeigt werden, dass bei allen diesen Maßen die grobe Clusterstrukter recht stabil ist.

4.3.2 CAM

Die Nummern der Relationen:

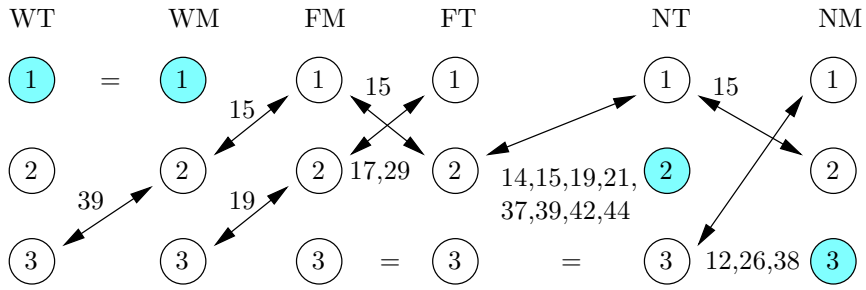
Transport	1	Standort	24
Lagerbestand	2	APLGrp_Station_Zuo	25
Reservierung	3	Teilladung	26
Wartung_Zuo	4	Palettentyp_Zuo	27
Lagerspiel	5	Palettenverfolgung	28
Arbeitsplan_Zuo	6	Verteilung	29
Arbeitsgang	7	Wartungsplan_Zuo	30
Arbeitsgang_Zuo	8	Gruppierung	31
Werkzeugeinsatz	9	Struktur	32
Mitarbeiter_Zuo	10	Pruefplan_Zuo	33
NC_Programm	11	Ersatzteil_Zuo	34
Werkzeug_Zuo	12	Vorranggraph	35

Auftrag	13	Arbeitsgruppierung	36
Mitarbeiterbelegung	14	Zeitereignis	37
Auftragkopf	15	Werkzeugbelegung	38
Auftragsarbeitsplan	16	MitarbeiterMaschinenbelegung	39
Auftragsarbeitsgang	17	NC_Sequenz_Zuo	40
Auftragsarbeitsgang_Zuo	18	Position	41
Maschinenbelegung	19	Mitarbeiter_Unterbrechung_Zuo	42
Auftrag_Bereich_Zuo	20	Arbeitsgangbereich_Zuo	43
Unterbrechung	21	Unterbrechung_Schicht_Zuo	44
Lager_Station_Zuo	22	Erzeugt	45
Transportgruppierung	23		

Die folgende Tabelle zusammen mit dem folgenden Bild zeigt eine plausible Zerlegung in jeweils 3 Cluster für jede der 6 Methoden in übersichtlicher Weise:

Clusternummer	Methode	Relationen im Cluster
1	Ward-Proximität, beide Metriken (WT,WM)	1, 2, 4, 6, 9, 10, 22, 23, 25, 27, 28, 30, 32, 34, 36, 45
2	Nearest Neighbour, Tanimoto-Metrik (NT)	3, 5, 13, 16, 20, 31, 43
3	Nearest Neighbour, Simple-Metrik (NM)	7, 8, 11, 18, 24, 33, 35, 40, 41

Dies sind die jeweils kleinsten Cluster. Im Bild sind diese farbig hervorgehoben. Relationen, die nicht in diesen Clustern enthalten sind, werden je nach Methode anderen Clustern zugeordnet, wie genau zeigt das Bild.

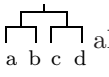


Die Pfeile geben an, welche Relationen bei einem Methodenwechsel von einem Cluster in einen anderen wechseln. Z.B. verlässt Relation 15 bei einem Wechsel von NT zu NM Cluster 1 und wandert in Cluster 2; Cluster 1 nimmt

dafür aber die Relationen 12, 26, 38 aus Cluster 3 neu auf. Die Cluster erweisen sich als erstaunlich stabil, abgesehen vom Übergang zwischen FT und NT, bei dem 8 Relationen zwischen Cluster 1 und 2 wechseln.

Als Fazit ist festzustellen, dass sich beide Ähnlichkeitsmaße gleich gut zur Clusterbildung eignen, während bei den Proximitätsmaßen *Ward* und *Furthest Neighbour* gute Übereinstimmung zeigen. Nur bei *Nearest Neighbour* gibt es deutlichere Abweichungen.

Um einen Eindruck davon zu vermitteln, wie sehr sich die ausgewählten Cluster auf naheliegende Weise „aufdrängen“, wird hier das vollständige Dendogramm für die Methode FM gezeigt.

Der Übersichtlichkeit halber ist die Darstellung zweidimensional, dabei wurde ein Dendogramm 

als Matrix $\begin{pmatrix} a & b \\ c & d \end{pmatrix}$ dargestellt:

$$\left(\begin{array}{c} \left(\begin{array}{c} \left(\begin{array}{cc} \begin{pmatrix}(16,43)\\(3,5)\end{pmatrix} & \begin{matrix}(13,20)\\31\end{matrix} \end{array}\right) \\ (21,44,37) \end{array}\right) \\ \left(\begin{array}{c} \left(\begin{array}{ccc} \left(\begin{array}{cc} \begin{pmatrix}9,10,25,36,15\\6,32\end{pmatrix} & 34 \end{array}\right) & 45 & (4,30) \\ (22,27) & 1 & (2,28) \end{array}\right) \\ (14,42) \end{array}\right) \\ \left(\begin{array}{cc} \begin{pmatrix}7,8,33,41\\11,40\\12\end{pmatrix} & \begin{matrix}(18,24,35,26)\\38\end{matrix} \end{array}\right) \end{array}\right) \quad \begin{pmatrix}19\\39\end{pmatrix} \end{array}\right)$$

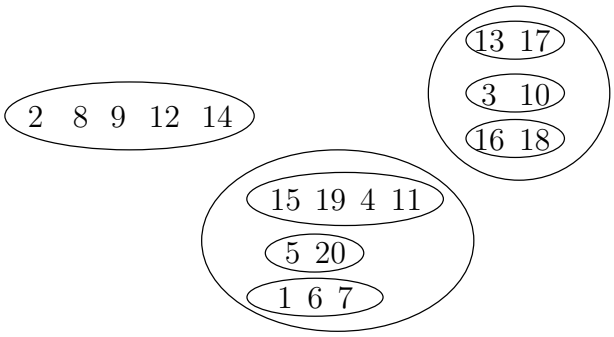
Cluster 1 und Cluster 3 sind gut zu erkennen (oben Mitte in blau bzw. unten in grün), während Cluster 2 die verbleibenden Relationen zugeordnet wurden – einschließlich der beiden „Ausreißer“ 19 und 39.

4.3.3 Beschaffungslogistik

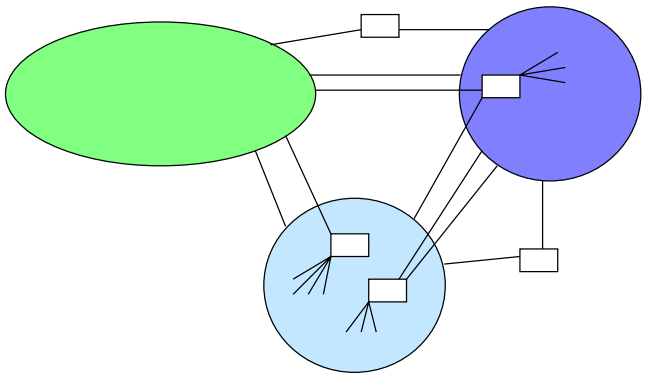
Wiederum sind die Relationen durchnummeriert:

1	Beschaffungsbeleg	11	Lieferant-Text-Zuo
2	Beschaffungsbelegposition	12	Lieferanteninformationsquelle
3	Lieferantenkondition	13	Lieferantenkondition-Text-Zuo
4	Kontierung	14	Kontierung-Zuo
5	Lieferantenmerkmal-Zuo	15	Einkaeufergruppierung
6	Lieferanten-Zuo	16	Materialmerkmal-Zuo
7	Beschaffungsweg-Zuo	17	Fremdgut-Text-Zuo
8	Zuo	18	Fremdgut-Beschaffungsauftrag-Zuo
9	Zeitreihe-1	19	Fremdgut-Zuo
10	Zeitreihe-2	20	Lieferant-Beschaffungsauftragstyp-Zuo

Die folgenden Cluster wurden mit der Tanimoto-Metrik und der Proximität nach Ward gebildet:



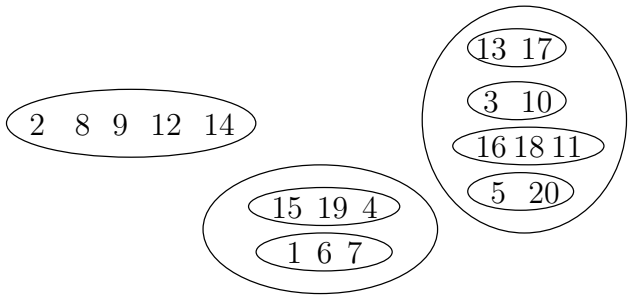
Die Güte dieser Clusterung zeigt das nächste Bild, das zwar die Cluster nur noch als Ellipsen zeigt, dafür aber noch die Verbindungen zwischen diesen darstellt. Verbindende Linien stellen Abhängigkeiten zwischen Relationen in den Clustern dar, Rechtecke stehen für Entitäten, von denen Relationen abhängig sind. Man findet nur wenige Linien, die von Cluster zu Cluster gehen, wenn die Entitäten „richtig“ den Clustern zugeordnet werden:



Nur für zwei explizit gezeigte Entitäten ist keine klare Zuordnung zu einem Cluster möglich, und nur die in den Clustern gezeigten Entitäten werden überhaupt aus einem anderen Cluster heraus referenziert.

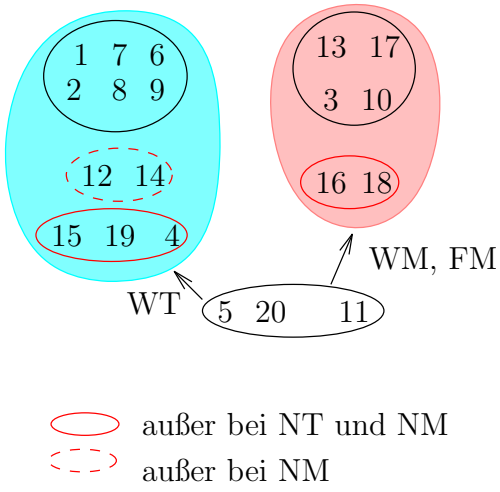
Es wird deutlich, dass mit der Clusterung nach Ähnlichkeitswerten eine bessere Anordnung der Entitäten und Relationen im ER-Diagramm erreicht werden kann.

Mit der M-Metrik entsteht:



Beide Clusterungen unterscheiden sich nur an wenigen Stellen.

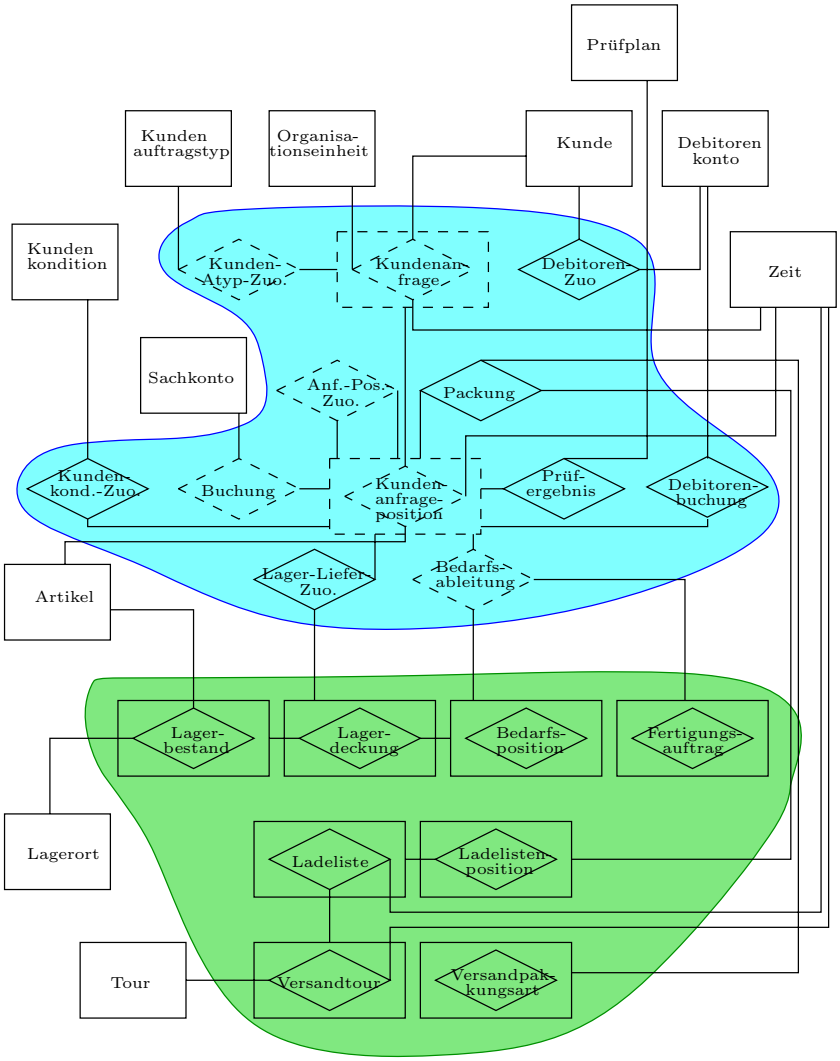
Das folgende Bild stellt die beiden stabilsten Cluster für die sechs verschiedenen Methoden der Clusterbildung dar, den einen in cyan, den anderen in rosa. Schwarz umrandet sind jene Relationsnummern, die bei jeder Methode im Cluster enthalten sind. Für die rot umrandeten ist angegeben, bei welcher Methode sie nicht mehr enthalten sind. Drei Relationen gehören nur bei drei Methoden zu diesen Clustern, die Pfeile geben Details an.



Man erkennt einen klaren „harten Kern“ in beiden Clustern sowie eine Menge von Relationen, die mal dem einen, mal dem anderen Cluster zugeordnet ist, und die deshalb auch als eigener Cluster aufgefasst werden kann.

4.3.4 Vertriebsabwicklung

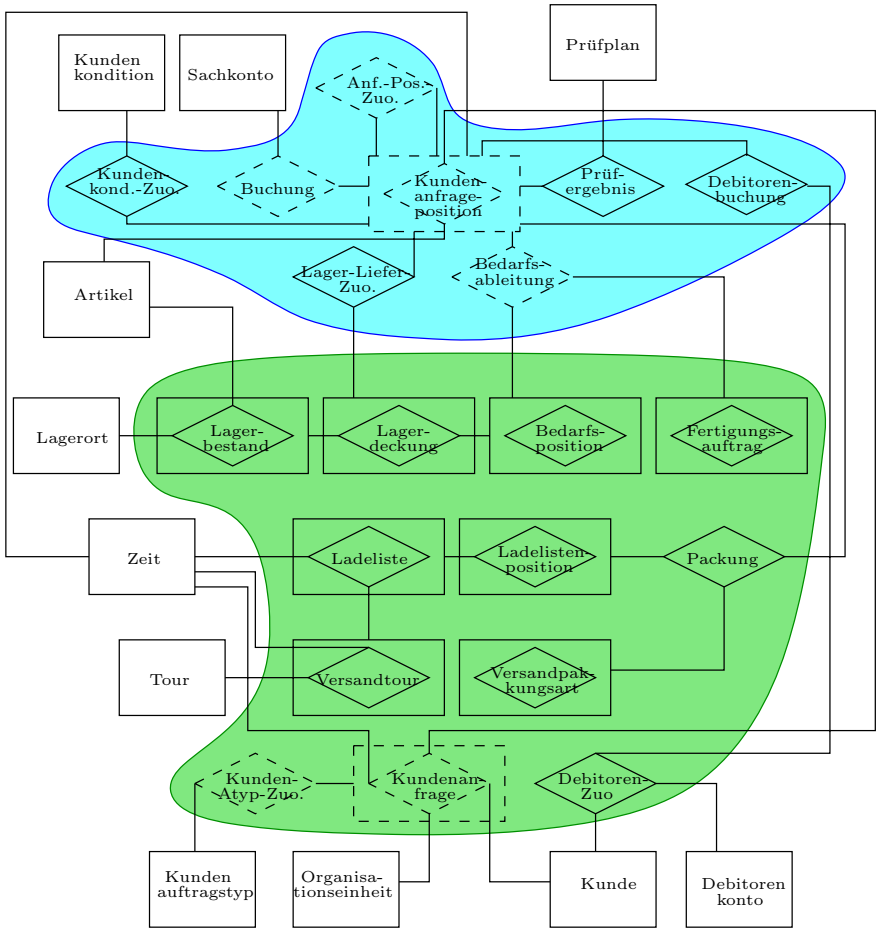
Die Methode **WT** soll diesmal im Detail gezeigt werden. Man erhält die folgende Zerlegung der Relationenmenge in zwei große farbige Cluster:



Der untere Cluster enthält dabei noch weitere hier nicht gezeigte, aber im Originalmodell vorhandene Relationen. Jede Zeile in den Clustern stellt ein Untercluster dar. Im Vergleich mit dem Originalmodell[4] wird der Gewinn an Übersichtlichkeit deutlich, die um die Cluster angeordneten Entitäten machen die Verwandtschaft der Relationen gut sichtbar, auch lassen sich klare Schnittstellen zwischen den beiden Clustern erkennen.

Das folgende Bild verwendet die M-Metrik (**WM**). Zeilen sind dabei keine Untercluster mehr:

[4][SC97], S. 458f, Abb. B.II.25, oder Seite 91



Überraschenderweise werden nun die Relationen `Kundenanfrage` und `Kundenanfrageposition` verschiedenen Clustern zugeordnet. Die Trennung wirkt geringfügig plausibler und die Schnittstellen zwischen den beiden Clustern sind verbessert, wenn man auch die Entitäten den richtigen Clustern zuordnet, und dabei insbesondere die Entität `Zeit` dem unteren Cluster.

Per Augenschein liegen aber Verbesserungen nahe, so sind offenbar `Lager-Liefer-Zuo` und `Bedarfsableitung` im unteren Cluster besser aufgehoben. Dennoch, die Clusterung ergibt auch hier eine Verbesserung der Anordnung gegenüber der Vorlage in der Originalarbeit und ist zumindest ein guter Ausgangspunkt für weitere Optimierungen.

4.3.5 Gesamtmodell

Die vier bisher betrachteten Datenmodelle sind Teile eines Gesamtmodells. Im folgenden sollen sie zusammengefügt werden. Man erwartet, dass die

Clusteranalyse für die vereinigten Datenmodelle wieder auf die vier Teilmodelle führt. Falls nicht, erhält man evtl. Aufschluss über andere Gliederungsmöglichkeiten des Gesamtmodells.

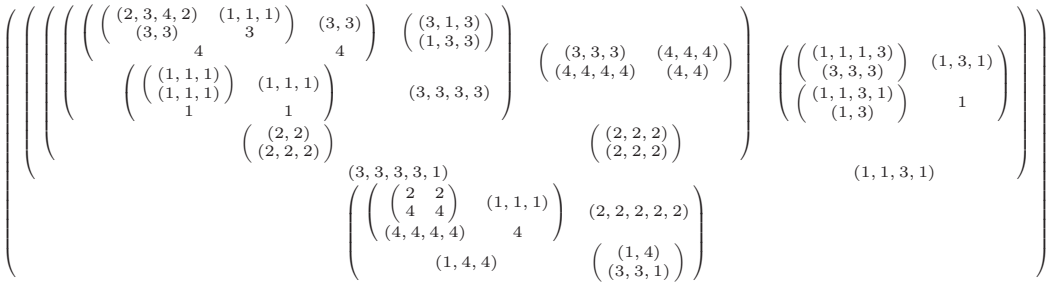
Die folgende Abbildung zeigt das komplette Dendogramm für alle 129 Relationen bei Verwendung der Tanimoto-Metrik und des Proximitätsmaßes von Ward. Es sind dabei der Übersicht halber nicht mehr die Relationsnummern gezeigt. Stattdessen steht für jede Relation nur die Nummer des Teilmodells, in dem sie vorkommt. Die Teilmodellnummern sind[5]:

1. für Bedarfs-, Zeit- und Kapazitätsplanung, ([SC97], S. 176, Abb. B.I.66, S. 254f, Abb. B.I.133)

2. für Beschaffungslogistik, ([SC97], S. 418f, Abb. B.II.07)

3. für CAM ([SC97], S. 366f, Abb. B.I.221) und

4. für Vertriebsabwicklung ([SC97], S. 458f, Abb. B.II.25).



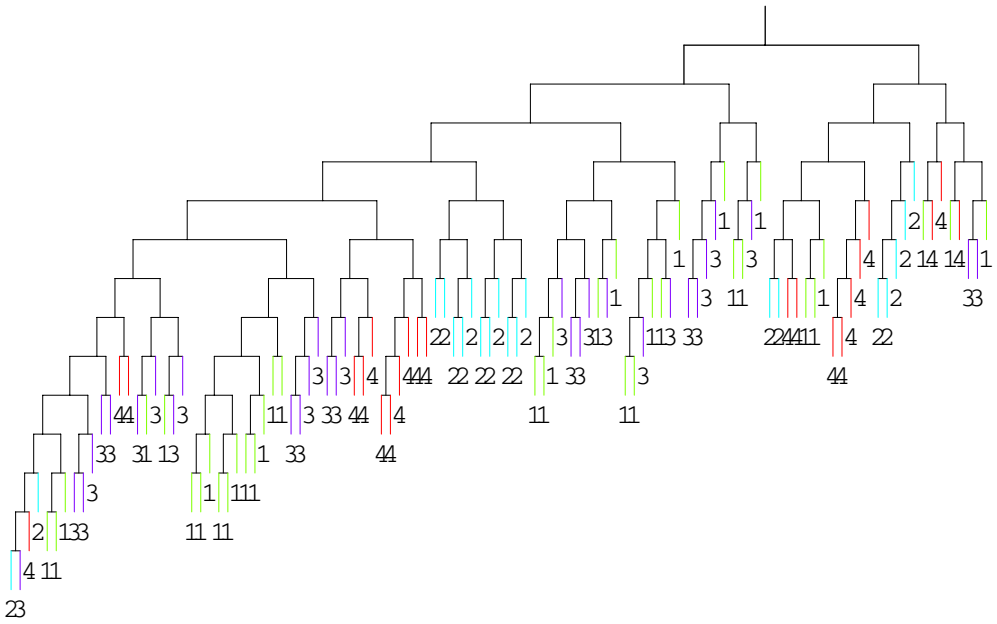
Man erkennt deutlich, dass sich die Relationen eines Modells auch in Clustern zusammenfügen, dass aber auch häufig Relationen verschiedener Teilmodelle zusammengefasst werden. Dies ist klar der Fall für die Modelle 1 und 3, also Bedarfsplanung etc. und CAM. Die Clusterung mit anderen Proximitätsmaßen und Metriken zeigt ein ähnliches Verhalten (ohne Abbildung). Betrachtet man die einzelnen Relationen, so findet man einige, die bei allen Clusterungsvarianten ein und demselben *anderen* Teilmodell zugeordnet werden. Die Clusteranalyse nach Ähnlichkeitsmetriken kann also konkrete Hinweise zur Umorganisation der Teilmodelle geben.

[5]Wegen Überschneidungen der Teilmodelle kommen einige Relationen in mehreren Teilmodellen vor, sie wurden willkürlich einem Datenmodell zugeordnet

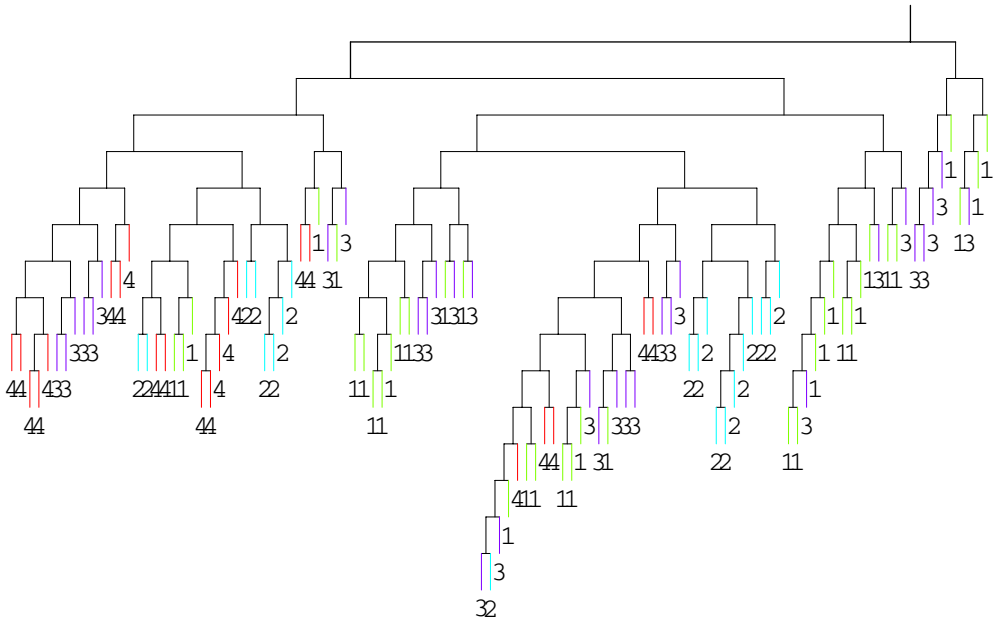
Im Einzelnen handelt es sich dabei z.B. um `Prüfplan_Zuo` und `NC_Sequenz_Zuo`, die der Bedarfsplanung statt dem CAM-Teilmodell zugeordnet werden; `Primärbedarfsposition` wird der Vertriebsabwicklung statt der Bedarfsplanung, `Auftragsposition` und `Auftragskopf` (beide stehen für eine Gruppe von strukturidentischen Relationen) werden Vertrieb statt Bedarfsplanung, und `AbgesKundenAuftrag` und `Kunden_AbgesAuftragstyp_Zuo` werden der Beschaffungslogistik statt dem Vertrieb zugeordnet.

Es wird aber auch deutlich, dass nicht die gesamten Teilmodelle – auch nicht im Wesentlichen – als Cluster auftauchen, sondern stets Untermodelle, die dann wechselnde Beziehungen eingehen. Dabei ist die graphische Darstellung in Matrixform noch vergleichsweise übersichtlich. Zeigt man die von der Clusteranalyse gelieferte Dendogrammstruktur, so sind die Teilmodelle nur schwer als Teilstruktur zu erkennen. Die folgenden Bilder zeigen die Cluster als vollständige Dendogramme. Abgebildet sind der Reihe die Methoden **WT**, **WM** und **FM**. Die anderen Varianten eignen sich nicht für diese Darstellung, da sie sehr lange Ketten enthalten.

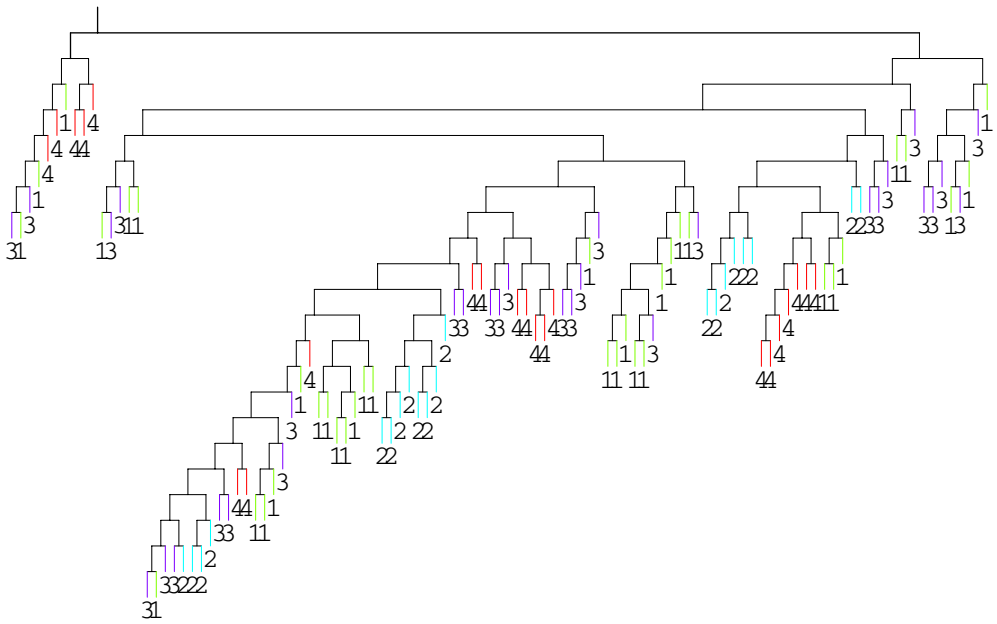
WT:



WM:



FM:

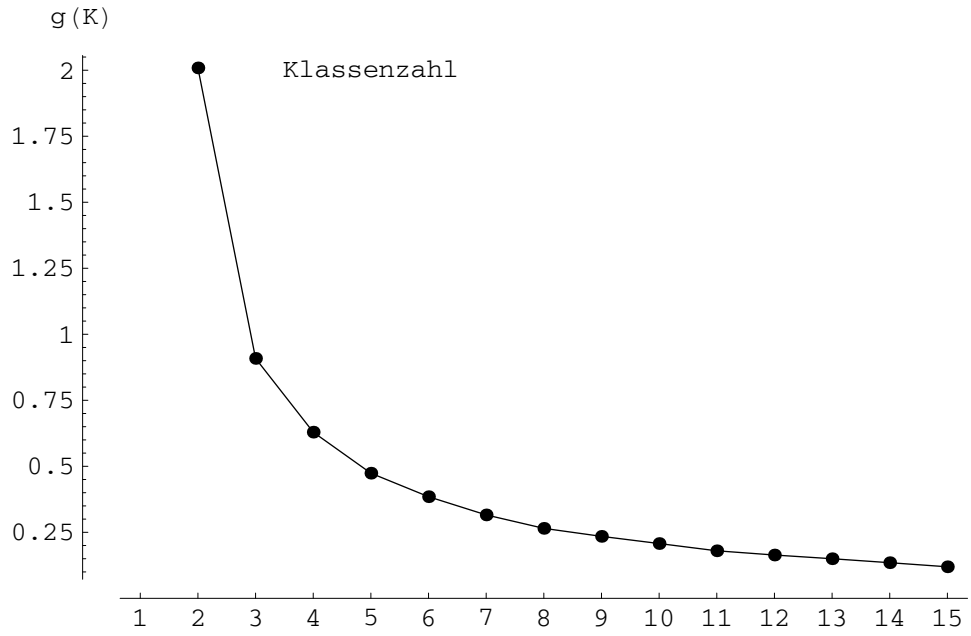


Die Teilmodelle sind zusätzlich mit Farben kodiert:

- Bedarfsplanung etc. (1) in grün,
- Beschaffungslogistik (2) in cyan,
- CAM (3) in magenta und
- Vertrieb (4) in rot.

4.4 Bestimmung der optimalen Anzahl von Clustern

Das Verfahren der Clusteranalyse geht von einelementigen Clustern aus, die so lange miteinander vereinigt werden, bis nur noch ein Cluster übrig ist. Zu jedem Zeitpunkt liegt damit eine disjunkte Zerlegung der Objekte in Cluster vor. Die Cluster einer solchen Zerlegung heißen Klassen. Deren Anzahl liegt also zwischen der Anzahl der Objekte und 1. Die bestgeeignete Gruppierung der Objekte in Klassen ist gesucht. Um diese zu finden, gibt es verschiedene Verfahren, die, ausgehend von einer Heterogenitätsfunktion für Klassen, einen Gütewert für eine Klassifikation bestimmen. Die Einteilung in Klassen versucht, eine gute Kombination von kleiner Klassenzahl und hoher Güte der Klassifikation zu finden, siehe z.B. [EV01]. Da hier ein großer Interpretationsspielraum besteht, soll nur ein Beispiel gegeben werden, nämlich die aus dem Dendogramm zur Ward-Proximität und zur Tanimoto-Metrik (WT) entstehende Graphik, die die Güte der Klassenbildung gegen die Anzahl Klassen aufträgt:



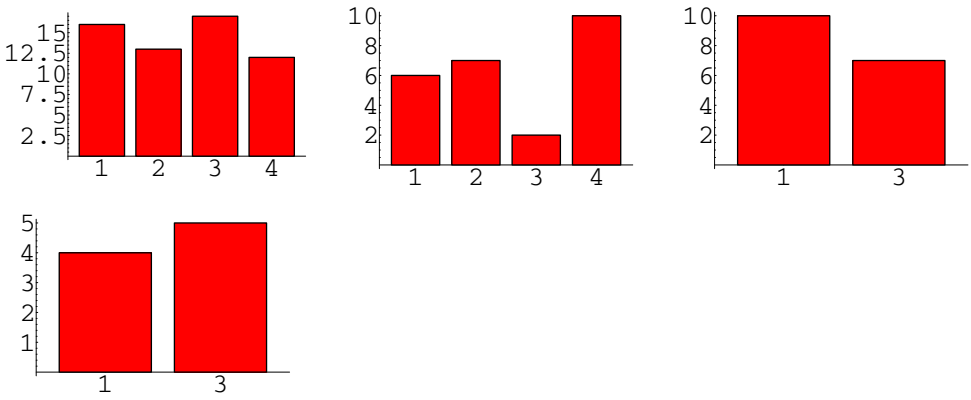
Hierbei wurden verwendet: die maximale Distanz $h(K)$ innerhalb einer Klasse als Heterogenitätsmaß und die gewichtete Summe der Heterogenitäten über alle Klassen $g(\mathcal{K})$ als Maßzahl der Güte der Klassenzerlegung, in Formeln:

$$\begin{aligned} h(K) &:= \max_{R_1,R_2\in K} d(R_1,R_2), \\ g(\mathcal{K}) &:= \frac{1}{w(\mathcal{K})}\sum_{K\in\mathcal{K}} h(K), \text{ mit} \\ w(\mathcal{K}) &:= \sum_{K_1,K_2\in\mathcal{K}} v(K_1,K_2), \text{ wobei} \\ v(K_1,K_2) &:= \frac{1}{|K_1||K_2|}\sum_{\substack{R_1\in K_1\\ R_2\in K_2}} d(R_1,R_2) \text{ ist.} \end{aligned}$$

Einer hohen Güte entsprechen kleine Werte von $g(\mathcal{K})$. $g(\mathcal{K})$ nimmt mit zunehmender Klassenzahl ab und erreicht den Wert Null, wenn jede Relation eine eigene Klasse bildet. Man sucht deshalb eine geeignete nicht zu große Klassenzahl mit befriedigender Güte aus.

Die entstehende Graphik weist evtl. einen deutlichen Knick, den „Ellenbogen“ auf, hinter welchem eine Erhöhung der Klassenzahl nur noch eine geringe Zunahme der Güte bewirkt. Dieser Knick kann als Ort der optimalen Klasseneinteilung gesehen werden.

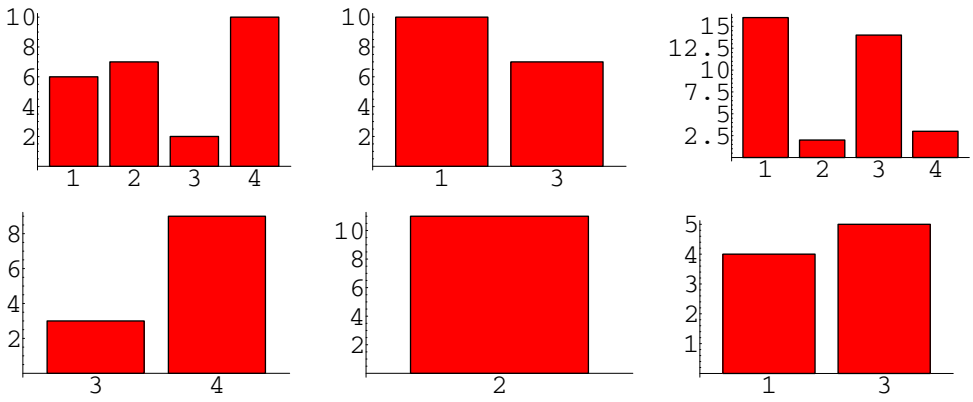
Freundlicherweise findet sich ein deutlicher Ellbogen bei der Klassenzahl vier, was gerade die Anzahl Teilmodelle ist, die Gegenstand der Clusteranalyse waren!



Insofern hat die Clusteranalyse die richtige Anzahl wiedergefunden. Allerdings bleibt es bei dem Ergebnis, das die Zuordnung der Relationen zu den

Klassen nicht mit der Zerlegung in die vorgegebenen Teilmodelle identisch ist. Es zeigt sich also, dass diese Art der Clusteranalyse anhand der Ähnlichkeitswerte zu vernünftigen Ergebnissen führen und gleichzeitig Hinweise auf eine Umorganisation der ER-Modelle geben kann.

Man könnte sich allerdings auch dafür entscheiden, den Ellbogen bei der Klassenzahl 6 zu sehen. In diesem Fall zeigt das folgende Bild, wie sich die Relationen in den Klassen auf die Teilmodelle verteilen:



Man erkennt, dass die Teilmodelle 1 und 3 nicht einzeln vorkommen, sondern gemeinsam den Hauptbestandteil von dreien der gebildeten Klassen stellen, dass Teilmodell 4 in verschiedenen Kombinationen vorkommt, insbesondere zusammen mit Relationen aus Teilmodell 3, und Teilmodell 2 stellt alle Mitglieder einer der neuen Klassen.

Zur Dokumentation sind hier die Relationen in den Klassen komplett aufgelistet:

1. AbgesKundenauftrag, AbgesKundenauftragsposition, Auftragskopf, Auftragsposition, Beschaffungsbeleg, Beschaffungsbelegposition, Lagerdeckung, Lagerspiel, Primaerbedarfsposition, Reservierung, Bedarfsableitung, Beschaffungsweg_Zuo, Buchung, Debitorenbuchung, Folgeauftrag, Gruppierung, Kontierung_Zuo, KundenAbgesauftragstyp_Zuo, Kundenkonditionen_Zuo, Lager_Liefer_Zuo, Lieferanteninformationsquelle, Packung, Pruefergebnis, Zeitreihe_1, Zuo
2. Arbeitsgang, Arbeitsgang_Zuo, Mitarbeiter_Zuo, Mitarbeiterbelegung, NC_Programm, Werkzeug_Zuo, Werkzeugeinsatz, Arbeitsgruppierung, Bereichsgruppierung, MitarbeiterMaschinenbelegung, Mitarbeiter_Unterbrechung_Zuo, NC_Sequenz_Zuo, Pruefplan_Zuo, Verfahren_Zuo, Vorranggraph, Werkzeugbelegung, APLGrp_Station_Zuo

3. Arbeitsplan_Zuo, Arbeitsplanreihenfolge, Auftrag, Auftrag_Bereich_Zuo, Auftragkopf, Fertigungsauftrag, Fertigungsauftragskopf, Kontierung, Lagerbestand, Struktur, Teil_Merkmaltyp_Zuo, Teilstrecke, Transport, Wartung_Zuo, Arbeitsgang_Komponente_Zuo, Arbeitsgangbereich_Zuo, Arbeitsplan_Werk_Zuo, Ersatzteil_Zuo, Erzeugt, Werksgruppierung, Debitoren_Zuo, Dispositionsstufe_Zuo, Lager_Station_Zuo, Lager_Zuo, Lieferant_Text_Zuo, Merkmalgruppierung, Palettentyp_Zuo, Palettenverfolgung, Struktur_Merkmal_Zuo, Struktur_Variante_Zuo, Teil_Variante_Zuo, Tour_Zuo, Transportgruppierung, Verteilung, Wartungsplan_Zuo
4. Ladeliste, Ladelisteposition, Packung_Zuo, Transport_Zuo, Transportzusammenstellung, Unterbrechung, Versandpackungsart, Versandtour, Versandtransporteinheit, Versandtransportmittel, Unterbrechung_Schicht_Zuo, Zeitereignis
5. Lieferantenkondition, Einkaeufergruppierung, Fremdgut_Beschaffungsauftrag_Zuo, Fremdgut_Text_Zuo, Fremdgut_Zuo, Lieferant_Beschaffungsauftragtyp_Zuo, Lieferanten_Zuo, Lieferantenkondition_Text_Zuo, Lieferantenmerkmal_Zuo, Materialmerkmal_Zuo, Zeitreihe_2
6. Auftrag_Arbeitsgang_Zuo, Auftragsarbeitsgang, Auftragsarbeitsgang_Zuo, Auftragsarbeitsplan, Maschinenbelegung, Belastung, Position, Standort, Teilladung

Im Anhang B, Seite 86, finden sich Graphiken, die diese Klasseneinteilung darstellen und anhand derer man beurteilen kann, ob und wie gut die Klassenbildung mit fachlichen Gliederungen in Einklang steht.

Der Ellbogen bei der Klassenzahl 4 taucht auch auf, wenn zwei alternative Heterogenitätsmaße verwendet werden, nämlich

$$h_1(K) := \frac{1}{2|K|} \sum_{\substack{R_1,R_2 \in K \\ R_1 \neq R_2}} d(R_1,R_2), \quad \text{(BStandard)}$$

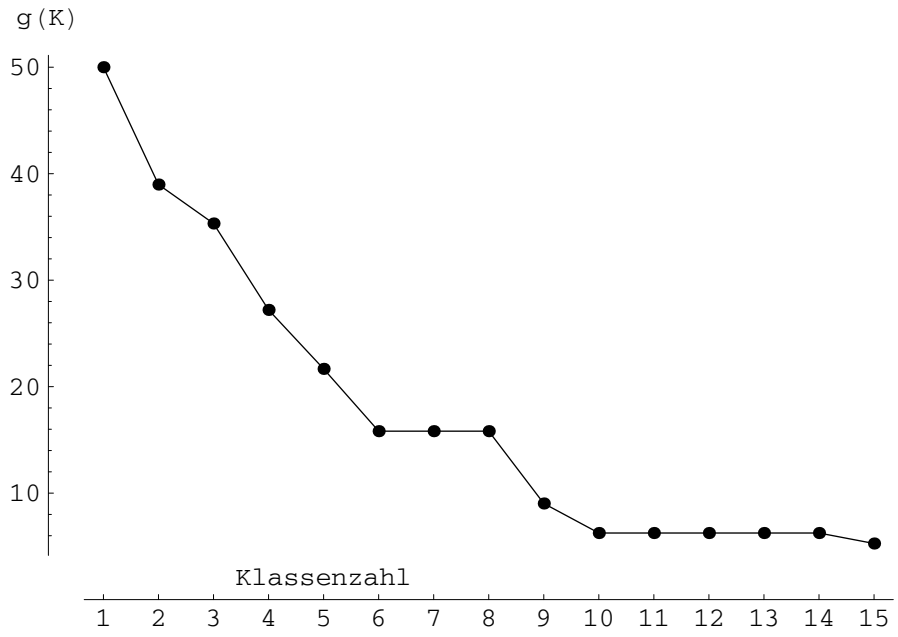
$$h_2(K) := \frac{1}{|K|(|K|+1)} \sum_{\substack{R_1,R_2 \in K \\ R_1 \neq R_2}} d(R_1,R_2), \quad \text{(durchschn. Abstand)}$$

Als andere Gütemaße wurden betrachtet

$$g_1(\mathcal{K}) := \sum_{K \in \mathcal{K}} h(K), \quad \text{(Summe der Heterog.)}$$

$$g_2(\mathcal{K}) := \max_{K \in \mathcal{K}} h(K), \quad \text{(Maximum der Heterog.)}$$

Sie erwiesen sich nur in Verbindung mit dem *BStandard* als Heterogenitätsmaß sinnvoll. g_1 zeigte dabei keinen ausgeprägten Ellbogen im Bereich unter 15 Klassen, g_2 hingegen ergab einen stufenförmigen Verlauf:



Hier kommen als Klassenzahlen 6 oder 10 in Frage.

Kombiniert man g_1 mit h_2, so ist die Güte nicht einmal mehr monoton fallend:

Schließlich kann man noch andere Maße für die Verschiedenheit von Klassen verwenden:

$$\begin{aligned} v_1 &:= \min_{\substack{R_1 \in K_1 \\ R_2 \in K_2}} d(R_1, R_2) \\ v_2 &:= \max_{\substack{R_1 \in K_1 \\ R_2 \in K_2}} d(R_1, R_2) \end{aligned}$$

Deren Verwandschaft zu den beiden Proximitätsmaßen *Furthest* und *Nearest Neighbour* ist offensichtlich, weshalb sie in Verbindung mit diesen verwendet werden sollten. Da sich diese Proximitätsmaße (je nach Metrik) wegen ihrer Neigung zur Bildung langer Ketten nicht so gut zur Clusterbildung eignen wie das Maß von Ward, sind sie nicht weiter verfolgt worden.

4.5 Zuordnung der Entitäten zu Clustern und Klassen

Es sind mindestens zwei Möglichkeiten denkbar, auch die Entitäten in die Clusterung einzubeziehen:

1. Man kann die Entitäten als Relationen auffassen, die nur von sich selbst abhängig sind, im Sinne von

   $$\hat{E}(E) := \{E\}.$$

   Damit können Ähnlichkeitswerte zwischen Entitäten und Relationen bestimmt werden (zwei verschiedene Entitäten haben stets Ähnlichkeit 0), und die Entitäten können dann wie andere Relationen behandelt werden.

2. Man führt Clusterungen ohne Entitäten durch und ordnet anschließend jede Entität dem Cluster zu, zu dem es die größte Anzahl Beziehungen hat. Hierbei kann es häufig vorkommen, dass eine Entität zu zwei oder mehr Clustern die gleiche, maximale Anzahl Beziehungen hat.

Für das Teilmodell Beschaffungslogistik ist die erste Methode durchgeführt worden. Bildet man aus dem Dendogramm vier Klassen, so sehen diese ohne die Entitäten wie folgt aus:

1. Kontierung, Lieferant_Text_Zuo, Einkaeufergruppierung, Fremdgut_Zuo

2. Beschaffungsbeleg, Beschaffungsbelegposition, Beschaffungsweg_Zuo, Zuo, Zeitreihe_1, Lieferanteninformationsquelle, Kontierung_Zuo

3. Lieferantenkondition, Zeitreihe_2, Lieferantenkondition_Text_Zuo, Materialmerkmal_Zuo, Fremdgut_Text_Zuo, Fremdgut_Beschaffungsauftrag_Zuo

4. Lieferantenmerkmal_Zuo, Lieferanten_Zuo, Lieferant_Beschaffungsauftragtyp_Zuo

Mit den Entitäten ergibt sich

1. Kontierung, Lieferant_Text_Zuo, Einkaeufergruppierung, Fremdgut_Zuo, *Bestellanforderung*, *Beschaffungsweg*, *Zusatzdaten*, *Periodenraster*, *Konto*, *Mitarbeiter*, *Einkaeufer*, *Fertigungsauftrag*, *Kostenstelle*, *Kostentraeger*, *Einkaufsgruppe*

2. Beschaffungsbeleg, Beschaffungsbelegposition, Beschaffungsweg_Zuo, Zuo, Zeitreihe_1, Lieferanteninformationsquelle, Kontierung_Zuo, *Zeit*, *Organisationseinheit*

3. Lieferantenkondition, Zeitreihe_2, Lieferantenkondition_Text_Zuo, Materialmerkmal_Zuo, Fremdgut_Text_Zuo, *Lieferantentext*, *Materialmerkmal*, *Fremdgut*

4. Lieferantenmerkmal_Zuo, Lieferanten_Zuo, Fremdgut_Beschaffungsauftrag_Zuo, Lieferant_Beschaffungsauftragtyp_Zuo, *Lieferantenmerkmal*, *Lieferant*, *Beschaffungsauftragstyp*

Hinzugekommene Entitäten sind kursiv geschrieben. An einer Stelle hat sich die Klassenzugehörigkeit einer Relation geändert: `Fremdgut_Beschaffungsauftrag_Zuo` ist von der 3. in die 4. Klasse gewandert.

Die Klassenbildung in diesem Beispiel ist kritisch zu sehen. Insbesondere die erste Klasse besteht aus vier schlecht zusammenpassenden Relationen, bekommt aber eine große Zahl von Entitäten zugeordnet. Das hat seine Ursache in den geringen Ähnlichkeitswerten, die Entitäten zu Relationen aufweisen, weshalb sie erst spät in das Dendogramm aufgenommen werden. Diese Vorgehensweise neigt also dazu, viele Entitäten der letzten „Lumpensammler"-Klasse zuzuordnen.

Bei der zweiten Methode, Entitäten den Klassen zuzuordnen, erhält man als Ergebnis

1. Kontierung, Lieferant_Text_Zuo, Einkaeufergruppierung, Fremdgut_Zuo, *Konto, Einkaeufer, Kostenstelle, Kostentraeger, Einkaufsgruppe,*

2. Beschaffungsbeleg, Beschaffungsbelegposition, Beschaffungsweg_Zuo, Zuo, Zeitreihe_1, Lieferanteninformationsquelle, Kontierung_Zuo, *Lieferant, Beschaffungsweg, Zusatzdaten, Periodenraster, Zeit, Organisationseinheit, Konto, Kostenstelle, Kostentraeger, Fremdgut*

3. Lieferantenkondition, Zeitreihe_2, Lieferantenkondition_Text_Zuo, Materialmerkmal_Zuo, Fremdgut_Text_Zuo, Fremdgut_Beschaffungsauftrag_-Zuo, *Periodenraster, Lieferantentext, Beschaffungsauftragstyp, Materialmerkmal*

4. Lieferantenmerkmal_Zuo, Lieferanten_Zuo, Lieferant_Beschaffungsauftragtyp_Zuo, *Lieferantenmerkmal, Beschaffungsauftragstyp, Einkaufsgruppe*

Die unterstrichenen Entitäten sind nicht klar zuzuordnen, da sie zu beiden Clustern gleich viele Beziehungen haben. Diese Methode ergibt eine deutlich sinnvollere Zuordnung zu den gegebenen Klassen - unabhängig davon, wie die Klassen selbst zu beurteilen sind.

4.6 Zusammenfassung

Die in den vorigen Abschnitten diskutierten Beispiele zeigen deutlich, dass sich mittels der Clusterbildung auch aus Ähnlichkeitswerten, die weder 0 noch 1 sind, ein deutlicher Nutzen für eine übersichtlichere Gestaltung und Anordnung der Entitäten und Relationen eines Datenmodells gewinnen lässt. Eine klare Empfehlung für die Auswahl des Verfahrens zur Clusterbildung und zur optimalen Auswahl der Klassenzahl wäre wie folgt:

Metrik: Tanimoto-Metrik

Proximitätsmaß: Wardsches Maß

Heterogenitätsmaß: maximale Distanz innerhalb einer Klasse

Gütemaß: gewichtete Summe der Heterogenitäten

Die Ergebnisse einer Clusteranalyse sind aber im Kern weitgehend stabil gegen eine Änderung der Methode. Durch Vergleich der Ergebnisse bei Einsatz verschiedener Varianten lassen sich noch Verbesserungen durch manuelle Nacharbeit erreichen.

5 Implementierung

Die beschriebenen Verfahren zur Berechnung von Ähnlichkeitswerten, zur Suche nach strukturidentischen Relationen und zur Bildung von Clustern wurden mit Mathematica® implementiert. Der Quellcode liegt in Form eines Mathematica-*Notebooks* vor. Die Verwendung erfolgt einfach dadurch, dass das Notebook in einer Mathematica-Sitzung ausgeführt wird.

Die Übergabe des Datenmodells wurde sehr einfach gehalten und ist von der besonderen Modellierungsvariante unabhängig. Die meisten Funktionen erwarten als Eingabe eine Liste von Paaren `{R,E}`, wobei ein solches Paar dafür steht, dass die Relation R über der Entität E definiert ist. Ist die Relation über mehreren Entitäten definiert, so sind entsprechend viele Paare mit R als linkem Eintrag in die Liste aufzunehmen. Das gilt auch dann, wenn R über mehreren Exemplaren derselben Entität definiert ist. Zwischen identifizierenden und nicht identifizierenden Beziehungen wird nicht unterschieden. Die Relationen und Entitäten können durch Zahlen, durch Zeichenketten oder durch Symbole repräsentiert werden.

Bei der Ausführung des Notebooks werden von mehreren Alternativen für die verwendete Metrik, die Proximitätsfunktion, das Heterogenitäts- und das Gütemaß die jeweils letzte im Quellcode aufgelistete Definition zur Default-Einstellung. Werden andere Funktionen gewünscht, so sind die entsprechenden Definitionen von Hand auszuführen, dadurch wird der Default überschrieben.

Es folgt eine kommentierte Auflistung des Quellcodes. Anschließend wird ein Mathematica-Notebook als Beispiel abgedruckt, in welchem die meisten Funktionen mit Eingabe und Ausgabe gezeigt werden.

Vorbereitungen

Notwendige Packages

```
Needs["DiscreteMath`Combinatorica`"]

Needs["DiscreteMath`Tree`"]

Needs["LinearAlgebra`MatrixManipulation`"]
```

Hilfsfunktion zum durchgreifenden Sortieren, korrektem First, korrektem Flatten, sowie viele andere

```
SortListsOnly[l_List]:=l//laufSort

SortListsOnly[a_]:=a/;AtomQ[a]

firstOrEmpty[l_List]:=First[l]

firstOrEmpty[{}]:={}

flattenOrAtom[l_List]:=l//Flatten

flattenOrAtom[a_]:=a/;AtomQ[a]

laufSort[l_]:=Flatten[l//Sort//Split//Sort,1]

MultiIntersection[l1_List,l2_List]:=
  Module[{nl,f},f[x_]:={First[#],Length[#]}&/@Split[Sort[x]];
    nl=Sort[Join[Flatten[Map[f,{l1,l2}],1]]];nl=Split[nl,#[[1]]===#2[[1]]&];
    Flatten[Cases[nl,{{x_,m_},{x_,n_}}:->Table[x,{m}]],1]]

MultiUnion[l1_List,l2_List]:=
  Module[{nl,f},f[x_]:={First[#],Length[#]}&/@Split[Sort[x]];
    nl=Sort[Join[Flatten[Map[f,{l1,l2}],1]]];nl=Split[nl,#[[1]]===#2[[1]]&];
    Flatten[{Cases[nl,{{x_,m_},{x_,n_}}:->Table[x,{n}]],
        Cases[nl,{{x_,m_}}:->Table[x,{m}]]},2]]

aehnMaxStruktur[ents1_,ents2_]:=Module[{level1,level2},
    level1=
      Flatten[Subsets/@Select[Level[ents1,{0,Infinity}],\[Not]AtomQ[#]&],1]//
        Union;level2=
      Flatten[Subsets/@Select[Level[ents2,{0,Infinity}],\[Not]AtomQ[#]&],1]//
```

```
        Union;Sort[level1\[Intersection]level2,
        Length[#1//flattenOrAtom]>Length[#2//flattenOrAtom]&]//firstOrEmpty
    ]

nestedEntsRule[n_,ae_]:=n->(Select[ae,#[[1]]==n&][[1,2]])

applyNestedRules[rs_,ae_]:={#[[1]],#[[2]]/.rs}&/@ae

Avg[l_]:=(Plus@@l)/Length[l]

het[{a_},_]:=0

DistanceMatrix[arrs_]:=Module[{d},
    d=tableDistances[arrs]//N;
    ReplacePart[d,\[Infinity],Table[{i,i},{i,1,Dimensions[d][[1]]}]]
    ]

gew[Cf_,DM_]:=
  Plus@@(Table[ver[Cf[[i]],Cf[[j]],DM],{i,1,Length[Cf]},{j,i+1,Length[Cf]}]//
        Flatten)

ver[c1_,c2_,DM_]:=
  Plus@@(Table[DM[[c1[[i]],c2[[j]]]],{i,1,Length[c1]},{j,1,Length[c2]}]//
           Flatten)/Length[c1]/Length[c2]

ExprPlot[expr_]:=Module[{ex,g,xlist,xhilf,xshift,nx},g=ExprPlot0[expr,0,0,1];
    xlist=Cases[g,{a_,_}/;AtomQ[a]->a,Infinity]//Union//Sort;
    nx=Length[xlist]//Range;
    xhilf={xlist,nx}//Transpose;
    xshift=Cases[xhilf,{0,a_}->a];
    xrules=#[[1]]->#[[2]]&/@({xlist,nx-xshift[[1]]}//Transpose);
    Graphics[g/.xrules]]

ExprPlot0[f_[children__],x_,y_,n_]:=
  Block[{xl,xr,c,xi,gnew,gthis,i,dx},c={children};
    If[Length[c]==1,
      Return[Flatten[{Line[{{x,-y},{x,-y-1}}],
           ExprPlot0[First[c],x,y+1,1]}]]];
    xl=x-$TreeWidth^(-y) 2/n//N;
    xr=x+$TreeWidth^(-y) 2/n//N;
    dx=N[(xr-xl)/(Length[c]-1)];
    gnew=Table[
        If[!AtomQ[c[[i+1]]],
```

```
          ExprPlot0[c[[i+1]],xl+i dx,y+1,
            Length[c]],{Text[c[[i+1]],{xl+i dx,-y-1.5},{0,1},{0,1}]}],{i,0,
          Length[c]-1}];
    gthis=Table[xi=xl+i dx;
        { Line[{{xi,-y},{xi,-y-1}}],{Hue[0,1,0],Thickness[0.001]}},{i,0,
          Length[c]-1}];
    Flatten[{Line[{{xl,-y},{xr,-y}}],gthis,gnew}]]

ExprPlot0[e_,x_,y_,n_]:={Text[e,{x,-y}]}

$TreeWidth=2.1;
$TreeHeight=0.8;
```

Einstellungen (zur Änderung der Defaults)

Defaults sind: Tanimoto-Metrik, Proximität von Ward, maximale Distanz als Heterogenitätsmaß, gewichtete Summe der Heterogenitäten als Gütemaß.

Auswahl der Metrik

Simple-Matching-Metrik (M-Metrik)

```
tableDistances=tableMDistances
```

Tanimoto-Metrik (M-Metrik)

```
tableDistances=tableTanimoto
```

Proximitätsmaße für die Distanz zu einem Cluster

Nearest Neighbour (Complete Linkage)

```
newdist[a_,{n_,m_},D_,C_]:=Min[D[[a,m]],D[[a,n]]];
```

Furthest Neighbour (Single Linkage)

```
newdist[a_,{n_,m_},D_,C_]:=Max[D[[a,m]],D[[a,n]]];
```

Average Linkage ungewichtet

```
newdist[a_, {n_, m_}, D_, C_] :=
  (1\2) ((D[[a, m]] + D[[a, n]]));
```

Average Linkage gewichtet

```
newdist[a_, {n_, m_}, D_, C_] :=
    Module[{la, ln, lm, an, am, nm},
    la = Length[{C[[a]]} // Flatten];
ln = Length[{C[[n]]} // Flatten];
lm = Length[{C[[m]]} // Flatten];
an = D[[a, n]];
am = D[[a, m]];
nm = D[[n, m]]; (ln an + lm am)/(ln + lm)
    ]
```

Centroid

```
newdist[a_, {n_, m_}, D_, C_] :=
    Module[{la, ln, lm, an, am, nm}, la =
        Length[{C[[a]]} // Flatten];
ln = Length[{C[[n]]} // Flatten];
lm = Length[{C[[m]]} // Flatten];
an = D[[a, n]];
am = D[[a, m]];
nm = D[[n, m]];
(ln an + lm am)/(ln + lm) - (ln lm nm)/((ln + lm))^2
 ]
```

Median

```
newdist[a_, {n_, m_}, D_, C_] :=
    (1/2) ((D[[a, m]] + D[[a, n]]) - (1/4)D[[n, m]]
```

Ward

```
newdist[a_, {n_, m_}, D_, C_] :=
    Module[{la, ln, lm, an, am, nm},
    la = Length[{C[[a]]} // Flatten];
ln = Length[{C[[n]]} // Flatten];
```

```
lm = Length[{C[[m]]} // Flatten];
an = D[[a, n]];
am = D[[a, m]];
nm = D[[n, m]];
(((la + ln)) an + ((la + lm)) am - la nm)/(la + ln + lm))
```

Heterogenitätsmaße

durchschnittliche Distanz

```
het[c_,DM_]:=
  Table[DM[[c[[i]],c[[j]]]],{i,1,Length[c]},{j,i+1,
            Length[c]}]/.{\[Infinity]->0}//Flatten//Avg
```

mittlere Distanz für BStandard

```
het[c_,DM_]:=(Plus@@(Table[
                DM[[c[[i]],c[[j]]]],{i,1,Length[c]},{j,i+1,
                  Length[c]}]/.{\[Infinity]->0}//Flatten))/Length[c]
```

maximale Distanz

```
het[c_,DM_]:=
  Table[DM[[c[[i]],c[[j]]]],{i,1,Length[c]},{j,i+1,
          Length[c]}]/.{\[Infinity]->0}//Max
```

Gütefunktionen

Summe der Heterogenitäten

```
guet[Cf_,DM_]:=Plus@@(het[#,DM]&/@Cf//Flatten)
```

Maximum der Heterogenitäten

```
guet[Cf_,DM_]:=Max@@(het[#,DM]&/@Cf//Flatten)
```

Summe der Heterogenitäten, gewichtet

```
guet[Cf_,DM_]:=Plus@@(het[#,DM]&/@Cf//Flatten)/gew[Cf,DM]
```

Funktionen zur Analyse des Datenmodells

Das Datenmodell wird übergeben in einer „Arrows“ genannten Datenstruktur.
Aufbau: {{r1,e1},{r2,e2},{r3,e3},...}.
Jeder Eintrag {r1,e1} entspricht einer Relation r1, die über der Entität e1 definiert ist. Ist eine Relation über mehreren Entitäten definiert, so gibt es entsprechend viele Einträge. Parallele Beziehungen führen zu identischen Einträgen.

Funktionen für den Anwender: Ähnlichkeitswerte, Metriken, Strukturen

```
nestedEntitiesOfRelations
Rückgabe: Liste der Gestalt {{r1,ents1}{r2,ents2},...}, bestehend aus
Paaren mit einer Relation (erstes Element) und der geschachtelten
Entitätsstruktur, auf der die Relation definiert ist (zweites
Element).
Argument ist die Liste der Arrows.

entitiesOfRelations
Rückgabe: Liste der Gestalt {{r1,ents1}{r2,ents2},...}, bestehend aus
Paaren mit einer Relation (erstes Element) und der Liste der
Entitäten, auf der die Relation definiert ist (zweites Element).
Argument ist die Liste der Arrows.

rangOfRelations
Rückgabe: Liste der Gestalt {{r1,rang1}{r2,rang2},...}, bestehend aus
Paaren mit einer Relation (erstes Element) und dem Rang der Relation
(zweites Element).
Argument ist die Liste der Arrows.

structuredIdentities
Rückgabe: Liste, bestehend aus Mengen strukturidentischer Relationen
(erstes Element) und der geschachtelten Entitätsstruktur, auf der die
Relation(-enmenge) definiert ist (zweites Element).
Strukturidentität beachtet Schachtelung.
Argument ist die Liste der Arrows.

unstructuredIdentities
Rückgabe: Liste, bestehend aus Mengen strukturidentischer Relationen
(erstes Element) und der Liste der Entitäten, auf der die
Relation(-enmenge) definiert ist (zweites Element).
```

```
Strukturidentität beachtet Schachtelung nicht.
Argument ist die Liste der Arrows.

tableSimilarityValues
Rückgabe: Tabelle der unstrukturierten Ähnlichkeitswerte, sortiert
nach Relationsnummern.  Relationsnummern sind nicht Bestandteil der
Tabelle.
Argument ist die Liste der Arrows.

tableStructuredValues
Rückgabe: Tabelle der strukturierten Ähnlichkeitswerte, sortiert nach
Relationsnummern.  Relationsnummern sind nicht Bestandteil der
Tabelle.
Argument ist die Liste der Arrows.

tableTanimoto
Rückgabe: Tabelle der Tanimoto-Distanzen, sortiert nach
Relationsnummern.  Relationsnummern sind nicht Bestandteil der
Tabelle.
Argument ist die Liste der Arrows.

tableMDistances
Rückgabe: Tabelle der M-Distanzen, sortiert nach Relationsnummern.
Relationsnummern sind nicht Bestandteil der Tabelle.
Argument ist die Liste der Arrows.

similarityValue,
structuredValue,
commonStructure,
Tanimoto,
MDistance.
Funktionen, die Ähnlichkeitswerte einzeln abfragen etc.
Argumente sind: Relationsnummer, Relationsnummer, Arrows.
(Nummern müssen nicht konsekutiv sein, sondern so, wie sie in Arrows
vorkommen.)

intermedRelations
Rückgabe: Liste der intemediären Relationen.
Argument ist die Liste der Arrows.

nestedEntitiesOfRelations[arrs_]:=Module[{ae,rs,an},
```

```
    ae={#[[1,1]],#[[2]]}&/@
        Transpose/@
          Split[Sort[arrs,#1[[1]]<#2[[1]]&],#1[[1]]==#2[[1]]&];
    rs=nestedEntsRule[#,ae]&/@(ae//Transpose)[[1]];
    an={#[[1]],SortListsOnly//@#[[2]]}&/@
        FixedPoint[applyNestedRules[rs,#]&,ae];
    Return[an]
    ]

entitiesOfRelations[arrs_]:={#[[1]],#[[2]]//Flatten//Sort}&/@
    nestedEntitiesOfRelations[arrs]

rangOfRelations[arrs_]:={#[[1]],#[[2]]//Depth}&/@
    nestedEntitiesOfRelations[arrs]

structuredIdentities[arrs_]:=Module[{an,ans,anssplit},
    an=nestedEntitiesOfRelations[arrs];
    ans=Sort[an,OrderedQ[{#1[[2]],#2[[2]]}]&];
    anssplit=
      Select[{#[[1]],(#[[2]]//Union)[[1]]}&/@
          Transpose/@Split[ans,#1[[2]]==#2[[2]]&],Length[#[[1]]]>1&];
    Return[anssplit]
    ]

unstructuredIdentities[arrs_]:=Module[{an,ans,anssplit},
    an=entitiesOfRelations[arrs];
    ans=Sort[an,OrderedQ[{#1[[2]],#2[[2]]}]&];
    anssplit=
      Select[{#[[1]],(#[[2]]//Union)[[1]]}&/@
          Transpose/@Split[ans,#1[[2]]==#2[[2]]&],Length[#[[1]]]>1&];
    Return[anssplit]
    ]

tableSimilarityValues[arrs_] :=
    Module[{r, ents, l,
        t1},
r = (((arrs // Transpose))[[1]] // Union) // Sort;
ents = ((entitiesOfRelations[arrs] // Transpose))[[2]];
l = r // Length;
t1 = Table[Length[MultiIntersection[ents[[r]], ents[[s]]]]/
    Length[MultiUnion[ents[[r]], ents[[s]]]],
    {r, 1, l}, {s, 1, l}]]]
```

```
tableStructuredValues[arrs_]:=Module[{r,ents,structs,l,t1},
    r=(arrs//Transpose)[[1]]//Union//Sort;
    ents=(entitiesOfRelations[arrs]//Transpose)[[2]];
    structs=(nestedEntitiesOfRelations[arrs]//Transpose)[[2]];
    l=r//Length;
    t1=Table[
        Length[aehnMaxStruktur[structs[[r]],structs[[s]]]//Flatten]/
          Length[MultiUnion[ents[[r]],ents[[s]]]],{r,1,l},{s,1,l}]
    ]

tableTanimoto[arrs_]:=1-tableSimilarityValues[arrs]

tableMDistances[arrs_]:=Module[{r,ents,l,t1},
    r=(arrs//Transpose)[[1]]//Union//Sort;
    ents=(entitiesOfRelations[arrs]//Transpose)[[2]];
    l=r//Length;
    t1=Table[
        Length[MultiUnion[ents[[r]],ents[[s]]]]-
          Length[MultiIntersection[ents[[r]],ents[[s]]]],{r,1,l},{s,1,l}]
    ]

similarityValue[i_,j_,arrs_]:=Module[{rels,ents,r,s},
    rels=(arrs//Transpose)[[1]]//Union//Sort;
    ents=(entitiesOfRelations[arrs]//Transpose)[[2]];
    r=Position[rels,i][[1,1]];
    s=Position[rels,j][[1,1]];
    Length[MultiIntersection[ents[[r]],ents[[s]]]]/
      Length[MultiUnion[ents[[r]],ents[[s]]]]
    ]

structuredValue[i_,j_,arrs_]:=Module[{rels,ents,structs,r,s},
    rels=(arrs//Transpose)[[1]]//Union//Sort;
    ents=(entitiesOfRelations[arrs]//Transpose)[[2]];
    structs=(nestedEntitiesOfRelations[arrs]//Transpose)[[2]];
    r=Position[rels,i][[1,1]];
    s=Position[rels,j][[1,1]];
    Length[aehnMaxStruktur[structs[[r]],structs[[s]]]//Flatten]/
      Length[MultiUnion[ents[[r]],ents[[s]]]]
    ]

commonStructure[i_,j_,arrs_]:=Module[{rels,structs,r,s},
    rels=(arrs//Transpose)[[1]]//Union//Sort;
    structs=(nestedEntitiesOfRelations[arrs]//Transpose)[[2]];
```

```
    r=Position[rels,i][[1,1]];
    s=Position[rels,j][[1,1]];
    aehnMaxStruktur[structs[[r]],structs[[s]]]
    ]

Tanimoto[i_,j_,arrs_]:=1-similarityValue[i,j,arrs]

MDistance[i_,j_,arrs_]:=Module[{rels,ents,r,s},
    rels=(arrs//Transpose)[[1]]//Union//Sort;
    ents=(entitiesOfRelations[arrs]//Transpose)[[2]];
    r=Position[rels,i][[1,1]];
    s=Position[rels,j][[1,1]];
    Length[MultiUnion[ents[[r]],ents[[s]]]]-
      Length[MultiIntersection[ents[[r]],ents[[s]]]]
    ]

intermedRelations[arrs_]:=Module[{structs,r,l,diag,nondiag,result},
    structs=(nestedEntitiesOfRelations[arrs]//Transpose)[[2]];
    r=(arrs//Transpose)[[1]]//Union//Sort;
    l=r//Length;
    diag=Table[aehnMaxStruktur[structs[[i]],structs[[i]]],{i,1,l}];
    nondiag=
      Flatten[Table[
          aehnMaxStruktur[structs[[i]],structs[[j]]],{i,1,l-1},{j,2,l}],1];
    result=Select[Complement[nondiag,diag],(#//Flatten//Length)>1&]
    ]
```

Funktionen für den Anwender: Clusterbildung, Güte, Klassenzahl, Graphiken

```
Clustering
Bildet Cluster.
Aufruf: Clusterung[Arrows]
Rückgabe: {Dendogramm, {Cluster, Güten}}

QualityClasses
Erstellt Gütediagramm.
Argumente: Ausgabe von Clustering (, maximale Zahl von
Klassen in der Graphik)

Dendogramm
Zeigt Dendogramm als Graphik.
```

```
Argumente: Ausgabe von Clustering

listClasses
Klassen aus Clusterbildung in gewünschter Anzahl anzeigen.
Argumente: Ausgabe von Clustering, Klassenzahl

chainity
Kettenhaftigkeitszahl des Dendogramms.
Argumente: Ausgabe von Clustering

Clustering[arrs_]:=
  Clustering[DistanceMatrix[arrs],(arrs//Transpose)[[1]]//Union//Sort]

Clustering[DM_,C_]:=
  Module[{R,Rneu,min,p,pi,pj,pa,pb,Cl,Cf,Clneu,Cfneu,Dh,DMneu,Guete,Ellbogen,
      crules,firstRow},
    R=Range[Length[C]];
    Cl=C;                    (* Cluster in Dendogrammform *)
    Cf={#}&/@C; (* Cluster als Mengen,
      flat! Am Ende ist alles in einer Menge! *)
    crules=#[[1]]->#[[2]]&/@({Cl,R}//Transpose);
    Ellbogen = {{Cf,0}}; (* Daten f\[UDoubleDot]r Ellbogendiagramm *)
    Dh=DM;
    While[Length[R]>1,min=Min[Dh];p={pi,pj}=Position[Dh,min]//First;
      pa=Min[p];
      pb=Max[p];
      Rneu=Most[R];
      firstRow=
        Extract[Table[newdist[j,p,Dh,Cl],{j,1,Length[R]}],{#}&/@
            Complement[R,p]];
      Clneu=Prepend[Extract[Cl,{#}&/@Complement[R,p]],{Cl[[pi]],Cl[[pj]]}];
      Cfneu =
        Prepend[Extract[Cf,{#}&/@Complement[R,p]],{Cf[[pi]],Cf[[pj]]}//
            Flatten];
      Dh=Drop[Dh,{pa},{pa}];
      Dh=Drop[Dh,{pb-1},{pb-1}];
      DMneu=
        BlockMatrix[{{{{\[Infinity]}},{firstRow}},{{firstRow}//Transpose,
              Dh}}];
      Dh=DMneu;
      R=Rneu;
      Cl=Clneu;
      Cf = Cfneu;
```

```
      (* If[Length[R]<20,Print[Dh]]; *)
      (* G\[UDoubleDot]te berechnen *)
      Guete=guet[Cf/.crules,DM];
      Ellbogen = Prepend[Ellbogen,{Cf,Guete}];
      (* Print[Ellbogen]; *)
      (* weiter gehts *)
      ];
    Return[{Cl,Ellbogen//Transpose}]
    ]

QualityClasses[Cl_,kk_:0]:=Module[{GG,plot1,plot2,units,k,pts},
    GG=Cl[[2,2]]//Rest;
    k=If[kk==0,k=GG//Length,kk];
    pts={Range[2,k+1],Take[GG,k]}//Transpose;
    plot1=
      ListPlot[pts,PlotJoined->True,PlotRange->All,
        DisplayFunction->Identity];
    plot2=
      ListPlot[pts,PlotJoined->False,PlotRange->All,
        PlotStyle->PointSize[0.02],DisplayFunction->Identity];
    Show[plot1,plot2,AxesLabel->{"Klassen","g(K)"},
      Ticks->{Range[2,k+1],Automatic},
      DisplayFunction->$DisplayFunction]
    ]

Dendogramm[Cl_]:=Show[Cl[[1]]//ExprPlot,PlotRange->All]

listClasses[Cl_,k_]:=Cl[[2,1]][[k]]

chainity[Cl_] := Module[{m, l},
    l = Cl[[1, 1]];
m = Length[l // Flatten];
(((a G0[l] + b /. Solve[{a ((n - 1)) + b == 1,
                      a (n/2) Log[2, n] + b == 0}, {a, b}]) /.
               n -> m))[[1]]] // Evaluate) // N)

G0[{a_List,b_List}]:=
  G0[a]+G0[b]+Min[Length[{a}//Flatten],Length[{b}//Flatten]]

G0[{a_,b_List}]:=1+G0[b]/;AtomQ[a]

G0[{a_List,b_}]:=1+G0[a]/;AtomQ[b]
```

```
G0[{a_,b_}]:=1/;AtomQ[a]\[And]AtomQ[b]
```

```
arrs1 = {{1, 6}, {1, 7}, {1, 8}, {2, 7}, {2, 8}, {2, 9}, {10, 3},
  {10, 4}, {3, 6}, {3, 7}, {3, 8}, {4, 8}, {4, 5}, {5, 6}, {5, 7}}
```

$$\begin{pmatrix} 1 & 6 \\ 1 & 7 \\ 1 & 8 \\ 2 & 7 \\ 2 & 8 \\ 2 & 9 \\ 10 & 3 \\ 10 & 4 \\ 3 & 6 \\ 3 & 7 \\ 3 & 8 \\ 4 & 8 \\ 4 & 5 \\ 5 & 6 \\ 5 & 7 \end{pmatrix}$$

```
entitiesOfRelations[arrs1]
```

$$\begin{pmatrix} 1 & \{6, 7, 8\} \\ 2 & \{7, 8, 9\} \\ 3 & \{6, 7, 8\} \\ 4 & \{6, 7, 8\} \\ 5 & \{6, 7\} \\ 10 & \{6, 6, 7, 7, 8, 8\} \end{pmatrix}$$

```
nestedEntitiesOfRelations[arrs1]
```

$$\begin{pmatrix} 1 & \{6, 7, 8\} \\ 2 & \{7, 8, 9\} \\ 3 & \{6, 7, 8\} \\ 4 & \{8, \{6, 7\}\} \\ 5 & \{6, 7\} \\ 10 & \{\{8, \{6, 7\}\}, \{6, 7, 8\}\} \end{pmatrix}$$

```
structuredIdentities[arrs1]
```

({1, 3} {6, 7, 8})

```
unstructuredIdentities[arrs1]
```

({1, 3, 4} {6, 7, 8})

```
tableSimilarityValues[arrs1]
```

$$\begin{pmatrix} 1 & \frac{1}{2} & 1 & 1 & \frac{2}{3} & \frac{1}{2} \\ \frac{1}{2} & 1 & \frac{1}{2} & \frac{1}{2} & \frac{1}{4} & \frac{2}{7} \\ 1 & \frac{1}{2} & 1 & 1 & \frac{2}{3} & \frac{1}{2} \\ 1 & \frac{1}{2} & 1 & 1 & \frac{2}{3} & \frac{1}{2} \\ \frac{2}{3} & \frac{1}{4} & \frac{2}{3} & \frac{2}{3} & 1 & \frac{1}{3} \\ \frac{1}{2} & \frac{2}{7} & \frac{1}{2} & \frac{1}{2} & \frac{1}{3} & 1 \end{pmatrix}$$

```
tableStructuredValues[arrs1]
```

$$\begin{pmatrix} 1 & \frac{1}{2} & 1 & \frac{2}{3} & \frac{2}{3} & \frac{1}{2} \\ \frac{1}{2} & 1 & \frac{1}{2} & \frac{1}{4} & \frac{1}{4} & \frac{2}{7} \\ 1 & \frac{1}{2} & 1 & \frac{2}{3} & \frac{2}{3} & \frac{1}{2} \\ \frac{2}{3} & \frac{1}{4} & \frac{2}{3} & 1 & \frac{2}{3} & \frac{1}{2} \\ \frac{2}{3} & \frac{1}{4} & \frac{2}{3} & \frac{2}{3} & 1 & \frac{1}{3} \\ \frac{1}{2} & \frac{2}{7} & \frac{1}{2} & \frac{1}{2} & \frac{1}{3} & 1 \end{pmatrix}$$

```
tableTanimoto[arrs1]
```

$$\begin{pmatrix} 0 & \frac{1}{2} & 0 & 0 & \frac{1}{3} & \frac{1}{2} \\ \frac{1}{2} & 0 & \frac{1}{2} & \frac{1}{2} & \frac{3}{4} & \frac{5}{7} \\ 0 & \frac{1}{2} & 0 & 0 & \frac{1}{3} & \frac{1}{2} \\ 0 & \frac{1}{2} & 0 & 0 & \frac{1}{3} & \frac{1}{2} \\ \frac{1}{3} & \frac{3}{4} & \frac{1}{3} & \frac{1}{3} & 0 & \frac{2}{3} \\ \frac{1}{2} & \frac{5}{7} & \frac{1}{2} & \frac{1}{2} & \frac{2}{3} & 0 \end{pmatrix}$$

```
tableMDistances[arrs1]
```

$$\begin{pmatrix} 0 & 2 & 0 & 0 & 1 & 3 \\ 2 & 0 & 2 & 2 & 3 & 5 \\ 0 & 2 & 0 & 0 & 1 & 3 \\ 0 & 2 & 0 & 0 & 1 & 3 \\ 1 & 3 & 1 & 1 & 0 & 4 \\ 3 & 5 & 3 & 3 & 4 & 0 \end{pmatrix}$$

```
rels = (arrs1 // Transpose)[[1]] // Union // Sort
```

{1, 2, 3, 4, 5, 10}

```
commonStructure[1, 5, arrs1]
```

{6, 7}

```
MDistance[1, 5, arrs1]
```

1

```
Tanimoto[1, 5, arrs1]
```

$\frac{1}{3}$

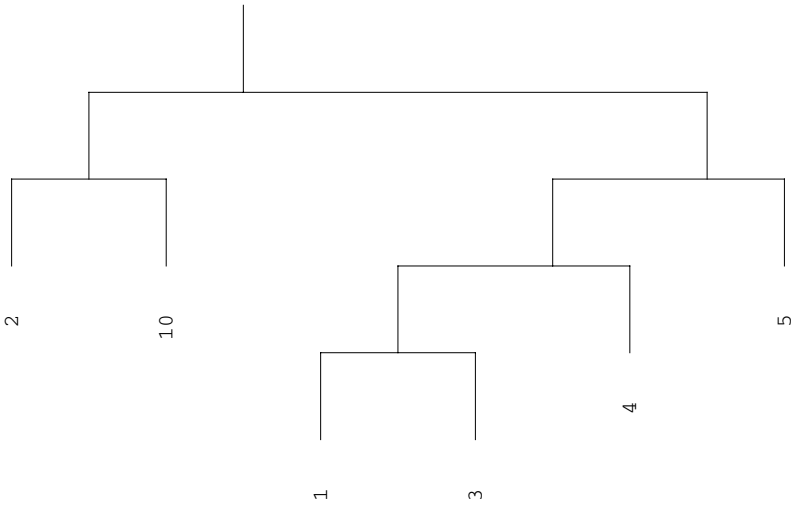
```
intermedRelations[arrs1]
```

(7 8)

```
Cl = Clustering[arrs1];
```

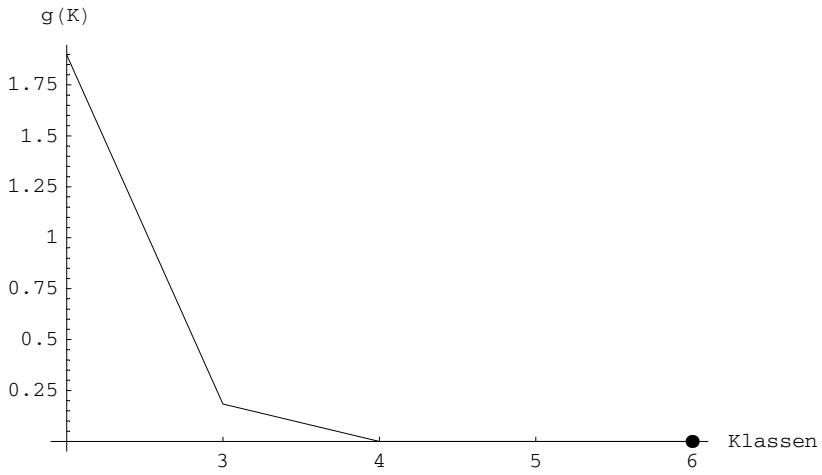
Power::infy : Infinite expression $\frac{1}{0}$ encountered. More…

```
Cl // Dendogramm
```



- Graphics -

QualityClasses[Cl]



- Graphics -

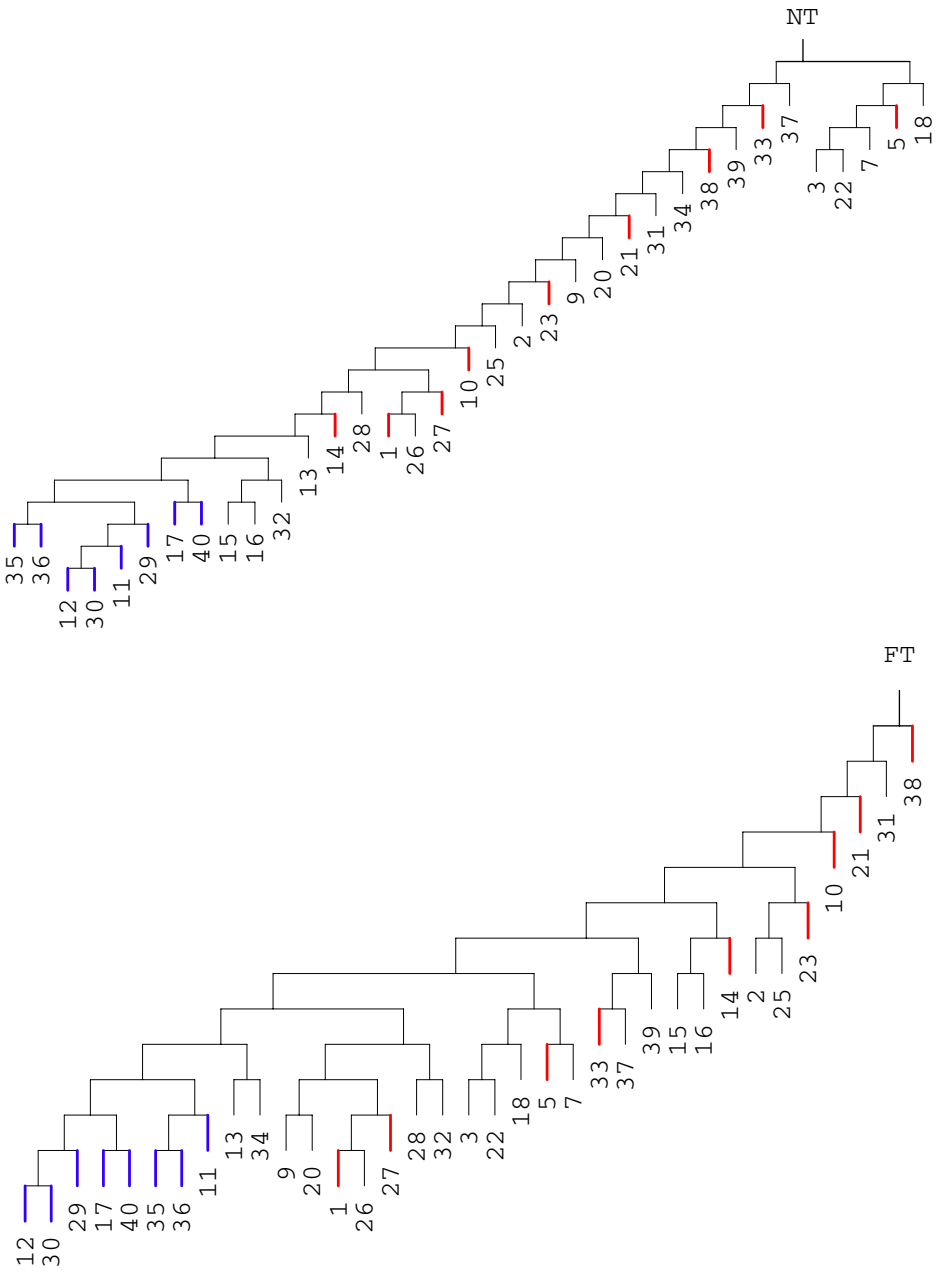
listClasses[Cl, 3]

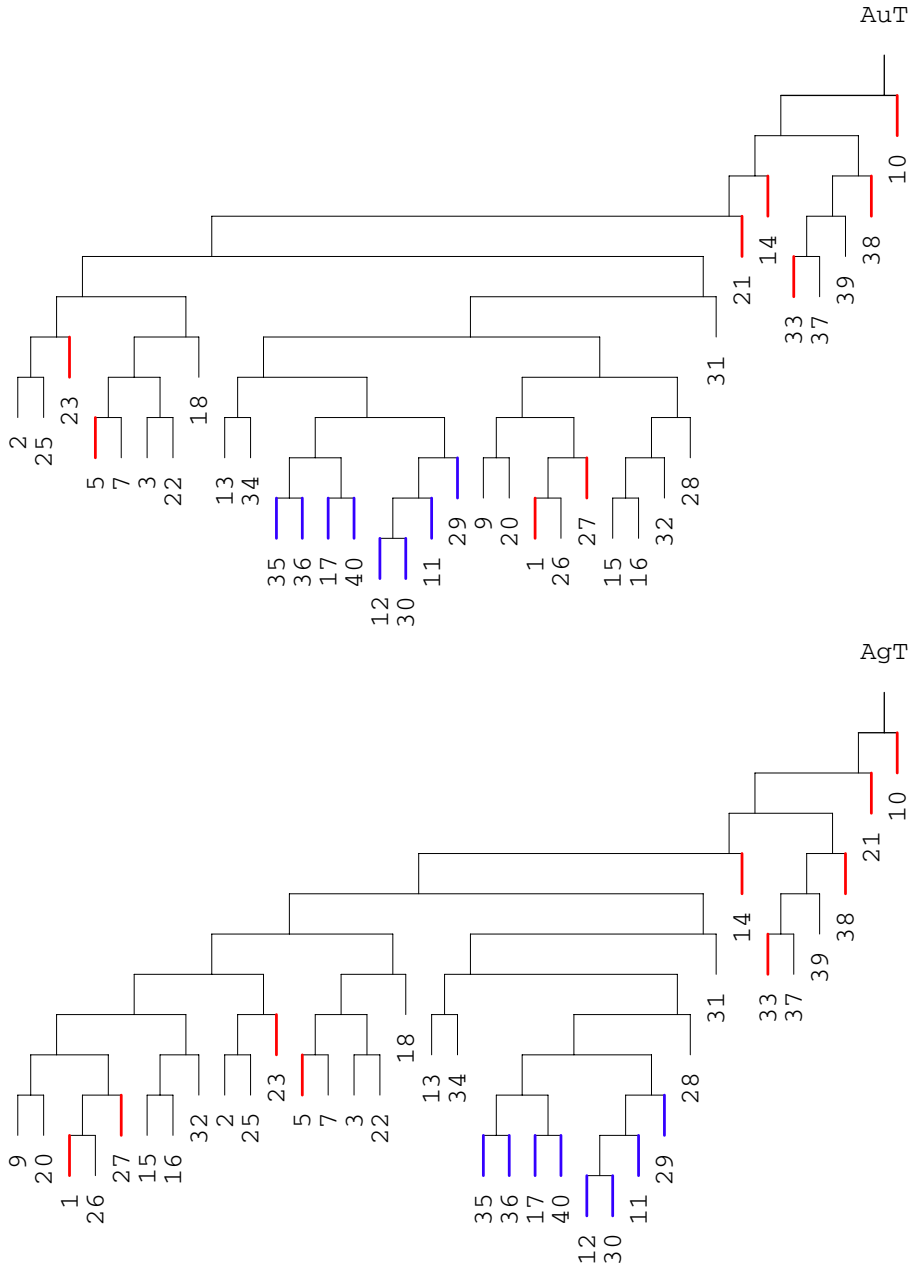
{{1, 3, 4, 5}, {2}, {10}}

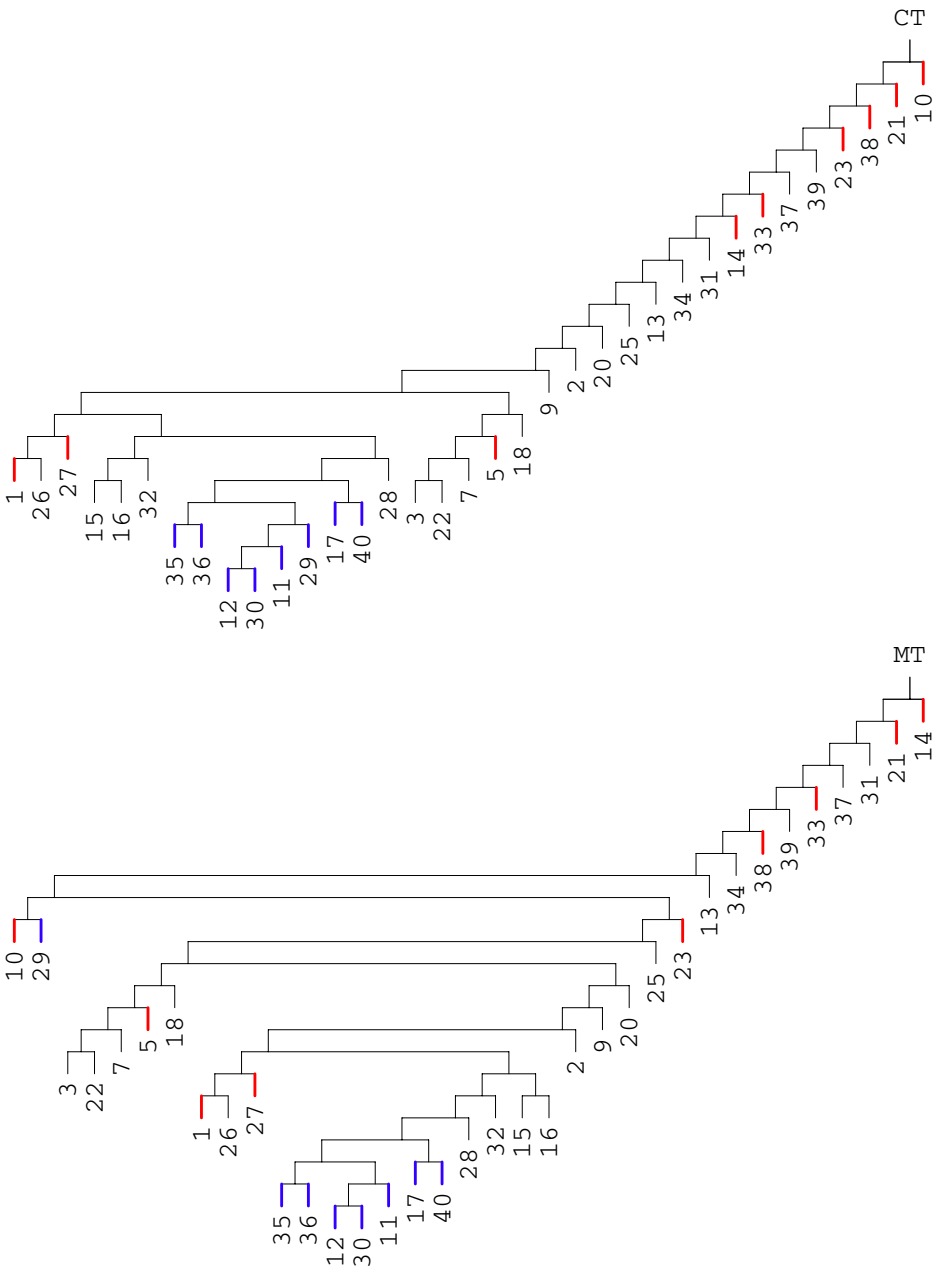
chainity[Cl]

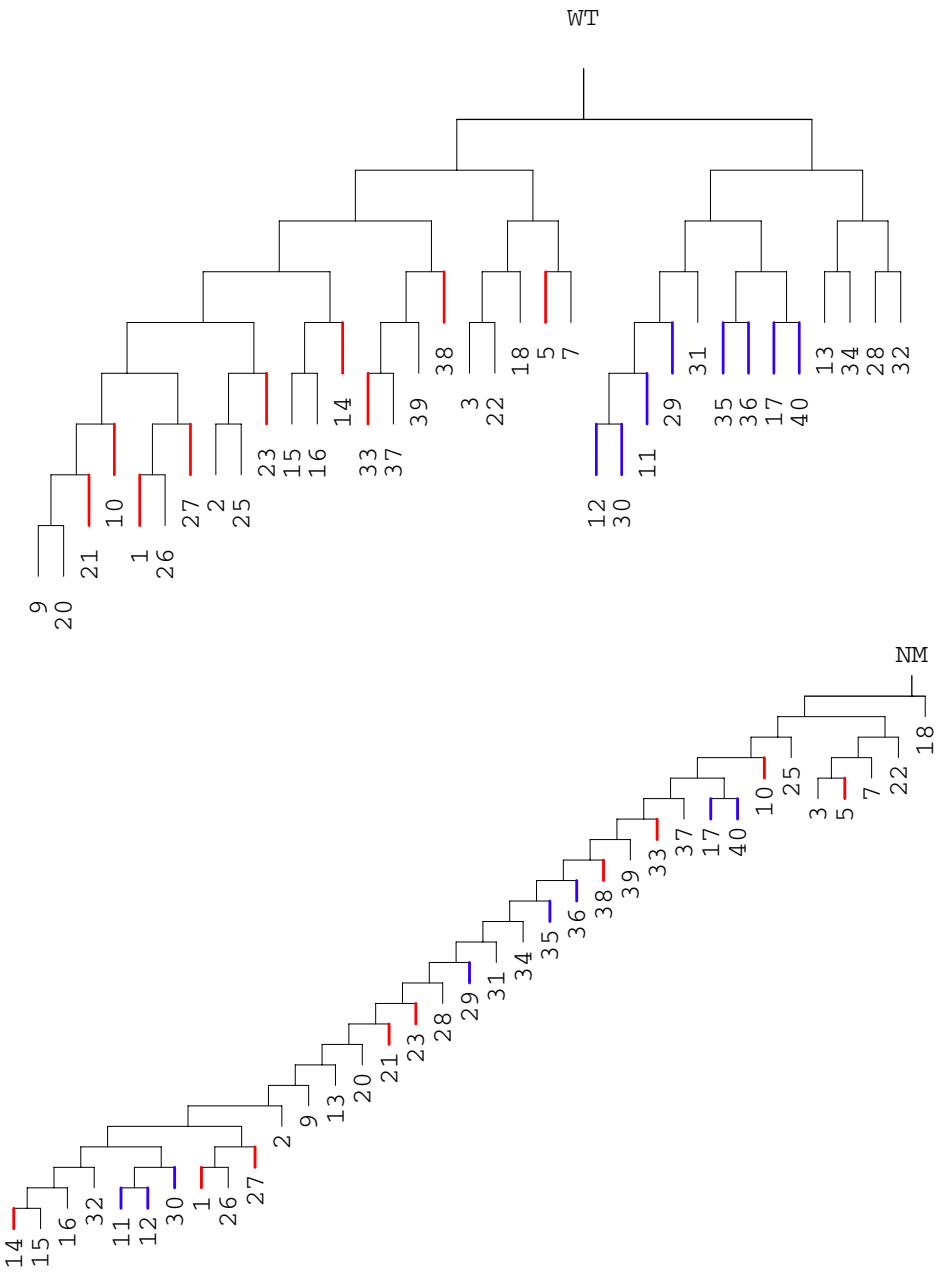
0.637009

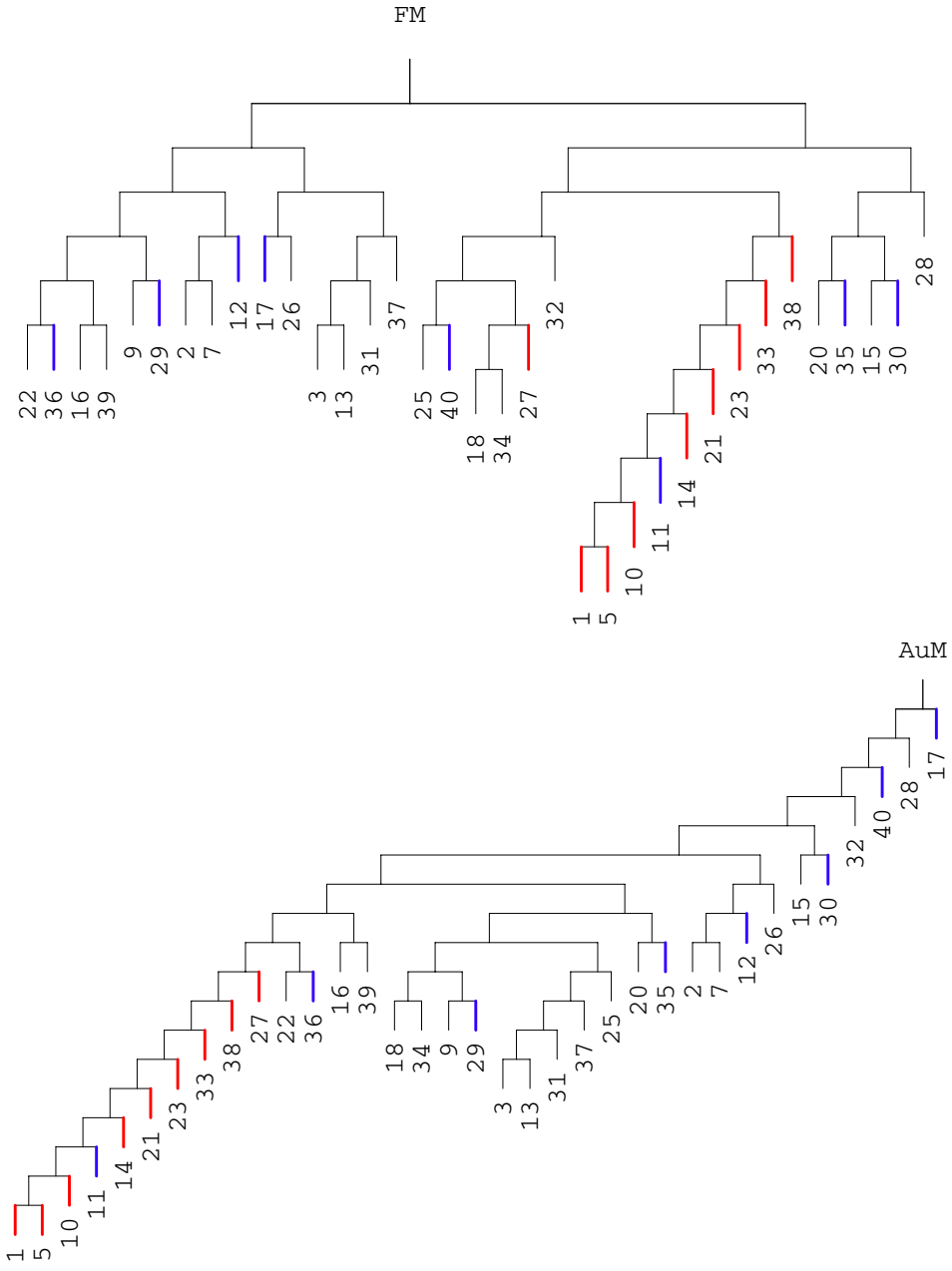
A Dendogramme für Bedarfsplanung etc.

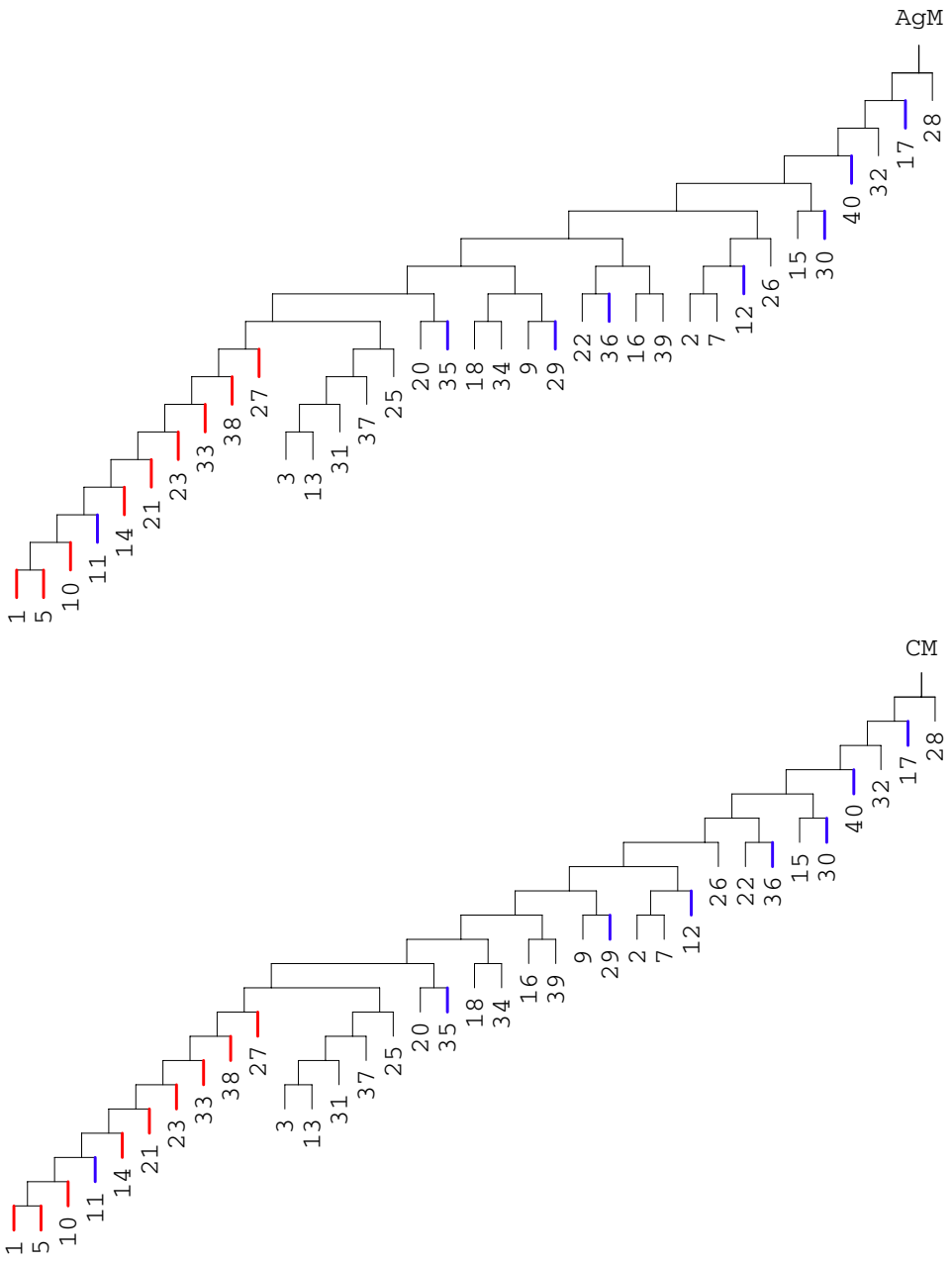


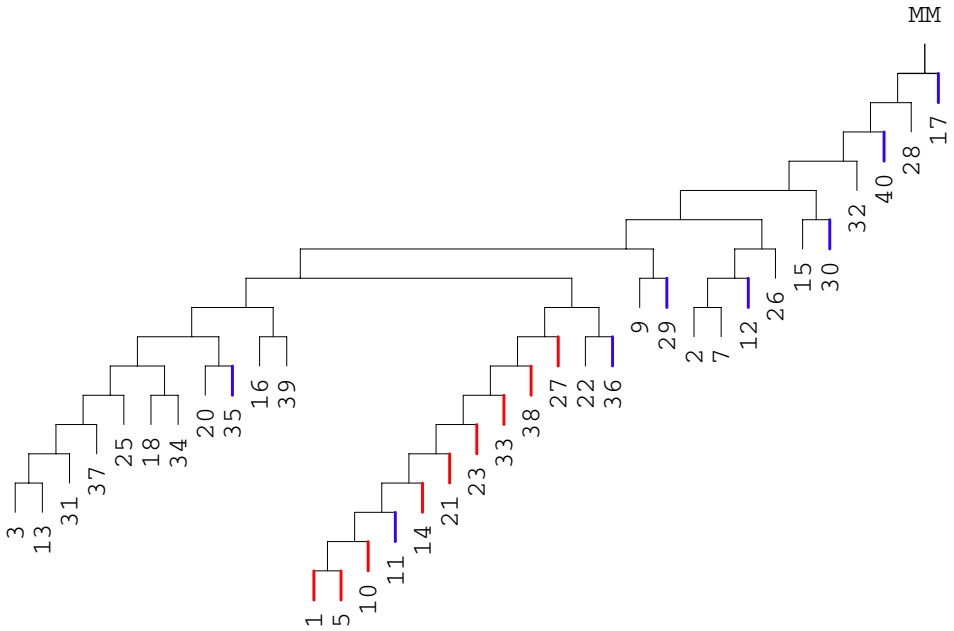

AuT
10
14
38
21
39
33
37
31
23
2
25
18
5
7
3
22
13
34
29
35
36
17
40
11
12
30
9
20
27
1
26
32
15
16
28
AgT
10
21
38
14
39
33
37
31
18
23
32
2
25
5
7
3
22
13
34
28
9
20
27
15
16
1
26
29
35
36
17
40
11
12
30

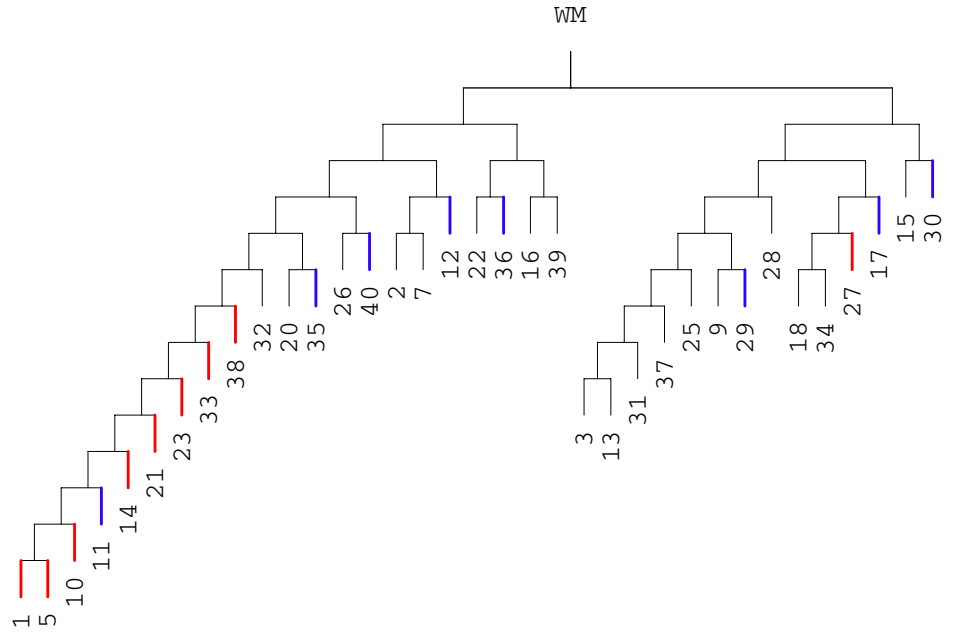

CT
10 21 38 23 39 37 33 14 31 34 13 25 20 2 9 18 5 7 22 3 28 40 17 29 11 30 12 36 35 32 16 15 27 26 1
MT
14 21 31 37 33 39 38 34 13 23 25 20 9 2 18 5 7 22 3 16 15 32 28 40 17 11 30 12 36 35 27 26 1 29 10


WT
9
20
21
10
1
26
27
2
25
23
15
16
14
33
37
39
38
3
22
18
5
7
12
30
11
29
31
35
36
17
40
13
34
28
32
NM
14
15
16
32
11
12
30
1
26
27
2
9
13
20
21
23
28
29
31
34
35
36
38
39
33
37
17
40
10
25
3
5
7
22
18


FM
22 36 16 39 9 29 2 7 12 17 26 3 13 31 37 25 40 18 34 27 32 1 5 10 11 14 21 23 33 38 20 35 15 30 28
AuM
1 5 10 11 14 21 23 33 38 27 22 36 16 39 18 34 9 29 3 13 31 37 25 20 35 2 7 12 26 15 30 32 40 28 17

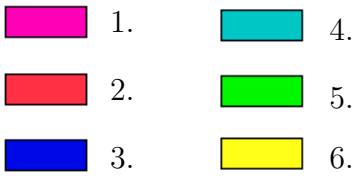

AgM
1
5
10
11
14
21
23
33
38
27
3
13
31
37
25
20
35
18
34
9
29
22
36
16
39
2
7
12
26
15
30
40
32
17
28
CM
1
5
10
11
14
21
23
33
38
27
3
13
31
37
25
20
35
18
34
16
39
9
29
2
7
12
26
22
36
15
30
40
32
17
28


MM
17
28
40
32
30
15
26
12
7
2
29
9
36
22
27
38
33
23
21
14
11
10
5
1
16
39
35
20
34
18
25
37
31
13
3


WM
30
15
17
27
34
18
29
9
25
37
31
13
3
28
12
22
36
16
39
7
2
40
26
35
20
32
38
33
23
21
14
11
10
5
1

B Zuordnung Teilmodelle - Klassen

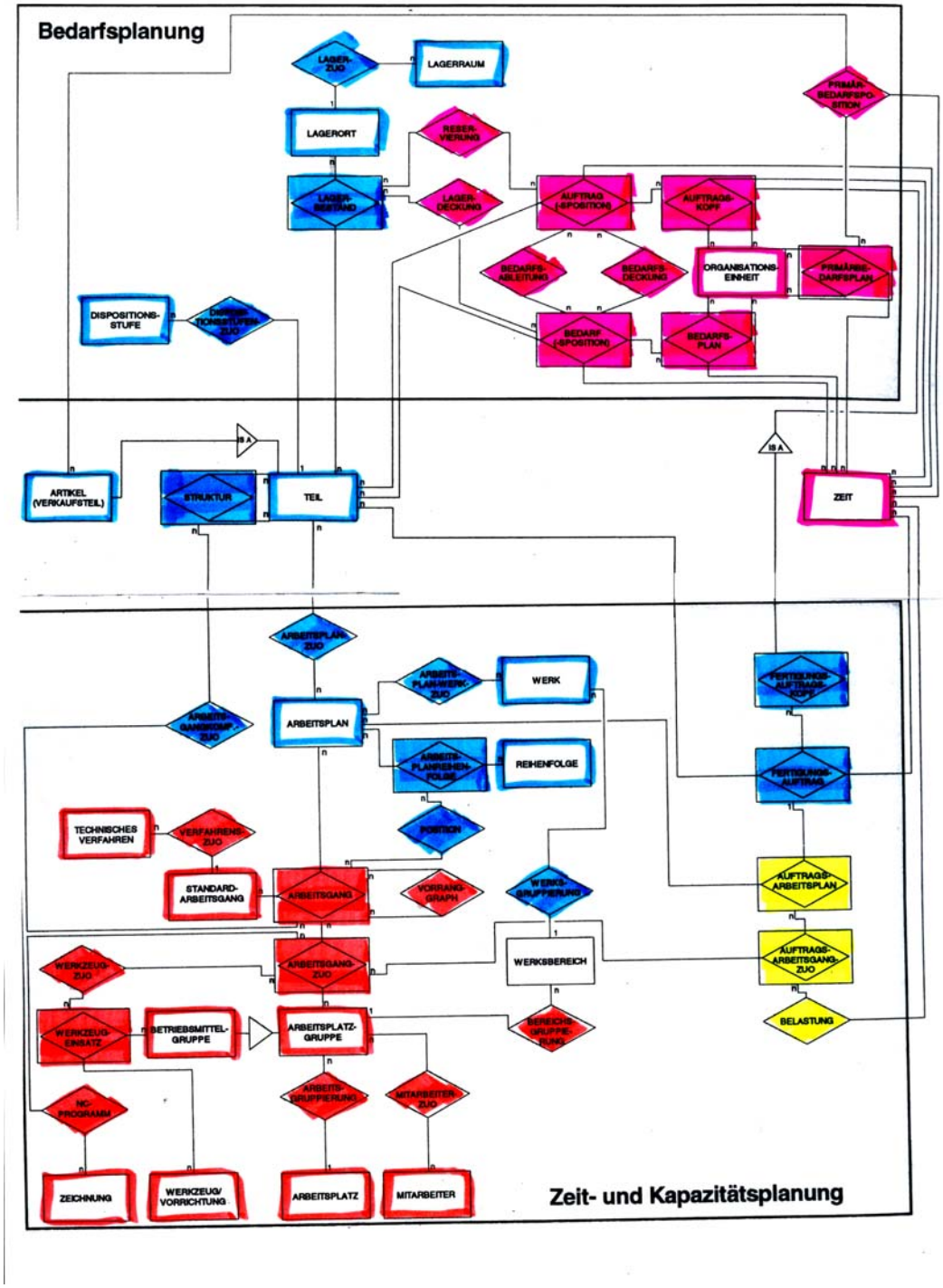
Auf den folgenden Seiten finden sich Abbildungen der Teilmodelle aus [SC97], in denen die Relationen und die Entitäten den 6 Klassen aus der Clusteranalyse durch eine farbliche Kodierung zugeordnet werden. Relationen sind ausgefüllt, Entitäten farbig umrandet. Die Entitäten wurden dabei derjenigen Klasse zugeordnet, zu der sie die meisten Beziehungen haben. Einige wenige Entitäten wurden nicht eingefärbt, da sich die Zuordnung so nicht eindeutig bestimmen ließ. Die Zuordnung von Farben zu Klassen ist der folgenden Abbildung zu entnehmen:

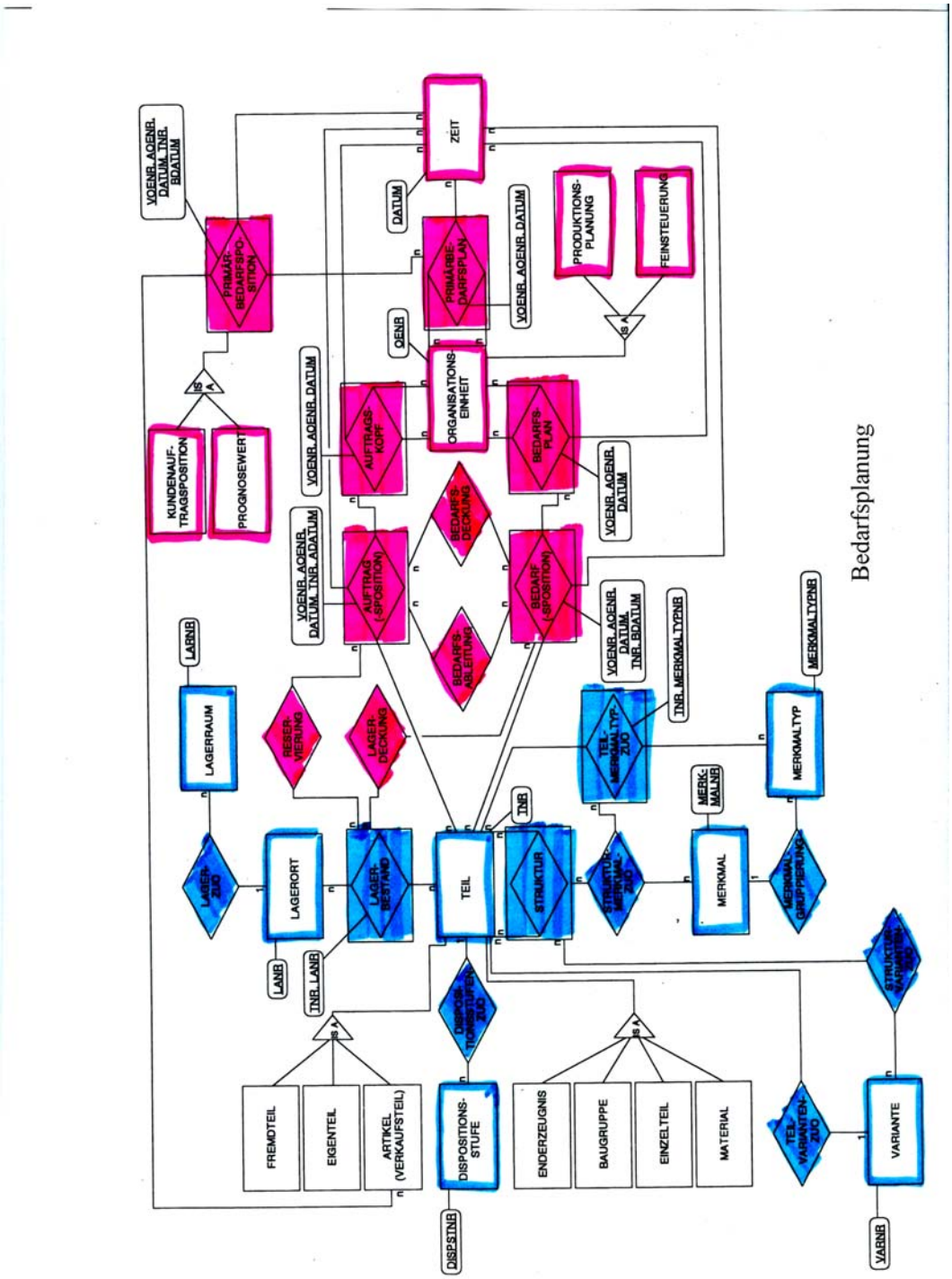


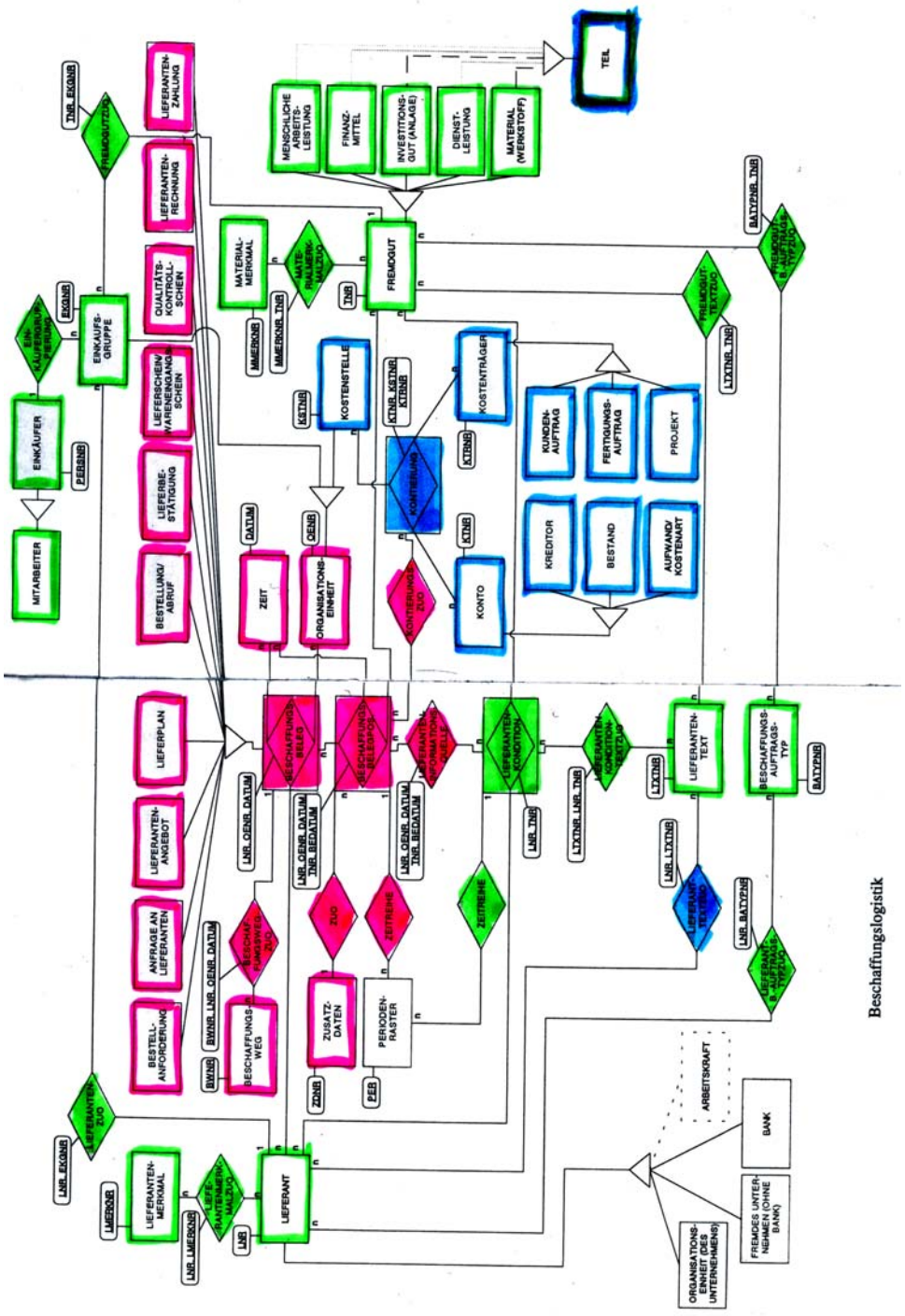
Wenn nur 4 Klassen gebildet werden, dann fallen die Nummern 3, 4 und 5 in eine Klasse zusammen, das sind die blau-grünen Farben.

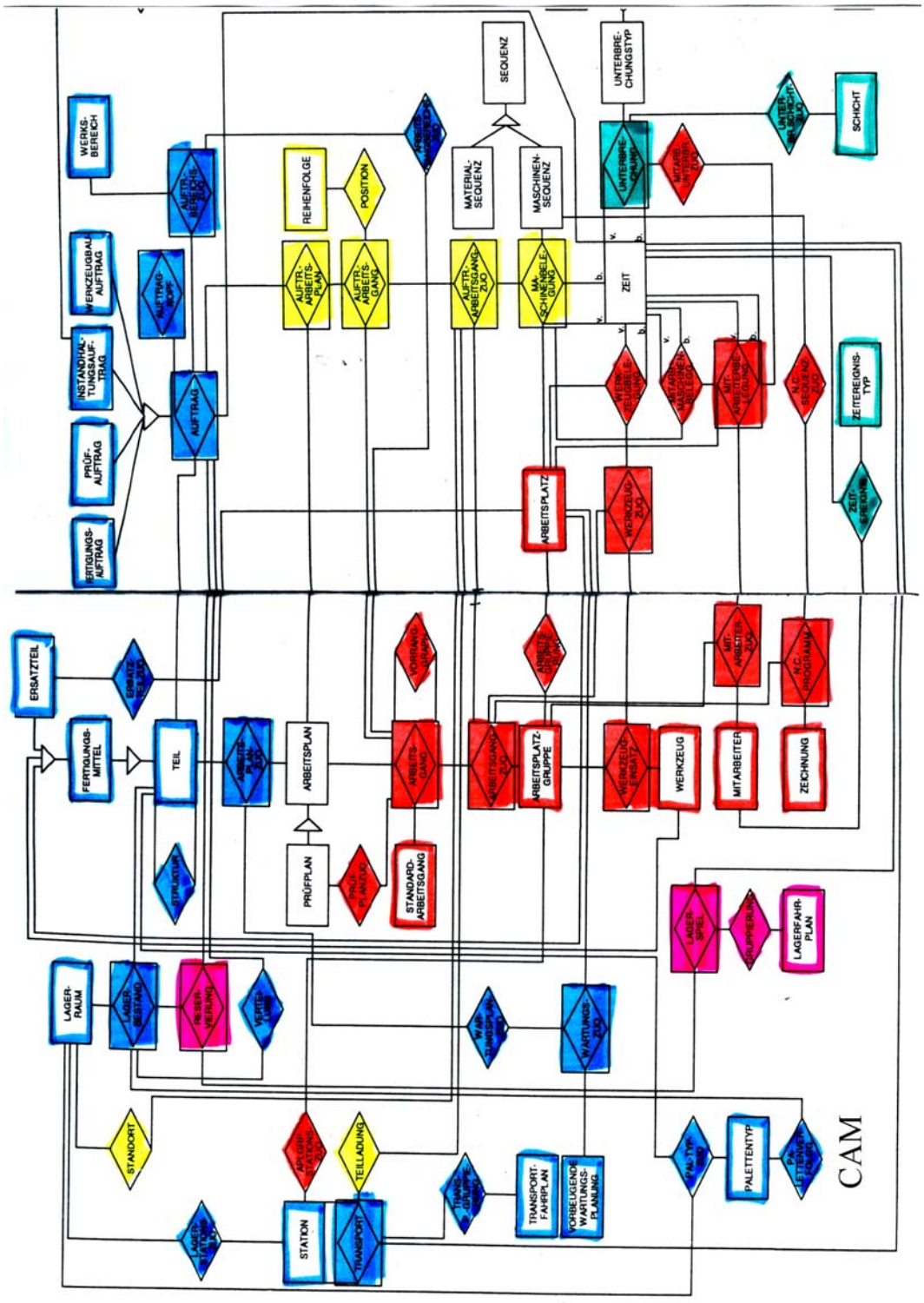
Es findet sich:

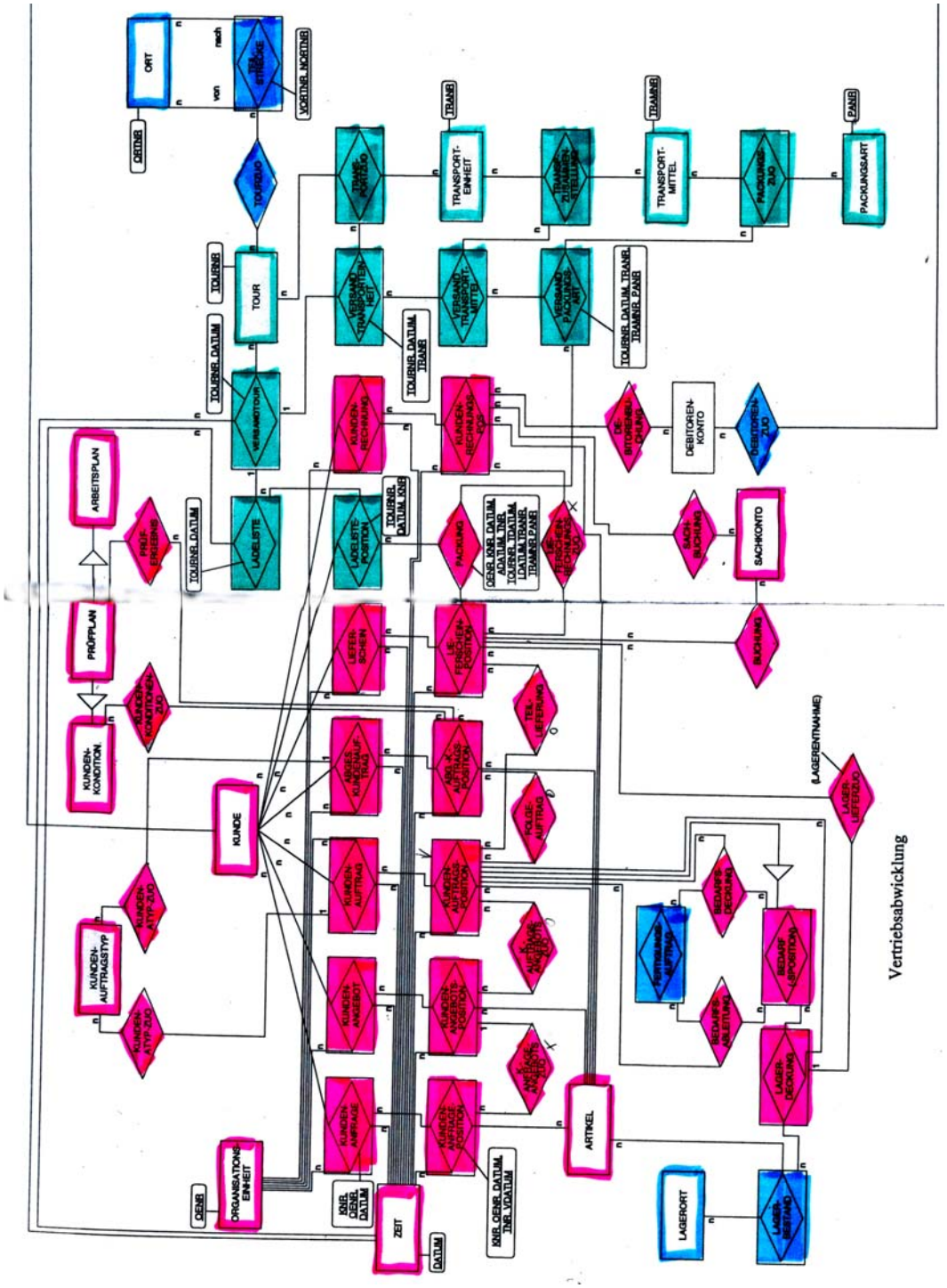
- Bedarfsplanung, Zeit- und Kapazitätsplanung auf Seite 87,
- Bedarfsplanung in detaillierterer Darstellung auf Seite 88,
- Beschaffung auf Seite 89,
- CAM auf Seite 90,
- Vertrieb auf Seite 91.


Bedarfsplanung
LAGER-ZUO
LAGERRAUM
LAGERORT
RESER-VIERUNG
LAGER-BESTAND
LAGER-DECKUNG
AUFTRAG (-SPOSITION)
AUFTRAGS-KOPF
PRIMÄR-BEDARFSPO-SITION
BEDARFS-ABLEITUNG
BEDARFS-DECKUNG
ORGANISATIONS-EINHEIT
PRIMÄRBE-DARFSPLAN
DISPOSITIONS-STUFE
DISPOSITIONSSTUFEN-ZUO
BEDARF (-SPOSITION)
BEDARFS-PLAN
IS A
ARTIKEL (VERKAUFSTEIL)
STRUKTUR
TEIL
ZEIT
ARBEITSPLAN-ZUO
ARBEITS-PLANERSTEL-ZUO
WERK
ARBEITS-GANGKOMP.-ZUO
ARBEITSPLAN
ARBEITS-PLANFOLGE-FOLGE
REIHENFOLGE
FERTIGUNGS-AUFTRAGS-KOPF
FERTIGUNGS-AUFTRAG
TECHNISCHES VERFAHREN
VERFAHRENS-ZUO
POSITION
STANDARD-ARBEITSGANG
ARBEITSGANG
VORRANG-GRAPH
WERKS-GRUPPIERUNG
AUFTRAGS-ARBEITSPLAN
WERKZEUG-ZUO
ARBEITSGANG-ZUO
WERKSBEREICH
AUFTRAGS-ARBEITSGANG-ZUO
WERKZEUG-EINSATZ
BETRIEBSMITTEL-GRUPPE
ARBEITSPLATZ-GRUPPE
BEREICHS-GRUPPE-ZUO
BELASTUNG
NC-PROGRAMM
ARBEITS-GRUPPIERUNG
MITARBEITER-ZUO
ZEICHNUNG
WERKZEUG/ VORRICHTUNG
ARBEITSPLATZ
MITARBEITER
Zeit- und Kapazitätsplanung


Bedarfsplanung
ZEIT
DATUM
VOENR.AOENR.DATUM.TNR.BDATUM
PRIMÄR-BEDARFSPOSITION
PRIMÄRBEDARFSPLAN
VOENR.AOENR.DATUM
PRODUKTIONS-PLANUNG
FEINSTEUERUNG
IS A
KUNDENAUFTRAGSPOSITION
PROGNOSEWERT
OENR
ORGANISATIONSEINHEIT
AUFTRAGSKOPF
BEDARFSPLAN
VOENR.AOENR.DATUM
VOENR.AOENR.DATUM.TNR.ADATUM
AUFTRAG(-SPOSITION)
BEDARFSDECKUNG
BEDARFSABLEITUNG
BEDARF(-SPOSITION)
VOENR.AOENR.DATUM.TNR.BDATUM
LARNB
LAGERRAUM
RESERVIERUNG
LAGERDECKUNG
TNR.MERKMALTYPNR
TEIL-MERKMALTYP-ZUO
MERKMALTYPNR
MERKMALTYP
LAGER-ZUO
LAGERORT
LAGER-BESTAND
TEIL
TNR
STRUKTUR
STRUKTUR-MERKMAL-ZUO
MERK-MALNR
MERKMAL
MERKMAL-GRUPPIERUNG
LANB
TNR.LANB
DISPOS-TEILEARTEN-ZUO
STRUKTUR-VARIANTEN-ZUO
FREMDTEIL
EIGENTEIL
ARTIKEL (VERKAUFSTEIL)
DISPOSITIONS-STUFE
ENDERZEUGNIS
BAUGRUPPE
EINZELTEIL
MATERIAL
TEIL-VARIANTEN-ZUO
VARIANTE
DISPSTNR
VARNR


TEIL
MENSCHLICHE ARBEITS-LEISTUNG
FINANZ-MITTEL
INVESTITIONS-GUT (ANLAGE)
DIENST-LEISTUNG
MATERIAL (WERKSTOFF)
FREMDGUT
FREMDGUTZUO
LIEFERANTEN-ZAHLUNG
LIEFERANTEN-RECHNUNG
QUALITÄTS-KONTROLL-SCHEIN
LIEFERSCHEIN/ WARENEINGANGS-SCHEIN
LIEFERBE-STÄTIGUNG
BESTELLUNG/ ABRUF
MATERIAL-MERKMAL
MATE-RIALMERK-MALZUO
MMERKNR
MMERKNR_TNR
EINKÄUFER-GRUPPE
EINKÄUFER-GRUPPE
EKGNR
PERSNR
EINKÄUFER
MITARBEITER
KOSTENSTELLE
KSTNR
KOSTENTRÄGER
KTNR_KSTNR
KTNR
KONTIERUNG
KONTIERUNGS-ZUO
KONTO
KUNDEN-AUFTRAG
FERTIGUNGS-AUFTRAG
PROJEKT
KREDITOR
BESTAND
AUFWAND/ KOSTENART
DATUM
ZEIT
ORGANISATIONS-EINHEIT
ONR
FREMDGUT-TEXTZUO
LTXTNR_TNR
FREMDGUT-AUFTRAGS-TYPZUO
BATYPNR_TNR
LIEFERPLAN
LIEFERANTEN-ANGEBOT
ANFRAGE AN LIEFERANTEN
BESTELL-ANFORDERUNG
BESCHAFFUNGS-BELEG
BESCHAFFUNGS-BELEGPOS
LIEFERANTEN-INFORMATIONS-QUELLE
LIEFERANTEN-KONDITION
LIEFERANTEN-KONDITION-SORTZUO
LIEFERANTEN-TEXT
BESCHAFFUNGS-AUFTRAGS-TYP
LNR_ONR_DATUM
LNR_ONR_DATUM
TNR_BEDATUM
LNR_ONR_DATUM
TNR_BEDATUM
LNR_TNR
LTXTNR_LNR_TNR
LNR_LTXTNR
LTXTNR
BATYPNR
LIEFERANT-TEXTZUO
LNR_BATYPNR
LIEFERANT-AUFTRAGS-TYPZUO
BESCHAF-FUNGSWEG-ZUO
BWNR_LNR_ONR_DATUM
BWNR
BESCHAFFUNGS-WEG
ZUO
ZEITREIHE
ZEITREIHE
ZEITREIHE
ZUSATZ-DATEN
ZDNR
PERIODEN-RASTER
PER
LIEFERANTEN-ZUO
LNR_EKGNR
LIEFERNR
LIEFERANTEN-MERKMAL
LNR_LMERKNR
LIEFE-RANTENMERK-MALZUO
LNR
LIEFERANT
ARBEITSKRAFT
BANK
FREMDES UNTER-NEHMEN (OHNE BANK)
ORGANISATIONS-EINHEIT (DES UNTERNEHMENS)
Beschaffungslogistik


WERKS-BEREICH
WERKZEUGBAU-AUFTRAG
INSTANDHAL-TUNGSAUF-TRAG
PRÜF-AUFTRAG
FERTIGUNGS-AUFTRAG
AUFTRAG
AUFTRAG-HOPF
AUFTR.-BEREICHS-ZUO
REIHENFOLGE
POSITION
AUFTR.-ARBEITS-PLAN
AUFTR.-ARBEITS-GANG
SEQUENZ
MATERIAL-SEQUENZ
MASCHINEN-SEQUENZ
AUFTR.-ARBEITSGANG-ZUO
MA-SCHINENBELE-GUNG
ZEIT
UNTERBRE-CHUNGSTYP
UNTERBRE-CHUNG
MITARB.-UNTERBR.-ZUO
UNTER-BR.-SCHICHT-ZUO
SCHICHT
WERK-ZEUGBELE-GUNG
MITARB.-MASCHINEN-BELEGG
MIT-ARBEITERBE-LEGUNG
N.C.-SEQUENZ-ZUO
ZEITEREIGNIS-TYP
ZEIT-EREIGNIS
ARBEITSPLATZ
WERKZEUG-ZUO
ERSATZTEIL
ERSATZ-TEIL-ZUO
FERTIGUNGS-MITTEL
TEIL
STRUKTUR
ARBEITS-PLAN-ZUO
ARBEITSPLAN
PRÜFPLAN
PRÜF-PLANZUO
STANDARD-ARBEITSGANG
VORRANG-GRAPH
ARBEITS-GANG
ARBEITSGANG-ZUO
ARBEITSPLATZ-GRUPPE
WERKZEUG-EINSATZ
WERKZEUG
MIT-ARBEITER-ZUO
MITARBEITER
N.C.-PROGRAMM
ZEICHNUNG
LAGER-SPIEL
GRUPPIERUNG
LAGERFAHR-PLAN
LAGER-RAUM
LAGER-BESTAND
RESER-VIERUNG
VERTEI-LUNG
WAR-TUNGSPLAN-ZUO
WARTUNGS-ZUO
STANDORT
APLGRP-STATIONS-ZUO
TEILLADUNG
TRANS-GRUPPE-RUNG
TRANSPORT-FAHRPLAN
VORBEUGENDE WARTUNGS-PLANUNG
PAL-TYP-ZUO
PALETTENTYP
PA-LETTENVER-FOLGE
LAGER-STATIONS-BLOCK
STATION
TRANSPORT
CAM


Vertriebsabwicklung

Literatur

[BA96] Backhaus et al.: Multivariate Analyse-Methoden. Springer 1996

[BA86] Batini, Carlo; Lenzerini, Maurizio; Navathe, Shamkant B.: A Comparitive Analysis of Methodologies for Database Schema Integration. ACM Computing Surveys 18 (1986) 4, 323-364

[BE81] Siegfried Bergs: Optimalität bei Cluster-Analysen. Diss. Münster 1981

[CH76] Chen, Peter Pin-Shan: The Entity-Relationship Model - Toward a Unified View of Data. ACM Transactions on Database Systems 1 (1976) 1, S.9-36

[EV01] Brian S. Everitt, Sabine Landau, Morven Leese: Cluster analysis. London [u.a.] : Arnold [u.a.], 2001

[FE05] Fettke, Peter; Loos, Peter: Zur Idenfikation von Strukturanalogien in Datenmodellen. Wirtschaftsinformatik 47 (2005) 2, S. 89-100

[JA02] Helmut Jarosch: Datenbankentwurf - Eine beispielorientierte Einführung für Studenten und Praktiker. Vieweg 2002

[LO97] P. Loos: Capture More Data Semantic Through The Expanded Entity-Relationship Model, Arbeitsbericht des Instituts für Wirtschaftsinformatik, Nr. 53, Münster, März 1997

[RA92] Otto Rauh, Eberhard Stickel: Entity Tree Clustering - A Method for Simplifying ER Designs. In: Günther Permul, A. Min Tjoa (Eds.): Entity-Relationship Approach - ER'92, 11th International Conference on the Entity-Relationship Approach, Karlsruhe, Germany, October 7-9, 1992. Lecture Notes in Computer Science 645, Springer 1992, S. 62–78

[SI93] Sinz, E.: Datenmodellierung im Strukturierten Entity-Relationship-Modell (SERM), in: Müller-Ettrich, G. (Hrsg.): Fachliche Modellierung von Informationssystemen - Methoden, Vorgehen, Werkzeuge. Bonn - Paris 1993, S. 63–126

[SC97] Scheer, August-Wilhelm: Wirtschaftsinformatik - Referenzmodelle für industrielle Geschäftsprozesse. Berlin et al. 1997