Problemreduktion durch Transformation am Beispiel des Erweiterten Euklidschen Algorithmus

Wolfgang Windsteiger

JKU Linz, A-4040 Linz, Austria

Kurzfassung

Transformation beschreibt im Wesentlichen die *algorithmische Rückführung* eines Problems auf ein einfacheres Problem durch sogenannte Transformationsfunktionen. Ist für das einfachere Problem ein Lösungsalgorithmus bekannt, so bekommen wir dadurch automatisch auch einen Lösungsalgorithmus für das ursprüngliche Problem. Wir beschreiben die Problemlösetechnik der "Reduktion durch Transformation" anhand des Erweiterten Euklidschen Algorithmus für ganzzahlige Eingabewerte durch Rückführung auf den Erweiterten Euklidschen Algorithmus für natürliche Zahlen.

1 Einleitung

Problemreduktion durch Transformation ist eine wichtige Problemlösestrategie, vor allem—aber nicht nur—in der Mathematik. Der Hauptnutzen einer Transformation ist die Wiederverwendung von bekanntem Wissen. In einer algorithmischen Betrachtung der Mathematik führt Reduktion durch Transformation auf die Wiederverwendbarkeit von bekannten mathematischen Algorithmen. Das Analogon in der Welt des Programmierens ist der Unterprogrammaufruf.

In dieser Arbeit wollen wir die mathematischen Hintergründe der Reduktion durch Transformation noch einmal genauer beleuchten. Das mathematische Wissen hinter einer Transformation eines Problems P auf ein einfacheres Problem $\overline{P}$ ist in [AlgMeth04] als abstraktes Schema der Form

$$\textbf{wenn } \overline{P}_{\mathrm{pre}[x],y} \textbf{ dann } P_{x,\mathrm{post}[y]} \tag{1}$$

eingeführt, wobei hier $P_{x,y}$ für ein Problem mit Input x und Output y steht. Das Problem P mit Input x lässt sich demnach auf das einfachere Problem $\overline{P}$ mit "vorbehandeltem Input" $\mathrm{pre}[x]$ zurückführen. Die Lösung des vereinfachten Problems y muss dann mit einer Operation post "nachbehandelt" werden, sodass schlussendlich $\mathrm{post}[y]$ als Lösung des Originalproblems P betrachtet werden kann. Ist weiters mit $\overline{g}$ ein Lösungsalgorithmus für $\overline{P}$ bekannt, so ist sofort auch ein Lösungsalgorithmus für P bekannt, nämlich $g[x] := \mathrm{post}[\overline{g}[\mathrm{pre}[x]]]$.

Wir können nun verschiedene Aspekte dieses Ansatzes betrachten:

- Bei bekannter Reduktion, d.h. P, $\overline{P}$, pre und post sind *bekannt*, kann die Korrektheit des Algorithmus g bewiesen werden, indem wir (1) beweisen.
- Die Art der Reduktion ist *nicht bekannt*, d.h. mindestens eines von $\overline{P}$, pre oder post sind unbekannt, somit ist auch g unbekannt. Wir können nun versuchen, einen Lösungsalgorithmus g zu erfinden, indem wir einen Beweis von (1) beginnen und im Zuge des Beweises versuchen, $\overline{P}$, pre oder post gerade so zu wählen, dass der Beweis erfolgreich beendet werden kann.
- Sobald wir mathematischem Wissen der Form (1) begegnen, muss uns ins Auge springen, dass wir damit die Lösung des Problems P auf die Lösung des Problems $\overline{P}$ zurückführen können.

Der Erweiterte Euklidsche Algorithmus (EEA) löst das Problem, zu zwei gegebenen natürlichen Zahlen x_1 und x_2 drei Zahlen y_1, y_2 und y_3 so zu finden, dass y_3 der größte gemeinsame Teiler (ggT) von x_1 und x_2 ist und dass gilt $x_1\, y_1 + x_2\, y_2 = y_3$. Eine sinnvolle Verallgemeinerung dieser Problemstellung betrachtet ganzzahlige Eingabewerte. Anstelle der Entwicklung eines *neuen Algorithmus* für die verallgemeinerte Problemstellung werden wir zeigen, dass die Problemstellung für ganzzahligen Input durch Transformation auf die Problemstellung für natürliche Zahlen als Eingabewerte reduziert werden kann, was automatisch einen Lösungsalgorithmus für ganzzahligen Input liefert.

Wir werden in Abschnitt 2 die Problemstellung anhand eines Beispiels veranschaulichen. In Abschnitt 3 werden wir die in [AlgMeth04] abstrakt gezeigte Problemreduktion im Beispiel des Erweiterten Euklidschen Algorithmus durchführen und deren Korrektheit beweisen. Abschnitt 4 beinhaltet die konkrete Umsetzung des in Abschnitt 3 erhaltenen Algorithmus in einer Mathematica Implementierung.

2 Ein Beispiel

Eine mögliche Realisierung des Erweiterten Euklidschen Algorithmus für natürliche Zahlen als Eingabewerte in Mathematica könnte etwa wie folgt aussehen.

```
ggTℕ[{x1_Integer?NonNegative, x2_Integer?NonNegative}] := ggTℕ[{1, 0, x1}, {0, 1, x2}]
```

```
ggTℕ[X1 : {_, _, _}, {_, _, 0}] := X1
```

```
ggTℕ[X1 : {_, _, x1_Integer?NonNegative}, X2 : {_, _, x2_Integer?Positive}] :=
  With[{q = Quotient[x1, x2]},
    ggTℕ[X2, X1 − q ∗ X2]]
```

```
ggTℕ[{203, 112}]
```

```
{5, −9, 7}
```

Ein Korrektheitsbeweis für diesen Algorithmus ist beispielsweise in [LinAlg04] zu finden. Natürlich gilt für die Ausgabewerte

$$203 * 5 + 112 * (-9) = 7\ . \tag{2}$$

Wollen wir nun analog dazu den ggT von -203 und 112 und eine entsprechende Darstellung als Linearkombination der Eingabewerte berechnen, so gehen wir dazu wie folgt vor: der ggT von -203 und 112 ist ebenfalls 7, und wir können den ggT darstellen als

$$7 \underset{\uparrow\,(2)}{=} 203 * 5 + 112 * (-9) = -203 * (-5) + 112 * (-9)\ .$$

Die Vermutung liegt nahe, dass wir also in den Koeffizienten der Linearkombination nur das Vorzeichen abhängig vom Vorzeichen der Eingabewerte anpassen müssen. Ein Beispiel ist natürlich kein Beweis, wir werden daher diese Idee exakt verifizieren müssen. Es zeigt sich, dass das Schema der Reduktion durch Transformation genau diese Situation widerspiegelt, die Korrektheitsaussage orientiert sich daher an (1).

3 Der Erweiterte Euklidsche Algorithmus

Wir verwenden die Notation aus [AlgMeth04] für exakte Problemspezifikationen und Problemreduktionen. Wir betrachten nun das Problem $\overline{P}_{\langle x_1,x_2\rangle,\langle y_1,y_2,y_3\rangle}$ des Berechnens des ggT zweier natürlicher Zahlen mitsamt den Koeffizienten der Lineardarstellung des ggT.

Gegeben: x_1, x_2
sodass: $x_1, x_2 \in \mathbb{N}_0$.
Gesucht: y_1, y_2, y_3
sodass: $y_3 \in \mathbb{N}_0 \wedge y_1, y_2 \in \mathbb{Z} \wedge \text{ist-ggT}[y_3, x_1, x_2] \wedge x_1\, y_1 + x_2\, y_2 = y_3$.

Die analoge Problemstellung über den ganzen Zahlen (wir schreiben dafür $P_{\langle x_1,x_2\rangle,\langle y_1,y_2,y_3\rangle}$) lautet dann:

Gegeben: x_1, x_2
sodass: $x_1, x_2 \in \mathbb{Z}$.
Gesucht: y_1, y_2, y_3
sodass: $y_3 \in \mathbb{N}_0 \wedge y_1, y_2 \in \mathbb{Z} \wedge \text{ist-ggT}[y_3, x_1, x_2] \wedge x_1\, y_1 + x_2\, y_2 = y_3$.

Wie in [AlgMeth04] beschrieben, steht eine Problemspezifikation $P_{x,y}$ für die Aussage "wenn x die Eingabebedingung des Problems erfüllt, dann erfüllen x und y die Ausgabebedingung des Problems", somit steht im konkreten Beispiel $\overline{P}_{\langle x_1,x_2\rangle,\langle y_1,y_2,y_3\rangle}$ für

$$x_1, x_2 \in \mathbb{N}_0 \Rightarrow y_3 \in \mathbb{N}_0 \wedge y_1, y_2 \in \mathbb{Z} \wedge \text{ist-ggT}[y_3, x_1, x_2] \wedge x_1\, y_1 + x_2\, y_2 = y_3$$

bzw. $P_{\langle x_1,x_2\rangle,\langle y_1,y_2,y_3\rangle}$ für

$$x_1, x_2 \in \mathbb{Z} \Rightarrow y_3 \in \mathbb{N}_0 \wedge y_1, y_2 \in \mathbb{Z} \wedge \text{ist-ggT}[y_3, x_1, x_2] \wedge x_1\, y_1 + x_2\, y_2 = y_3 \ .$$

Wie in Abschnitt 2 gezeigt, vermuten wir die für die Problemreduktion notwendigen Operationen pre und post wie folgt:

$$\text{pre}[\langle x_1, x_2\rangle] := \langle |x_1|, |x_2|\rangle \quad \text{bzw.}$$

$$\text{post}[\langle x_1, x_2\rangle, \langle y_1, y_2, y_3\rangle] := \langle \text{Sign}[x_1]\, y_1, \text{Sign}[x_2]\, y_2, y_3\rangle \ .$$

Dabei bezeichnet Sign die bekannte Signum-Funktion, und die Multiplikation der y_i mit $\text{Sign}[x_i]$ entspricht genau der in Abschnitt 2 beschriebenen Anpassung der Vorzeichen der Koeffizienten der Linearkombination abhängig vom Vorzeichen der Eingabewerte. Wir sehen in diesem Beispiel, dass die Nachbehandlung des Resultats in post auch vom Input x_1 und x_2 abhängig ist, d.h. wir müssen das in (1) gezeigte Schema für eine Reduktion durch Transformation verallgemeinern zu

$$\textbf{wenn}\ \overline{P}_{\text{pre}[x],y}\ \textbf{dann}\ P_{x,\text{post}[x,y]} \ . \qquad (3)$$

Das Schema für einen Lösungsalgorithmus, der durch Reduktion durch Transformation entsteht, ist dann entsprechend $g[x] := \text{post}[x, \overline{g}[\text{pre}[x]]]$. Wir können nun die *Korrektheit der Reduktion* des ggT-Problems für ganze Zahlen auf das ggT-Problem für natürliche Zahlen durch eine mathematisch beweisbare Aussage formulieren.

Satz 1 (Korrektheit des EEA über $\mathbb{Z}$)

$$\forall_{x_1,x_2,y_1,y_2,y_3} (|x_1|, |x_2| \in \mathbb{N}_0 \Rightarrow y_3 \in \mathbb{N}_0 \wedge y_1, y_2 \in \mathbb{Z} \wedge \text{ist-ggT}[y_3, |x_1|, |x_2|] \wedge |x_1|\, y_1 + |x_2|\, y_2 = y_3) \Longrightarrow$$
$$(x_1, x_2 \in \mathbb{Z} \Rightarrow y_3 \in \mathbb{N}_0 \wedge y_1 \text{ Sign}[x_1],$$
$$y_2 \text{ Sign}[x_2] \in \mathbb{Z} \wedge \text{ist-ggT}[y_3, x_1, x_2] \wedge x_1 \text{ Sign}[x_1]\, y_1 + x_2 \text{ Sign}[x_2]\, y_2 = y_3)$$

Beweis: Wir wählen zuerst x_1, x_2, y_1, y_2, y_3 b.a.f. und nehmen an

$$|x_1|, |x_2| \in \mathbb{N}_0 \Rightarrow y_3 \in \mathbb{N}_0 \wedge y_1, y_2 \in \mathbb{Z} \wedge \text{ist-ggT}[y_3, |x_1|, |x_2|] \wedge |x_1|\, y_1 + |x_2|\, y_2 = y_3\,. \tag{4}$$

Zu beweisen ist nun

$$x_1, x_2 \in \mathbb{Z} \Rightarrow y_3 \in \mathbb{N}_0 \wedge y_1 \text{ Sign}[x_1],$$
$$y_2 \text{ Sign}[x_2] \in \mathbb{Z} \wedge \text{ist-ggT}[y_3, x_1, x_2] \wedge x_1 \text{ Sign}[x_1]\, y_1 + x_2 \text{ Sign}[x_2]\, y_2 = y_3\,,$$

daher nehmen wir $x_1, x_2 \in \mathbb{Z}$ an und zeigen

$$y_3 \in \mathbb{N}_0 \wedge y_1 \text{ Sign}[x_1], y_2 \text{ Sign}[x_2] \in \mathbb{Z} \wedge \text{ist-ggT}[y_3, x_1, x_2] \wedge x_1 \text{ Sign}[x_1]\, y_1 + x_2 \text{ Sign}[x_2]\, y_2 = y_3\,. \tag{5}$$

Wegen unserer Annahme $x_1, x_2 \in \mathbb{Z}$ ist $|x_1|, |x_2| \in \mathbb{N}_0$ immer wahr, daher ist die Annahme (4) aussagenlogisch äquivalent zu

$$\underbrace{y_3 \in \mathbb{N}_0}_{(6.1)} \bigwedge \underbrace{y_1, y_2 \in \mathbb{Z}}_{(6.2)} \bigwedge \underbrace{\text{ist-ggT}[y_3, |x_1|, |x_2|]}_{(6.3)} \bigwedge \underbrace{|x_1|\, y_1 + |x_2|\, y_2 = y_3}_{(6.4)}\,. \tag{6}$$

Um (5) zu beweisen, zeigen wir die die einzelnen Bestandteile der Konjunktion separat.
Beweis von $y_3 \in \mathbb{N}_0$: Klar wegen Annahme (6.1).
Beweis von $y_1 \text{ Sign}[x_1], y_2 \text{ Sign}[x_2] \in \mathbb{Z}$: Einfach wegen Annahme (6.2) und zusätzlichem Wissen über Sign (auf das wir hier nicht näher eingehen).
Beweis von ist-ggT$[y_3, x_1, x_2]$: Durch Equivalence Rewriting aus (6.3) mit zusätzlichem Wissen über ggT:

$$\forall_{g,a,b} \text{ ist-ggT}[g, a, b] \Leftrightarrow \text{ist-ggT}[g, |a|, |b|]\,. \tag{7}$$

Beweis von $x_1 \text{ Sign}[x_1]\, y_1 + x_2 \text{ Sign}[x_2]\, y_2 = y_3$ $(*)$: Wir wissen

$$\forall_x\, x \text{ Sign}[x] = |x|\,, \tag{8}$$

daher zeigen wir $(*)$ mit Equality Rewriting:

$$x_1 \text{ Sign}[x_1]\, y_1 + x_2 \text{ Sign}[x_2]\, y_2 \underset{\uparrow (8)}{=} |x_1|\, y_1 + |x_2|\, y_2 \underset{\uparrow (6.4)}{=} y_3\,.$$

□

4 Realisierung in einem Mathematica Programm

Wir können nun die in Abschnitt 3 eingeführten Funktionen pre und post als Mathematica Programme implementieren. Man beachte dabei, dass zwischen der mathematischen Formulierung in Abschnitt 3 und den lauffähigen Mathematica Programmen kaum ein Unterschied besteht, abgesehen von der Verwendung der Mengenklammern in Mathematica für Listen (=Tupel).

```
pre[{x1_, x2_}] := {Abs[x1], Abs[x2]}
```

```
post[{x1_, x2_}, {y1_, y2_, y3_}] := {Sign[x1] y1, Sign[x2] y2, y3}
```

Laut Problemreduktionsschema ist ein Lösungsalgorithmus für unser Problem der Berechnung des ggT zweier ganzer Zahlen mitsamt den Koeffizienten der Lineardarstellung des ggT gegeben durch

```
ggTℤ[X : {_Integer, _Integer}] := post[X, ggTℕ[pre[X]]]
```

```
ggTℤ[{−203, 112}]
```

```
{−5, −9, 7}
```

5 Zusammenfassung

Wir beschreiben in dieser Arbeit die Problemlösetechnik der Reduktion durch Transformation am Beispiel des Erweiterten Euklidschen Algorithmus für ganze Zahlen. Das Hauptaugenmerk liegt dabei auf dem engen Zusammenhang zwischen dem Korrektheitsbeweis der Transformation und der tatsächlichen Realisierung des Algorithmus in Form eines Computerprogramms. Die im Korrektheitsbeweis verwendeten Funktionen pre und post führen direkt zu einem Lösungsalgorithmus nach dem Algorithmenschema für Reduktion durch Transformation.

Alternativ dazu können wir das allgemeine Schema der Reduktion durch Transformation auch dazu verwenden, um bisher nicht bekannte Algorithmen zu entdecken. Anstatt *vorher* die konkreten Funktionen pre und post bekanntzugeben und dann den Korrektheitsbeweis zu führen, würden wir hier die beiden Funktionen zunächst unbekannt belassen. Im Laufe des Korrektheitsbeweises ergeben sich dann Bedingungen an die Funktionen pre und post, die für einen vollständigen Beweis notwendig sind. Aus diesen Bedingungen lassen sich meist Rückschlüsse über die konkrete Realisierung der Funktionen pre und post ziehen.

Literatur

[AlgMeth04] W.Windsteiger. *Algorithmische Methoden 1*. Vorlesungsskriptum JKU Linz und Tutoriumsunterlagen, Wintersemester 2004/05, 2004.

[LinAlg04] G.Pilz. *Lineare Algebra und Analytische Geometrie I*. Vorlesungsskriptum JKU Linz, Wintersemester 2004/05, 2004.