Berechnung von Normalformen

10

In diesem Kapitel geht es um Algorithmen, die ein gegebenes Schema in Normalform zerlegen. Dazu führen wir als erstes den Armstrong Kalkül ein, mit dem die Hülle einer Menge von funktionalen Abhängigkeiten berechnet werden kann. Dann betrachten wir einen Algorithmus zur Berechnung der Hülle einer Menge von Attributen unter einer Menge von funktionalen Abhängigkeiten. Das heisst, wir berechnen all diejenigen Attribute, welche von einer gegebenen Attributmenge funktional abhängig sind. Weiter geben wir einen Algorithmus an, um eine minimale Überdeckung einer Menge von funktionalen Abhängigkeiten zu berechnen. Schliesslich untersuchen wir einen Zerlegungsalgorithmus um ein Schema verlustfrei in BCNF zu zerlegen, sowie einen Synthesealgorithmus um ein Schema verlustfrei und abhängigkeitserhaltend in die dritte Normalform zu zerlegen.

10.1 Armstrong-Kalkül

Die Hülle F^+ einer Menge F von funktionalen Abhängigkeiten wurde in Definition 9.12 unter Bezugnahme auf Erfüllbarkeit semantisch eingeführt. Hier studieren wir den *Armstrong-Kalkül*. Dieser liefert ein syntaktisches Verfahren, welches die Berechnung von F^+, ausgehend von F, ermöglicht.

In diesem Kapitel bezeichnet U die Menge aller Attribute, die in den involvierten Relationenschemata vorkommen. Ausgangspunkt für den Armstrong-Kalkül bildet eine Menge F von funktionalen Abhängigkeiten mit Attributen aus U. Sind $X, Y \subseteq U$, so schreiben wir $F \vdash X \to Y$, um auszudrücken, dass sich die funktionale Abhängigkeit $X \to Y$ aus F ableiten lässt.

Definition 10.1 (Armstrong-Kalkül). Gegeben sei eine Menge F von funktionalen Abhängigkeiten über U. Dann wird $F \vdash X \to Y$ für $X, Y \subseteq U$ induktiv definiert durch:

T. Studer, *Relationale Datenbanken*,
https://doi.org/10.1007/978-3-662-58976-2_10

1. Elemente von F. Ist $X \to Y$ ein Element von F, so gilt $F \vdash X \to Y$.
2. Reflexivität. Ist Y eine Teilmenge von X, so gilt $F \vdash X \to Y$.
3. Augmentation. Aus $F \vdash X \to Y$ und $Z \subseteq U$ folgt $F \vdash XZ \to YZ$.
4. Transitivität. Aus $F \vdash X \to Y$ und $F \vdash Y \to Z$ folgt $F \vdash X \to Z$.

Beispiel 10.2. Wir betrachten nochmals die Situation aus Beispiel 9.28 mit

$$U := \{\texttt{Autor}, \texttt{Jahrgang}, \texttt{Titel}\}$$

$$F_1 := \{\texttt{Autor} \to \texttt{Jahrgang}\}.$$

Somit gilt mit Regel *Elemente von F*

$$F_1 \vdash \texttt{Autor} \to \texttt{Jahrgang}.$$

Dann folgt mit Regel *Augmentation* (für $Z = \{\texttt{Autor}, \texttt{Titel}\}$)

$$F_1 \vdash \{\texttt{Autor}, \texttt{Titel}\} \to \{\texttt{Autor}, \texttt{Jahrgang}, \texttt{Titel}\}.$$

Die nächsten beiden Theoreme besagen, dass Herleitbarkeit im Armstrong-Kalkül und logische Folgerung für funktionale Abhängigkeiten übereinstimmen.

Theorem 10.3 (Armstrong-Kalkül, Korrektheit). *Ist F eine Menge von funktionalen Abhängigkeiten über U, so gilt für alle $X, Y \subseteq U$*

$$F \vdash X \to Y \quad \Longrightarrow \quad F \models X \to Y.$$

Dieses Theorem wird durch einfache Induktion nach der Länge der Herleitung im Armstrong-Kalkül gezeigt. Ist die funktionale Abhängigkeit $X \to Y$ ein Element von F oder gilt $Y \subseteq X$, so ist $F \models X \to Y$ trivialerweise erfüllt. Wurde $F \vdash X \to Y$ durch Augmentation oder Transitivität erschlossen, so folgt die Behauptung aus der Induktionsvoraussetzung mit Hilfe von Tupelberechnungen.

Anmerkung 10.4. Für die Korrektheit des Armstrong-Kalküls ist es wesentlich, dass wir die Erfüllung einer funktionalen Abhängigkeit via (9.1) definieren und nicht (analog zu unique Constraints) die Bedingung

$$s[X] = t[X] \quad \Longrightarrow \quad s[Y] \simeq t[Y] \tag{10.1}$$

verwenden. Um dies zu zeigen, definieren wir das Schema

$$\mathscr{S} := (\texttt{BuchId}, \texttt{Autor}, \texttt{Jahrgang})$$

mit den funktionalen Abhängigkeiten

$$F := \{\texttt{BuchId} \to \texttt{Autor}, \texttt{Autor} \to \texttt{Jahrgang}\}.$$

Wir betrachten nun folgende Instanz von $\mathscr{S}$:

Werke

BuchId	Autor	Jahrgang
1	null	1751
1	null	1765

Wir nehmen nun an, dass die Erfüllbarkeit von funktionalen Abhängigkeiten mit Hilfe von (10.1) definiert ist. Damit finden wir:

1. die Relation `Werke` erfüllt $\texttt{BuchId} \to \texttt{Autor}$,
2. die Relation `Werke` erfüllt $\texttt{Autor} \to \texttt{Jahrgang}$,
3. aber die Relation `Werke` erfüllt $\texttt{BuchId} \to \texttt{Jahrgang}$ nicht.

Das heisst, mit (10.1) ist die Transitivitätsregel des Armstrong-Kalküls nicht korrekt.

Theorem 10.5 (Armstrong-Kalkül, Vollständigkeit). *Ist F eine Menge von funktionalen Abhängigkeiten über U, so gilt für alle $X, Y \subseteq U$*

$$F \models X \to Y \quad \Longrightarrow \quad F \vdash X \to Y.$$

Wir verzichten hier auf den Beweis dieses Vollständigkeitssatzes. Stattdessen betrachten wir sofort eine unmittelbare Folgerung aus der Korrektheit und Vollständigkeit des Armstrong-Kalküls.

Korollar 10.6 (Charakterisierung der Hülle F^+). *Ist F eine Menge von funktionalen Abhängigkeiten über U, so gilt*

$$F^+ \overset{\textit{Definition } 9.12}{=} \{X \to Y \mid F \models X \to Y\} \overset{\textit{Theorems } 10.3,\ 10.5}{=} \{X \to Y \mid F \vdash X \to Y\}.$$

Beispiel 10.7. Im Beispiel 10.2 haben wir im Armstrong-Kalkül

$$F_1 \vdash \{\texttt{Autor}, \texttt{Titel}\} \to \{\texttt{Autor}, \texttt{Jahrgang}, \texttt{Titel}\}$$

hergeleitet. Mit obigem Korollar folgt nun daraus

$$\{\texttt{Autor}, \texttt{Titel}\} \to \{\texttt{Autor}, \texttt{Jahrgang}, \texttt{Titel}\} \in F_1^+.$$

Es folgen einige Deduktionsregeln für den Armstrong-Kalkül, die recht wichtig sind und sich mit den Regeln aus Definition 10.1 leicht beweisen lassen.

Theorem 10.8. *Ist F eine Menge von funktionalen Abhängigkeiten über U, so gilt für alle $X, Y, Z \subseteq U$:*

1. Vereinigung. $F \vdash X \to Y$ *und* $F \vdash X \to Z \quad \Longrightarrow \quad F \vdash X \to YZ,$
2. Zerlegung. $F \vdash X \to Y$ *und* $Z \subseteq Y \quad \Longrightarrow \quad F \vdash X \to Z.$
3. Einfachheit. $F \vdash X \to Y_1Y_2 \ldots Y_n \quad \Longleftrightarrow \quad F \vdash X \to Y_i$ *für alle* $1 \leq i \leq n.$

Die dritte Aussage dieses Theorems folgt unmittelbar aus der Vereinigungs- und Zerlegungseigenschaft. Sie besagt, dass jede funktionale Abhängigkeit durch funktionale Abhängigkeiten mit einelementigen rechten Seiten ausgedrückt werden kann.

Definition 10.9 (Einfache funktionale Abhängigkeit). Funktionale Abhängigkeiten mit einelementigen rechten Seiten werden als *einfache* funktionale Abhängigkeiten bezeichnet.

10.2 Hüllenberechnungen

Häufig ist es für das Design einer Datenbank wesentlich, ob für eine gegebene Menge F von funktionalen Abhängigkeiten und für Attributmengen X und Y die Beziehung $F \vdash X \to Y$ gilt. Da wir andererseits wissen, dass

$$F \vdash X \to Y \quad \Longleftrightarrow \quad X \to Y \in F^+,$$

könnten wir im Prinzip die Frage, ob $F \vdash X \to Y$ der Fall ist, dadurch beantworten, dass wir die Menge F^+ systematisch auflisten. Dieses Vorgehen ist jedoch in der Regel unrealistisch, da F^+ für eine n-elementige Menge F im schlimmsten Fall $O(2^n)$ viele Elemente besitzen kann.

Beispiel 10.10. Wir betrachten die Menge

$$F := \{A \to B_1,\ A \to B_2,\ \ldots,\ A \to B_n\}.$$

Durch mehrfache Anwendung der Vereinigungseigenschaft aus Theorem 10.8 erhalten wir dann sehr leicht, dass F^+ mindestens $2^n - 1$ Elemente besitzt.

In Definition 9.12 haben wir die Hülle von einer Menge von funktionalen Abhängigkeiten betrachtet. Jetzt definieren wir die Hülle von einer Menge von Attributen.

Definition 10.11 (Attributhülle X^+). Gegeben seien eine Menge F von funktionalen Abhängigkeiten über der universellen Menge U sowie eine Menge $X \subseteq U$. Die *Attributhülle X^+ von X unter F* ist dann definiert durch

$$X^+ := \{A \in U \mid F \vdash X \to A\}.$$

Im folgenden Lemma geben wir nun eine weitere charakterisierende Eigenschaft von X^+ an und betrachten im Anschluss daran einen Algorithmus zur Berechnung von X^+.

Lemma 10.12. *Für alle Mengen F von funktionalen Abhängigkeiten über der universellen Menge U sowie für alle Mengen $X, Y \subseteq U$ gilt*

$$F \vdash X \to Y \quad \Longleftrightarrow \quad Y \subseteq X^+.$$

Beweis. Es sei Y die Attributmenge $\{A_1, A_2, \ldots, A_n\}$. Wir zeigen nun zuerst die Richtung von links nach rechts. Es gelte also $F \vdash X \to Y$. Aufgrund der Zerlegungseigenschaft aus Theorem 10.8 folgt $F \vdash X \to A_i$ für alle $1 \leq i \leq n$. Dies ergibt $Y \subseteq X^+$.

Um die Richtung von rechts nach links zu zeigen, nehmen wir $Y \subseteq X^+$ an. Mit der Definition von X^+ folgt daraus $F \vdash X \to A_i$ für alle $1 \leq i \leq n$. Mit der Vereinigungseigenschaft aus Theorem 10.8 erhalten wir also unmittelbar $F \vdash X \to Y$. □

Zusammen mit Theorem 10.5 erlaubt uns dieses Lemma also, die Frage nach $F \models X \to Y$ auf die Frage nach $Y \subseteq X^+$ zu reduzieren. Diese Überlegung bildet die Grundlage des folgenden Algorithmus.

Algorithmus zur Berechnung von X^+

Gegeben seien eine universelle Menge U von Attributen, eine Menge F von funktionalen Abhängigkeiten über U sowie eine Menge $X \subseteq U$. Folgender Algorithmus berechnet die Attributhülle X^+ von X unter F:

```
EINGABE: F, X
         R := X;   alt_R := ∅;
         WHILE R ≠ alt_R DO
            alt_R := R;
            FOR EACH V → W IN F DO
                  IF V ⊆ R THEN R := R ∪ W
AUSGABE: X⁺ := R
```

Es ist leicht zu sehen, dass der oben angegebene Algorithmus im schlimmsten Fall quadratisch in der Anzahl der Elemente von F ist. Es gibt sogar einen etwas komplizierteren Algorithmus zur Berechnung der Attributhülle von X unter F, der nur linear in der Grösse von F ist.

Beispiel 10.13. Bezeichne U die Menge der Attribute $\{A, B, C, D, E, F, G\}$, F die Menge der funktionalen Abhängigkeiten

$$\{\, AC \to G,\ BD \to CE,\ E \to A,\ G \to BF \,\}$$

und X die Teilmenge $\{C, E\}$ von U. Dann nimmt R bei der Berechnung von X^+ im obigen Algorithmus der Reihe nach folgende Werte an:

$$\{C, E\}, \quad \{C, E, A\}, \quad \{C, E, A, G\}, \quad \{C, E, A, G, B, F\}.$$

10.3 Minimale Überdeckungen

In diesem Abschnitt betrachten wir Überdeckung und Äquivalenz von Mengen funktionaler Abhängigkeiten, beschäftigen uns mit einem Äquivalenztest und studieren minimale Überdeckungen.

Definition 10.14. Gegeben seien Mengen F und G von funktionalen Abhängigkeiten über U.

1. G *überdeckt* F, in Zeichen $F \leq G$, falls $F^+ \subseteq G^+$.
2. F und G sind *äquivalent*, falls F und G sich gegenseitig überdecken, das heisst

$$F \simeq G \quad :\Longleftrightarrow \quad F \leq G \wedge G \leq F.$$

Aufgrund früherer Überlegungen wissen wir, dass in der Regel die Äquivalenz von F und G **nicht** dadurch getestet werden sollte, dass wir F^+ und G^+ berechnen und auf Gleichheit untersuchen. Stattdessen betrachten wir zuerst ein Lemma und dann ein effizienteres Verfahren zur Abklärung von Äquivalenzen.

Lemma 10.15. *Gegeben seien Mengen F und G von funktionalen Abhängigkeiten über U. Dann gilt*

$$F \subseteq G^+ \quad \Longleftrightarrow \quad F \leq G.$$

Beweis. Die Richtung von rechts nach links ist offensichtlich. Um die Umkehrung zu beweisen, nehmen wir $F \subseteq G^+$ an. Damit gilt

$$F^+ = \{Y \to Z \mid F \vdash Y \to Z\}$$
$$\subseteq \{Y \to Z \mid G^+ \vdash Y \to Z\} = (G^+)^+ = G^+.$$

Daraus folgt mit der vorhergehenden Definition sofort die Behauptung, dass $F \leq G$ gilt. □

Algorithmen für Überdeckungs- und Äquivalenztest

Den Ausgangspunkt bilden Mengen F und G von funktionalen Abhängigkeiten über U. Mit folgendem Algorithmus können wir entscheiden, ob die Menge G die Menge F überdeckt:

EINGABE: F, G.

Überprüfe für jedes $Y \to Z \in F$, ob $Y \to Z \in G^+$ durch:

Berechne Y^+ unter G und teste $Z \subseteq Y^+$.

Falls dies immer der Fall ist,

AUSGABE: $F \leq G$; anderenfalls AUSGABE: $F \not\leq G$.

Damit erhalten wir natürlich auch einen Algorithmus für den Test der Äquivalenz von F und G, indem wir sowohl $F \leq G$ als auch $G \leq F$ überprüfen.

Lemma 10.16. *Jede Menge F von funktionalen Abhängigkeiten über U ist zu einer Menge G äquivalent, die nur einfache funktionale Abhängigkeiten enthält, d. h. funktionale Abhängigkeiten mit einelementigen rechten Seiten.*

Beweis. Wir gehen aus von einer Menge F von funktionalen Abhängigkeiten über U und definieren

$$G_F := \{X \to A \mid \exists Y(X \to Y \in F \text{ und } A \in Y)\}.$$

Aufgrund der Zerlegungseigenschaft folgt $X \to A \in F^+$ für alle A und Y mit $A \in Y$ und $X \to Y \in F$. Folglich ist $G_F \subseteq F^+$.

Ist andererseits Y die Menge $\{A_1, A_2, \ldots, A_n\}$ und gilt

$$X \to A_i \in G_F \quad \text{für alle } 1 \leq i \leq n,$$

so folgt mit der Vereinigungseigenschaft auch $X \to Y \in G_F^+$. Daraus folgt $F \subseteq G_F^+$.

Aus $G_F \subseteq F^+$ und $F \subseteq G_F^+$ folgt mit Lemma 10.15, dass $F \simeq G_F$ gilt. Da G_F offensichtlich eine Menge von einfachen funktionalen Abhängigkeiten ist, haben wir unser Lemma bewiesen. □

Definition 10.17. Eine Menge F von funktionalen Abhängigkeiten über U heisst *minimal*, falls folgende Bedingungen erfüllt sind:

1. F enthält nur einfache funktionale Abhängigkeiten.
2. Keine funktionale Abhängigkeit in F ist redundant, d. h.

$$(\forall X \to A \in F)(F \setminus \{X \to A\} \not\simeq F). \tag{M2}$$

3. Kein Attribut auf der linken Seite einer funktionalen Abhängigkeit aus F ist redundant, d. h.

$$(\forall X \to A \in F)(\forall Y \subsetneq X)(F \setminus \{X \to A\} \cup \{Y \to A\} \not\simeq F). \tag{M3}$$

Man kann direkt sehen, ob F nur aus einfachen funktionalen Abhängigkeiten besteht. Die Überprüfung auf Redundanz der funktionalen Abhängigkeiten von F geschieht durch das Testen auf

$$A \in X^+ \text{ unter } F \setminus \{X \to A\}$$

für alle $X \to A \in F$. Schliesslich kann man auf Redundanz bei den Attributen auf der linken Seite testen, indem wir für alle $X \to A \in F$ und $Y \subsetneq X$ feststellen, ob

$$A \in Y^+ \text{ unter } F.$$

Diese drei Punkte liefern damit ein Verfahren, um abzuklären, ob F minimal ist.

Definition 10.18. Gegeben sei eine Menge F von funktionalen Abhängigkeiten über U. Eine *minimale Überdeckung von* F ist eine minimale Menge von funktionalen Abhängigkeiten, welche zu F äquivalent ist.

In einem nächsten Schritt geben wir nun einen Algorithmus an, der für jede Menge F von funktionalen Abhängigkeiten über U eine minimale Überdeckung $\texttt{MU}(F)$ von F berechnet.

Algorithmus für minimale Überdeckungen

EINGABE: Eine Menge F von funktionalen Abhängigkeiten über U.

1. Spalte alle nicht-einfachen funktionalen Abhängigkeiten in F auf. Nenne die neue Menge F'.
2. Entferne sukzessive alle redundanten Attribute im Sinne von (M3) der vorhergehenden Definition aus F'. Nenne die neue Menge F''.
3. Entferne sukzessive alle redundanten funktionalen Abhängigkeiten im Sinne von (M2) der vorhergehenden Definition aus F''. Nenne die neue Menge F'''.

AUSGABE: $\mathtt{MU}(F) := F'''$.

Lemma 10.19. *Für jede Menge F von funktionalen Abhängigkeiten über U ist $\mathtt{MU}(F)$ eine minimale Überdeckung von F.*

Wir verzichten auf die explizite Angabe eines Beweises dieses Lemmas, da er sich recht direkt aus der Beschreibung des Algorithmus zur Berechnung von $\mathtt{MU}(F)$ und den begleitenden Überlegungen ergibt.

Beispiel 10.20. Wir gehen aus von einer Menge F von funktionalen Abhängigkeiten über der Attributmenge $\{A, B, C\}$ mit

$$F = \{ AB \to C,\ A \to AB,\ B \to A \}.$$

Nun wenden wir den obigen Algorithmus an.

1. Schritt: Die einzige nicht-einfache funktionale Abhängigkeit in F ist $A \to AB$. Wir spalten diese auf und erhalten so die Menge

$$F' = \{ AB \to C,\ A \to A,\ A \to B,\ B \to A \}.$$

2. Schritt: Wir testen, ob A redundant in $AB \to C$ ist. Dazu überprüfen wir, ob

$$C \in \{B\}^+ \text{ unter } F'$$

gilt. Da $\{B\}^+$ unter F' die Menge $\{A, B, C\}$ ist, ist dies der Fall. Daher ist A redundant in $AB \to C$ und wird gestrichen. Weitere redundante Attribute auf linken Seiten gibt es nicht. Also erhalten wir

$$F'' = \{ B \to C,\ A \to B,\ B \to A,\ A \to A \}.$$

3. Schritt: Wir entfernen alle redundanten funktionalen Abhängigkeiten aus F''. Offensichtlich gilt

$$A \in \{A\}^+ \text{ unter } F'' \setminus \{A \to A\}.$$

Folglich ist $A \to A$ in F'' redundant und wird entfernt. Weitere Redundanzen gibt es dann nicht mehr. Es folgt also

$$\texttt{MU}(F) = \{ B \to C,\ A \to B,\ B \to A \}.$$

Anmerkung 10.21. Sei F eine Menge von funktionalen Abhängigkeiten. Dann kann es mehrere minimale Überdeckungen von F geben. D. h. im Allgemeinen ist die minimale Überdeckung von F nicht eindeutig.

Betrachte zum Beispiel die beiden Mengen von funktionalen Abhängigkeiten über der Attributmenge $\{A, B, C\}$:

$$F_1 := \{A \to B,\ B \to A,\ A \to C,\ C \to A\},$$
$$F_2 := \{A \to B,\ B \to C,\ C \to A\}.$$

Beide Mengen sind offensichtlich minimal und es gilt $F_1^+ = F_2^+$.

10.4 Zerlegungen in BCNF und 3NF

In Theorem 9.34 haben wir gesehen:

1. jedes Schema kann verlustfrei in BCNF zerlegt werden;
2. jedes Schema kann verlustfrei und abhängigkeitserhaltend in 3NF zerlegt werden.

In diesem Abschnitt wollen wir nun Algorithmen angeben, um solche Zerlegungen zu erzeugen. Dabei werden wir Attributmengen wie folgt als Schemata betrachten: Eine Attributmenge X kann für ein beliebiges Schema stehen, welches alle Attribute aus X enthält.

Algorithmus für die BCNF-Zerlegung

Gegeben seien ein Relationenschema $\mathscr{S}$ mit einer Menge von funktionalen Abhängigkeiten F. Der folgende Algorithmus berechnet eine Zerlegung $\mathscr{Z}$ von $\mathscr{S}$, welche in BCNF bezüglich F ist.

EINGABE: $\mathscr{S}$, F

$\mathscr{Z} := \{\mathscr{S}\}$

WHILE es gibt $\mathscr{S}_i \in \mathscr{Z}$ mit $\mathscr{S}_i$ nicht in BCNF bez. $\Pi_{\mathscr{S}_i}(F)$ DO

wähle ein solches $\mathscr{S}_i$

wähle disjunkte $X, Y, Z \subseteq \mathscr{S}_i$ mit

$$X \cup Y \cup Z = \mathscr{S}_i \text{ und}$$

$$X \to Y \in F^+ \text{ und}$$

$$X \to A \notin F^+ \text{ für alle } A \in Z$$

$\mathscr{Z} := \big(\mathscr{Z} \setminus \{\mathscr{S}_i\}\big) \cup \big\{X \cup Y,\ X \cup Z\big\}$

AUSGABE: $\mathscr{Z}$

Die Grundidee dieses Algorithmus ist folgende. Wähle ein Schema $\mathscr{S}_i$ welches die BCNF Bedingungen verletzt bezüglich einer funktionalen Abhängigkeit $X \to Y$. Das Schema $\mathscr{S}_i$ wird dann gemäss Abb. 10.1 in zwei Schemata zerlegt.

Weil $X \to Y \in F^+$ erfüllt sein muss, garantiert Lemma 9.19, dass diese Zerlegung von $\mathscr{S}_i$ in $\{X \cup Y,\ \mathscr{S}_i \setminus Y\}$ verlustfrei ist. Dieser Schritt wird solange wiederholt, bis alle Schemata der Zerlegung in BCNF sind.

Beispiel 10.22. Wir beginnen mit dem Schema aus Beispiel 9.32, welches BCNF verletzt. Wir betrachten also

$$\mathscr{S}_3 = (\texttt{Stadt}, \texttt{Str}, \texttt{PLZ})$$
$$F_3 = \big\{\{\texttt{Stadt}, \texttt{Str}\} \to \{\texttt{PLZ}\},\quad \{\texttt{PLZ}\} \to \{\texttt{Stadt}\}\big\}.$$

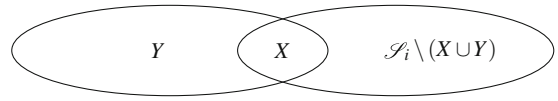
Wir beginnen nun mit $\mathscr{Z} := \{\mathscr{S}_3\}$. Es gilt $\mathscr{S}_3 \in \mathscr{Z}$ ist nicht in BCNF bezüglich F_3. Wir wählen nun

$$X := \{\texttt{PLZ}\} \qquad Y := \{\texttt{Stadt}\} \qquad Z := \{\texttt{Str}\}.$$

Es gilt $X \cup Y \cup Z = \mathscr{S}_3$ und $X \to Y \in F_3^+$ und $X \to \texttt{Str} \notin F_3^+$. Wir setzen $\mathscr{Z}$ neu auf

$$\big\{(\texttt{PLZ}, \texttt{Stadt}),\ (\texttt{PLZ}, \texttt{Str})\big\}.$$

Abb. 10.1 BCNF Zerlegung



Jetzt sind alle Elemente von $\mathscr{Z}$ in BCNF und der Algorithmus gibt die Zerlegung $\mathscr{Z}$ als Resultat zurück.

Es lässt sich zeigen, dass ein Algorithmus, der iterativ ein Schema in zwei Schemata zerlegt, im Allgemeinen nicht eine abhängigkeitserhaltende Zerlegung erzeugen kann. Somit benötigen wir eine etwas komplexere Methode um eine abhängigkeitserhaltende Zerlegung in 3NF zu finden. Dabei benutzen wir vorgängig Lemma 10.19, um zu einer gegebenen Menge F von funktionalen Abhängigkeiten eine minimale Überdeckung zu berechnen.

Algorithmus für die 3NF-Zerlegung

Gegeben seien ein Schema $\mathscr{S}$ sowie eine *minimale* Menge F von funktionalen Abhängigkeiten. Für jede Abhängigkeit $X \to A \in F$ definieren wir ein Schema $\mathscr{S}_{X\to A}$, so dass

$$\mathscr{S}_{X\to A} = X \cup \{A\}$$

im mengentheoretischen Sinn.

EINGABE: $\mathscr{S}$, F
$\mathscr{Z} := \{\mathscr{S}_{X\to A} \mid X \to A \in F\}$
IF kein $\mathscr{S}_{X\to A} \in \mathscr{Z}$ enthält Schlüssel für $\mathscr{S}$ bez. F THEN
wähle Schema $\mathscr{K}$, welches Schlüssel für $\mathscr{S}$ ist
$\mathscr{Z} := \mathscr{Z} \cup \{\mathscr{K}\}$
AUSGABE: $\mathscr{Z}$

Die so erhaltene Ausgabemenge $\mathscr{Z}$ ist dann die gewünschte Zerlegung von $\mathscr{S}$ in 3NF bezüglich F.

Beispiel 10.23. Wir betrachten das Schema aus Beispiel 9.30, welches nicht in dritter Normalform ist. Wir haben

$$\mathscr{S}_2 = (\texttt{BuchId}, \texttt{Autor}, \texttt{Jahrgang}, \texttt{Titel})$$
$$F_2 = \big\{\{\texttt{BuchId}\} \to \{\texttt{Autor}, \texttt{Jahrgang}, \texttt{Titel}\}, \{\texttt{Autor}\} \to \{\texttt{Jahrgang}\}\big\}.$$

Zuerst berechnen wir die minimale Überdeckung $\texttt{MU}(F_2)$ von F_2.

1. Durch Aufspalten erhalten wir

$$F_2' := \{\, \texttt{BuchId} \to \texttt{Autor},\ \texttt{BuchId} \to \texttt{Jahrgang}$$
$$\texttt{BuchId} \to \texttt{Titel},\ \texttt{Autor} \to \texttt{Jahrgang} \,\}.$$

2. Redundante Attribute entfernen. Die linken Seiten sind alle ein-elementig. Es können also keine Attribute entfernt werden. Somit haben wir

$$F_2'' := F_2'.$$

3. Redundante Abhängigkeiten entfernen. Wir finden

$$\texttt{Jahrgang} \in \texttt{BuchId}^+ \text{ unter } F_2'' \setminus \{\texttt{BuchId} \to \texttt{Jahrgang}\}.$$

   Somit gilt

$$F_2'' \setminus \{\texttt{BuchId} \to \texttt{Jahrgang}\} \simeq F_2''$$

   und wir setzen

$$\texttt{MU}(F_2) := F_2''' := F_2'' \setminus \{\texttt{BuchId} \to \texttt{Jahrgang}\}.$$

Als Input für den 3NF-Algorithmus verwenden wir nun $\mathscr{S}_2$ und $\texttt{MU}(F_2)$. Im ersten Schritt erhalten wir so die Zerlegung

$$\mathscr{Z} = \big\{\, \{\texttt{BuchId}, \texttt{Autor}\},\ \{\texttt{BuchId}, \texttt{Titel}\},\ \{\texttt{Autor}, \texttt{Jahrgang}\} \,\big\}.$$

Das Schema $\{\texttt{BuchId}, \texttt{Autor}\} \in \mathscr{Z}$ enthält einen Schlüssel für $\mathscr{S}_2$ bezüglich $\texttt{MU}(F_2)$. Somit müssen wir im zweiten Schritt kein Schema hinzufügen.

Damit ist $\mathscr{Z}$ eine verlustfreie und abhängigkeitserhaltende Zerlegung des Schemas $\mathscr{S}_2$ in die dritte Normalform.

Anmerkung 10.24. Die Zerlegung $\mathscr{Z}$ im obigen Beispiel ist offensichtlich nicht optimal. Die beiden Schemata

$$\{\texttt{BuchId}, \texttt{Autor}\} \text{ und } \{\texttt{BuchId}, \texttt{Titel}\}$$

könnten einfach zusammengefasst werden.

Dieser Effekt entsteht, weil eine minimale Überdeckung gemäss Definition aus *einfachen* funktionalen Abhängigkeiten besteht. Wir könnten nach der Berechnung von $\texttt{MU}(F_2)$ die Abhängigkeiten mit gleichen linken Seiten zusammenfassen zu

$$\mathrm{MU}'(F_2) := \big\{ \{\texttt{BuchId}\} \to \{\texttt{Autor}, \texttt{Titel}\},\ \{\texttt{Autor}\} \to \{\texttt{Jahrgang}\} \big\}$$

und diese Menge als Input für den NF3-Algorithmus verwenden. Dies würde etwas bessere Ergebnisse liefern.

Anmerkung 10.25. Es gibt eine weitere Konstellation, in welcher der 3NF Algorithmus nicht zu einer optimalen Zerlegung führt. Wir betrachten nochmals das Schema

$$\begin{aligned} \mathscr{S}_3 &= (\texttt{Stadt}, \texttt{Str}, \texttt{PLZ}) \\ F_3 &= \big\{ \{\texttt{Stadt}, \texttt{Str}\} \to \{\texttt{PLZ}\}, \quad \{\texttt{PLZ}\} \to \{\texttt{Stadt}\} \big\}. \end{aligned}$$

Der Algorithmus liefert die Zerlegung

$$\big\{(\texttt{Stadt}, \texttt{Str}, \texttt{PLZ}),\ (\texttt{Stadt}, \texttt{PLZ})\big\}.$$

Diese Zerlegung erfüllt zwar die 3NF Bedingungen, ist aber nicht optimal. Offensichtlich ist das Schema (Stadt, PLZ) im Schema (Stadt, Str, PLZ) enthalten und damit überflüssig. Wir könnten den Algorithmus verbessern, indem wir am Ende noch überprüfen, ob es $\mathscr{S}', \mathscr{S}'' \in \mathscr{Z}$ gibt, so dass $\mathscr{S}' \subsetneq \mathscr{S}''$ gilt. Falls dies der Fall ist, entfernen wir $\mathscr{S}'$ aus $\mathscr{Z}$.

Beachte, dass in diesem Beispiel die funktionale Abhängigkeit

$$\{\texttt{PLZ}\} \to \{\texttt{Stadt}\} \tag{10.2}$$

für beide Schemata der Zerlegung gilt. Das heisst, sowohl

$$(\texttt{Stadt}, \texttt{Str}, \texttt{PLZ}) \quad \text{als auch} \quad (\texttt{Stadt}, \texttt{PLZ})$$

müssen (10.2) erfüllen. Damit können wir (Stadt, PLZ) aus der Zerlegung entfernen, ohne dass Abhängigkeiten verloren gehen.

Beispiel 10.26. In diesem Beispiel zeigen wir, weshalb wir möglicherweise ein Schema $\mathscr{K}$, welches ein Schlüssel für $\mathscr{S}$ ist, zu der Zerlegung in 3NF hinzufügen müssen. Wir betrachten ein Schema

$$\mathscr{S} := (\,\underline{A},\ \underline{B},\ C\,)$$

mit der Menge von funktionalen Abhängigkeiten

$$F := \{\, B \to C \,\}.$$

Wir wenden nun den Algorithmus zur Zerlegung in 3NF an. Nach dem ersten Schritt erhalten wir

$$\mathscr{Z} = \big\{\{B, C\}\big\},$$

da $B \to C$ die einzige funktionale Abhängigkeit in F ist. Wir sehen, dass die Menge $\mathscr{Z}$ nach diesem ersten Schritt noch nicht alle Attribute aus $\mathscr{S}$ abdeckt. Die Menge $\mathscr{Z}$ ist also noch keine Zerlegung von $\mathscr{S}$. Der Algorithmus korrigiert dies im 2. Schritt, indem er noch einen Schlüssel $\mathscr{K}$ für $\mathscr{S}$ zu $\mathscr{Z}$ hinzufügt. Der einzige Schlüssel für $\mathscr{S}$ ist $\{A, B\}$ und wir erhalten somit

$$\mathscr{Z} = \big\{\{B, C\},\ \{A, B\}\big\}$$

als 3NF–Zerlegung des Schemas $\mathscr{S}$.

Mit diesem Beispiel beschliessen wir dieses Buch und hoffen, dass Sie, liebe Leserinnen und Leser, das Wissen über relationale Datenbanken persistent gespeichert haben und effizient darauf zugreifen können.

Weiterführende Literatur[1]

1. Armstrong, W.W.: Dependency structures of data base relationships. In: IFIP Congress, S. 580–583 (1974)
2. Bernstein, P.A.: Synthesizing third normal form relations from functional dependencies. ACM Trans. Database Syst. **1**(4), 277–298 (1976). https://doi.org/10.1145/320493.320489
3. Biskup, J., Dayal, U., Bernstein, P.A.: Synthesizing independent database schemas. In: Proceedings of the 1979 ACM SIGMOD International Conference on Management of Data, SIGMOD '79, 143–151. ACM (1979). https://doi.org/10.1145/582095.582118
4. Tsou, D.M., Fischer, P.C.: Decomposition of a relation scheme into Boyce-Codd normal form. SIGACT News **14**(3), 23–29 (1982). https://doi.org/10.1145/990511.990513

[1] Armstrong [1] präsentierte seinen Kalkül zur Beschreibung von funktionalen Abhängigkeiten 1974. Bernstein [2] und Biskup et al. [3] studieren Synthesealgorithmen, um verlustfreie und abhängigkeitserhaltende Zerlegungen in dritte Normalform zu erzeugen. Die Dekomposition eines Schemas in Boyce-Codd Normalform wird unter anderem von Tsou und Fischer [4] untersucht.