Ludwig Maximilians Universität München
Institut für Informatik
Lehr- und Forschungseinheit für Datenbanksysteme

Skript zur Vorlesung

Knowledge Discovery in Databases

im Wintersemester 2007/2008

Kapitel 9: Data Warehousing und Generalisierung

http://www.dbs.ifi.lmu.de/Lehre/KDD

9. Data Warehousing und Generalisierung

Inhalt dieses Kapitels

9.1 Einleitung

Motivation

- Analysen nicht auf den detaillierten Daten, sondern auf aggregierten Daten
- Aufgabe: kompakte Beschreibung einer gegebenen Datenmenge
  - deutlich kleinere Menge von Datensätzen
  - mit Attributwerten auf abstrakterem Niveau

  ➡ Generalisierung
- Anwendungen

  Online Analytical Processing (OLAP)

  automatische Zusammenfassung von Daten z.B. für Manager

  Transformation einer Datenbank für andere Data-Mining-Verfahren
- Zwei Typen von Generalisierung

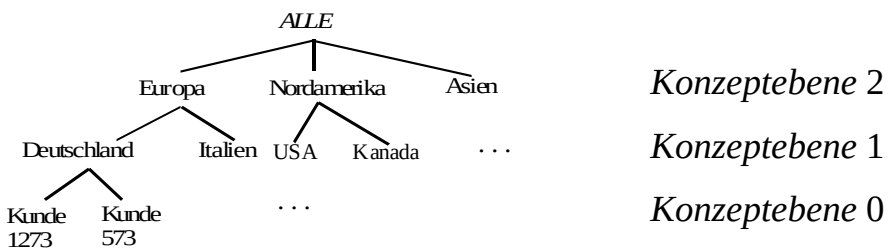
  manuell bzw. automatisch

9.1 Einleitung

Grundbegriffe

- *Wertebereiche* D_i, $1 \le i \le d$:

  logisch zusammengehörige Mengen von Werten mit $ALLE \in D_i$
- *Relation R*, $R \subseteq D_1 \times D_2 \times ... \times D_d$, mit den Attributen $A_1, \ldots, A_d$
- *Konzepthierarchie* für D_i (bzw. für A_i): Baum mit folgenden Eigenschaften
  - die Knoten repräsentieren Werte aus D_i
  - die Wurzel repräsentiert den speziellen Wert *ALLE*
  - die Kanten repräsentieren eine „is-a“-Beziehung



Grundbegriffe

- *Generalisierung* der Relation R in Bezug auf das Attribut A_i:

  Ersetzung aller Werte von A_i durch ihren direkten Vorgänger in der Konzepthierarchie von D_i

- *Spezialisierung*: Ersetzung aller A_i-Werte durch einen ihrer direkten Nachfolger in der Konzepthierarchie von D_i

- *Aggregierung* einer Menge T von Tupeln, $T \subseteq R$, die auf den Attributen $A_1, \ldots, A_c$, $1 \leq c \leq d$, übereinstimmen:

  liefert ein Tupel $(a_1, \ldots, a_c, Op(\{a_{c+1}(t) \mid t \in T\}), \ldots, Op(\{a_d(t) \mid t \in T\}))$

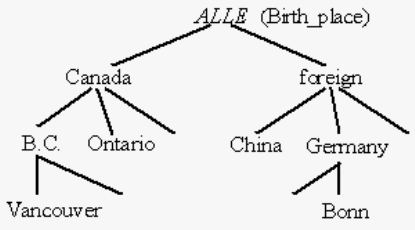
  

  Op ist ein arithmetischer Operator

Beispiel

Name§	Sex§	Age§	Birth_Place§	Department§	Position§	Salary§	Support§
Anderson§	female§	26§	Burnaby§	CompSc§	secretary§	26.000§	1§
Benson§	male§	45§	Vancouver§	ElectEng§	full_prof§	63.000§	1§
. . .§	. . .§	. . .§	. . .§	. . .§	. . .§	. . .§	. . .§
Young§	male§	38§	Bonn§	CivilEng§	assoc_prof§	55.400§	1§

ursprüngliche Relation



Aggregierung
- von Attribut Support
- mit Operator „+“

generalisierte Relation§

Sex§	Age§	Birth_Place§	Salary§	Support§
male§	old§	Canada§	high§	200§
female§	young§	Canada§	low§	35§
male§	mid_age§	foreign§	medium§	137§

generalisierte Relation

9.2 Data Cubes

Grundbegriffe

Data Warehouse (*DW*)

- eine dauerhafte, integrierte Sammlung von Daten aus unterschiedlichen Quellen zum Zweck der Analyse bzw. Entscheidungsunterstützung

  [Chaudhuri & Dayal 1997]

- eine themenorientierte, integrierte, zeit-variante und permanente Sammlung von Daten zur Entscheidungsunterstützung

  [Inmon 1996]

Data Warehousing

der Prozess des Aufbaus und der Nutzung eines Data Warehouse

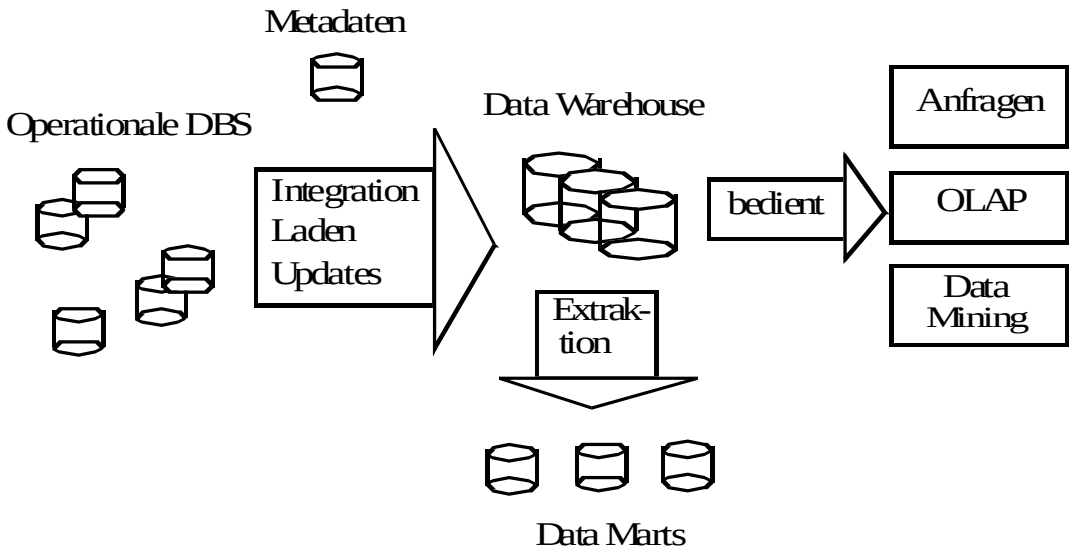
9.2 Data Warehouses

Vergleich Data Warehouse ↔ operationales DBS

	operationales DBS	**Data Warehouse**
Ziel	Abwicklung des Geschäfts	Analyse des Geschäfts
Focus auf	Detail-Daten	aggregierten Daten
Versionen	nur aktuelle Daten	gesamte Historie der Daten
DB-Größe	~ 1 GB	~ 1 TB
DB-Operationen	Updates und Anfragen	nur Anfragen
Zugriffe pro Op.	~ 10 Datensätze	~ 1.000.000 Datensätze
Leistungsmaß	Durchsatz	Antwortzeit

zwei getrennte Systeme

Architektur eines Data Warehouse



Entwicklung eines Data Warehouse

Integration

- einfache Transformationen
- Nutzen von Anwendungswissen zur Vereinheitlichung ähnlicher Daten
- Überprüfen von Konsistenzbedingungen

Laden

- Aggregierung der Daten (mit Operatoren wie z.B. SUM, AVG, COUNT)
- Erzeugen von Indexstrukturen

Updates

- keine On-Line-Updates des DW
- Batch Updates in Zeiten, in denen das DW nicht verfügbar sein muß
- inkrementelle Updates der aggregierten Daten

9.2 Data Cubes

Motivation

- multidimensionales Modell für ein Data Warehouse
- Materialisierung gewisser generalisierter Daten

  Generalisierung der Blätter der Konzepthierarchie direkt auf den Wert *ALLE*

Grundbegriffe

- *Dimensionen* (unabhängige Attribute)

  z.B. Produkt oder Kunde

  mit entsprechender Konzepthierarchie
- *Maße* (abhängige Attribute)

  z.B. Preis oder Gewinn
- Dimensionen: spannen einen multidimensionalen Datenraum auf
- Zellen: enthalten die aggregierten Maße für den entsprechenden Teilraum

9.2 Data Cubes

Grundbegriffe

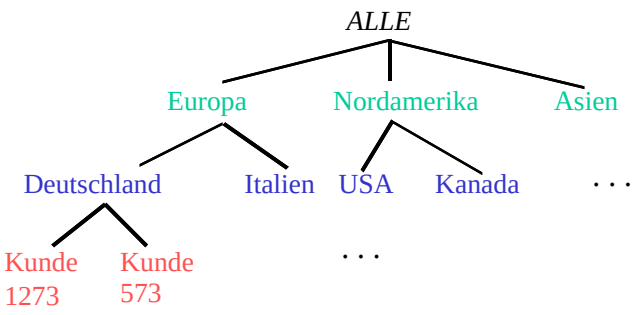
- im allgemeinen mehrere Attribute pro Dimension

  z.B. Dimension Kunde: Attribute Region, Nation und KundenID
- *Hierarchieschema*:

  organisiert die Attribute einer Dimension

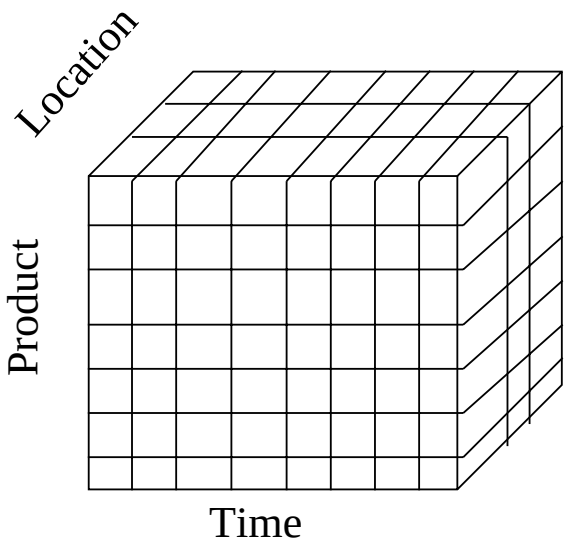
  Konzepthierarchie ist Instanz eines solchen Hierarchieschemas

Beispiel

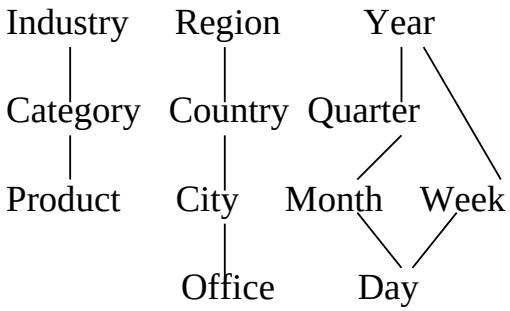


9.2 Data Cubes

Beispiel

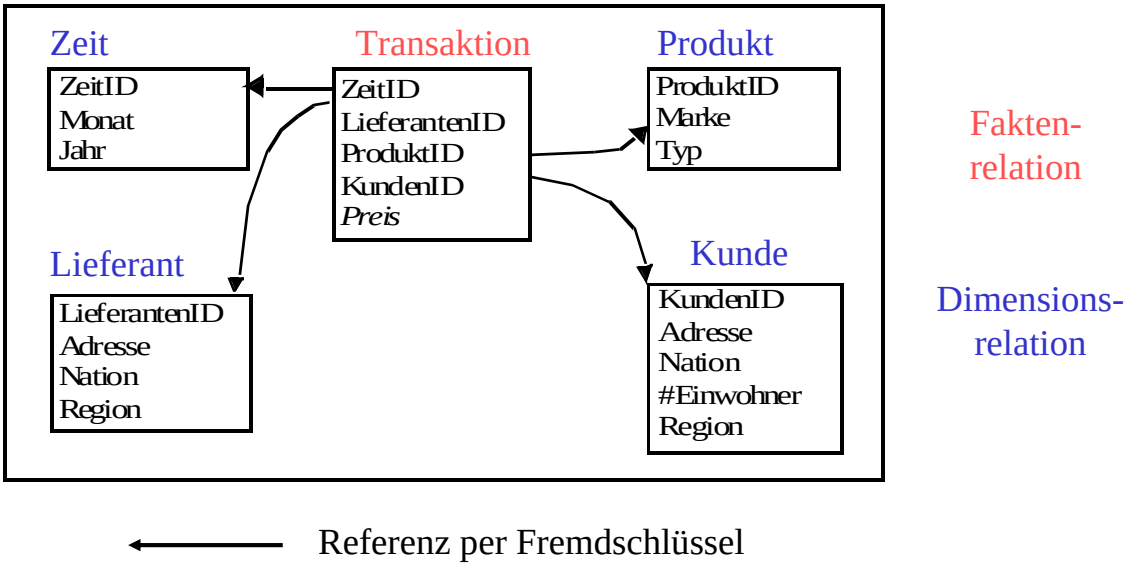


Hierarchieschemata



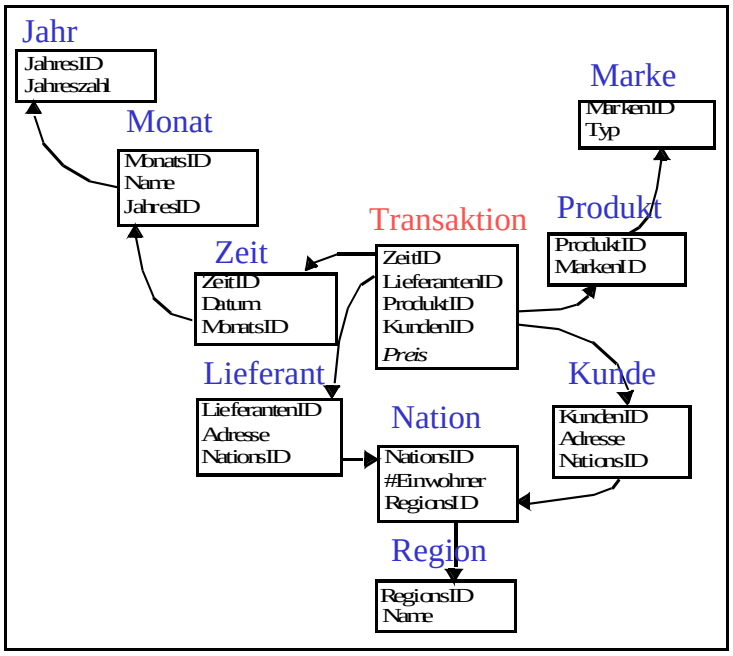
9.2 Implementierung von Data Cubes

Sternschema



9.2 Implementierung von Data Cubes

Schneeflockenschema



Faktenrelation

Dimensionsrelation

Referenz per Fremdschlüssel

9.2 Implementierung von Data Cubes

SQL-Anfragen

- Beispiel: Data Cube mit den Dimensionen Produkt, Lieferant und Kunde

  Datensätze in der Relation `Transaktion`

- alle Zellen der Form (Produkt = *p*, Lieferant = *ALLE*, Kunde = *k*) z.B. durch

```
SELECT ProduktID, KundenID, SUM(Preis)
FROM Transaktion
GROUP BY ProduktID, KundenID ;
```

- Repräsentation der SQL-Anfragen durch ihre GROUP BY-Attribute, z.B.

```
(p,l,k) = GROUP BY ProduktID, LieferantenID, KundenID
          liefert alle Basis-Zellen des Data Cube
() = kein GROUP BY, liefert ein einziges Tupel
```

9.3 Effiziente Anfragebearbeitung in Data Cubes

Ansätze

Einsatz spezieller Indexstrukturen

- Bitmap Indices

  Unterstützung multi-dimensionaler Selektionen

- Join Indices

  Unterstützung des Joins zwischen Fakten- und Dimensionsrelationen

Materialisierung häufiger Anfragen / Sichten

- Berücksichtigung von Abhängigkeiten zwischen den Anfragen
- Auswahl der zu materialisierenden Anfragen

  Speicherplatz ist begrenzt!

9.3 Bitmap Indices

Grundbegriffe [Labio, Quass & Adelberg 1997]

- *Invertierte Listen*

  für jedes Attribut ein Index

  Einträge des Index: (Attributwert, [Rec_1, Rec_2, . . ., Rec_m])

  Kombination der Ergebnislisten der Indices mit Hilfe von $\cap$
- Variation der invertierten Listen

  Bitvektor statt Liste der Datensätze
- *Bitmap Indices*

  pro Datensatz ein Bit: der Datensatz hat den Attributwert ja / nein?

  Kombination der Ergebnisse der einzelnen Indices sehr effizient
- Nachteil

  nicht geeignet für Attribute mit großen Wertebereichen

9.3 Bitmap Indices

Beispiel

Cust	Region	Type
1	Asia	Retail
2	Europe	Dealer
3	Asia	Dealer
4	America	Retail
5	Europe	Dealer

Relation

Index für Region

RecID	Asia	Europe	America
1	1	0	0
2	0	1	0
3	1	0	0
4	0	0	1
5	0	1	0

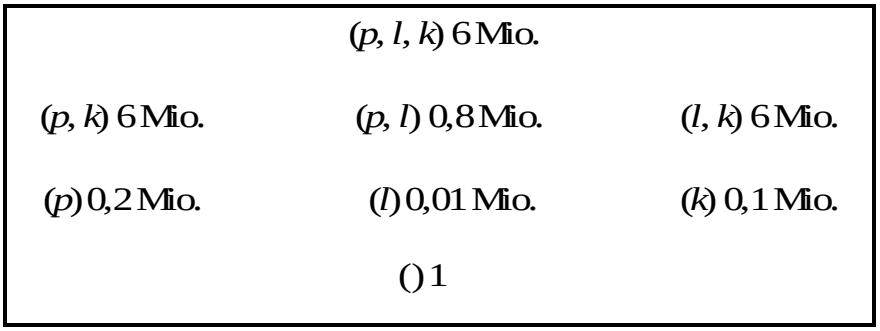
Index für Type

RecID	Retail	Dealer
1	1	0
2	0	1
3	0	1
4	1	0
5	0	1

9.3 Materialisierung von Sichten

Motivation

- Beispiel: Data Cube mit den Dimensionen Produkt, Lieferant und Kunde
- insgesamt 6 Mio. Datensätze in der Faktenrelation



(*a*, . . .): Sicht mit GROUP BY *a* (...) x: Sicht besitzt x Tupel

9.3 Materialisierung von Sichten

Motivation

- Anfrage: (p)
- Kosten der Anfragebearbeitung:

  proportional zur Zahl zu verarbeitender Tupel
- Varianten der Anfragebearbeitung:

  mit materialisiertem (p): 0,2 Mio. Tupel

  mit materialisiertem (p, k): 6 Mio. Tupel
- Varianten der Materialsierung

  alle 8 Sichten: kostet 19,11 Mio. Tupel Speicherplatz

  6 Sichten, ohne (p, k) und (l, k): kostet 7,11 Mio. Tupel Speicherplatz

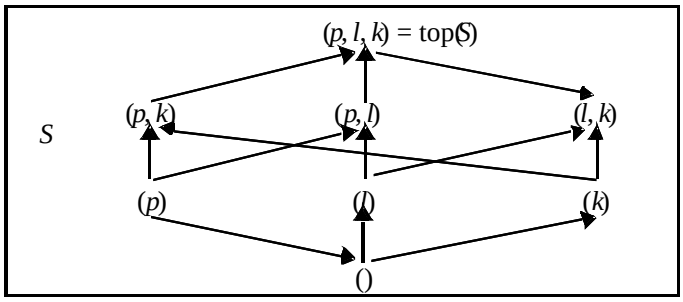
identischer Effizienzgewinn

9.3 Materialisierung von Sichten

Grundbegriffe [Harinarayan, Rajaraman & Ullman 1996]

- S: Menge von Sichten über einer gegebenen Relation, $s_1, s_2 \in S$
- s_1 ist *ableitbar* von s_2, notiert als $s_1 \preccurlyeq s_2$:

  die Anfrage nach s_1 kann vollständig mit Hilfe von s_2 beantwortet werden
- im folgenden: nur solche Mengen S, die in Bezug auf $\preccurlyeq$ einen *Verband* bilden
- $top(S) \in S$: $\forall s \in S: s \preccurlyeq top(S)$



$s_1 \longrightarrow s_2$: $s_1 \Uparrow s_2$

Grundbegriffe

Annahmen

- Anfragen nur eine der Sichten aus S
- gleiche Wahrscheinlichkeit für alle Anfragen
- Beantwortung einer Anfrage q:
  durch kleinste materialisierte Sicht $s \in S$ mit $q \preceq s$
- Kosten der Anfragebearbeitung: Zahl der zu verarbeitenden Tupel

Problemstellung

- Gegeben: für jede der Sichten ihre Größe (Anzahl der Tupel)
- Gesucht:
  diejenige Teilmenge von S mit der Kardinalität k, die die durchschnittlichen Kosten der Anfragebearbeitung minimiert

Grundbegriffe

- *Vorteil* einer Sicht s in Bezug auf die Menge M bereits materialisierter Sichten:
  Reduktion der Kosten zur Beantwortung aller Anfragen
- $(S, \preceq)$: Verband von Sichten, $v, w \in S$ und $M \subseteq S$
  $u \in M$: die bezüglich $\preceq$ kleinste Sicht, für die gilt $w \preceq u$
- für $w \preceq v$ ist der *Vorteil* von v in Bezug auf w und M, bezeichnet als $B_w(v, M)$:

$$B_w(v, M) = \begin{cases} Kosten_u(w) - Kosten_v(w) & falls\ Kosten_v(w) < Kosten_u(w) \\ 0 & andernfalls \end{cases}$$

- *Vorteil* von v in Bezug auf M, bezeichnet als $B(v, M)$:

$$B(v, M) = \sum_{w \in S, w \preceq v} B_w(v, M)$$

Algorithmen

- optimaler Algorithmus

  betrachtet alle Teilmengen von S mit der Kardinalität k

  bestimmt die durchschnittlichen Kosten der Anfragebearbeitung

  wählt die optimale Teilmenge aus

  ineffizient für größere Mengen S

- heuristischer Algorithmus

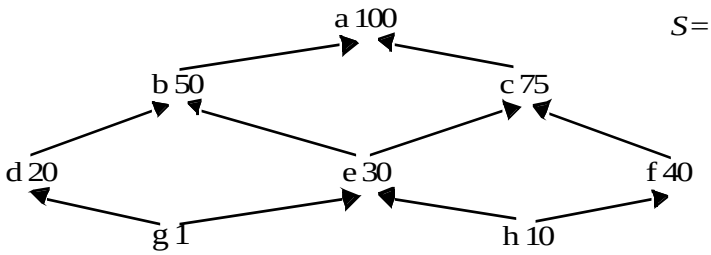
  wählt in jedem Schritt jeweils nur eine weitere Sicht aus

  wählt jeweils die Sicht v mit dem größten Vorteil B(v, M)

  liefert mindestens 63% des Gesamtvorteils des optimalen Algorithmus

Beispiel



$S = \{a,b,c,d,e,f,g,h\}$

$k = 4$

v	B(v,S) für Schritt 1	B(v,S) für Schritt 2	B(v,S) für Schritt 3
b	50 x 5 = 250	x	x
c	25 x 5 = 125	25 x 2 = 50	25 x 1 = 25
d	80 x 2 = 160	30 x 2 = 60	30 x 2 = 60
e	70 x 3 = 210	20 x 3 = 60	20 x 2 + 10 = 50
f	60 x 2 = 120	60 + 10 = 70	x
g	99 x 1 = 99	49 x 1 = 49	49 x 1 = 49
h	90 x 1 = 90	40 x 1 = 40	30 x 1 = 30

9.4 Attributorientierte Induktion

Motivation

Data Cube

manuelle Generalisierung

Manuelle Generalisierungsoperationen

- *Roll-Up*: Übergang zur nächsthöheren Generalisierungsebene
- *Drill-Down*: Übergang zur nächstniedrigeren Generalisierungsebene

Nachteil des manuellen Ansatzes

unklar ist:

- welche Dimensionen sollen benutzt werden?
- welcher Generalisierungsgrad ist sinnvoll?

⇒ automatische Generalisierung

9.4 Attributorientierte Induktion

Grundbegriffe

- Relation R, $R \subseteq D_1 \times D_2 \times ... \times D_d$, mit den Attributen $A_1, . . ., A_d$
- $|R|$: *Anzahl der Tupel* der Relation R
- für jedes Attribut A_i eine Konzepthierarchie C_i
- m_i: *Anzahl verschiedener Werte* des Attributs A_i in R
- *Generalisierungsgrad von* A_i, bezeichnet als g_i:
  die gemeinsame Konzeptebene der Werte von A_i
- *Generalisierungsgrad von* R, bezeichnet als G_R:

$$G_i = \sum_{i=1}^{d} g_i$$

- *Support* eines Tupels:
  Anzahl der Tupel aus R, die zu diesem Tupel generalisiert wurden

9.4 Attributorientierte Induktion

Grundbegriffe

- *Basisrelation R*:

  alle Attribute besitzen die Konzeptebene 0

- *generalisierte Relation R*:

  $G_R > 0$

- für jede generalisierte Relation:

  zusätzliches Attribut, das den *Support* des jeweiligen Tupels mißt

- *vollständig generalisierte Relation in Bezug auf T*, bezeichnet als R_T:

  $|R| \leq T$ (*tupelzahl-orientierte Generalisierung*) bzw.
  $\forall i, 1 \leq i \leq d: m_i \leq T$ (*attributwertzahl-orientierte Generalisierung*)

- Gesucht:

  eine möglichst große, vollständig generalisierte Relation in Bezug auf *T*

9.4 Algorithmus LCHR

Algorithmus [Han, Cai & Cercone 1993]

- Input: Relation *R*, Integer *T*
- Methode

  solange mehr als *T* Tupel in *R*
  wähle das nächste Attribut A_i aus;
  wenn die Werte von $A_i \neq$ ALLE sind
  für jedes Tupel aus *R*
  ersetze den Wert des Attributs A_i im Tupel durch seinen Vorgänger in C_i;
  eliminiere redundante Tupel und aktualisiere den Support der verbleibenden Tupel von *R*;
  sonst entferne das Attribut A_i von *R*

- Output : vollständig generalisierte Relation in Bezug auf *T*

9.4 Algorithmus LCHR

Auswahl des nächsten Attributs

Ziel: minimaler Generalisierungsgrad der endgültigen Relation

- wähle Attribut, das Anzahl der Tupel möglichst stark reduziert
- Heuristik: wähle Attribut A_i mit dem maximalen m_i

Ziel: ähnlicher Generalisierungsgrad aller Attribute

- wähle Attribut A_i mit minimalem Generalisierungsgrad g_i

Benutzerdefinierte Auswahl

- Nutzung von Anwendungswissen
- benutzerdefinierte Prioritäten für die Auswahl der Attribute

kritisch für die Qualität des Ergebnisses

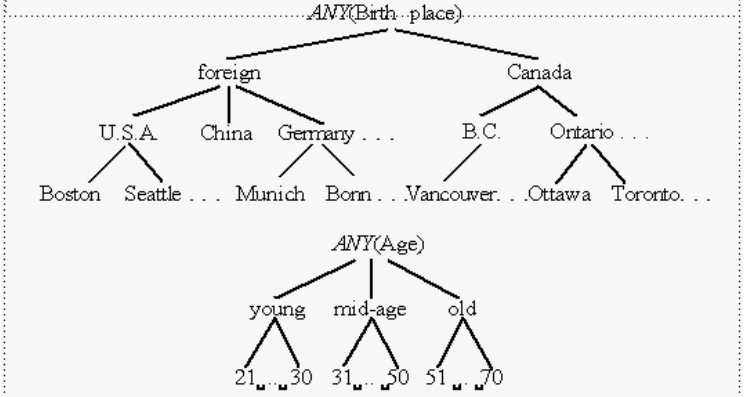
9.4 Algorithmus LCHR

Beispiel

Name§	Sex§	Age§	Birth place§	Department§	Position§	Salary§
Anderson§	female§	26§	Burnaby§	computer sc§	secretary§	26.000§
Bach§	male§	38§	Ottawa§	electrical eng§	lab manager§	41.000§
Barton§	female§	30§	Toronto§	chemistry§	junior lecturer§	28.000§
Benson§	male§	45§	Vancouver§	computer sc§	full professor§	63.000§
. . .§	. . .§	. . .§	. . .§	. . .§	. . .§	. . .§
Winton§	male§	38§	Seattle§	civil eng§	assoc. professor§	55.400§
Young§	male§	55§	Bonn §	german§	full professor§	68.000§

Kanadische Universitäten

Basisrelation



zwei der Konzepthierarchien

9.4 Algorithmus LCHR

Professoren nach einer Generalisierung§

Sex§	Age§	Birth place§	Department§	Position§	Salary§	Support§
male§	45§	Vancouver§	computer sc§	full professor§	63000§	1§
. . .§	. . .§	. . .§	. . .§	. . .§	. . .§	. . .§
male§	38§	Seattle§	civil eng§	assoc. professor§	55400§	1§
male§	55§	Bonn §	german§	full professor§	68000§	1§

Selektion der Professoren

Professoren nach zwei Generalisierungen§

Sex§	Age§	Birth place§	Department§	Position§	Salary§	Support§
male§	45§	B.C.§	computer sc§	full professor§	63000§	1§
. . .§	. . .§	. . .§	. . .§	. . .§	. . .§	. . .§
male§	38§	U.S.A.§	civil eng§	assoc. professor§	55400§	1§
male§	55§	Germany §	german§	full professor§	68000§	1§

$G_R = 1$

$G_R = 2$

Professoren vollständig generalisiert§

Sex§	Age§	Birth place§	Salary§	Support§
male§	old§	Canada§	high§	20§
male§	old§	foreign§	high§	15§
male§	mid-age§	Canada§	medium§	75§
male§	mid-age§	foreign§	medium§	130§
female§	mid-age§	Canada§	medium§	25§

vollständig verallgemeinerte Relation mit $T = 5$

9.4 Algorithmus FIGR

Algorithmus [Carter & Hamilton 1998]

Idee

benutze *d*-dimensionales Array *M*, das für jede Kombination von *möglichen* Attributwerten den Support repräsentiert

Methode

- Transformation *R* --> *M*

  für jedes Tupel aus *R* den Zähler der entsprechenden Zelle in *M* inkrementieren

- erster Durchlauf von *M*

  für jede Dimension die tatsächlich auftretenden Attributwerte zählen
  falls > *T*, generalisiere Attributwerte der jeweiligen Dimension genügend

- Transformation *M* --> *Gen*

  Matrix Gen besitzt für jeden der generalisierten Werte einen Eintrag
  zweiter Durchlauf von *M*, um Zähler der Zellen in *Gen* zu setzen

9.4 Algorithmus FIGR

Diskussion

- Laufzeitaufwand von LCHR

  $O(i \cdot n \cdot \log n)$ mit $n = |R|$ und i = Anzahl der Iterationen

- Laufzeitaufwand von FIGR

  $O(n + m)$ mit $n = |R|$ und $m = \prod_{i=1}^{d} |D_i|$

- Effizienz

  FIGR wesentlich effizienter, solange nicht $m >> n$

- Skalierbarkeit

  FIGR nur für kleine Werte von d und für kleine $|D_i|$ anwendbar, da sonst die Datenstruktur M nicht in den Hauptspeicher paßt

- Anwendbarkeit

  Zahl der gewünschten Tupel einfacher zu spezifieren als Zahl der gewünschten Attributwerte

9.4 Attributorientierte Induktion

Mit Vorverarbeitung und Evaluation der Ergebnisse

Name	Gender	Major	Birth-Place	Birth_date	Residence	Phone #	GPA
Jim Woodman	M	CS	Vancouver,BC, Canada	8-12-76	3511 Main St., Richmond	687-4598	3.67
Scott Lachance	M	CS	Montreal, Que, Canada	28-7-75	345 1st Ave., Richmond	253-9106	3.70
Laura Lee	F	Physics	Seattle, WA, USA	25-8-70	125 Austin Ave., Burnaby	420-5232	3.83
…	…	…	…	…	…	…	…
Removed	Retained	Sci,Eng, Bus	Country	Age range	City	Removed	Excl, VG,..

Basisrelation

Vorverarbeitung

Gender	Major	Birth_region	Age_range	Residence	GPA	Count
M	Science	Canada	20-25	Richmond	Very-good	16
F	Science	Foreign	25-30	Burnaby	Excellent	22
…	…	…	…	…	…	…

vollständig verallgemeinerte Relation

Birth_Region / Gender	Canada	Foreign	Total
M	16	14	30
F	10	22	32
Total	26	36	62

Präsentation des Ergebnisses

9.4 Anwendung: Klassifikation

Methode

- Nutzung der attributorientierten Generalisierung
- Gegeben

  zwei Relationen *Target Class* und *Contrasting Class*

  eine ganze Zahl *T*
- Gesucht

  Klassifikationsregeln und ihre Konfidenz
- *überlappendes Tupel*:

  ist sowohl in der Target Class als auch in der Contrasting Class enthalten

9.4 Anwendung: Klassifikation

Methode

Simultane Generalisierung

gleichzeitige Generalisierung der Tupel der beiden Klassen,

bis die Target Class höchstens noch *T* Tupel enthält

Nach jeder Generalisierung

Markieren der überlappenden Tupel

Generieren von Klassifikationsregeln

aus den beiden vollständig verallgemeinerten Relationen

Bestimmung ihrer Konfidenz aus dem Support der überlappenden Tupel

6.4 Anwendung: Klassifikation

Regeltypen

Charakterisierungsregeln

$$\forall X,\ target\_class(X) \Rightarrow condition(X)\quad [t : t\_weight]$$

➡ notwendige Bedingung

Diskriminanzregeln

$$\forall X,\ target\_class(X) \Leftarrow condition(X)\quad [d : d\_weight]$$

➡ hinreichende Bedingung

Beschreibungsregeln

$$\forall X,\ target\_cla\ ss(X) \Leftrightarrow$$
$$condition_{1}(X)[t : w_1, d : w'_1] \vee ... \vee condition_{n}(X)[t : w_n, d : w'_n]$$

➡ notwendig und hinreichende Bedingung

9.4 Anwendung: Klassifikation

Beispiel

Professoren vollständig generalisiert (Target Class)

Sex	Age	Birth place	Salary	Support	Mark
male	old	Canada	high	20	
male	old	foreign	high	15	
male	mid-age	Canada	medium	75	*
male	mid-age	foreign	medium	130	*
female	mid-age	Canada	medium	25	

Markierung überlappender Tupel

Instruktoren vollständig generalisiert (Contrasting Class)

Sex	Age	Birth place	Salary	Support	Mark
male	young	Canada	low	30	
male	mid-age	Canada	medium	25	*
female	young	Canada	low	12	
male	mid-age	foreign	medium	10	*

Sex = male ∧ Age = old
∧ Birth Place = Canada
∧ Salary = high

→ Professor (100%)

Sex = male ∧ Age = mid-age
∧ Birth Place = Canada
∧ Salary = medium

→ Professor (75%)

9.5 Inkrementelle attributorientierte Induktion

Motivation

- Updates der Basisrelation
- erfordern Updates aller generalisierten Relationen
- *inkrementelle* Updates der generalisierten Relationen:

  keine Anwendung des Generalisierungs-Algorithmus auf der

  aktualisierten Basisrelation
- Anforderungen an inkrementelle attributorientierte Generalisierung
  - *Effizienz*: signifikant kürzere Laufzeit
  - *Korrektheit*: identisches Ergebnis wie bei Anwendung des nicht-inkrementellen Algorithmus auf die aktualisierte Basisrelation

9.5 Inkrementelle attributorientierte Induktion

Inkrementelle Generalisierung mit Algorithmus LCHR

[Ester & Wittmann 1998]

R_{Gen}: ein beliebige generalisierte Relation von R

Inkrementelle Einfügungen

- einzufügendes Tupel analog R_{Gen} generalisieren und in R_{Gen} einfügen (R_{Gen}')
- falls $|R_{Gen}'| > T$, Algorithmus LCHR auf R_{Gen}' anwenden

Inkrementelle Löschungen

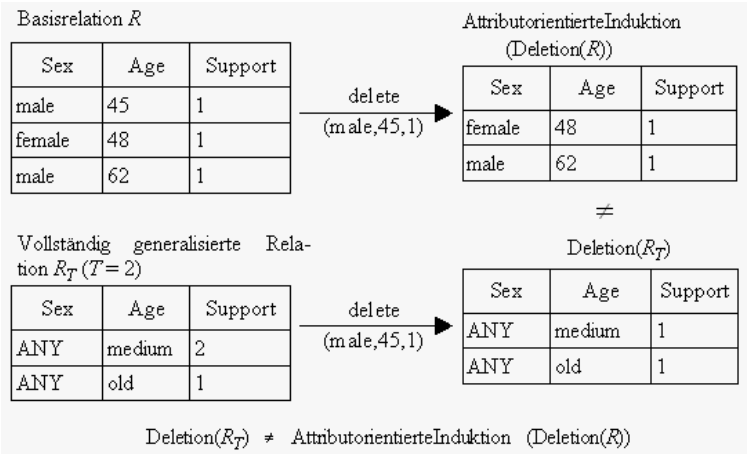
- zu löschendes Tupel analog R_{Gen} generalisieren und aus R_{Gen} löschen (R_{Gen}')
- nach diesem Update gilt immer $|R_{Gen}'| \leq T$

aber: Problem der Übergeneralisierung

9.5 Inkrementelle attributorientierte Induktion

Übergeneralisierung

- vollständig generalisierte Relation R_T heißt *übergeneralisiert*:
  Generalisierungsebene größer als bei Anwendung des nicht-inkrementellen Generalisierungsalgorithmus auf die aktualisierte Basisrelation
- Beispiel



9.5 Inkrementelle attributorientierte Induktion

Inkrementelle Generalisierung mit Ankerrelation

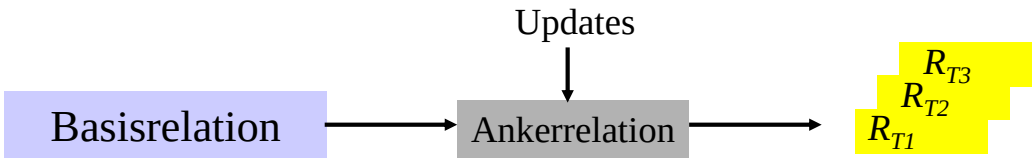
Trade-Off

- zur Erzielung der optimalen Effizienz
  Updates direkt auf der vollständig generalisierten Relation R_T durchführen
- zur Vermeidung der Übergeneralisierung
  Updates auf der Basisrelation R durchführen
  ⇒ als Kompromiß: Updates auf Ankerrelation

Methode

- *Ankerrelation*:
  generalisierte Relation mit einer Generalisierungsebene zwischen R und R_T
- Einfügungen und Löschungen auf der Ankerrelation ausgeführen
- Generalisierungs-Algorithmus auf die aktualisierte Ankerrelation anwenden

Inkrementelle Generalisierung mit Ankerrelation



R_{TAnc}

vollständig generalisierte Relationen

Methode

- Ankerrelation soll nur für alle vollständig generalisierten Relationen nötige Generalisierungen enthalten:

$$\forall i: T_{Anc} > T_i$$

- Spezifikation der Größe der Ankerrelation durch *Ankerreduktionsfaktor* (*ARF*):
  Anteil an der Kardinalität der Basisrelation

$$ARF = \frac{|R|}{T_{Anc}}$$

Inkrementelle Generalisierung mit Algorithmus FIGR

[Carter & Hamilton 1998]

Einfügungen

- entsprechende Zelle von *M* um 1 inkrementieren und die Attributwerte des neuen Tupels entsprechend *Gen* generalisieren
- falls nun für mindestens eines der Attribute $m_i > T$ gilt: dieses Attribut von *Gen* um eine Stufe generalisieren

Löschungen

- entsprechende Zelle von *M* um 1 dekrementieren
- falls der Wert dieser Zelle nun gleich 0 ist: die vollständig generalisierte Relation *Gen* neu aus *M* generalisieren
- andernfalls das gelöschte Tupel entsprechend *Gen* generalisieren und die entsprechende Zelle in *Gen* um 1 dekrementieren