

# Abschnitt 1calc

11. Juni 2000

Dieses Werk ist urheberrechtlich geschützt; alle Rechte mit Ausnahme des folgenden sind vorbehalten:

*Studenten dürfen für ihre persönlichen Lernzwecke dieses Dokument ausdrucken und auf ihren Rechnern Kopien der entsprechenden Datei vorhalten.*

Ausdrücklich verboten wird jede andere Verwendung von Ausdrucken und Dateikopien; insbesondere darf dieses Skriptum nicht in irgendeiner Weise vertrieben werden und es dürfen keine Kopien der Datei auf irgendwelchen „Skriptenservern“ angelegt werden. Gegen die Anlage von Verweisen („hypertext links“) auf diese Datei ist selbstverständlich nichts einzuwenden.

Copyright © Herbert Klaeren, 1999.

# Inhaltsverzeichnis

<b>1</b>	<b>Einführung</b>	<b>9</b>
1.1	Was ist Informatik? . . . . .	9
1.2	Geschichte der Programmierung . . . . .	17
<b>2</b>	<b>Induktive Definitionen</b>	<b>21</b>
2.1	Natürliche Zahlen . . . . .	21
2.2	Wortmengen . . . . .	28
2.3	Syntaktische Beschreibungsmittel . . . . .	32
2.4	Terme . . . . .	40
2.5	Aufgaben . . . . .	47
<b>3</b>	<b>Algorithmen und Programme</b>	<b>49</b>
3.1	Algorithmus . . . . .	49
3.2	Programme . . . . .	54
3.2.1	Ausdrücke, Namen . . . . .	55
3.2.2	Fallunterscheidungen . . . . .	59
3.2.3	Rekursive Definitionen . . . . .	60
3.3	Aufgaben . . . . .	62
<b>4</b>	<b>Abstrakte Datentypen</b>	<b>93</b>
4.1	Einführung . . . . .	93
4.2	Boolesche Werte . . . . .	95
4.3	Zähler . . . . .	98
4.4	Listen . . . . .	101
4.5	Bäume . . . . .	108
4.6	Aufgaben . . . . .	116
<b>5</b>	<b>Logische Kalküle</b>	<b>117</b>
5.1	Ein Kalkül für die Aussagenlogik . . . . .	118
5.2	Ein Kalkül für die Prädikatenlogik . . . . .	122
5.3	Der Reduktionskalkül $RC_1$ . . . . .	125
5.4	Der $\lambda$ -Kalkül . . . . .	127
<b>A</b>	<b>Mathematische Grundlagen</b>	<b>119</b>
A.1	Mengen, Relationen, Abbildungen . . . . .	119

A.2	Formale Logik . . . . .	124
	A.2.1 Aussagenlogik . . . . .	124
	A.2.2 Prädikatenlogik . . . . .	126
A.3	Halbordnungen . . . . .	128
A.4	Aufgaben . . . . .	129

## Kapitel 5

### Logische Kalküle

Als *Implementierungen* eines abstrakten Datentyps betrachten wir alle Algebren, deren Operationen die definierenden Gleichungen erfüllen. In Definition 4.3 haben wir ein semantisches Kriterium dafür angeführt.

Wollen wir nun die Frage studieren, welche Gleichungen zwischen Termen *allgemeingültig* sind, so gestaltet sich dies als technisch nicht einfach: Wir müssen immer *alle Modelle* betrachten und, um die Gültigkeit jeder einzelnen Gleichung nachzuweisen, *alle Variablenbelegungen*. Da hilft es nicht viel, daß wir den letzten Teil der Aufgabe durch strukturelle Induktion unterstützen können. Der Teil der mathematischen Logik, der sich mit solchen Fragestellungen beschäftigt, heißt *Modelltheorie*. In dem schon häufiger angesprochenen Zwiespalt Syntax–Semantik befindet sich die Modelltheorie auf der Seite der Semantik.

Einer Behandlung durch Computer sind aber eher symbolische Verfahren zugänglich, bei denen wir im wesentlichen syntaktisch operieren: ein Term wird so lange in seine Teilterme strukturiert, bis wir bei atomaren Termen angekommen sind; danach wird entsprechend der zuvor ermittelten Struktur wieder bis herauf zum kompletten Term argumentiert. Gelegentlich werden Teilterme durch andere, als *äquivalent* bewiesene Terme ersetzt. Wir haben dies im vergangenen Abschnitt (Beispiel 4.6) bereits als „Rechnen in einem Gleichungssystem“ informell ausgenutzt.

Hier wollen wir diesen Prozeß auf eine solide formale Basis stellen; deshalb beschäftigen wir uns mit etwas *Beweistheorie*. Dazu gehen wir zunächst von den Gleichungen über Datentypen wieder auf einen übersichtlicheren Bereich zurück, nämlich die Aussagenlogik.

## 5.1 Ein Kalkül für die Aussagenlogik

Der Umgang mit formalen Beweisen erfordert ein strengeres Vorgehen als in der Modelltheorie, da kleine Änderungen an der *Sprache*, in der unsere Sätze und Beweise formuliert sind, häufig große Änderungen an der Form der Beweise mit sich ziehen. In der Modelltheorie ist das anders; dort lassen sich neue Sprachelemente meist einfach durch neue Regeln beschreiben.

Wir definieren induktiv die Sprache  $\mathcal{L}_1$  der Aussagenlogik wie folgt:

**Definition 5.1** Sei  $V$  eine endliche Menge *aussagenlogischer Variablen*,  $V = \{P_1, P_2, \dots, P_N\}$  mit  $N > 0$ ,  $\mathcal{C} = \{\top, \perp\}$  die Menge der *aussagenlogischen Konstanten*. Sei außerdem  $J = \{\neg, \wedge, \vee\}$  die Menge der *logischen Junktoren*. Die Sprache  $\mathcal{L}_1$  der *aussagenlogischen Formeln* ist die kleinste Menge  $\mathcal{L}_1 \subseteq \Sigma^*$  mit  $\Sigma = V \cup \mathcal{C} \cup J \cup \{(, )\}$  mit den Eigenschaften:

1.  $\mathcal{C} \subseteq \mathcal{L}_1$
2. Ist  $L \in \mathcal{C} \cup \mathcal{V}$ , so sind  $L \in \mathcal{L}_1$  und  $\neg L \in \mathcal{L}_1$ . Die Formeln entsprechend Punkt (1) und (2) heißen auch *Literale*.
3. Sind  $A \in \mathcal{L}_1$  und  $B \in \mathcal{L}_1$ , so sind auch  $(A \wedge B) \in \mathcal{L}_1$  und  $(A \vee B) \in \mathcal{L}_1$ .

Sind Mißverständnisse ausgeschlossen, können Klammern beim Hinschreiben weggelassen werden.

Als Konvention halten wir fest, daß wir Literale durch die Buchstaben  $L, M, N$  und Formeln durch  $A, B, C, D, F, G$  bezeichnen.

Gegenüber der in Abschnitt A.2 beschriebenen Sprache der Aussagenlogik fällt auf, daß Negation hier nur für Literale vorgesehen ist. Dies stellt aber keine echte Einschränkung dar, wie man sich leicht überzeugt. Daß wir nunmehr neue Symbole für die Wahrheitswerte eingeführt haben, wobei  $\top$  (sprich: „top“) für „wahr“ steht und  $\perp$  (sprich: „bottom“) für „falsch“, hat mit der bereits im Abschnitt 2.3 angesprochenen Trennung von Metasprache und Objektsprache zu tun: Im Augenblick ist die Sprache der Aussagenlogik für uns eine Objektsprache; wir möchten aber trotzdem logisch (also metasprachlich) darüber reden können.

Wie läßt sich nun der Begriff eines Beweises in dieser Sprache formalisieren? Zunächst einmal werden einige grundlegende Aussagen als wahr

festgelegt; diese nennt man *Axiome*. Dann muß es einen Weg geben, aus schon bewiesenen Aussagen andere zu schließen. Dazu dienen die sogenannten *Regeln*. Die Kombination von Sprache, Axiome und Regeln nennt man einen logischen *Kalkül*.

Eine beliebte Klasse von Kalkülen sind die sogenannten *Sequenzkalküle*, in denen sich Beweise besonders einfach finden lassen. Sequenzkalküle beschäftigen sich statt mit einzelnen Formeln allgemeiner mit Sequenzen von Formeln.

Eine *Sequenz* ist eine endliche Multimenge von Formeln in  $\mathcal{L}_1$ . Eine einzelne Formel wird mit der entsprechenden einelementigen Sequenz identifiziert. In Erweiterung unserer Konventionen wollen wir vereinbaren, daß Sequenzen durch  $\Gamma, \Pi, \Delta$  etc. bezeichnet werden. Die semantische Vorstellung ist, daß eine Sequenz als Disjunktion ihrer Formeln betrachtet werden soll, d.h. die Sequenz  $A, B, C$  steht für „A oder B oder C“.

Die Axiome des Kalküls  $SC_1$  sind die folgenden:

$$\top, \Gamma \quad (5.1)$$

$$\neg \mathcal{P}_i, \mathcal{P}_i, \Gamma \quad (5.2)$$

Diese Axiome stimmen mit unseren semantischen Vorstellungen überein: „wahr oder  $\Gamma$ “ ist offensichtlich für alle  $\Gamma$  wahr, ebenso für jede Aussage  $\mathcal{P}$  die Aussage „ $\mathcal{P}$  oder nicht  $\mathcal{P}$ “. Beachte, daß (5.2) wie ein einziges Axiom aussieht, aber in Wirklichkeit für  $N$  Axiome steht.

Die Regeln in  $SC_1$  sehen folgendermaßen aus:

$$\frac{A, B, \Gamma}{A \vee B, \Gamma} \quad (5.3)$$

$$\frac{A, \Gamma \quad B, \Gamma}{A \wedge B, \Gamma} \quad (5.4)$$

Diese Regeln sind folgendermaßen zu lesen:

Über dem „Bruchstrich“, der hier jetzt *Inferenzstrich* genannt wird, befinden sich *Voraussetzungen* oder *Prämissen*. In einem Beweis sind dies entweder Axiome oder „schon bewiesene“ Sequenzen. Unter dem Inferenzstrich befindet sich die *Schlussfolgerung* oder *Konsequenz* der Regel, d.h. also die Sequenz, die bewiesen wird. Für  $A$  und  $B$  können dabei beliebige Formeln eingesetzt werden.

Auch hier ist die semantische Vorstellung einleuchtend: Ist „A oder B oder  $\Gamma$ “ bewiesen, so offenbar auch „ $A \vee B$  oder  $\Gamma$ “. Für einen Beweis von  $A \wedge B$  muß es separate Beweise für A und B geben. Wir möchten jedoch nochmals betonen, daß diese semantische Vorstellung hier nur unterstützend wirken kann, denn wir operieren in einem Kalkül rein symbolisch, d.h. also syntaktisch.

Ein Beweis in  $SC_1$  ist ein Baum, in dem Regeln angewendet werden, und an dessen Blättern Axiome stehen. Als Beispiel beweisen wir die Formel  $(\mathcal{P}_1 \wedge \mathcal{P}_2) \vee (\neg \mathcal{P}_1 \vee \neg \mathcal{P}_2)$ :

$$\frac{\frac{\frac{\mathcal{P}_1, \neg \mathcal{P}_1, \neg \mathcal{P}_2}{\mathcal{P}_1 \wedge \mathcal{P}_2, \neg \mathcal{P}_1, \neg \mathcal{P}_2} \text{ (5.2)}}{\mathcal{P}_1 \wedge \mathcal{P}_2, \neg \mathcal{P}_1 \vee \neg \mathcal{P}_2} \text{ (5.3)}}{(\mathcal{P}_1 \wedge \mathcal{P}_2) \vee (\neg \mathcal{P}_1 \vee \neg \mathcal{P}_2)} \text{ (5.3)}$$

Um alle „Beweismittel“ sozusagen „gerichtsfest“ vorzulegen, haben wir hier die verwendeten Axiome sowie Schlüsse durch die entsprechenden Nummern gekennzeichnet. Bei der Anwendung von (5.2) in der rechten Hälfte wird deutlich, daß wir die Sequenzen auch umsortieren dürfen; deshalb deren Definition als Multimenge.

Es gibt nun also zwei Arten der Wahrheit: die vorherige, intuitive Vorstellung von *Wahrheit* in der Modelltheorie, und der neue Begriff der *Beweisbarkeit*. Sie müssen ab sofort unterschieden werden: „A ist wahr im Modell  $\mathcal{B}$ “ wird geschrieben als:

$$\models_{\mathcal{B}} A$$

Ist A in *allen Modellen* wahr, so schreiben wir

$$\models A$$

„A ist beweisbar“ wird geschrieben als:

$$\vdash A$$

Aus dem neuen Begriff der Beweisbarkeit ergeben sich noch weitere Definitionen, bezogen auf beliebige Kalküle:

- Ein logischer Kalkül mit Negation heißt *konsistent*, wenn es keine Formel A gibt, für die sich sowohl A als auch  $\neg A$  beweisen läßt.

- Ein logischer Kalkül *korrekt*, gilt:

$$\text{falls } \vdash A, \text{ so } \models A$$

- Ein logischer Kalkül ist *vollständig*, wenn gilt:

$$\text{falls } \models A, \text{ so } \vdash A$$

Offensichtlich können logische Kalküle, die nicht konsistent sind, auch nicht korrekt sein, denn die Formel  $A \wedge \neg A$  wäre in einem inkonsistenten Kalkül beweisbar, aber sie kann in keinem Modell wahr sein.

**Satz 5.2**  $SC_1$  ist konsistent, korrekt und vollständig.

**Beweis: Korrektheit** Sei  $\gamma : V \rightarrow \{W, F\}$  eine Variablenbelegung. Entsprechend Satz 2.31 setzen wir  $\gamma$  zunächst fort auf  $\mathcal{L}_1$ , indem wir die Konstanten und logischen Junktoren in der natürlichen Weise interpretieren. Durch Oder-Verknüpfung setzen wir  $\gamma^\#$  induktiv auf endliche Sequenzen fort.

Für alle Axiome  $\Gamma$  gilt offensichtlich  $\gamma^\#(\Gamma) = W$ .

Gelte nun  $\gamma^\#(A, B, \Gamma) = W$ . Dann muß  $\gamma^\#(A) = W, \gamma^\#(B) = W$  oder  $\gamma^\#(\Gamma) = W$  gelten. Wie auch immer die Situation sein mag, es wird in jedem Fall  $\gamma^\#(A \vee B) = W$  oder  $\gamma^\#(\Gamma) = W$  gelten. Die Regel (5.3) erhält also die Wahrheit.

Gelte nun  $\gamma^\#(A, \Gamma) = W$  und  $\gamma^\#(B, \Gamma) = W$ . Wenn nun  $\gamma^\#(\Gamma) = W$  nicht gilt, so muß  $\gamma^\#(A) = W$  und  $\gamma^\#(B) = W$  gelten. Dann gilt aber auch  $\gamma^\#(A \wedge B) = W$ . Auch die Regel (5.4) erhält also die Wahrheit.

In einem Beweisbaum haben deshalb zunächst alle Axiome — also alle Blätter — den Wahrheitswert  $W$ . Wahrheit vererbt sich in den Regeln aber von oben nach unten; durch strukturelle Induktion folgt also, daß auch die Sequenz ganz unten im Beweisbaum den Wahrheitswert  $W$  haben muß.

**Vollständigkeit** Sei  $\Gamma$  eine Sequenz mit  $\gamma^\#(\Gamma) = W$  für alle Variablenbelegungen  $\gamma$ .

Dann läßt sich mit  $\Gamma$  als Wurzel ein Baum der gleichen Art wie ein Beweisbaum konstruieren, indem systematisch alle Formeln durch Anwendung der Regeln (5.4) und (5.3) zerlegt werden, bis nur noch Literale übrig sind.

Analog zum vorigen Korrektheitsbeweis läßt sich leicht sehen, daß Wahrheit auch von unten nach oben vererbt wird.

Die Blätter des Baums haben also unter allen Variablenbelegungen den Wahrheitswert  $W$ . Angenommen, ein solches Blatt  $\Gamma$  hätte weder die Form (5.1) noch die Form (5.2). Dann besteht es nur aus  $\perp$  und Literalen der Form  $\mathcal{P}_i$  und  $\neg\mathcal{P}_i$ , wobei  $\mathcal{P}_i$  und  $\neg\mathcal{P}_i$  nicht gleichzeitig auftreten. ( $\perp$  allein ist offensichtlich nicht möglich.) Dann läßt sich aber eine Belegung  $\beta$  konstruieren mit  $\beta(\mathcal{P}_i) = F$  für  $\mathcal{P}_i \in \Gamma$  und  $\beta(\mathcal{P}_j) = W$  für  $\neg\mathcal{P}_j \in \Gamma$ . Für deren Fortsetzung  $\beta^\#$  gilt dann  $\beta^\#(\Gamma) = F$  im Widerspruch zur Voraussetzung.

**Konsistenz** Sei  $A$  eine Formel mit  $\vdash A$  und  $\vdash \neg A$ . Aus der Korrektheit folgt dann, daß für jede Belegung  $\gamma$  gilt  $\gamma^\#(A) = W = \gamma^\#(\neg A)$ . Das ist aber unmöglich, da  $\gamma^\#(\neg A) = \text{if } \gamma^\#(A) = W \text{ then } F \text{ else } W$ .

□

## 5.2 Ein Kalkül für die Prädikatenlogik

$SC_1$  läßt sich auf die Prädikatenlogik erweitern. Das Ergebnis wird der Kalkül  $SPC_{1,\infty}$  sein. Zunächst zur Sprache dieses Kalküls,  $\mathcal{L}_{1,\infty}$ :

- An die Stelle der aussagenlogischen Variablen tritt eine endliche Menge von *Prädikaten*  $\{\mathcal{P}_1, \mathcal{P}_2, \dots, \mathcal{P}_N\}$ . Jedes Prädikat  $\mathcal{P}_i$  hat eine zugewiesene *Stelligkeit*  $\alpha(i)$ .
- Zu den atomaren Bestandteilen der Sprache kommen noch eine endliche Menge von *Konstanten*  $\{\mathcal{C}_1, \dots, \mathcal{C}_M\}$  und eine abzählbar unendliche Menge von *Variablen*  $\{\mathcal{V}_1, \dots, \mathcal{V}_n, \dots\}$  hinzu.
- Die Literale sind nun  $\top$ ,  $\perp$ ,  $\mathcal{P}_i(b_1, \dots, b_{\alpha(i)})$  und  $\neg\mathcal{P}_i(b_1, \dots, b_{\alpha(i)})$  wobei jedes  $b_j$  eine Konstante oder eine Variable ist.
- Formeln sind nun die Literale,  $(A \wedge B)$  und  $(A \vee B)$  für Formeln  $A$  und  $B$  sowie zusätzlich für eine Formel  $A$  und eine Variable  $x$  die Formeln  $\forall x A$  und  $\exists x A$ .

Unsere Bezeichnungskonvention wird dahingehend erweitert, daß  $x, y$  und  $z$  Variablen bezeichnen und  $c$  und  $d$  Konstanten.  $a$  und  $b$  bezeichnen wahlweise Konstanten oder Variablen.

Im Umgang mit den Quantoren  $\forall$  und  $\exists$  treten nun zwei zentrale Begriffe auf, nämlich der der *freien* und der *gebundenen* Variablen. Eine Variable  $x$  heißt *frei* in einer Formel  $A$  gdw. es ein Vorkommen von  $x$  außerhalb jeglicher Teilformel  $\forall xA$  bzw.  $\exists xA$  gibt. Wir schreiben in diesem Fall  $x \in \text{free}(A)$ . Beispiele:

$$\begin{aligned} x &\in \text{free}(\mathcal{P}_1(x)) \\ x &\notin \text{free}(\forall xA) \\ x &\in \text{free}(\mathcal{P}_1(x) \wedge (\forall x\mathcal{P}_2(x))) \end{aligned}$$

Umgekehrt ist eine Variable *gebunden* in einer Formel  $A$ , wenn es ein Vorkommen von  $x$  innerhalb einer Teilformel  $\forall xB$  oder  $\exists xB$  gibt. (Eine Variable kann in einer Formel gleichzeitig frei und gebunden vorkommen, z.B. die Variable  $x$  in unserem letzten Beispiel.)

In Scheme ist `lambda` auch eine Art Quantor; die Begriffe „frei“ und „gebunden“ haben in Scheme exakt die gleiche Bedeutung wie in der Prädikatenlogik.

Zum Hantieren mit prädikatenlogischen Formeln ist es wichtig zu erkennen, daß die Variablen beliebig gegeneinander austauschbar sind, solange die Zugehörigkeit von Variablenvorkommen zu ihren Quantoren dadurch unberührt bleibt. Intuitiv hat die Formel  $\forall x\mathcal{P}_1(x)$  die gleiche Bedeutung wie  $\forall y\mathcal{P}_1(y)$ . Auch die Formeln  $(\forall x\mathcal{P}_1(x)) \vee (\exists x\mathcal{P}_2(x))$  und  $(\forall x\mathcal{P}_1(x)) \vee (\exists y\mathcal{P}_2(y))$  haben die gleiche Bedeutung, aber  $\forall x\mathcal{P}_1(x, y)$  und  $\forall x\mathcal{P}_1(x, x)$  haben ganz unterschiedliche Bedeutung.

Es lohnt sich deswegen, den Begriff der Substitution zu formalisieren. Für eine Variable  $x$  und eine Konstante  $c$  bezeichnet  $A[x \mapsto c]$  die Formel  $A$ , in der  $x$  durch  $c$  ersetzt wurde. Für eine Variable  $y$  ist die Substitution  $A[x \mapsto y]$  induktiv definiert:

- Ist  $x \notin \text{free}(A)$ , so ist  $A[x \mapsto y] := A$ .
- Ist  $A$  ein Literal und  $x \in \text{free}(A)$ , so entsteht  $A[x \mapsto y]$  aus  $A$ , indem dort jedes Vorkommen von  $x$  durch  $y$  ersetzt wird.
- Falls  $A = C \wedge D$  und  $x \in \text{free}(A)$ , so ist  $A[x \mapsto y] := C[x \mapsto y] \wedge D[x \mapsto y]$ .
- Falls  $A = C \vee D$  und  $x \in \text{free}(A)$ , so ist  $A[x \mapsto y] := C[x \mapsto y] \vee D[x \mapsto y]$ .

- Falls  $A = \forall zB$  mit  $z \neq y$  und  $x \in \text{free}(A)$ , so ist  $A[x \mapsto y] := \forall z(B[x \mapsto y])$ .
- Falls  $A = \exists zB$  mit  $z \neq y$  und  $x \in \text{free}(A)$ , so ist  $A[x \mapsto y] := \exists z(B[x \mapsto y])$ .
- Falls  $A = \forall yB$ ,  $x \in \text{free}(A)$  und  $z$  eine Variable ist, die in  $A$  nicht vorkommt, so ist  $A[x \mapsto y] := \forall z(B[y \mapsto z][x \mapsto y])$ .
- Falls  $A = \exists yB$ ,  $x \in \text{free}(A)$  und  $z$  eine Variable ist, die in  $A$  nicht vorkommt, so ist  $A[x \mapsto y] := \exists z(B[y \mapsto z][x \mapsto y])$ .

Eine Überlegung zeigt, daß es in den letzten beiden Fällen auch möglich ist, den Quantor über  $x$  zu definieren und dann in der Formel  $B$  einfach  $x$  durch  $y$  und  $y$  durch  $x$  zu ersetzen. Beispiel: Sei  $A := \forall y(\mathcal{P}_1(x, y) \vee \forall z\exists x\mathcal{P}_2(x, y, z))$ . Dann gilt:

$$A[x \mapsto y] = \forall x(\mathcal{P}_1(y, x) \vee \forall z\exists y\mathcal{P}_2(y, x, z))$$

Die Regeln von  $\text{SPC}_{1,\infty}$  ergänzen die Regeln von  $\text{SC}_1$  um die folgenden:

$$\frac{A[x \mapsto b], \exists xA, \Gamma}{\exists xA, \Gamma} \quad (5.5)$$

$$\frac{A[x \mapsto y], \Gamma}{\forall xA, \Gamma} \quad \text{wobei } y \notin \text{free}(\forall xA) \cup \text{free}(\Gamma) \quad (5.6)$$

Die Intuition hinter den beiden Regeln ist vielleicht nicht ganz offensichtlich:

Die erste Regel beschäftigt sich mit dem Beweis der Existenz eines  $x$  mit der Eigenschaft  $A$ . Dieser Beweis ist auf jeden Fall dann geführt, wenn ein konkretes Beispiel für  $x$  gefunden ist, eine Konstante.

Die zweite Regel beschäftigt sich mit dem Beweis, daß  $A$  für alle  $x$  gilt. Wenn es gelingt, die Aussage  $A$  zu beweisen, in der  $x$  durch eine frische Variable  $y$  ersetzt wird, so hat man offensichtlich die Aussage  $A$  für ein Ding  $x$  bewiesen, ohne irgendwelches Wissen über  $x$  zu besitzen. Damit kann man dann aber schließen, daß die Aussage für alle  $x$  gilt.

**Satz 5.3**  $\text{SPC}_{1,\infty}$  ist korrekt und konsistent.

Während der Beweis dieses Satzes nicht allzu schwierig ist, übersteigt der Beweis der Vollständigkeit die Mittel dieser Vorlesung. Es sei trotzdem gesagt:

**Satz 5.4**  $SPC_{1,\infty}$  ist vollständig.

### 5.3 Der Reduktionskalkül $RC_1$

$SC_1$  und  $SPC_{1,\infty}$  sind sogenannte *Inferenzkalküle*: Beweise sind Bäume, in denen Anwendungen von Regeln durch Inferenzstriche getrennt sind. Es gibt jedoch noch andere Methoden, logische Kalküle aufzubauen. Die sogenannten *Reduktionskalküle* benutzen das Prinzip der algebraischen Vereinfachung: Eine logische Formel wird schrittweise durch die Anwendung von Gleichungen vereinfacht.

Der Kalkül  $RC_1$  ist, wie  $SC_1$ , eine Formalisierung der Aussagenlogik und benutzt die gleiche Sprache,  $\mathcal{L}_1$ . Kommt am Ende  $\top$  heraus, ist die Formel eine Tautologie. Hier sind die Regeln von  $SC_1$ :

**Definition 5.5 (Regeln von  $RC_1$ )**

$$\top \vee A \triangleright \top \quad A \vee \top \triangleright \top \quad (5.7)$$

$$\top \wedge A \triangleright A \quad A \wedge \top \triangleright A \quad (5.8)$$

$$(5.9)$$

$$L \vee A_1 \vee \dots \vee A_n \vee \neg L \vee B \triangleright \top \quad (5.10)$$

$$(A \vee B) \vee C \triangleright A \vee (B \vee C) \quad (5.11)$$

$$(A \wedge B) \vee C \triangleright (A \vee C) \wedge (B \vee C) \quad (5.12)$$

$$C \vee (A \wedge B) \triangleright (C \vee A) \wedge (C \vee B) \quad (5.13)$$

Wie sind diese Regeln zu verstehen? Eine logische Formel läßt sich mit Hilfe einer Reduktionsregel dann vereinfachen, wenn sie auf die linke Seite der Regel paßt. Sie heißt dann ein *Redex* (aus *reducible expression* abgekürzt) und wird durch die Entsprechung der rechten Regelseite ersetzt. Hier ein Beispiel:

$$E_1 := (\mathcal{P}_1 \wedge \mathcal{P}_2) \vee (\neg \mathcal{P}_1 \vee \neg \mathcal{P}_2)$$

Diese Formel paßt auf die linke Seite von Regel 5.12 und läßt sich dementsprechend folgendermaßen mit  $A = \mathcal{P}_1$ ,  $B = \mathcal{P}_2$  und  $C = (\neg\mathcal{P}_1 \vee \neg\mathcal{P}_2)$  reduzieren:

$$E_2 := (\mathcal{P}_1 \vee (\neg\mathcal{P}_1 \vee \neg\mathcal{P}_2)) \wedge (\mathcal{P}_2 \vee (\neg\mathcal{P}_1 \vee \neg\mathcal{P}_2))$$

Es ist leicht zu beweisen — z.B. in  $SC_1$  — daß bei den Reduktionsregeln jeweils linke und rechte Seite äquivalent sind. Auf diese Art und Weise bildet  $\triangleright$  eine Relation auf  $\mathcal{L}_1$ .

Leider läßt sich die obige Formel nicht weiter reduzieren — sie paßt auf keine linke Regelseite. Das ist insbesondere deswegen bedauerlich, da es sich um eine Tautologie handelt. Jedoch läßt sich  $E_2$  als  $E_2 = E_2^{(1)} \wedge E_2^{(2)}$  schreiben und beide *Teilformeln*  $E_2^{(1)}$  und  $E_2^{(2)}$  passen auf die Regel 5.10. Wendet man Regel 5.10 auf die beiden Teilformeln an, so kommt folgende Formel heraus:

$$E_3 := \top \wedge \top$$

Diese läßt sich wiederum mit Regel 5.8 zu  $\top$  reduzieren, und fertig ist der Beweis.

Wir müssen also  $\triangleright$  zu einer anderen Relation  $\blacktriangleright$  ausbauen, die auch auf Subtermen von Formeln arbeiten kann, m.a.W. Redexe im Inneren einer Formel finden kann:

**Definition 5.6 (Erweiterung von  $\triangleright$  auf Subterme)**  $\blacktriangleright$  ist die Erweiterung von  $\triangleright$  auf Subterme: Sei  $A$  eine Formel aus  $\mathcal{L}_1$ , in der eine andere Formel  $B$  an einer bestimmten Stelle vorkommt. Sei des weiteren  $C$  derart, daß  $B \triangleright C$  gilt. Sei des weiteren  $D$  eine Formel, die mit  $A$  übereinstimmt, außer, daß an der Stelle des Vorkommens von  $B$  stattdessen  $C$  steht. Dann gilt  $A \blacktriangleright D$ .

Diese Definition macht  $\blacktriangleright$  übrigens zur Ableitungsrelation eines *Termersetzungssystems*.

Nun läßt sich Beweisbarkeit in  $RC_1$  definieren:

**Definition 5.7 (Beweisbarkeit in  $RC_1$ )** Eine Formel  $A$  ist in  $RC_1$  beweisbar (geschrieben  $RC_1 \vdash A$  oder einfach nur  $\vdash A$ ), wenn  $A \blacktriangleright^* \top$  gilt.

Dabei ist  $\blacktriangleright^*$  der *transitiv-reflexive Abschluß* von  $\blacktriangleright$ :  $A \blacktriangleright^* B$  gilt genau dann, wenn entweder  $A = B$  (darum „reflexiv“), oder wenn es Formeln  $A_1, \dots, A_n$  gibt, so daß

$$A \blacktriangleright A_1 \blacktriangleright \dots \blacktriangleright A_n \blacktriangleright B$$

gilt. (Damit wird  $\blacktriangleright^*$  auch transitiv.)

Diese drei Zutaten — Reduktionsregeln, Erweiterung auf Subterme und reflexiv-transitiver Abschluß — machen einen Reduktionskalkül aus.

**Satz 5.8**  $RC_1$  ist korrekt, konsistent und vollständig.

Beweis:

Für  $A \blacktriangleright B$  sind  $A$  und  $B$  offensichtlich immer äquivalent. Daraus folgen Konsistenz und Korrektheit.

Der Beweis für die Vollständigkeit ist etwas aufwendiger und wird darum nur skizziert. Prinzipiell schwierig ist er allerdings nicht:

Die Reduktionsregeln 5.11, 5.12 und 5.13 für sich gesehen überführen eine Formel in die Form:

$$(L_1^{(1)} \vee (L_2^{(1)} \vee (\dots \vee L_{k_1}^{(1)})) \dots) \wedge \dots \wedge (L_1^{(n)} \vee (L_2^{(n)} \vee (\dots \vee L_{k_n}^{(n)})) \dots)$$

(Diese Form heißt auch *konjunktive Normalform*.) Eine Formel dieser Form ist genau dann wahr, wenn alle Teilformeln

$$(L_1^{(j)} \vee (L_2^{(j)} \vee (\dots \vee L_{k_j}^{(j)})) \dots)$$

wahr sind. Eine solche Teilformel ist aber genau dann wahr, wenn sie entweder  $\top$  enthält oder zwei Literale  $L_1^{(j)}$  und  $L_m^{(j)}$  enthält mit  $L_1^{(j)} = L_m^{(j)}$ . Genau diese beiden Fälle werden aber von den Regeln 5.7 und 5.10 abgedeckt, so daß sich diese Teilformeln zu  $\top$  reduzieren lassen. Regel 5.8 besorgt dann den Rest.

## 5.4 Der $\lambda$ -Kalkül

## Literaturverzeichnis

- [Bau68] BAUMANN, RICHARD: *Algol-Manual der Alcor-Gruppe*. R. Oldenbourg, München/Wien, 3. Auflage, 1968.
- [Bau89] BAUER, F. L.: *100 Jahre Peano-Zahlen*. Informatik-Spektrum, 12:340–341, 1989.
- [End72] ENDERTON, H. B.: *A Mathematical Introduction to Logic*. Academic Press, 1972.
- [GKP89] GRAHAM, R. L., D. E. KNUTH und O. PATASHNIK: *Concrete Mathematics*. Addison-Wesley, 1989.
- [Hal69] HALMOS, P.: *Naive Mengenlehre*. Vandenhoeck & Ruprecht, Göttingen, 1969.
- [KCR98] KELSEY, RICHARD, WILLIAM CLINGER und JONATHAN REES: *Revised<sup>5</sup> Report on the Algorithmic Language Scheme*. SIGPLAN Notices, 33(9):26–76, 1998.
- [Kla83] KLAEREN, HERBERT: *Algebraische Spezifikation — Eine Einführung*. Springer Verlag, Berlin-Heidelberg-New York, 1983.
- [Knu73] KNUTH, D. E.: *The Art of Computer Programming*, volume 3. Addison-Wesley, 1973. Sorting and Searching.
- [Mes71] MESCHKOWSKI, H.: *Einführung in die moderne Mathematik*, Band 75/75a der Reihe *Hochschultaschenbücher*. BI, 1971.
- [MHR80] METROPOLIS, N., J. HOWLETT, and GIAN-CARLO ROTA (editors): *A History of Computing in the Twentieth Century*. Academic Press, 1980.
- [Rie91] RIESE, ADAM: *Rechnung auff der Linihen und Federn /Auff allerley handthirung gemacht / durch Adam Risen (Faksimile 2. Aufl. Erfurt 1532)*. Magistrat der Stadt Erfurt, 1991.
- [Wex81] WEXELBLAT, RICHARD L. (editor): *History of Programming Languages*, New York, 1981. Academic Press.