Vortrag 3. Jahrestagung GI 1973

in Proc. 3. Jahrestagung der GI, Springer LNCS 1, pages 363-371, 1973.

DATENSTRUKTUREN UND Q-SYSTEME - EINE MATHEMATISCHE STUDIE

H.-D. Ehrich

1. Einleitung

Beim Entwurf verallgemeinerter Datenbanksysteme spielt die strukturelle Unabhängigkeit der Daten eine große Rolle: die Organisation der gespeicherten Daten soll (bis zu einem gewissen Grade) unabhängig sein von der logischen Beschreibung der Daten auf der Benutzerebene. Wenn der Benutzer nicht mit der aktuellen Speicherorganisation befaßt ist, kann er in seinen Programmen und Anfragen auch nicht auf diese Bezug nehmen. Das heißt, er kann sich nur formal auf Daten, Beziehungen zwischen Daten und Operationen auf Daten beziehen. Es ist dann Aufgabe des Systems, die Daten und die Beschreibung der Daten zentral zu verwalten und die Operationen wie Suche, Änderung etc. zentral zu steuern.

Damit das System diese Aufgaben wahrnehmen kann, müssen die Daten, die Beziehungen zwischen Daten, die Operationen auf Daten und die Abbildungen zwischen verschiedenen Darstellungen formal beschrieben werden.

In dem vorliegenden Beitrag wird ein mathematisches Modell für Datenstrukturen und Anfrageoperationen untersucht. Nach der Definition und Erläuterung dieses "Attribut-Modells" im nächsten Abschnitt werden im Abschnitt 3 dessen Beziehungen zum Relationen-Modell von Codd [1] untersucht. Dann werden im Abschnitt 4 "Q-Systeme", das sind kombinierte Systeme (Datenstrukturen, Anfrageoperationen), definiert,und deren Relevanz für die Untersuchung praktischer Speicher- und Zugriffsorganisationen wird erörtert. Auf der Grundlage eines plausiblen Äquivalenzbegriffs für Q-Systeme werden zum Schluß äquivalente Umformungen von Q-Systemen betrachtet.

2. Das Attribut-Modell

Die Grundkonzeption des Datenstrukturmodells, welches hier präzisiert werden soll, findet sich mehrfach in der Literatur. Insbesondere sei auf den "relational data file" von Salton [6] und die Datenstruktur der assoziativen Sprache LEAP von Feldman und Rovner [2,5] hingewiesen.

Definition 2.1.: Eine Datenstruktur ist ein Quadrupel $D=(\Omega,A,V,\rho)$, wobei Ω, A und V endliche Mengen von Objekten, Attributen bzw. Werten sind. ρ ist eine dreistellige Relation auf Ω, A und V : $\rho \subset \Omega \times A \times V$. Folgende Bedingungen werden gefordert:

1. $pr_1(\rho)=\Omega$ 2. $pr_2(\rho)=A$ 3. $pr_3(\rho)=V$

Hierbei bezeichnet $pr_i(\rho)$ die Projektion der i-ten Komponente von ρ, $i=1,2,3$, also die Menge der Elemente, die in der i-ten Komponente von ρ vorkommen.

Die Beziehung $(\omega,a,v)\varepsilon\rho$ wird folgendermaßen interpretiert:

"Das Attribut a des Objekts ω hat den Wert v"

oder

"Das Objekt ω hat die Eigenschaft (a,v)"

Beispiel 1 :

(Karl , arbeitet an , Entwurf)
(Fritz , arbeitet an , Programmierung)
(Hans , leitet , Dokumentation)
(Emil , arbeitet an , Dokumentation)
(Emil , arbeitet unter , Hans)
.....

Alle Eigenschaften eines festen Objekts fassen wir zu einem Satz zusammen.

Definition 2.2.: Sei $D=(\Omega,A,V,\rho)$ eine Datenstruktur und $\omega\varepsilon\Omega$. Der Satz von ω ist

$$R_\omega := \{ (a,v) \mid (\omega,a,v)\varepsilon\rho \}$$

Durch die Äquivalenzrelation $\omega\equiv\omega' :\Leftrightarrow pr_1(R_\omega)=pr_1(R_{\omega'})$ wird eine Klasseneinteilung $K=\{\Omega_1,\ldots\Omega_p\}$ auf Ω definiert, in der Objekte mit "gleichem Satzformat" zu Äquivalenzklassen zusammengefaßt werden:

$$\Omega = \bigcup_{i=1}^{p} \Omega_i \quad , \quad \Omega_i \cap \Omega_j = \emptyset \text{ für } 1 \leq i < j \leq p$$

Wegen der Bedingung 1 der Definition 2.1. ist $pr_1(R_\omega)\neq\emptyset$ für alle $\omega\varepsilon\Omega$, und aufgrund der Definition der Äquivalenzrelation $\equiv$ können wir jeder Objektklasse Ω_i, $1\leq i\leq p$, eindeutig die Attributmenge $A_i := pr_1(R_\omega)$ für $\omega\varepsilon\Omega_i$ zuordnen. Wegen der Bedingung 2 der Definition 2.1. ist dann

$$A = \bigcup_{i=1}^{p} A_i \quad .$$

Damit können wir die Relation ρ in p paarweise disjunkte Relationen

$$\rho_i \subset \Omega_i \times A_i \times V$$

zerlegen, sodaß gilt:

$$\rho = \bigcup_{i=1}^{p} \rho_i \quad , \quad \rho_i \cap \rho_j = \emptyset \text{ für } 1 \leq i < j \leq p$$

Für die Einschränkungen ρ_i von ρ gilt dann, daß es zu jedem Paar $(\omega,a) \in \Omega_i \times A_i$ einen Wert $v \in V$ gibt, so daß $(\omega,a,v) \in \rho_i$ ist.

Wir unterscheiden zwischen "einfachen" und "komplexen" Datenstrukturen. Auf der Grundlage der obigen Überlegungen definieren wir folgendes:

Definition 2.3.: Eine Datenstruktur $D=(\Omega,A,V,\rho)$ heißt einfach, wenn die Teilrelationen ρ_i für $1 \leq i \leq p$ Funktionen der Form

$$\rho_i : \Omega_i \times A_i \to V$$

sind.

Einfache Datenstrukturen sind demnach diejenigen, in denen zu jedem Attribut jedes Objektes höchstens ein Wert gehört. Der Zweck der obigen Überlegungen wird im nächsten Abschnitt deutlich, in dem die Beziehungen zum Relationen-Modell von Codd [1] näher untersucht werden. Damit wird an einem Beispiel gezeigt, wie andere Darstellungen von Datenstrukturen aus dem Attribut-Modell formal hergeleitet werden können.

3. Zusammenhang mit dem Relationen-Modell

Zur Unterscheidung in diesem Abschnitt nennen wir die Datenstrukturen des Attribut-Modells "A-Strukturen" und diejenigen des Relationen-Modells "R-Strukturen", und wir präzisieren Codd's Modell folgendermaßen:

Definition 3.1.: Eine R-Struktur ist ein Paar $D^*=(\mathcal{W},\mathcal{R})$, wobei gilt:

1. $\mathcal{W} = \{V_1,\ldots,V_n\}$ ist eine endliche Menge von Wertebereichen (attribute sets in [1]).
2. $\mathcal{R} = \{R_1,\ldots,R_p\}$ ist eine endliche Menge von Relationen,

   $R_i \subset V_{i1} \times V_{i2} \times \cdots \times V_{ik_i}$ für $i=1,\ldots,p$ und $V_{ij} \in \mathcal{W}$ für $j=1,\ldots,k_i$
3. Die folgende Bedingung ist immer erfüllt:

$$\bigcup_{i=1}^{n} V_i = \bigcup_{i=1}^{p} \bigcup_{j=1}^{k_i} pr_j(R_i)$$

Die letztere Bedingung besagt, daß es in einer R-Struktur keine überflüssigen Werte gibt, die in keiner Relation vorkommen.

Zunächst fällt auf, daß sich A-Strukturen als spezielle R-Strukturen auffassen lassen:

$$D=(\Omega,A,V,\rho) \quad \hat{=} \quad D^*=(\{\Omega,A,V\} \ , \ \{\rho\})$$

Umgekehrt läßt sich aber auch jede R-Struktur unter bestimmten, kaum einschränkenden Bedingungen durch eine spezielle, nämlich einfache

A-Struktur darstellen. Um dies zu zeigen, führen wir zwei Transformationen

$$\phi : \{\text{R-Strukturen}\} \to \{\text{einf. A-Strukt.}\}$$
$$\psi : \{\text{einf. A-Strukt.}\} \to \{\text{R-Strukturen}\}$$

ein:

I. Sei $D^* = (\mathcal{D}, \mathcal{R})$ gegeben. Dann definieren wir

$$\phi(D^*) = D = (\Omega, A, V, \rho)$$

folgendermaßen:

$$\Omega := \{ (r, R) \mid r \varepsilon R \varepsilon \mathcal{R} \}$$

Jedem Wertebereich $V_i \varepsilon \mathcal{D}$ ordnen wir ein Attribut $a(V_i)$ zu:

$$A := \{a(V_1), \ldots, a(V_n)\}$$

Für die Wertemenge V setzen wir

$$V := \bigcup_{i=1}^{n} V_i$$

Die Tripelmenge ρ definieren wir folgendermaßen: ist $\omega=(r,R_i)$ und gehört zur j-ten Komponente der Relation R_i der Wertebereich V_{ij} sowie zur j-ten Komponente des k_i-Tupels $r \varepsilon R_i$ der Wert $v \varepsilon V_{ij}$, genau dann ist

$$(\omega, a(V_{ij}), v) \varepsilon \rho$$

Da in jeder Komponente jeder Relation der R-Struktur D^* nur ein Wert steht, ist $\phi(D^*)$ offenbar eine einfache A-Struktur.

II. Sei $D=(\Omega, A, V, \rho)$ eine einfache A-Struktur. Dann definieren wir

$$\psi(D) = D^* = (\mathcal{D}, \mathcal{R})$$

folgendermaßen: jedem Attribut $a_j \varepsilon A$, $j=1,\ldots,n$, ordnen wir den Wertebereich

$$V(a_j) := \{ v \varepsilon V \mid \exists \omega \varepsilon \Omega \; ; \; (\omega, a_j, v) \varepsilon \rho \}$$

zu. Wir setzen

$$\mathcal{D} := \{V(a_1), \ldots, V(a_n)\}$$

Sei $K=\{\Omega_1, \ldots, \Omega_p\}$ die im vorigen Abschnitt auf Ω eingeführte Klasseneinteilung, und sei

$$A_i = \{a_{i1}, \ldots, a_{ik_i}\} \quad , \quad i=1,\ldots,p \quad ,$$

die zu Ω_i gehörige Attributmenge (s.o.). Sei ferner

$$\rho_i(\omega, a_{ih}) = v_{ih}$$

für alle $\omega \varepsilon \Omega_i$, $i=1,\ldots,p$ und $h=1,\ldots,k_i$. Dann setzen wir für $i=1,..,p$

$$R_i := \{(v_{i1}, \ldots, v_{ik_i}) \mid \exists \omega \varepsilon \Omega_i \; \forall h=1,\ldots,k_i : \rho(\omega, a_{ih}) = v_{ih}\}$$

Es ist $R_i \subset V(a_{i1}) \times \cdots \times V(a_{ik_i})$, und wir setzen

$$\mathcal{R} := \{R_1, \ldots, R_p\}$$

Dann ist $\psi(D) = D^* = (\mathcal{W}, \mathcal{R})$ eine R-Struktur.

Diese Transformationen ϕ und ψ hängen auf folgende Weise zusammen:

Ist D^* eine R-Struktur, in der zwei verschiedene Relationen niemals für alle entsprechenden Komponenten gleiche Wertebereiche haben, so ist

$$\psi(\phi(D^*)) \simeq D^*$$

(Die Isomorphie $\simeq$ bedeutet wie üblich die Gleichheit bis auf eineindeutige Bezeichnungsänderungen.)

Ist umgekehrt D eine einfache A-Struktur, in der verschiedene Objekte immer verschiedene Sätze haben, so ist

$$\phi(\psi(D)) \simeq D$$

Beispiel 2 : Sei D^* gegeben durch folgende Relationen:

PERS	Name	Vater	Beruf
	Karl	Kurt	Lehrer
	Emil	Kurt	Arzt

TEIL	Teilnr.	verw.in	Anz.
	1oo746	XQR2	5

D sei folgende A-Struktur:

(X , Name , Karl)	(Y , Beruf , Arzt)
(X , Vater , Kurt)	(T , Teilnr. , 1oo746)
(X , Beruf , Lehrer)	(T , verw.in , XQR2)
(Y , Name , Emil)	(T , Anz. , 5)
(Y , Vater , Kurt)	

Dann ist $\phi(D^*)=D$ und $\psi(D)=D^*$.

4. Q-Systeme

Retrieval-Operationen sind in gewissem Sinne grundlegend für eine ganze Reihe weiterer Operationen auf Datenstrukturen. Die Formalisierung von Retrieval-Operationen in diesem Abschnitt geht mehr von der Sicht des Implementierers aus als von der des Benutzers. Es besteht nicht die Absicht, hiermit eine Grundlage zu schaffen für die Konstruktion einer benutzerorientierten Anfragesprache. Vielmehr wird angestrebt, ein allgemeines Konzept für die Darstellung von Speicher- und Zugriffsorganisationen zu entwickeln. Komplexe Benutzeranfragen werden in der Regel in eine Reihe von Einzelzugriffen aufgelöst, und es sind diese Einzelzugriffe, welche hier untersucht werden sollen.

Sei $D=(\Omega,A,V,\rho)$ eine Datenstruktur.

Definition 4.1.: Eine Frage ist eine zweistellige Relation $q \subset A\times V$. Die Antwort auf q in D ist

$$\alpha_D(q) := \{\ \omega\in\Omega \mid q \subset R_\omega\ \}$$

Die Länge von q ist $|q|$, und Fragen der Länge 1 heißen Elementarfragen.

Fragen sind also Kombinationen von Eigenschaften, und die Antwort besteht aus allen Objekten, welche alle geforderten Eigenschaften haben. Fragen sind Mengen und lassen sich als solche durch Mengenoperationen verknüpfen. Für die Vereinigung und den Durchschnitt von Fragen gelten folgende Gestze:

Lemma 4.1.: Sei $D=(\Omega,A,V,\rho)$ eine Datenstruktur. Für alle Fragen $q,q'\subset A\times V$ gilt:

$$1.\ \alpha_D(q\cap q') \supset \alpha_D(q)\cup\alpha_D(q')$$
$$2.\ \alpha_D(q\cup q') = \alpha_D(q)\cap\alpha_D(q')$$

Beweis: 1. $\omega\in\alpha_D(q)\cup\alpha_D(q') \Rightarrow q\subset R_\omega \vee q'\subset R_\omega \Rightarrow q\cap q'\subset R_\omega \Rightarrow \omega\in\alpha_D(q\cap q')$

2. $\omega\in\alpha_D(q\cup q') \Leftrightarrow q\cup q'\subset R_\omega \Leftrightarrow q\subset R_\omega \wedge q'\subset R_\omega \Leftrightarrow \omega\in\alpha_D(q)\cap\alpha_D(q')$

Aufgrund der zweiten Beziehung läßt sich die logische Konjunktion von Fragen als Vereinigung der Fragen darstellen.

Im folgenden sollen kombinierte Systeme, bestehend aus Datenstrukturen und zugehörigen Mengen von "Standardfragen", im Zusammenhang betrachtet werden.

Definition 4.2.: Ein Q-System ist ein Paar $\mathcal{Q}=(D,Q)$, wobei $D=(\Omega,A,V,\rho)$ eine Datenstruktur und $Q\subset\mathcal{P}(A\times V)$ eine Menge von Standardfragen ist.

Für Probleme der Implementierung von Q-Systemen sind die folgenden speziellen Eigenschaften von Bedeutung:

Definition 4.3.: Ein Q-System $\mathcal{Q}=(D,Q)$ heißt

1. Tabelle $:\Leftrightarrow \forall q,q'\in Q : pr_1(q) = pr_1(q')$
2. elementar $:\Leftrightarrow \forall q\in Q : |q| = 1$
3. boolesch $:\Leftrightarrow V\subset\{0,1\}\wedge pr_2(Q)\subset\{1\}$

Hierbei ist $pr_2(Q) := \bigcup_{q\in Q} pr_2(q)$.

Elementare Tabellen sind unmittelbar der Implementierung durch herkömmliche und bekannte Methoden zugänglich, wie z.B. sequentielle und index-sequentielle Organisation durch Ordnung der Sätze nach den Werten

des Schlüsselattributs, auf das sich alle Fragen beziehen. Ferner kommen hier Streuspeicherung (hash coding), Suchbäume etc. in Frage. Ein Speicher- und Zugriffssystem mit invertierten Files für einige Attribute läßt sich als elementares Q-System auffassen. Es gibt fortgeschrittenere Organisationsmethoden, wie sie z.B. von Ray-Chaudhury [4] und anderen dort zitierten Autoren beschrieben wurden, die sich als boolesche Q-Systeme interpretieren lassen. Die Methode der kombinierten Indizes von Lum [3] gibt Beispiele für Q-Systeme, welche weder Tabellen noch elementar noch boolesch sind.

Wir wollen untersuchen, ob und ggf. wie sich Q-Systeme umformen lassen, um sie u.U. in eine für die Implementierung bessere Form bringen zu können. Für die Untersuchung solcher Umformungen benötigen wir einen plausiblen Äquivalenzbegriff.

Definition 4.4.: Zwei Q-Systeme $\mathcal{Q}_1=(D_1,Q_1)$ und $\mathcal{Q}_2=(D_2,Q_2)$ heißen äquivalent, in Zeichen $\mathcal{Q}_1 \sim \mathcal{Q}_2$, genau dann, wenn folgendes gilt ($\alpha_i := \alpha_{D_i}$, $i=1,2$) :

1. $\alpha_1(Q_1) \cup \alpha_2(Q_2) \subset \Omega_1 \cap \Omega_2$
2. $\exists \kappa: Q_1 \twoheadrightarrow Q_2 \; \forall q_1 \in Q_1 \; : \; \alpha_1(q_1) = \alpha_2(\kappa(q_1))$

Hierbei ist $\alpha(Q) := \bigcup_{q \in Q} \alpha(q)$.

Grob gesagt sind Q-Systeme äquivalent, wenn auf gleiche Fragen gleiche Antworten gegeben werden. Auf der Grundlage dieses Äquivalenzbegriffs erhalten wir folgende Ergebnisse:

Satz 4.1.: Zu jeder Tabelle $\mathcal{T}=(D,Q)$ gibt es eine äquivalente Tabelle $\mathcal{T}^*=(D^*,Q^*)$, deren Attributmenge höchstens ein Attribut enthält (welche also insbesondere elementar ist) .

Beweis: Sei $D=(\Omega,A,V,\rho)$ und $A=\{a_1,\ldots,a_n\}$. Ist $Q=\emptyset$, so ist die leere Tabelle $\mathcal{T}^*=((\emptyset,\emptyset,\emptyset,\emptyset),\emptyset)$ äquivalent zu $\mathcal{T}$. Ist $Q=\{\emptyset\}$, so ist $\mathcal{T}^*=((\Omega,\emptyset,\emptyset,\emptyset),\{\emptyset\})$ äquivalent zu $\mathcal{T}$. Q enthalte nun mindestens eine nichtleere Frage, und o.B.d.A. sei $pr_1(q)=\{a_1,\ldots,a_p\}$, $1 \leq p \leq n$, für alle $q \in Q$. Dann setzen wir $a^* := (a_1,\ldots,a_p)$ und ordnen jeder Frage $q \in Q$, $q=\{(a_1,v_1),\ldots,(a_p,v_p)\}$, das Paar $\kappa(q) := \{(a^*,v^*)\}$ zu, wobei $v^* := (v_1,\ldots,v_p) \in V^p$ ist. Sei

$$Q^* := \{ \kappa(q) \mid q \in Q \}$$
$$\rho^* := \{ (\omega,a^*,v^*) \mid \forall i=1,\ldots,p : (\omega,a_i,v_i) \in \rho \}$$
$$\Omega^* := pr_1(\rho^*)$$
$$V^* := pr_3(\rho^*)$$

Dann ist die Tabelle $\mathcal{T}^*=(D^*,Q^*)$ mit $D^*=(\Omega^*,\{a^*\},V^*,\rho^*)$ äquivalent zu $\mathcal{T}$. Denn mit $\alpha:=\alpha_D$ und $\alpha^*:=\alpha_{D^*}$ gilt:

$$\begin{aligned}\alpha(Q) &= \{\ \omega\in\Omega \mid \exists q\in Q : q\subset R_\omega\ \}\\ &\subset \{\ \omega\in\Omega \mid \forall i=1,\ldots,p\ \exists v_i\in V\ :\ (\omega,a_1,v_i)\in\rho\ \}\\ &= \Omega^*\end{aligned}$$

Offenbar ist $\alpha^*(Q^*)\subset\Omega^*$ und $\Omega^*\subset\Omega$, so daß die erste Bedingung für die Äquivalenz erfüllt ist.

Für alle $q\in Q$, $q=\{(a_1,v_1),\ldots,(a_p,v_p)\}$ ist $\kappa(q)=\{(a_1,\ldots,a_p),(v_1,\ldots,v_p)\}$, und es gilt:

$$\begin{aligned}\alpha(q) &= \{\ \omega \mid q\subset R_\omega\ \} = \{\ \omega \mid \forall i=1,\ldots,p : (\omega,a_1,v_1)\in\rho\ \}\\ &= \{\ \omega \mid (\omega,(a_1,\ldots,a_p),(v_1,\ldots,v_p))\in\rho^*\ \}\\ &= \{\ \omega \mid \kappa(q)\subset R^*_\omega\ \}\\ &= \alpha^*(\kappa(q))\end{aligned}$$

Hierbei ist R^*_ω der Satz von ω in D^*. Daß κ eine bijektive Abbildung ist, ist klar, so daß auch die zweite Bedingung für die Äquivalenz erfüllt ist.

Satz 4.2.: Zu jedem Q-System $\mathcal{Q}=(D,Q)$ gibt es ein äquivalentes Q-System $\mathcal{Q}^*=(D^*,Q^*)$, welches elementar und boolesch ist, und dessen Datenstruktur D^* einfach ist.

Beweis: Sei $\alpha:=\alpha_D$ und $\alpha^*:=\alpha_{D^*}$. Wir setzen $\Omega^*:=\alpha(Q)$, $A^*:=Q$ und $V^*:=\mathrm{pr}_3(\rho^*)$, wobei gilt:

$$\begin{aligned}\rho^* := &\{\ (\omega,q,1) \mid q\in Q,\ \omega\in\alpha(q)\ \}\\ &\{\ (\omega,q,0) \mid q\in Q,\ \omega\in\Omega^*-\alpha(q)\ \}\end{aligned}$$

Offenbar ist ρ^* eine Funktion von $\Omega^*\times A^*$ in V^*, d.h. die Datenstruktur $D^*=(\Omega^*,A^*,V^*,\rho^*)$ ist einfach.

Wir setzen nun $Q^*:=\{\{(q,1)\} \mid q\in Q\}$ und definieren die bijektive Abbildung $\kappa : Q\rightarrow\!\!\!\!\rightarrow Q^*$ durch $\kappa(q):=\{(q,1)\}$. Setzen wir nun $\mathcal{Q}^*:=(D^*,Q^*)$, so ist $\mathcal{Q}^*$ elementar und boolesch, und es ist $\mathcal{Q}\sim\mathcal{Q}^*$. Denn einerseits ist $\alpha(Q)=\Omega^*=\alpha^*(Q^*)$ und $\Omega^*\subset\Omega$, und andererseits gilt die folgende Äquivalenz:

$$\omega\in\alpha(q) \Leftrightarrow (\omega,q,1)\in\rho^* \Leftrightarrow \omega\in\alpha^*(\kappa(q))\ .$$

Literatur

1. Codd,E.F.: A relational model of data for large shared data banks. Comm. ACM 13 (1970), 377-387.
2. Feldman,J.A.-Rovner,P.D.: An ALGOL-based associative language. Comm. ACM 12 (1969), 439-449.

3. Lum,V.Y.: Multi-attribute retrieval with combined indexes. Comm. ACM 13 (1970), 660-665.

4. Ray-Chaudhury,D.K.: Combinatorial information retrieval systems for files. SIAM J. Appl. Math. 16 (1968), 973-992.

5. Rovner,P.D.-Feldman,J.A.: The LEAP language and data structure. Proc. IFIP 68, A.J.H.Morrell (ed), North-Holland, Amsterdam 1969 , pp. 579-585.

6. Salton,G.: Automatic information organization and retrieval. McGraw-Hill, New York 1968.