

2. Gleichungslogik und Termersetzung

Themen dieses Kapitels:

- Gleichungslogik: Syntax und Semantik
- Gleichungstheorien
- Beziehungen zu abstrakten algebraischen Datenstrukturen
- Grundlagen der Termersetzung:
 - Substitution, Unifikation
- Termersetzungsregeln und Termersetzungs-systeme (TES)
 - Normalformen, Konfluenz und Church-Rosser-Eigenschaft
 - Terminierung
 - Vervollständigung von TES
- TES und Beweisen in Gleichungstheorien
- TES und Funktionale Programmierung

Ergänzende Literatur

J. Avenhaus, *Reduktionssysteme*. Springer Lehrbuch, 1995
für Termersetzung

Ehrich, Gogolla, Lipeck, *Algebraische Spezifikation abstrakter Datentypen*, Teubner, Stuttgart 1989. [eher elementares Lehrbuch]

Wirsing: *Algebraic Specification*. Chapter 13 of *Handbook of Theoretical Computer Science*, vol. B: Formal Models and Semantics. Elsevier, Amsterdam 1990.

Umgehen mit Gleichungen – informell

“Leibniz-Gleichheit”: zwei Dinge sind gleich, wenn sie bezüglich aller ihrer Eigenschaften gleich sind.

Formal: $a = b$ falls $C[a] = C[b]$ für jeden “Kontext” C .

In anderen Worten: *In jedem Kontext kann Gleiches durch Gleiches ersetzt werden.*

Gleichungslogik/Gleichungskalküle: Formalisierung des Schließens mit Gleichungen – Ableiten von Gleichungen aus Gleichungen.

Übliches Vorgehen: Lösung von Gleichungsproblemen durch sukzessives Ersetzen von “Gleichem durch Gleiches”.

Beispiel: Lösung von Systemen linearer Gleichungen

Viele *algebraische Theorien* lassen sich mit Hilfe von Gleichungen axiomatisieren.

Beispiel: Gruppentheorie

e : Konstante

i : einstellige Operation

$_ \cdot _$: zweistellige Operation

Axiome:

$$x \cdot (y \cdot z) = (x \cdot y) \cdot z \quad (1)$$

$$e \cdot x = x \quad (2)$$

$$i(x) \cdot x = e \quad (3)$$

Assoziativität von \cdot

e Linksneutrales Element

Linksinverses Element

Beispiel (informell) für eine Ableitung der Gleichung

$$x \cdot e = x \quad (\text{Linksneutrales Element ist auch rechtsneutral})$$

Beweisidee: eine Seite der Gleichung so lange mit Hilfe von Instanzen der Axiome umformen, bis die andere Seite der Gleichung erreicht ist.

$$\begin{aligned} x \cdot e &= (e \cdot x) \cdot e \\ &= ((i(i(x)) \cdot i(x)) \cdot x) \cdot e \\ &= (i(i(x)) \cdot (i(x) \cdot x)) \cdot e \\ &= (i(i(x)) \cdot e) \cdot e \\ &= i(i(x)) \cdot (e \cdot e) \\ &= i(i(x)) \cdot e \\ &= i(i(x)) \cdot (i(x) \cdot x) \\ &= (i(i(x)) \cdot i(x)) \cdot x \\ &= e \cdot x \\ &= x \end{aligned}$$

Beobachtungen:

Die Gleichungen der Axiome werden in beiden Richtungen benutzt.

Welche Umformung am besten als nächste anzuwenden ist, ist für den Menschen häufig offensichtlich,

aber ein solcher Beweis erfordert gewisse Einsichten ('Tricks') – oder viel unnötiges 'Ausprobieren'.

Problem: Wie können solche Beweise mechanisiert werden?

formales System: \rightsquigarrow Gleichungslogik

Beweismechanismus: \rightsquigarrow Termersetzungssystem

Gleichungslogik: Syntax

vorläufige Form der Syntax:

Vokabular:

- *Signatur*: Eine Menge Σ bestehend aus Konstantensymbolen c_i und Funktionssymbolen f_j ; im allgemeinen wird Σ als endlich angenommen.

Jedes Funktionssymbol hat eine *Stelligkeit (arity)*, die die Anzahl der Argumente angibt; Konstanten können als nullstellige Funktionssymbole aufgefaßt werden.

- Eine Menge V von Variablen x_k ; im allgemeinen wird V als abzählbar unendlich angenommen.

Gleichungslogik: Syntax (2)

Terme (induktiv definiert):

1. Jede Variable x_k und jede Konstante c_i ist ein Term.
2. Sind t_1, \dots, t_n Terme und f_j ein n -stelliges Funktionssymbol, so ist auch $f_j(t_1, \dots, t_n)$ ein Term.
3. Alle Terme werden auf diese Weise gebildet.

$T_\Sigma(V)$: Menge der (wohlgeformten) Terme über vorgegebenen Mengen Σ und V (' Σ -Terme')

Im folgenden wird $T_\Sigma(V)$ meistens zu T abgekürzt, wenn Σ und V aus dem Kontext klar sind.

Terme werden häufig als *Bäume* dargestellt.

Grundterm: ein Term, der keine Variablen enthält, d.h. ein Term

$$t \in T_\Sigma(\emptyset)$$

Gleichungslogik: Gleichheitsaxiome

Gleichungen: die atomaren Formeln der Gleichungslogik

$$t_1 = t_2$$

wobei t_1, t_2 Terme sind.

Grundgleichung: Gleichung von Grundtermen

Die Gleichheitsrelation ist eine *Äquivalenzrelation*, d.h. ist

reflexiv: $x = x$

symmetrisch: $x = y \Rightarrow y = x$

transitiv: $x = y \wedge y = z \Rightarrow x = z$

außerdem gilt *Substitutivität*, d.h. die Gleichheitsrelation ist eine *Kongruenzrelation* bzgl. der termbildenden Funktionssymbole aus Σ (Σ -Kongruenz):

Für jedes n -stellige Funktionssymbol f und alle Terme x_i und y_i ($1 \leq i \leq n$) gilt:

Aus $x_i = y_i$ für alle $1 \leq i \leq n$ folgt

$$f(x_1, \dots, x_n) = f(y_1, \dots, y_n).$$

Gleichungslogik: mehrsortige Syntax

Allgemeinere Form der Syntax: Termen werden unterschiedliche *Sorten* zugeordnet analog den Typen für Ausdrücke in (streng) getypten Programmiersprachen (z.B. Pascal, Modula-2).

Signatur (Vokabular):

- \mathcal{S} eine (endliche, oder höchstens abzählbare) Menge von *Sortensymbolen* S_i ;
 \mathcal{S} soll ein Symbol *Bool* für die Sorte der *Wahrheitswerte* enthalten. *Signatur* über \mathcal{S} : \mathcal{S} -typisierte Funktionen,

- Eine Menge Σ von Konstantensymbolen c_i und Funktionssymbolen f_j ;
jedem der Symbole ist eine sortierte *Stelligkeit* ('*Signatur*') zugeordnet:

$c : S_i$ für eine Konstante

$f^{(n)} : S_1 \times \dots \times S_n \rightarrow S_{n+1}$ für eine n-stellige Funktion

- Eine (höchstens abzählbar unendliche) Menge V von sortierten Variablen:

$v_i \in V_S$ Variable vom Typ S , für jedes $S \in \mathcal{S}$

(genauer: V ist eine \mathcal{S} -indizierte Familie $\{V_S\}_{S \in \mathcal{S}}$ von (disjunkten) Variablenmengen)

Gleichungslogik: mehrsortige Syntax (2)

Terme: wie vorher induktiv definiert, aber das Bilden von Termen muß sorten-korrekt sein

Notation: $t : S_i$ soll bedeuten 'der Term t hat die Sorte S_i '

1. Jede Variable $x_k : S_i$ und jede Konstante $c_i : S_i$ ist ein Term der Sorte S_i .
2. Für Terme $t_i : S_i$ ($i \in \{1, \dots, n\}$) und ein n -stelliges Funktionssymbol $f : S_1 \times \dots \times S_n \rightarrow S_{n+1}$ ist $f(t_1, \dots, t_n)$ ein Term der Sorte S_{n+1} .
3. Alle Terme werden auf diese Weise gebildet.

wie vorher:

$T_{\Sigma}^{(S)}(V)$ Menge der (wohlgeformten) Terme (der Sorte S) über einer vorgegebenen Signatur Σ und Variablenmenge V

Gleichungslogik: mehrsortige Syntax (3)

Gleichungen:

$$t_1 = t_2$$

wobei t_1, t_2 Terme *derselben Sorte* sind.

Warnung: Die Terminologie ist nicht einheitlich. Man spricht von “mehrsortiger Logik” (*many-sorted logic*) oder “getypter Logik”; wir werden hier häufig “Sorte” und “Typ” als Synonyme benutzen.

Gleichungslogik: Semantik

Eine Bedeutung von Termen und Gleichungen wird durch deren *Interpretation* \mathcal{I} in einer \mathcal{S} -sortierten Struktur angegeben: Die Interpretation \mathcal{I} ordnet zu

- jedem Sortensymbol $S \in \mathcal{S}$ eine *nichtleere* Menge M_S (“Trägermenge”, *carrier*, *domain* von S);
über die Beziehungen zwischen den M_S , insbesondere Disjunktheit, werden i.a. keine weiteren Annahmen gemacht.
- der Sorte *Bool* die Menge $M_{Bool} := \{\mathbf{W}, \mathbf{F}\}$
- jedem Konstanten-Symbol $c : S$ eine Konstante $\mathcal{I}[c] \in M_S$
- jedem Funktionssymbol $f : S_1 \times \dots \times S_n \rightarrow S_{n+1}$ eine Funktion
$$\mathcal{I}[f] : M_{S_1} \times \dots \times M_{S_n} \rightarrow M_{S_{n+1}}$$

Gleichungslogik: Semantik (2)

\mathcal{I} wird auf Grundterme und Grund-Gleichungen fortgesetzt:

- $\mathcal{I}[f(t_1, \dots, t_n)] := \mathcal{I}[f](\mathcal{I}[t_1], \dots, \mathcal{I}[t_n])$

- Das Gleichheitssymbol wird als Gleichheitsrelation auf der entsprechenden Trägermenge interpretiert:

$$\mathcal{I}[t_1 = t_2] := (\mathcal{I}[t_1] = \mathcal{I}[t_2])$$

d.h. die Interpretation der Gleichung ist der Wahrheitswert **W** genau dann, wenn t_1 und t_2 unter der Interpretation \mathcal{I} denselben Wert in der Trägermenge bezeichnen.

Eine *Variablenbelegung* (Variablenzuweisung) \mathcal{V} ordnet jeder Variablen $x : S$ einen Wert aus M_S zu. Eine Interpretation \mathcal{I} zusammen mit einer Variablenbelegung \mathcal{V} erlaubt es, auch Nicht-Grundtermen Werte zuzuordnen, d.h. sie zu interpretieren.

Gleichungstheorien

Eine Gleichungstheorie über einer Signatur wird definiert durch die Angabe von Gleichungen als weitere Axiome (zusätzlich zu den Gleichungsaxiomen).

Beispiel: Gruppentheorie als Gleichungstheorie, definiert durch die o.a. Signatur und Gleichungen.

Semantik:

Eine Interpretation ergibt ein *Modell* einer Gleichungstheorie, wenn die Gleichungsaxiome für alle möglichen Variablenbelegungen unter der Interpretation den Wert **W** zugeordnet bekommen, d.h. für alle Variablenwerte in den entsprechenden Trägermengen 'wahr sind'.

Viele algebraische Strukturen lassen sich wie die Gruppentheorie als Gleichungstheorien formalisieren.

↪ universelle Algebra

↪ Zusammenhang mit *algebraischen Spezifikationen* (wird später wieder aufgegriffen)

Substitution

- Eine *Substitution* ist eine Abbildung von einer endlichen Menge von Variablen in die Menge der Terme, erweitert auf die Menge aller Variablen durch die Identität.
- Eine *Grundsubstitution* bildet die Variablen ab auf Grundterme.

Eine Substitution wird häufig dargestellt als Menge (oder Folge) von Paaren,

$$\sigma = \{x \leftarrow t \mid \sigma(x) = t\}$$

Eine Substitution wird fortgesetzt auf Terme durch rekursiven Abstieg: z.B.

$$\sigma(f(x, y, z)) = f(\sigma(x), \sigma(y), \sigma(z))$$

d.h. eine Substitution wird allgemein angesehen als eine Abbildung von $T_{\Sigma}(V)$ nach $T_{\Sigma}(V)$.

Substitution (2)

Komposition von Substitutionen: wie normale Komposition von Abbildungen, d.h.

$$(\tau \cdot \sigma)(t) = \tau(\sigma(t))$$

Im mehrsortigen Kontext muß eine Substitution sortenkorrekt sein, d.h. die Sorten der Variablen und der ihnen zugeordneten Termen müssen übereinstimmen.

Unifikation

Ein *Unifikator* von zwei Termen s und t ist eine Substitution σ , die die beiden Terme “gleich macht”: $\sigma(s) \doteq \sigma(t)$.

Das Symbol \doteq bezeichnet hier *syntaktische Identität*: $t_1 \doteq t_2$ bedeutet, daß die Terme t_1 und t_2 syntaktisch nicht unterscheidbar sind.

Eine Substitution σ heißt *allgemeinster Unifikator* (*most general unifier*, MGU) von zwei Termen s und t , wenn gilt

- σ ist ein Unifikator von s und t .
- Für jeden Unifikator τ von s und t existiert eine Substitution σ' mit $\tau = \sigma' \cdot \sigma$.
d.h. τ läßt sich in gewisser Weise aus σ gewinnen.

Unifikation (2)

Unifikations-Algorithmus:

Zu Beginn ist $\sigma = \{\}$.

t_1, t_2 seien die zu unifizierenden Terme.

1. Wende σ auf t_1 und t_2 an.

2. Fallunterscheidung:

(a) $t_1 \doteq t_2$: Resultat ist σ .

(b) $t_1 \doteq v$, d.h. t_1 ist eine Variable: Falls v in t_2 vorkommt, ist das Resultat “nicht unifizierbar”; andernfalls setze $\sigma' := \{v \leftarrow t_2\}$.

(c) $t_2 \doteq v$, t_1 ist nicht eine Variable: Falls v in t_1 vorkommt, Resultat “nicht unifizierbar”; andernfalls setze $\sigma' := \{v \leftarrow t_1\}$.

(d) Weder t_1 noch t_2 ist eine Variable:

Falls die führenden Funktionssymbole in t_1 und t_2 übereinstimmen, berechne den Unifikator für die Argument-Terme s_1, \dots, s_n und r_1, \dots, r_n ; andernfalls ist das Resultat “nicht unifizierbar”.

3. Setze $\sigma := \sigma' \cdot \sigma$.

Unifikation (3)

Unifikation von Term-Listen s_1, \dots, s_n und r_1, \dots, r_n mit Anfangssubstitution σ :

Für $i = 1, \dots, n$ wiederhole:

1. Berechne Unifikator σ_1 für s_i und r_i .
2. Falls erfolgreich:
setze Schleife fort mit neuem σ für s_{i+1}, r_{i+1} fort.
3. Andernfalls Resultat "nicht unifizierbar".

Unifikation: Eigenschaften

Satz (Robinson): Wenn zwei Terme (bzw. atomare Formeln) unifizierbar sind, dann besitzen sie auch einen allgemeinsten Unifikator.

- Der Unifikationsalgorithmus konstruiert einen allgemeinsten Unifikator für zwei Terme, sofern es einen solchen gibt.
- Der Unifikationsalgorithmus terminiert immer: bei jedem Schritt wird eine Variable eliminiert.
- Wenn er erfolgreich terminiert, ist das Ergebnis ein allgemeinsten Unifikator.
- Der Unifikationsalgorithmus liefert ein *Entscheidungsverfahren* dafür, ob zwei Terme (bzw. atomare Formeln) unifizierbar sind.
- Der allgemeinste Unifikator (MGU) ist bis auf Variablenumbenennung eindeutig (genau genommen ist “der” MGU eine Äquivalenzklasse von Unifikatoren).

Unifikation: Eigenschaften (2)

- Bedeutung des “Occur-Checks” (v kommt nicht in t_i vor)
- Es gibt verschiedene Verallgemeinerungen der Unifikation, die z.B. Eigenschaften von Funktionen (wie Kommutativität und Assoziativität) mitberücksichtigen; letztere können das Unifikationsproblem entscheidend verändern.
- Unifikation spielt in vielen Gebieten der KI (und allgemein der Informatik) eine große Rolle; z.B. in anderen Beweisverfahren (Resolution - wird später behandelt), in prolog-artigen Programmiersprachen, bei der Analyse auf Typkorrektheit in bestimmten Klassen von Programmiersprachen, bei der Verarbeitung natürlicher Texten (“Unifikationsgrammatiken”).

Termersetzungssysteme

Ein *Termersetzungssystem* (TES) ist eine Relation

$$R \subset T_{\Sigma}(V) \times T_{\Sigma}(V)$$

$(l, r) \in R$ schreiben wir im allgemeinen als Regel: $l \rightarrow r$;

l und r sind die *linke* bzw. *rechte Seite* der Regel.

Ableitung mit einer Termersatzungsregel:

Seien $(l \rightarrow r) \in R$, $t \in T$, s ein Teilterm von t .

Gibt es eine Substitution σ , so daß $s \doteq \sigma(l)$, so kann s in t durch $\sigma(r)$ ersetzt werden:

$$t \rightarrow_R t' \text{ mit } t' \doteq t[s \leftarrow \sigma(r)]$$

“ t wird *reduziert* oder *abgeleitet* zu t' (in einem Schritt)

Termersetzungssysteme (2)

Mit Hilfe der Relation \rightarrow_R werden Termersetzungssysteme definiert:

\rightarrow_R^* : reflexiver und transitiver Abschluß von \rightarrow_R

d.h. $t_1 \rightarrow_R^* t_2$ gilt gdw. entweder $t_1 \doteq t_2$ oder es gibt Terme s_1, \dots, s_n ($n \geq 0$), so daß

$$t_1 \rightarrow_R s_1 \rightarrow_R \dots \rightarrow_R s_n \rightarrow_R t_2$$

\leftrightarrow_R^* : reflexiver, symmetrischer und transitiver Abschluß von \rightarrow_R

Im allgemeinen wird der Index R fortgelassen, da er aus dem Kontext klar ist.

Gleichungen und TES

Gleichungsmengen und Termersetzungssysteme können ineinander überführt werden:

- Aus dem Termersetzungssystem R entsteht ein Gleichungssystem E_R , indem jede Regel $l \rightarrow r$ durch die Gleichung $l = r$ ersetzt wird.
- Aus einer Gleichungsmenge E entsteht ein Termersetzungssystem R_E dadurch, daß jede Gleichung $s = t$ durch eine Regel $s \rightarrow t$ ersetzt wird.

Termersetzungssystem als Operationalisierung eines Gleichungskalkül:

Ausgangsgleichungen werden orientiert und als *Regeln* aufgefaßt, mit deren Hilfe Terme äquivalent umgeformt werden.

Die aus dem TES R_E abgeleitete Relation $\leftrightarrow_{R_E}^*$ ist identisch mit der durch E induzierten Kongruenzrelation \equiv_E auf Termen.

Eigenschaften von TES

Ein Termersetzungssystem R heißt *terminierend* (Noethersch), wenn es keine unendlichen Ableitungsfolgen bzgl. der Relation \rightarrow_R gibt.

Terminierung von Termersetzungssystemen ist unentscheidbar.

R heißt *Church-Rosser* (hat die Church-Rosser-Eigenschaft), wenn für alle Terme t_1 und t_2 mit $t_1 \leftrightarrow^* t_2$ ein Term t mit $t_1 \rightarrow^* t$ und $t_2 \rightarrow^* t$ existiert.

Ein Term t heißt *irreduzibel* (bezgl. R), wenn es kein t' mit $t \rightarrow_R t'$ gibt.

Ein Term t' heißt *Normalform* von t , falls $t \rightarrow^* t'$ gilt und t' irreduzibel ist.

Notation: $t \downarrow$

Eigenschaften von TES (2)

Satz: Eindeutigkeit von Normalformen

Für ein TES R mit der Church-Rosser-Eigenschaft gibt es für jedes t höchstens eine Normalform.

Satz: Existenz und Eindeutigkeit von Normalformen

Für R Church-Rosser und terminierend gibt es für jedes t genau eine Normalform.

Weiterhin gilt: $t_1 \leftrightarrow^* t_2$ gdw. die Normalformen von t_1 und t_2 identisch sind: $t_1 \downarrow \doteq t_2 \downarrow$

R heißt *konfluent*, wenn für alle $t_0, t_1, t_2 \in T$ mit $t_0 \rightarrow^* t_1$ und $t_0 \rightarrow^* t_2$ ein $t \in T$ existiert, so daß $t_1 \rightarrow^* t$ und $t_2 \rightarrow^* t$ gelten.

R heißt *lokal konfluent*, wenn für alle $t_0, t_1, t_2 \in T$ mit $t_0 \rightarrow t_1$ und $t_0 \rightarrow t_2$ ein $t \in T$ existiert, so daß $t_1 \rightarrow^* t$ und $t_2 \rightarrow^* t$ gelten.

Eigenschaften von TES (3)

Lokale Konfluenz ist schwächer als Konfluenz.

Satz (Äquivalenz von Church-Rosser und Konfluenz): Ein Termersetzungssystem R ist Church-Rosser genau dann, wenn es konfluent ist.

Satz (Äquivalenz von Church-Rosser und lokaler Konfluenz bei Terminierung): Ein terminierendes Termersetzungssystem R ist Church-Rosser genau dann, wenn es lokal konfluent ist.

Kritische Paare

$l_1 \rightarrow r_1, l_2 \rightarrow r_2$ seien variablen-disjunkte Regeln in R , s ein Teilterm von l_1 , σ MGU von s und l_2 . Die Terme

$$c_1 \doteq \sigma(r_1) \qquad c_2 \doteq \sigma(l_2[s \leftarrow r_2])$$

heißen *kritisches Paar* bezgl. R , entstanden durch Überlagerung von $l_2 \rightarrow r_2$ auf $l_1 \rightarrow r_1$ an der Stelle s .

Beispiel:

Regeln:

(1) $z + 0 \rightarrow z$

(2) $h(0) \rightarrow 0$

(3) $h((x + x) + y) \rightarrow x + h(y)$

($h =$ Halbierung)

Kritische Paare:

1. $(0 + h(y), h(0 + y)) \qquad (3 + 1)$

2. $(x + h(0), h(x + x)) \qquad (3 + 1)$

Kritische Paare (2)

Satz (Lokale Konfluenz und kritische Paare):

R ist genau dann lokal konfluent, wenn für jedes kritische Paar (c_1, c_2) ein Term t existiert, so daß $c_1 \rightarrow^* t$ und $c_2 \rightarrow^* t$

Korollar: Ein terminierendes Termersetzungssystem R ist genau dann konfluent, wenn für jedes kritische Paar (c_1, c_2) ein Term t existiert, so daß $c_1 \rightarrow^* t$ und $c_2 \rightarrow^* t$

Korollar: Für terminierende Termersetzungssysteme mit endlich vielen Regeln ist Konfluenz entscheidbar.

Termersetzungssysteme: Terminierung

Lemma: Ein Termersetzungssystem (TES) ist genau dann terminierend, wenn es keine unendlichen Ableitungsketten von *Grundtermen* gibt.

Gesucht: hinreichende Kriterien für Termination.

(strikte) partielle Ordnung auf einer Menge: irreflexiv, transitiv.

Eine Ordnung $>$ auf der Menge von Termen T ($= T_{\Sigma}(V)$) ist eine *Termordnung* (Terminationsordnung, *reduction ordering*), wenn gilt:

1. $>$ ist wohlfundiert (Noethersch): es gibt keine unendlichen absteigenden Ketten
2. $>$ ist monoton: falls $s > t$, dann $f(\dots, s, \dots) > f(\dots, t, \dots)$

Satz: Ein TES R ist genau dann terminierend, wenn es eine Termordnung $>$ auf der Menge der Terme T gibt mit $\sigma(l) > \sigma(r)$ für jede Regel $l \rightarrow r$ in R und jede Substitution σ .

Termersetzungssysteme: Terminierung (2)

Alternative Definition: zusätzliche Bedingung für Termordnung:

(3) $>$ ist kompatibel mit Substitutionen:

$s > t$ impliziert $\sigma(s) > \sigma(t)$ für alle Substitutionen σ .

Formulierung im Satz: “. . . mit $l > r$ für jede Regel $l \rightarrow r$ in R ”.

Aufgrund des Satzes genügt es, zur Bestimmung der Terminierung eines TES R nur die Regeln zu analysieren.

Es gibt viele Möglichkeiten, Termordnungen zu definieren. Eine Termordnung muß für jedes Regelsystem speziell angegeben werden; dies kann (nur) in gewissem Umfang automatisiert werden.

Simplifikationsordnungen

Eine Ordnungsrelation $>$ auf T heißt *Simplifikationsordnung*, falls sie monoton ist und für alle Terme $f(t_1, \dots, t_n)$ die *Subterm-Eigenschaft* gilt:

$$f(t_1, \dots, t_i, \dots, t_n) > t_i \quad \text{für alle } i.$$

Rekursive Pfad-Ordnung (RPO):

Unter RPO wird ein Term kleiner, wenn ein Unterterm u durch einen Term t' ersetzt wird, dessen äußeres Funktionssymbol kleiner ist als das von u und dessen Unterterme alle kleiner als u sind.

Sei $>$ Ordnung auf Operationssymbolen der Signatur. Die RPO $>_{RPO}$ auf T ist definiert durch:

$s >_{RPO} t$ gdw. $s \doteq f(s_1, \dots, s_m)$ und einer der folgenden Fälle:

- (1) $s_i >_{RPO} t$ oder $s_i \doteq t$ für ein i ($1 \leq i \leq m$)
- (2) $t \doteq g(t_1, \dots, t_n)$ und $f > g$ und $s >_{RPO} t_j$ für alle j ($1 \leq j \leq n$)
- (3) $t \doteq f(t_1, \dots, t_n)$ und $\{s_1, \dots, s_m\} \gg_{RPO} \{t_1, \dots, t_n\}$

\gg_{RPO} ist *Multimengenordnung*: Für Multimengen X, Y über einer Menge M mit Ordnung $>$:

$X \gg_{RPO} Y$ falls $X \neq Y$ und $\forall y \in Y - X. \exists x \in X - Y. x >_{RPO} y$

Lemma: Jede RPO ist eine Simplifikationsordnung.

Beispiel: Regelsystem für Umformung in DNF

- (1) $\text{not}(\text{not}(x)) \rightarrow x$
- (2) $\text{not}(x \text{ and } y) \rightarrow \text{not}(x) \text{ or } \text{not}(y)$
- (3) $\text{not}(x \text{ or } y) \rightarrow \text{not}(x) \text{ and } \text{not}(y)$
- (4) $x \text{ and } (y \text{ or } z) \rightarrow (x \text{ and } y) \text{ or } (x \text{ and } z)$
- (5) $(x \text{ or } y) \text{ and } z \rightarrow (x \text{ and } z) \text{ or } (y \text{ and } z)$

Ordnung auf Op-Symbolen: $\text{not} > \text{and} > \text{or}$

Lexikographische Pfadordnung (LPO):

Voraussetzung:

totale Ordnungsrelation (Präzedenz) $>$ auf den Operationssymbolen

Definition der LPO-Relation $>_{LPO}$ auf T :

$$s \doteq f(s_1, \dots, s_m) >_{LPO} g(t_1, \dots, t_n) \doteq t$$

gilt genau dann, wenn eine der folgenden Bedingungen erfüllt ist:

- (1) $s_i >_{LPO} t$ oder $s_i \doteq t$ für ein i
- (2) $f > g$ und $s >_{LPO} t_j$ für alle j
- (3) $f = g$ und $(s_1, \dots, s_m) >_{LPO}^{lex} (t_1, \dots, t_n)$ und $s >_{LPO} t_j$ für alle j

In (3) ist die Relation $>_{LPO}^{lex}$ definiert durch

$$(s_1, \dots, s_m) >_{LPO}^{lex} (t_1, \dots, t_n)$$

gdw.

- (i) $n = 0$ und $m > 0$, oder
- (ii) $s_1 >_{LPO} t_1$, oder
- (iii) $s_1 \doteq t_1$ und $(s_2, \dots, s_m) >_{LPO}^{lex} (t_2, \dots, t_n)$

Knuth-Bendix-Vervollständigung

Ziel: ein (terminierendes) TES so um weitere Regeln für kritische Paare zu ergänzen, daß es vollständig wird.

Darstellung als Menge von Regeln, die auf einer Menge von Gleichungen E und einer Menge von Regeln R operieren. E besteht aus der Menge der Ausgangsgleichungen bzw. kritischen Paaren.

$$(1) \quad \frac{E, R}{E \cup (c_1 = c_2), R} \quad \text{falls } (c_1, c_2) \text{ kritisches Paar bzgl. } R$$

$$(2) \quad \frac{E \cup (s = t), R}{E, R \cup (s \rightarrow t)} \quad \text{falls } s > t$$

$$(3) \quad \frac{E \cup (s = t), R}{E \cup (s' = t), R} \quad \text{falls } s \rightarrow_R s'$$

Knuth-Bendix-Vervollständigung (2)

$$(4) \quad \frac{E \cup (s \doteq t), R}{E, R}$$

$$(5) \quad \frac{E, R \cup (s \rightarrow t)}{E, R \cup (s \rightarrow t')} \quad \text{falls } t \rightarrow_R t'$$

$$(6) \quad \frac{E, R \cup (s \rightarrow t)}{E \cup (s' = t), R} \quad \text{falls } s \rightarrow_R s'$$

Regel (4) sagt, dass Gleichungen gestrichen werden können, deren Seiten zu identischen Termen reduziert worden sind, z.B. durch Anwendungen der Regel (3).

Beispiel für kritische Paare (s. oben):

Regeln:

$$(1) \quad z + 0 \rightarrow z$$

$$(2) \quad h(0) \rightarrow 0$$

$$(3) \quad h((x + x) + y) \rightarrow x + h(y)$$

($h =$ Halbierung)

Termordnung: $>_{RPO}$ mit Präzedenz $h > + > 0$

\leadsto Regeln sind mit $>_{RPO}$ korrekt orientiert

Kritische Paare:

$$1. \quad (0 + h(y), h(0 + y)) \quad (3 + 1)$$

$$2. \quad (x + h(0), h(x + x)) \quad (3 + 1)$$

Beispiel für kritische Paare (Forts.):

Reduktion der Gleichungen:

$$0 + h(y) = h(0 + y)$$

$$x = h(x + x)$$

Orientierung führt zu neuen Regeln:

$$(4) \quad h(0 + y) \rightarrow 0 + h(y)$$

$$(5) \quad h(x + x) \rightarrow x$$

\rightsquigarrow neue Regelmenge ist lokal konfluent
(kritische Paare müssen untersucht werden)

Beispiel: Vervollständigung für Gruppenaxiome mit LPO

\Rightarrow Übungen

Eigenschaften des Knuth-Bendix-Verfahrens

Reduktionsregeln sollen das resultierende TES möglichst effizient machen: die rechte Seite jeder Regel soll irreduzibel bzgl. R sein, die linke Seite irreduzibel bzgl. R ohne die betreffende Regel.

Mögliche Verhalten des Knuth-Bendix-Verfahrens:

1. Die Menge der Gleichungen ist leer: ein vollständiges (kanonisches) TES für das ursprüngliche Gleichungssystem ist konstruiert worden; das Verfahren terminiert.
2. Eine Gleichung kann nicht orientiert werden, ohne die Terminierungseigenschaft des TES zu zerstören: das Regelsystem kann nicht vervollständigt werden, das Verfahren bricht erfolglos ab.

Modifikationen des K-B-Verfahrens erlauben es, die Termordnung dynamisch zu verändern, um weitere Gleichungen orientieren zu können, aber auch dann kann dieser Fall eintreten.

3. Es werden neue Regeln generiert, ohne daß das Verfahren anhält: es wird ein unendliches Regelsystem generiert. (Dies kann insbesondere passieren bei einer Gleichungstheorie, die nicht entscheidbar ist.)