Reverse Polish Notation

Ergänzung zum Kapitel 5

Dies ist ein Ergänzungsartikel zum Buch *Automaten und Sprachen: Theoretische Informatik für die Praxis*, von Andreas Müller. Erschienen im Verlag Springer-Vieweg, ISBN 978-3-662-70145-4 (Softcover) und ISBN 978-3-662-70146-1 (eBook).

Website zum Buch: https://autospr.ch

Springer-Link: https://link.springer.com/book/10.1007/978-3-662-70146-1

Reverse Polish Notation

Andreas Müller

Zusammenfassung

Im Taschenrechner-Krieg in den Siebzigerjahren des zwanzigsten Jahrhunderts traten zwei ziemlich verschiedene Systeme für die Eingabe arithmetischer Ausdrücke gegeneinander an. Die einen Hersteller setzten auf eine Syntax möglichst nahe an der algebraischen Schreibweise, während Hewlett-Packard in seinen Tisch- und Taschenrechnern die umgekehrte polnische Notation (Reverse Polish Notation, RPN) verwendete.

1 Das Problem

Ein Taschenrechner wertet arithmetische Ausdrücke aus. Aufgrund seiner Größe kann er aber nur eine sehr einfache Benutzeroberfläche zur Verfügung stellen. Typischerweise stehen Tasten für die Eingabe von Ziffern, Dezimalpunkt und Operationszeichen zur Verfügung. Damit lassen sich bereits einfache arithmetische Ausdrücke eingeben. Die meisten einfachen Taschenrechner führen jedoch eine Operation aus, sobald klar ist, dass der zweite Operand vollständig ist. Gibt man zum Beispiel den Ausdruck

$$123 + 456 \times 789 \tag{1}$$

als

```
123+456×789
```

ein, wird die Summe 123 + 456 ausgewertet, sobald das Multiplikationszeichen eingegeben wird. Die nachfolgende Multiplikation wird also mit der Summe $123 + 456 = 579$ durchgeführt. Der Ausdruck (1) wird also als

$$(123 + 456) \times 789 \tag{2}$$

ausgewertet und erhält den Wert 456831, abweichend vom korrekten Wert 359907, der bei Anwendung der algebraischen Konvention, dass Multiplikationen vor Additionen ausgeführt werden müssen, gefunden wird. Ein solcher Taschenrechner ist also nicht geeignet für die zuverläßige Berechnung komplexer Ausdrücke, da die Gefahr besteht, dass sich der Benutzer nicht bewusst ist, ob der Taschenrechner die Operationsreihenfolge korrekt berücksichtigt.

Ein Teil der Schwierigkeit rührt daher, dass für die Auswertung komplexer Ausdrücke ausreichend Speicherplatz für Zwischenresultate bereitgestellt werden muss, den frühe Taschenrechner nicht zur Verfügung hatten. Dieses Problem wird noch verschärft, wenn der

Abbildung 1: Der Tischrechner HP9100A (1968, links) und der Taschenrechner HP-35 (1972, rechts) von Hewlett-Packard. Beide Rechner verwenden die umgekehrte polnische Notation als Eingabemethode.

Taschenrechner die Eingabe von Klammern erlaubt. Mit jeder öffnenden Klammer beginnt ein neuer Auswertungkskontext, in dem Speicherplatz für Zwischenresultate bereitgestellt werden muss. Im besten Fall kann der Taschenrechner also ein paar wenige Klammerebenen unterstützen. Damit entsteht das neue Problem, dass der Taschenrechner dem Benutzer irgendwie anzeigen können muss, ob er die Anzahl der zur Verfügung stehenden Klammerebenen ausgeschöpft hat.

Texas Instruments führte 1975 den programmierbaren Taschenrechner SR-52 ein, der das sogenannte *Algebraic Operating System* AOS™ [1] implementierte. Wegen beschränkter CPU-Resourcen konnten sich frühere Taschenrechner nur teilweise an die algebraischen Auswerteregeln für einen arithmetischen Ausdruck halten. Mit AOS™ konnte erstmals Klammern verarbeiten werden und der Operatorvorrang wurde eingehalten. Frühere Taschenrechner mussten also mit wesentlich einfacherer Hardware Rechnerfunktionen bereitstellen, die dem Benutzer trotzdem die korrekte Auswertung beliebig komplexer Ausdrücke ermöglichen.

2 Umgekehrte Polnische Notation

Im Jahre 1968 führte Hewlett-Packard den programmierbaren Tischrechner HP9100A ([3] und Abbildung 1) ein, der erstmals nicht nur die arithmetischen Grundoperationen beherrschte, sondern auch die üblichen transzendenten Funktionen wie Exponentialfunktion, Logarithmus und die trigonometrischen Funktionen. Für vollständige Syntaxanalyse von arithmetischen Ausdrücken wie beim AOS™ (Abschnitt 1) stand aber nicht ausreichend Speicher zur Verfügung.

Bei der üblichen algebraischen Notation stehen die Operationszeichen zwischen den Operanden, dies wird als *Infixnotation* bezeichnet. Werden die Operationen in einer der verbreiteten Programmiersprachen als Funktionen realisiert, zum Beispiel als `add(a, b)`, dann steht der Name der Funktion, `add`, vor den beiden Operationen `a` und `b`. In dieser Reihenfolge stehen die Operatoren vor den Operanden, man spricht von Prefixnotation. Sie

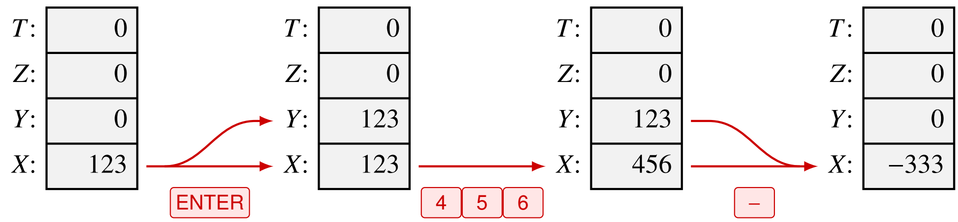

Abbildung 2: Bedienung des Stacks der umgekehrten polnischen Notation. Die ENTER-Taste kopiert den Inhalt von X nach Y und startet die Eingabe eines neuen Operanden. Die –-Taste löst die Subtraktion $Y - X$ aus.

wurde vom polnischen Mathematiker Jan Łukasiewicz als kompaktere Notation für die Logik und Arithmetik vorgeschlagen und heißt daher auch polnische Notation, *Łukasiewicz-Notation* und *Warschauer Notation*.

Beim HP9100A steht als Zwischenspeicher für Operanden ein Stapel mit drei Registern zur Verfügung, die als Z, Y und X bezeichnet werden. Die Operationen verwenden jeweils die letzten beiden Register. Sie werden ausgelöst, indem man die Operationstaste nach Eingabe der Operanden eingibt. Das Operationszeichen steht nicht zwischen den Operanden, sondern am Ende, man spricht von *Postfixnotation* oder auch *umgekehrte polnischer Notation* (UPN) oder *reverse polish notation* (RPN).

Als Trennzeichen für die Eingabe der Operanden steht die Taste ENTER zur Verfügung. Die Operanden 123 und 456 werden also durch die Tastenfolge

1 2 3 ENTER 4 5 6

eingegeben. Die Taste ENTER verschiebt den aktuellen Inhalt von X in das Register X und startet die Eingabe eines neuen Wertes im Register X. Falls Y bereits einen Wert enthält, dann wird der Inhalt nach Z kopiert. Man kann sich daher die Register auch als einen sehr kleinen Stack vorstellen. Die Eingabe ENTER fügt dem Stack ein neues Element hinzu, welches im Register X zu finden ist.

Da bei der Handrechnung von Subtraktionen der Subtrahend häufig unter dem Minuend geschrieben wird, wie auch bei Brüchen der Divisor unter dem Dividenden steht, ist es zweckmäßig, sich die Register so angeordnet vorzustellen. Im Gegensatz zur Darstellungsweise eines Stacks in Kapitel 7, bei der neue Elemente oben auf den Stack gelegt werden, wächst der Stack der Operanden nach unten, neue Elemente werden unten angefügt.

Der Taschenrechner HP-35 [2] war 1972 das erste Gerät im Taschenformat[1], welches die üblichen transzendenten Operationen bereitstellte und damit die Bedürfnisse eines technisch-wissenschaftlichen Benutzers abdecken konnte. Ursprünglich als ein Produkt für internen Gebrauch entwickelt, rechnete eine Marketingstudie mit einem möglichen Absatz von etwa 50000 Geräten. Tatsächlich würde über eine halbe Million Rechner verkauft, trotz des hohen Preises von 395 USD, was nach heutiger Kaufkraft etwa 2600 USD entsprechen würde. Auch der HP-35 verwendet die umgekehrte polnische Notation, allerdings mit vier Registern T, Z, Y und X.

[1] Angeblich sollen die Abmessungen des Geräts so gewählt worden sein, dass der Rechner in die Hemdtasche von Firmengründer Bill Hewlett passt. Die Nummer 35 geht darauf zurück, dass der Rechner 35 Tasten hat.

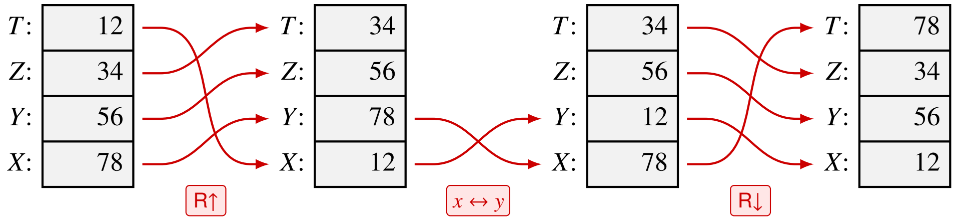

Abbildung 3: Erweiterte Stack-Operationen. Die Operation $x \leftrightarrow y$ vertauscht die Werte in den Registern X und Y. Die Operationen R↑ und R↓ vertauschen die Werte zyklisch nach oben bzw. unten. Diese Operation wird auch Rotation genannt.

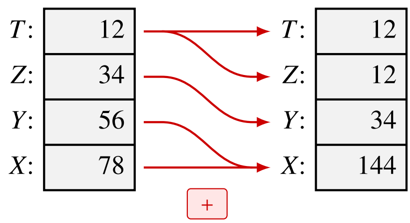

Abbildung 4: Eine zweistellige Operation wie die arithmetischen Operationen verknüpft die Werte in X und Y und legt das Resultat in X ab. Die darüberliegenden Wert werden ein Register nach unten kopiert. Der Wert in T bleibt erhalten.

Zusätzliche Funktionen (Abbildung 3) ermöglichen dem Benutzer, noch flexibler Nutzen aus den Registern des Stack zu ziehen. Die Operation $x \leftrightarrow y$ vertauscht den Inhalt der X- und Y-Register. Zugriff auf jeden beliebigen Wert des Stack kann durch Benutzer durch die Operationen R↑ und R↓, die die Werte auf im Stack zyklisch nach oben bzw. unten vertauschen oder rotieren.

Das T-Register hat noch eine weitere Funktion (Abbildung 4). Wenn die Werte im Stack als Folge einer arithmetischen Operation nach unten kopiert werden, bleibt der Wert im Register T erhalten. Das T-Register kann daher als Register für eine Konstanten verwendet werden, die immer wieder benötigt wird. Dies ist zum Beispiel nützlich beim Auswerten eines Polynoms. Das Polynom

$$p(x) = a_n x^n + a_{n-1} x^{n-1} + \cdots + a_2 x^2 + a_1 x + a_0$$

kann auch als

$$p(x) = ((\ldots(a_n x + a_{n-1})x + \cdots + a_2)x + a_1)x + a_0$$

geschrieben werden. Zur Auswertung wird der Stack zuerst mit den Tasten

$$x \;\boxed{\text{ENTER}}\;\boxed{\text{ENTER}}\;\boxed{\text{ENTER}}$$

mit dem Wert von x gefüllt. Durch die Tastenfolge

$$a_n \;\boxed{\times}\; a_{n-1} \;\boxed{+}\;\boxed{\times}\; a_{n-2} \;\boxed{+}\;\boxed{\times}\; \ldots\; a_2 \;\boxed{+}\;\boxed{\times}\; a_1 \;\boxed{+}\;\boxed{\times}\; a_0 \;\boxed{+}$$

wird der Wert des Polynoms berechnet, ohne dass x erneut eingegeben werden muss und ohne dass ein Zwischenresultat notiert werden muss.

Operationen mit nur einem Operanden wie der Vorzeichenwechsel oder der reziproke Wert $1/x$ verwenden das X-Register als Argument und legen das Resultat ebendort ab, ohne die übrigen Stackwerte zu verschieben. Spätere technisch-wissenschaftliche Rechner von Hewlett-Packard boten auch eine Funktion $\boxed{\to\mathsf{P}}$ für die Umrechung von kartesischen Koordinaten und Polarkoordinaten. Sie verwendet die Werte in X und Y als die Koordinaten x und y eines Punktes und berechnet den Polarwinkel φ im Register Y und den Radius r im Register X. Die Umkehrfunktion zu $\boxed{\to\mathsf{P}}$ ist $\boxed{\to\mathsf{R}}$.

Die Potemnzierungsoperation illustriert einen interessanten Konflikt der Vorstellung des umgekehrten Stack mit der Benutzererfahrung. Auf dem HP-35 steht die Taste $\boxed{x^y}$ zur Verfügung, welche den Wert im Register X mit dem Exponenten in Register Y potenziert. Dies passt zur Vorstellung, dass X der "untere" Wert ist, während Y "weiter oben" liegt. Es führt aber dazu, dass der Benutzer erst den Exponenten eingeben muss, dann erst die Basis. Die mathematische Notation präsentiert dem Benutzer aber erst die Basis und erst dann den Exponenten. Bei allen späteren Taschenrechnern wurde die Taste daher in $\boxed{y^x}$ umbenannt.

3 Die Auswertung des Syntaxbaumes mit RPN

In Abschnitt 5.3 wird erklärt, wie ein Syntaxbaum ausgewertet wird. Die umgekehrte polnische Notation definiert eine Aufgabenteilung zwischen dem Benutzer, der den Syntaxbaum des auszuwertenden Ausdrucks konstruiert und dem Rechner, der die Rechenoperationen ausführt (Abbildung 5). Jedesmal, wenn im Syntaxbaum eine Zahl angetroffen wird, wird diese als neues Element auf den Stack geschoben. Bei einem Knoten des Syntaxbaums, der einer arithmetischen Operation entspricht, muss nur noch die Rechenoperation ausgeführt werden, die Operanden befinden sich bereits auf dem Stack. Das Resultat der Operation wird das neue Element X auf dem Stack, steht also unmittelbar für die nächste Operation zur Verfügung.

Die Tatsache, dass auf dem Stack des Taschenrechners nur vier Ebenen zur Verfügung stehen, ist dabei ein leicht zu verschmerzender Kompromiss. Sollte ein Ausdruck so komplex geschachtelt sein, dass die vier Ebenen nicht ausreichen, dürfte es dem Benutzer auch schwer fallen, den Syntaxbaum zu überblicken. Derart komplizierte Ausdrücke können mit einem Taschenrechner nicht mehr zuverlässig ausgewertet werden und sollten mit einem Programm verarbeitet werden, welches beliebig komplexe Ausdrücke parsen kann, wie das in Abschnitt 7.3 beschriebene Calculator-Programm.

4 Weitere Produkte, die UPN einsetzen

Die Taschenrechner von Hewlett-Packard sind nicht die einzigen Produkte, die die umgekehrte polnische Notation verwenden. Die Seitenbeschreibungssprache PostScript ist Turing-vollständig und verwendet einen Stack für alle Funktionsaufrufe. Argumente werden auf den Stack geschoben, danach folgt der Funktionsaufruf. Die Operatoren `add`, `sub`, `mul` und `div` realisieren die arithmetischen Grundoperatoren. Für die Manipulation des

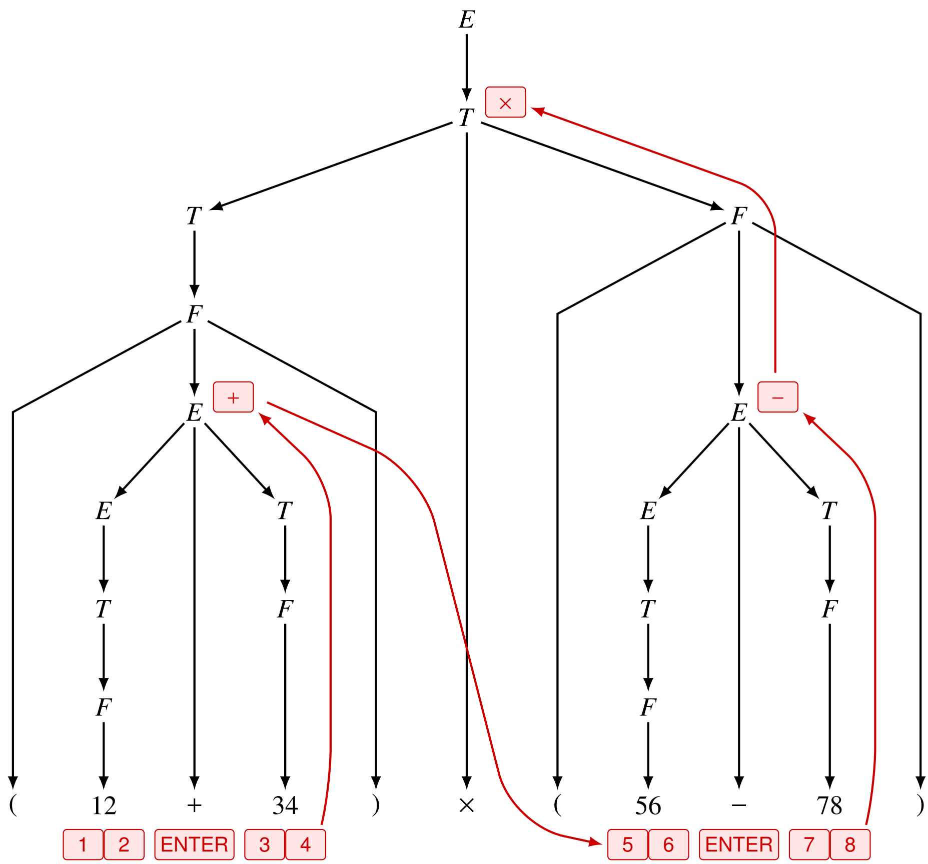

Abbildung 5: Syntaxbaum des Ausdrucks (12 + 34) × (56 − 78) in der Expression-Term-Factor-Grammatik. Zur Auswertung müssen die Operationen in der durch die roten Pfeile angedeuteten Reihenfolge eingegeben werden.

Stackinhalts stehen viele wie bei den HP-Taschenrechnern weitere Operatoren zur Verfügung, zum Beispiel **dup**, welcher wie ENTER den Wert zuoberst auf dem Stack dupliziert, `exch`, welcher wie $x \leftrightarrow y$ die obersten beiden Werte auf dem Stack vertauscht, und `roll`, welcher die obersten n Stackeinträge um i Elemente zyklisch vertauschen kann, und damit als Erweiterung von R↓ oder R↑ gesehen werden kann.

Die Programmiersprache FORTH entstand zunächst als ein integriertes System zur Steuerung eines Radioteleskops. Es erfreute sich dank der geringen Ansprüche an die Leistungsfähigkeit der Hardware in den Siebzigerjahren unter Astronomen einiger Beliebtheit. Forth verwendet einen Stack, die Operatoren `+`, `-`, `*` und `/` für die Arithmetik und Operatoren zur Manipulation mit den Namen `SWAP`, `DUP` und `ROT`. Forth erlangte einige Bedeutung als hardwareunabhängige Sprache für die plattformunabhängige Bootumgebung *Open Firmware*. Zusätzgeräte können dabei so eingebunden werden, dass sie den zur Bedienung des Gerätes nötigen Code als prozessorunabhängigen Forth-Code in

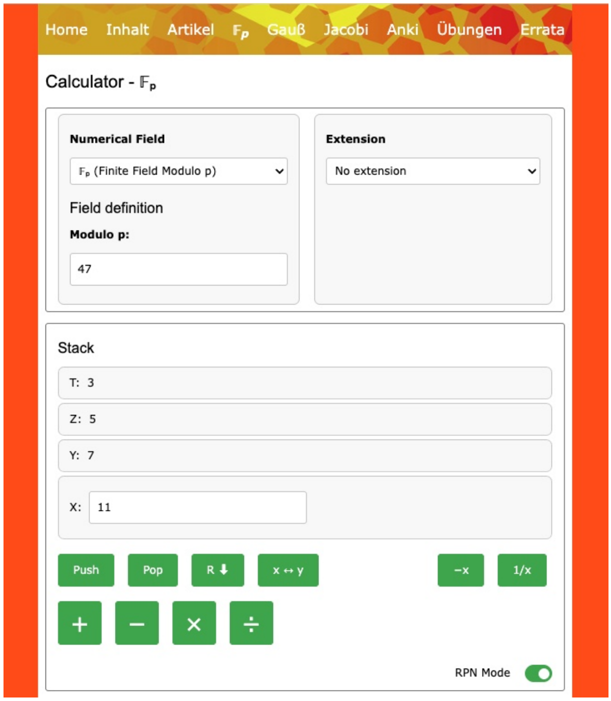

Abbildung 6: Rechner für die Körper $\mathbb{F}_p$ und $\mathbb{F}_p(\alpha)$ auf der Website `https://linalg.ch` bietet ein RPN-Interface an.

einem Festwertspeicher bereitstellen. Open Firmware verwendet diesen Code anstelle von separat zu installierenden Treibern. Open Firmware wurde auf SPARC-Rechnern von Sun Microsystems und später auch auf den PowerPC-Rechnern von Apple verwendet.

5 Ein Rechner für Arithmetik in $\mathbb{F}_p$

In der Kryptographie werden oft die endlichen Körper $\mathbb{R}_p$ oder die endlichen Erweiterungskörper $\mathbb{F}_{p^l} = \mathbb{F}_p(\alpha) = \mathbb{F}_p[X]/(m(X))$ verwendet, wobei $m(X)$ das Minimalpolynom von α ist. Während Addition, Subtraktion und Multiplikation in $\mathbb{F}_p$ einfach durchzuführen sind, verlangt die Division mach dem erweiterten euklidischen Algorithmus [4]. Für die Erweiterungskörpert $\mathbb{F}_p(\alpha)$ muss der Algorithmus im Polynomring $\mathbb{F}_p[X]$ durchgeführt werden, was die Kompmlexität weiter erhöht. Daher wurde von Simon Amberg und Ali Al-Kubaisi ein webbasierter Rechner auf `https://linalg.ch/fp` entwickelt, mit dem die Rechenoperationen in diesen Körpern in einem Webbrowser durchgeführt werden können. Neben einem einfachen Interface, in dem der Benutzer zwei Operationen eingeben und dann eine Operation auslösen kann, wurde auch ein RPN-Modus implementiert, welches in Abbildung 6 dargestellt ist.

Literatur

[1] *Algebraic Operationg System (AOS™)*. Dez. 2024. URL: http://www.datamath.org/Sci/WEDGE/SR-52_AOS.htm#Algebraic_Operating_System:.

[2] *HP-35*. Dez. 2024. URL: https://www.hpmuseum.org/hp35.htm.

[3] *HP 9100A*. Dez. 2024. URL: https://de.wikipedia.org/wiki/HP_9100A.

[4] Andreas Müller. *Lineare Algebra: Eine praxisorientierte Einführung. Mathematische Grundlagen, praxisrelevante Methoden und technische Anwendungen*. Springer-Vieweg, 2023. ISBN: 978-3-662-67865-7. DOI: https://doi.org/10.1007/978-3-662-67866-4.