Programmentwicklung durch Beweistransformation: Das Maximalsegmentproblem

Helmut Schwichtenberg

12. Dezember 1996

Es ist gut bekannt, daß man aus einem konstruktiven Beweis eines Existenzsatzes ein Programm zur Berechnung des als existent nachgewiesenen Objekts extrahieren kann. Diese Art der Programmierung bezeichnen Bates und Constable in [1] als "very high level programming". Sie ermöglicht es, den bewährten mathematisch-logischen Formalismus als Programmiersprache zu benutzen; auf diese Weise werden nachträgliche Korrektheitsbeweise überflüssig. Der höhere Aufwand bei der Erstellung von Programmen wird dadurch erträglicher, daß man Beweise von reinen Allsätzen nicht zu formalisieren braucht, falls man ihre Richtigkeit auf andere Weise eingesehen hat.

Wir wollen hier anhand eines einfachen Beispiels einen weiteren Aspekt dieser Art von Programmiersprache demonstrieren, nämlich die Möglichkeit, allgemeine Programme an spezielle Situationen anzupassen und sie dabei – auch extensional – zu verändern. Daß dies möglich ist, wurde zuerst von Goad in seiner Dissertation [2] bemerkt.

Unser Beispiel ist ein einfaches, aber reales Programmierproblem, das sogenannte Maximalsegmentproblem aus [1]: Zu einer gegebenen Folge $x_0, x_1, \ldots, x_n$ ganzer Zahlen ist ein maximales Segment zu finden, also eine durch Indizes i und k mit $i \leq k$ bestimmte Teilfolge aufeinander folgender Glieder, so daß die Summe $x_i + \cdots + x_k$ maximal ist.

Zunächst formulieren wir die Behauptung etwas allgemeiner und verwenden anstelle von $x_i + \cdots + x_k$ ein beliebiges Maß $\mathrm{seg}(i,k)$ für $x_i, \ldots, x_k$, also eine Abbildung $\mathrm{seg}\colon \mathbb{N} \times \mathbb{N} \to X$, wobei X eine beliebige (partiell) geordnete Menge ist. Gesucht sind dann i, k so daß $\mathrm{seg}(i,k)$ maximal in X ist. Ein Beispiel ist etwa $x_i \in \mathbb{Q}^+$ und $\mathrm{seg}(i,k) := \prod_{i \leq j \leq k} x_j$.

Natürlich kann man dieses Problem einfach durch Probieren aller Möglichkeiten lösen; dies sind $O(n^2)$ viele. Dieser trivialen Lösung entspricht auch der im folgenden angegebene Beweis für die allgemeine Behauptung. Genauer heißt dies: Aus dem konstruktiv geführten Beweis kann man einen Algorithmus ablesen, der dem simplen Durchprobieren entspricht.

Wir wollen dann anschließend zeigen, daß für unser konkretes Problem mit der Summe $x_i + \cdots + x_k$ als Maß der Beweis sich vereinfachen läßt, indem man an einer geeigneten Stelle die Monotonie der Summe verwendet. Aus diesem vereinfachten Beweis läßt sich ein besserer

Algorithmus ablesen, der nur noch $O(n)$ viele Schritte benötigt. Wir haben also hier durch Anpassen eines allgemeinen Beweises an eine spezielle Situation erreicht, daß der aus dem Beweis abzulesende Algorithmus wesentlich verbessert wurde.

Wir skizzieren zuerst den allgemeinen (und trivialen) Beweis. Zu zeigen ist

Spezifikation.

$$\forall n \exists i, k(i \leq k \leq n \wedge \forall i', k'[i' \leq k' \leq n \to \mathrm{seg}(i', k') \leq \mathrm{seg}(i, k)]).$$

Beweis durch Induktion über n. Der Induktionsanfang ist trivial, und im Schritt bestimmt man zunächst ein maximales Endsegment von $x_0, \ldots, x_n, x_{n+1}$ und bildet dann das Maximum von diesem maximalen Endsegment und dem nach IH gegebenen maximalen Segment. $\square$

Die erste Feststellung ist hier, daß man sowieso neben dem gesuchten maximalen Segment auch noch das maximale Endsegment berechnen muß. Es liegt deshalb nahe, die Spezifikation so zu erweitern, daß neben dem maximalen Segment auch noch das maximale Endsegment zu finden ist.

Erweiterte Spezifikation. *Für alle n gilt*

$$\exists i, k(i \leq k \leq n \wedge \forall i', k'[i' \leq k' \leq n \to \mathrm{seg}(i', k') \leq \mathrm{seg}(i, k)]) \tag{1}$$

$$\exists j{\leq}n \forall j'{\leq}n(\mathrm{seg}(j', n) \leq \mathrm{seg}(j, n)) \tag{2}$$

Beweis durch Induktion über n. *Fall* 0. Wähle $i = k = 0$. *Fall* $n + 1$. Nach IH haben wir bereits i_n, k_n sowie j_n.

Wir wollen zunächst j_{n+1} konstruieren, haben also zu zeigen

$$\exists j{\leq}n{+}1 \forall j'{\leq}n{+}1(\mathrm{seg}(j', n + 1) \leq \mathrm{seg}(j, n + 1)). \tag{3}$$

Da uns j_n hier offensichtlich nicht weiterhilft, verallgemeinern wir die Behauptung (3) so, daß wir sie durch eine Nebeninduktion beweisen können, nämlich zu

$$\forall m{\leq}n{+}1 \exists \ell{\leq}m \forall \ell'{\leq}m(\mathrm{seg}(\ell', n + 1) \leq \mathrm{seg}(\ell, n + 1)). \tag{3^+}$$

Den Beweis führen wir durch Nebeninduktion nach m, wobei n als Parameter festgehalten wird. *Fall* 0. Wähle $\ell = 0$. *Fall* $m + 1$. Nach NIH haben wir bereits ein ℓ_m. Ist $\mathrm{seg}(\ell_m, n) < \mathrm{seg}(m + 1, n)$, so sei $\ell_{m+1} := m + 1$; andernfalls setzen wir $\ell_{m+1} := \ell_m$. – Damit ist (3^+) bewiesen, und mit $m := n + 1$ können wir unser gesuchtes j_{n+1} definieren als $j_{n+1} := \ell_{n+1}$.

Jetzt können wir i_{n+1}, k_{n+1} konstruieren. Ist $\mathrm{seg}(j_{n+1}, n{+}1) < \mathrm{seg}(i_n, k_n)$, so nehmen wir das alte maximale Segment und setzen $i_{n+1} := i_n$, $k_{n+1} := k_n$; andernfalls ist das maximale Endsegment das maximale Segment und wir setzen $i_{n+1} := j_{n+1}$, $k_{n+1} := n + 1$. $\square$

Offenbar ist der diesem Beweis entsprechende Algorithmus quadratisch.

Wir wollen jetzt zeigen, wie man bei zusätzlichen Kenntnissen über die Eingangsdaten den Beweis vereinfachen kann. In unserer Ausgangssituation galt

$$(x_i + \cdots + x_k \leq x_j + \cdots + x_k) \to (x_i + \cdots + x_k + x_{k+1} \leq x_j + \cdots + x_k + x_{k+1}).$$

Wir betrachten deshalb den Spezialfall

$$\forall i, j, k[\mathrm{seg}(i, k) \leq \mathrm{seg}(j, k) \to \mathrm{seg}(i, k+1) \leq \mathrm{seg}(j, k+1)]. \tag{$*$}$$

Diese Zusatzannahme ermöglicht es uns, die Existenz (3) eines maximalen Endsegments direkter zu beweisen, und zwar

- ohne Verwendung der Verallgemeinerung (3^+) und der zu ihrem Beweis nötigen Nebeninduktion, aber
- mit Hilfe der IH – also j_n – und der Fallunterscheidung $\mathrm{seg}(j_n, n+1) < \mathrm{seg}(n+1, n+1)$ bzw. $\geq$.

Diesen durch die Zusatzannahme ($*$) ermöglichten verkürzten Beweis wollen wir jetzt genauer durchführen. Dazu modifizieren wir den obigen Beweis wie eben beschrieben.

Verkürzter Beweis. Zu zeigen ist wieder (1) und (2). Verwendet wird Induktion über n. *Fall* 0. Wähle $i = k = 0$. *Fall* $n + 1$. Nach IH haben wir bereits i_n, k_n sowie j_n.

Wir wollen zunächst j_{n+1} konstruieren, haben also zu zeigen

$$\exists j{\leq}n{+}1 \forall j'{\leq}n{+}1 (\mathrm{seg}(j', n+1) \leq \mathrm{seg}(j, n+1)). \tag{3}$$

Die IH liefert uns für beliebige $j' \leq n$

$$\mathrm{seg}(j', n) \leq \mathrm{seg}(j_n, n)$$

und deshalb aufgrund der Zusatzannahme ($*$) auch

$$\mathrm{seg}(j', n+1) \leq \mathrm{seg}(j_n, n+1). \tag{4}$$

Ist nun $\mathrm{seg}(j_n, n{+}1) < \mathrm{seg}(n{+}1, n{+}1)$, so sei $j_{n+1} := n{+}1$. Dann haben wir für beliebiges $j' \leq n + 1$ wie gewünscht stets $\mathrm{seg}(j', n+1) \leq \mathrm{seg}(j_{n+1}, n+1)$: im Fall $j' \leq n$ folgt dies aus (4) und der Fallunterscheidungsannahme, und im Fall $j' = n + 1$ ist es trivial.

Ist andererseits $\mathrm{seg}(n+1, n+1) \leq \mathrm{seg}(j_n, n+1)$, so setzen wir $j_{n+1} := j_n$. Dann haben wir für beliebiges $j' \leq n + 1$ wie gewünscht stets $\mathrm{seg}(j', n+1) \leq \mathrm{seg}(n+1, n+1)$: im Fall $j' \leq n$ ist dies gerade (4), und im Fall $j' = n + 1$ ist es die Fallunterscheidungsannahme.

Jetzt können wir wie vorher i_{n+1}, k_{n+1} konstruieren. □

Offenbar ist der diesem Beweis entsprechende Algorithmus linear.

Wir wollen an dieser Stelle kurz die Frage diskutieren, ob und inwieweit diese Beweistransformation automatisierbar ist. Ein vollautomatisches Verfahren ohne Hilfe durch den Benutzer scheint mir kaum erreichbar zu sein. Andererseits sollte es als Hilfe genügen, daß man die folgenden aus der informalen Analyse des Problems offensichtlichen Hinweise gibt.

1. Zu ersetzen ist der Beweisteil, der im Induktionsschritt die Existenz eines maximalen Endsegments liefert, also der zu (3) führt.

2. Der neu zu führende, kürzere Beweis von (3) soll – neben der Zusatzannahme $(*)$ und der IH – die Fallunterscheidung $\mathrm{seg}(j_n, n+1) < \mathrm{seg}(n+1, n+1)$ bzw. $\geq$ verwenden.

In unserem Beweissystem Minlog haben wir diese Fallstudie implementiert, und zwar mit folgendem Ergebnis. Aus dem Originalbeweis und dem unter der Zusatzannahme verkürzten Beweis haben wir Programme extrahiert, die die erwartete Form haben: der Originalbeweis liefert einen quadratischen und der verkürzte Beweis einen linearen Algorithmus. Ferner haben wir verifiziert, daß sich mit den angegebenen Hilfen der verkürzte Beweis aus dem Originalbeweis auch automatisch gewinnen läßt.

Literatur

[1] Joseph L. Bates and Robert L. Constable. Proofs as programs. *ACM Transactions on Programming Languages and Systems*, 7(1):113–136, January 1985.

[2] Christopher Alan Goad. *Computational uses of the manipulation of formal proofs.* PhD thesis, Stanford University, August 1980. Stanford Department of Computer Science Report No. STAN–CS–80–819.