Grundlagen der Mathematik

Inhaltsverzeichnis

1 Formale Sprachen

1.1 Abstrakte Syntaxbäume

Wenn ein mathematischer Ausdruck syntaktisch analysiert wurde, kann das Ergebnis dieser Analyse als ein sogenannter *abstrakter Syntaxbaum*, kurz *AST* für engl. *abstract syntax tree* dargestellt werden.

Dem Ausdruck »$2x$« ordnen wir z. B. den AST

```
(*,2,x)
```

zu und »$2x+4$« ist dementsprechend

```
(+,(*,2,x),4).
```

Hier ein paar Beispiele:

1. $f(x)$ wird zu `(f,x)`,
2. $f(x,y)$ wird zu `(f,x,y)`,
3. x^2+a wird zu `(+,(^,x,2),a)`,
4. $2(a+b)$ wird zu `(*,2,(+,a,b))`,
5. $a+b+c$ wird zu `(+,(+,a,b),c)`,
6. $a+b+c$ wird zu `(+,a,b,c)`,
7. $x^2=x$ wird zu `(=,(^,x,2),x)`,
8. $A\wedge B$ wird zu `(and,A,B)`.

Bei der Darstellung von abstrakten Syntaxbäumen im Computer gibt es zunächst zwei Varianten. In Lisp wird der AST zum Ausdruck »$f(x,y)$« als verkettete Liste

```
(f x y)
```

dargestellt. Hier kann nämlich mit `first` (auch `car` genannt) der Funktionsbezeichner `f` erhalten werden und mit `rest` (auch `cdr` genannt) die Liste der Argumente.

Bei der Verwendung von dynamischen Feldern anstelle von verketteten Listen sind die Operationen `first` und `rest` eventuell etwas ineffizienter. Hier ist die Darstellung

```
["f", ["x", "y"]]
```

sinnvoller. In einer effizienten statisch typisierten Programmiersprache ließe sich natürlich ein Datentyp für AST-Knoten formulieren, der auf extra die Bedürfnisse, welche sich ergeben, zugeschnitten ist.

2 Lambda-Kalkül

2.1 Variablensubstitution

Angenommen, bei `t` und `u` handelt es sich um Ausdrücke bzw. abstrakte Syntaxbäume. Dann bedeutet die Schreibweise

```
t [x:=u]
```

die Ersetzung jedes Vorkommens der Variable `x` im Baum `t` durch den Baum `u`. Verwendet man keine abstrakten Syntaxbäume, so müssen dabei jedoch im Zusammenhang mit Vorrangregeln die Regeln der Klammersetzung beachtet werden. Z. B. ist

$$(a*b)\,[b:=x+y]=a*(x+y) \tag{2.1}$$

und nicht $a*x+y$.

Eine solche Variablensubstitution soll außerdem nur für Variablen durchgeführt werden, welche *frei*, d. h. ungebunden vorkommen. Man spricht daher von einer *freien Variablensubstitution*. Variablen können ausschließlich durch lambda-Ausdrücke gebunden werden. Was ein lambda-Ausdruck ist, wird später erklärt.

2.2 Lambda-Ausdrücke

Einer Liste wie `[a,b,c,d]` ordnen wir den AST

```
([],a,b,c,d)
```

zu. Somit gehört zur einelementigen Liste `[x]` der Baum `([],x)`. Zur leeren Liste `[]` gehört der Baum `([])`, wobei die runden Klammern obligatorisch sind. Der Begriff *Liste* ist gleichbedeutend mit *Tupel*.

Einen AST der Form

```
(lambda, ([], x), t)
```

bzw. einen Ausdruck

$$\lambda([x],t) \tag{2.2}$$

wobei `t` ein beliebiger AST ist, wollen wir als lambda-Ausdruck bezeichnen. Anstelle von »$\lambda([x],t)$« schreiben wir kürzer »$|x|\ t$«. Alternative Schreibweisen sind »$\lambda x.\,t$« oder »$x\mapsto t$«.

Dabei verlangt man nun, dass $|x|$ schwächer bindet als alle anderen Operationen und rechtsassoziativ ist.

Rechtsassoziativ bedeutet, dass die Klammerung immer auf der rechten Seite gesetzt wird. Z. B. ist

$$|x|\ |y|\ x+y = |x|\ (|y|\ x+y). \tag{2.3}$$

Ein lambda-Ausdruck lässt sich nun auf einen AST *applizieren*. Man definiert

$$(|x|\ t)(u)\ :=\ (t\ [x := u]). \tag{2.4}$$

Z. B. ist

$$(|x|\ x^2)(4) = 4^2 = 16 \tag{2.5}$$

und

$$\begin{aligned} &(|x,y|\ x\cdot y)(a+b, c+d) \\ &= (x\cdot y)\ [x := a+b,\ y := c+d] \\ &= (a+b)(c+d). \end{aligned} \tag{2.6}$$

Applikation ist linksassoziativ. D. h. es gilt

$$f(x)(y) = (f(x))(y). \tag{2.7}$$

Man kann nun die freie Variablensubstitution präzise definieren. Für Ausdrücke s, t, u und Variablen x, y gelten folgende Regeln:

(s1) $x\ [x := u] = u$.
(s2) $y\ [x := u] = y$ wenn $x \neq y$.
(s3) $s(t)\ [x := u] = (s[x := u])(t[x := u])$.
(s4) $(|x|\ t)\ [x := u] = |x|\ t$.
(s5) $(|y|\ t)\ [x := u] = |y|\ (t[x := u])$ wenn $x \neq y$ und $\mathrm{FV}(u)$ disjunkt zu $\mathrm{BV}(|y|\ t)$ ist.

Mit $\mathrm{FV}(t)$ bezeichnet man die Menge der *freien Variablen* von t, engl. *free variables*. Dieser Operator ist wie folgt definiert:

1. $\mathrm{FV}(x) = \{x\}$ für jede Variable x.
2. $\mathrm{FV}(s(t)) = \mathrm{FV}(s) \cup \mathrm{FV}(t)$.
3. $\mathrm{FV}(|x|\ t) = \mathrm{FV}(t) \setminus \{x\}$.

Mit $\mathrm{BV}(t)$ bezeichnet man die Menge der *gebundenen Variablen* von t, engl. *bounded variables*. Dieser Operator ist wie folgt definiert:

1. $\mathrm{BV}(x) = \{\}$ für jede Variable x.
2. $\mathrm{BV}(s(t)) = \mathrm{BV}(s) \cup \mathrm{BV}(t)$.
3. $\mathrm{BV}(|x|\ t) = \mathrm{BV}(t) \cup \{x\}$.

Entsprechend können wir noch $\mathrm{AV}(t)$, die Menge aller Variablen von t definieren:

1. $\mathrm{AV}(x) = \{x\}$ für jede Variable x.
2. $\mathrm{AV}(s(t)) = \mathrm{AV}(s) \cup \mathrm{AV}(t)$.
3. $\mathrm{AV}(|x|\ t) = \mathrm{AV}(t) \cup \{x\}$.

Außerdem definieren wir noch $\mathrm{PV}(t)$, die Menge der in Ausdrücken präsenten Variablen, wie folgt:

1. $\mathrm{PV}(x) = \{x\}$ für jede Variable x.
2. $\mathrm{PV}(s(t)) = \mathrm{PV}(s) \cup \mathrm{PV}(t)$.
3. $\mathrm{PV}(|x|\ t) = \mathrm{PV}(t)$.

Man beachte dass eine Variable in einem Ausdruck sowohl frei als auch gebunden vorkommen kann. Mit anderen Worten: Eine disjunkte Zerlegung von $\mathrm{AV}(t)$ in $\mathrm{FV}(t)$ und $\mathrm{BV}(t)$ ist nicht immer möglich.

Nun gibt es auch lambda-Ausdrücke mit mehr als einem Argument. Wir können nun

$$(|x,y|\ t) := (|x|\ |y|\ t) \tag{2.8}$$

definieren. Man bezeichnet eine solche Umformung als *Currying* oder *Schönfinkeln*.

Bei mehr als einem Argument muss man in solchen Fällen aufpassen, wo die selben Variablen sowohl frei als auch gebunden vorkommen. Der Ausdruck

$$(|x,y|\ x+y)(y,x) \tag{2.9}$$

unterscheidet sich z. B. von

$$((x+y)[x := y])[y := x]. \tag{2.10}$$

Hier müssen beide Substitutionen »gleichzeitig« vorgenommen werden. Präziser gesagt müssen die gebundenen Variablen vor der Substitution umbenannt werden. Betrachten wir nämlich den geschönfinkelten Ausdruck

$$(|x|\ |y|\ x+y)(y)(x). \tag{2.11}$$

Hier erhält man nach Definition nun

$$((|y|x+y)[x := y])(x), \tag{2.12}$$

aber die Substitution kann wegen Regel (s5) nicht ausgeführt werden, denn $\mathrm{FV}(y) = \{y\}$ ist nicht disjunkt zu $\mathrm{BV}(|y|\ x+y) = \{y\}$. Benennt man die gebundene Variable y nun in a um, so ergibt sich

$$\begin{aligned} &((|a|\ x+a)[x := y])(x) \\ &= (|a|\ y+a)(x) = y+x. \end{aligned} \tag{2.13}$$

Wir hatten gesagt, Variablen können ausschließlich durch lambda-Ausdrücke gebunden werden. Variablenbindungen ohne lambda-Bindung müssen immer in solche mit lambda-Bindung umformuliert werden. Z. B. ist

$$\sum_{k=m}^{n} a_k = \mathrm{sum}(m, n, |k|\ a_k) \tag{2.14}$$

und

$$\int_a^b f(x)\,\mathrm{d}x = \mathrm{integral}(a, b, |x|\ f(x)). \tag{2.15}$$

Für die Quantoren der Prädikatenlogik ergibt sich:

$$\forall x{\in}M\,[P(x)] = \mathrm{all}(M, |x|\ P(x)), \tag{2.16}$$
$$\exists x{\in}M\,[P(x)] = \mathrm{any}(M, |x|\ P(x)). \tag{2.17}$$

Das bedeutet beim Allquantor z. B., dass der Baum

```
(all, (in,x,M), (P,x))
```

zum Baum

```
(all, M, (lambda, ([],x), (P,x)))
```

umformuliert wird.