Rekursion in der Sprache

Was ist Rekursion?

Was ist Rekursion?

- Wikipedia: "Als Rekursion (lat. recurrere 'zurücklaufen') bezeichnet man den Aufruf oder die Definition einer Funktion durch sich selbst."

- Bussmann (1990, 640): "Aus der Mathematik übernommener Begriff, der in der Linguistik die formalen Eigenschaft von Grammatiken bezeichnet, mit einem endlichen Inventar von Elementen und einer endlichen Menge von Regeln eine unendliche Menge von Sätzen zu erzeugen."

Rekursion in Mathematik und Programmierung

(1) *Beginn der Fibonacci-Folge:*
$0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, \ldots$

(2) *Rekursive Definition der Fibonacci-Folge:*

a. fib$(n) = 0$, wenn $n = 0$
b. fib$(n) = 1$, wenn $n = 1$
c. fib(n) = fib$(n-1)$+fib$(n-2)$, wenn $n \geq 2$

- Die Rekursion setzt ein in Bedingung (2-c): fib(n) wird definiert mit Bezug auf fib$(n-1)$ und fib$(n-2)$.
- Die Rekursion in (2) stoppt, wenn $n = 0$ oder $n = 1$.
- Rekursive Definitionen und Funktionen müssen einen **Bodensatz** (oder **Abbruchbedingung**) haben (siehe (2-a,b)), durch den die Rekursion gestoppt wird. Sonst kann nichts definiert oder berechnet werden ("Endlosschleife").

Rekursion in Mathematik und Programmierung 2

(3) *Fibonacci-Definition in Pascal:*
{INPUT: x ist eine natürliche Zahl ≥ 0}

```
function Fib(x: integer): integer;
begin
      if x = 0 then
      Fib := 0
      else if x = 1 then
      Fib := 1
      else
      Fib := Fib(x-1) + Fib(x-2)
end;
```

Rekursion in Mathematik und Programmierung 3

(4) *Euklidischer Algorithmus des GGT:*

a. ggt$(x, y) = x$, wenn $y = 0$
b. ggt$(x, y) =$ ggt$((x - y), y)$, wenn $x > y$
c. ggt$(x, y) =$ggt$(x, (y - x))$ wenn $x \leq y$

- GGT = größter gemeinsamer Teiler.
- Die Rekursion setzt ein in (4-b,c).
- Sie stoppt, wenn $y = 0$ (entsprechend der Abbruchbedingung (4-a)).

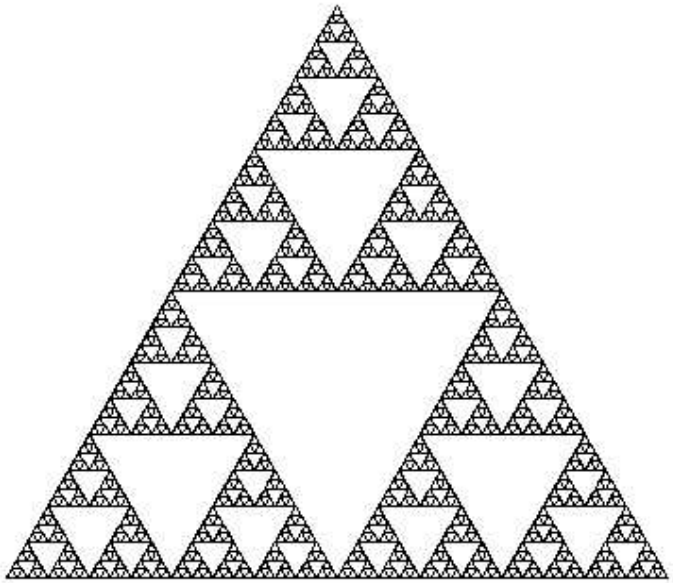
Rekursion in Mathematik und Programmierung 4

(5) *Euklidischer Algorithmus in Pascal:*

{INPUT: x ist eine natürliche Zahl ≥ 0}

```
function Euklid(x: integer, y: integer): integer;
begin
      if y = 0 then
      Euklid := x
      else if x > y then
      Euklid := Euklid((x-y),y)
      else
      Euklid := Euklid(x,(y-x))
end;
```

Rekursion in Mathematik und Programmierung 5



Sierpinski-Dreieck

- Rekursion ist hier sichtbar: die rekursiv konstruierten Dreiecke finden sich in der Gesamtstruktur wieder (das Dreieck enthählt **selbstähnliche** Strukturen).

Rekursion in Mathematik und Programmierung 6

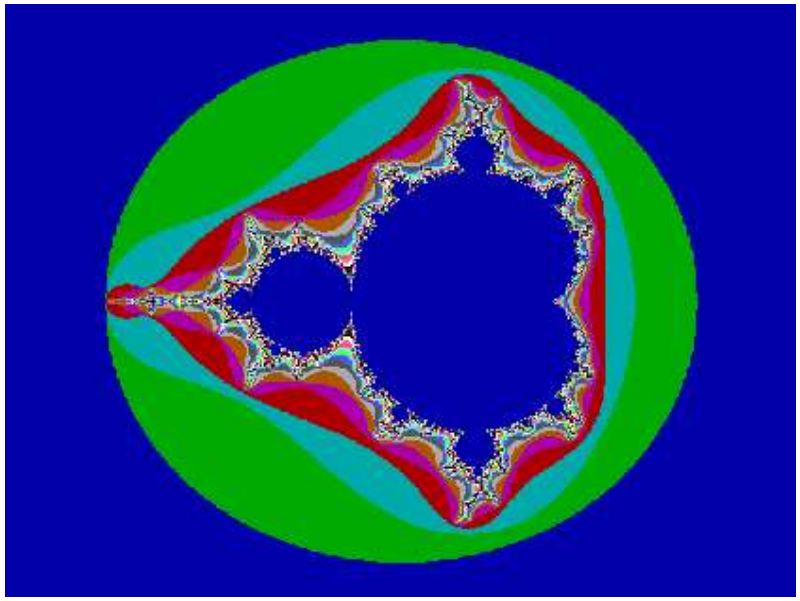
(6) *Algorithmus für Sierpinski-Dreieck:*

a. Zeichne ein gleichmäßiges Dreieck.
b. Verbinde die Mittelpunkte der Seiten; dadurch wird das ursprüngliche Dreieck in vier kongruente Teildreiecke zerlegt.
c. Entferne das mittlere der vier Teildreiecke; die anderen drei Teildreiecke bleiben übrig.
d. Wende Schritte (6-b-d) auf die drei briggebliebenen Teildreiecke an.

- Die Rekursion setzt ein im Schritt (6-d)

- Sie stoppt, wenn Schritt 4. nicht ausgeführt wird.

Rekursion in Mathematik und Programmierung 7

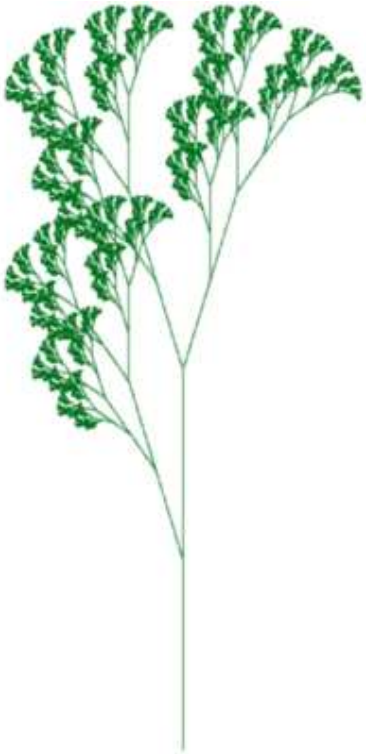


Mandelbrotmenge

Rekursion in der Natur

Romanescogemüse

Rekursion in der Natur 2



computererzeugte, selbstähnliche Pflanzenstruktur

Rekursion in Kunst und Dichtung

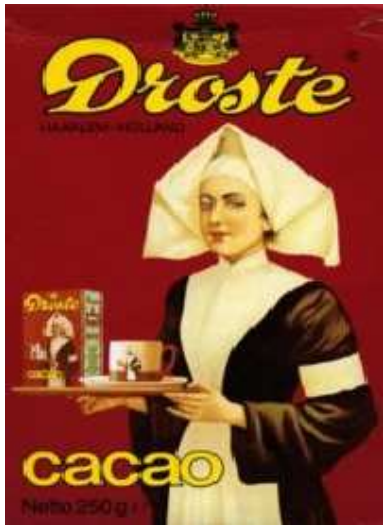
Ein Mops lief in die Küche (ohne Bodensatz!)

Ein Mops lief in die Küche
Und stahl dem Koch ein Ei
Da nahm der Koch den Löffel
Und schlug den Mops zu Brei
Da kamen viele Möpse
Und gruben ihm ein Grab
Und setzten drauf den Grabstein
Worauf geschrieben stand:

Ein Mops lief in die Küche
Und stahl dem Koch ein Ei
Da nahm der Koch den Löffel
Und schlug den Mops zu Brei
Da kamen viele Möpse
Und gruben ihm ein Grab
Und setzten drauf den Grabstein
Worauf geschrieben stand:

etc. ad infinitum

Rekursion in Kunst und Dichtung 2



"Droste"-Kakao-Werbung

e

Rekursion in der Sprache

- Chomsky (1963, 326f): "The generative grammar represents the information concerning sentence structure that is available [...] to one who has acquired the language. [...] there is no reasonable alternative to the conclusion that recursive rules specifying the correct solution are represented somehow in the brain [...]

- Hauser et al. (2002, 1569): "We submit that a distinction should be made between the faculty of language in the broad sense (FLB) and in the narrow sense (FLN). FLB includes [...] the computational mechanisms for recursion, providing the capacity to generate an infinite range of expressions from a finite set of elements. We hypothesize that FLN only includes recursion and is the only uniquely human component of the faculty of language."

Rekursion in der Morphologie

- Rekursion kommt in der **Morphologie** vor bei

  1. **Komposition**
  2. **Derivation**

(7) *Komposition:*

  a. [$_{N}$ Gesetz]
  b. [$_{N}$ [$_{N}$ Rahmen] [$_{N}$ Gesetz]]
  c. [$_{N}$ Hochschul [$_{N}$ [$_{N}$ Rahmen] [$_{N}$ Gesetz]]]

(8) *Derivation:*

  a. [$_{N}$ Grossvater]
  b. [$_{N}$ Ur [$_{N}$ Grossvater]]
  c. [$_{N}$ Ur [$_{N}$ Ur [$_{N}$ Grossvater]]]
  d. [$_{N}$ Ur [$_{N}$ Ur [$_{N}$ Ur [$_{N}$ Grossvater]]]]

Rekursion in der Syntax 1

- In der **Syntax** betrifft Rekursion den **Strukturaufbau** verschiedener Kategorien: NP, DP, PP, AP, VP, CP, . . .

(9) *NP-Rekursion:*

  a. [$_{NP}$ gute Zeit]
  b. [$_{NP}$ gute [$_{NP}$ alte Zeit]]
  c. [$_{NP}$ ferne [$_{NP}$ gute [$_{NP}$ alte Zeit]]]

(10) *DP-Rekursion:*

  a. [$_{DP}$ sein VW]
  b. [$_{DP}$ [$_{DP}$ dem Karl] sein VW]
  c. [$_{DP}$ [$_{DP}$ [$_{DP}$ dem Karl] seiner Oma] ihr VW]

(11) *PP-Rekursion:*

  a. [$_{PP}$ auf der Insel]
  b. [$_{PP}$ auf der Insel [$_{PP}$ mit dem Namen]]
  c. [$_{PP}$ auf der Insel [$_{PP}$ mit dem Namen [$_{PP}$ ohne Vokal]]]

Rekursion in der Syntax 2

(12) *AP-Rekursion (cf. (Abney 1987)):*

a. [$_{AP}$ old [$_{NP}$ days]]
b. [$_{AP}$ good [$_{AP}$ old [$_{NP}$ days]]]
c. [$_{AP}$ good [$_{AP}$ old [$_{AP}$ lonely [$_{NP}$ days]]]]

(13) *VP-Rekursion (cf. (Larson 1988)):*

a. [$_{VP}$ spoke to Mary]
b. [$_{VP}$ spoke [$_{VP}$ to Mary V on Tuesday]]
c. [$_{VP}$ spoke [$_{VP}$ to Mary V [$_{VP}$ in her house V on Tuesday]]]

(14) *CP-Rekursion:*

a. [$_{CP}$ dass er ein Lump ist]
b. [$_{CP}$ Maria glaubt, [$_{CP}$ dass er ein Lump ist]]
c. [$_{CP}$ Fritz sagt, [$_{CP}$ dass Maria glaubt, [$_{CP}$ dass er ein Lump ist]]]
d. [$_{CP}$ Karl behauptet, [$_{CP}$ dass Fritz sagt, [$_{CP}$ dass Maria glaubt, [$_{CP}$ dass er ein Lump ist]]]]

Rekursion in der Syntax 3

- **Überprüfung** (**Checking**) des Merkmals [wh] in der Syntax ist u.U. rekursiv zu definieren (bei **Pied-Piping**-Strukturen; (Heck 2004)).

(15) *Rekursive* wh-*Checking Definition:*
α ist in einer *wh*-Checkingkonfiguration mit einem Kopf γ, genau dann, wenn a. oder b. gelten.

a. α steht in Specγ.
b. β steht in SpecCγ und α ist in einer *wh*-Checkingkonfiguration mit dem Kopf von β.

(16) *Rekursives* wh-*Checking (Sells 1985):*

a. someone [whose lap] you spilled coffee on
b. someone [whose lawyer's lap] you spilled coffee on
c. someone [whose lawyer's mother's lap] you spilled coffee on
d. *someone [the lap of who(m)] you spilled coffee on

Rekursion in der Syntax 4

- Rekursion kann auch zu unerwünschten Effekten führen: Infinite Rekursion bei der Interpretation von **Antezedenz-beinhalteter Tilgung** (ACD) durch Kopieren des Antezedenz.

(17) *Infinite Rekursion bei ACD:*

a. Egbert $[_{VP}$ suspected everyone Horace did $[_{VP}$ – $]_2$ $]_3$
$\rightarrow$ (Kopiere VP_3 in VP_2)

b. Egbert $[_{VP}$ suspected everyone Horace did $[_{VP}$ suspected everyone Horace did $[_{VP}$ – $]_2$ $]_2$ $]_3$
$\rightarrow$ (Kopiere VP_2 in VP_2)

c. Egbert $[_{VP}$ suspected everyone Horace did $[_{VP}$ suspected everyone Horace did $[_{VP}$ suspected everyone Horace did $[_{VP}$ – $]_2$ $]_2$ $]_2$ $]_3$
$\rightarrow$ . . .

Rekursion in der Syntax 5

- Lösung: Diese infinite Rekursion wird verhindert, indem **Quantifier-Raising** (QR) die Struktur vor dem Kopieren verändert (May 1985).

(18) *Blockieren der infiniten Rekursion durch QR:*

a. Egbert $[_{VP}$ suspected $[_{NP}$ everyone Horace did $[_{VP}$ – $]_2$ $]_4$ $]_3$
$\rightarrow$ (QR NP_4)

b. $[_{NP}$ everyone Horace did $[_{VP}$ – $]_2$ $]_4$ Egbert $[_{VP}$ suspected t_4 $]_3$
$\rightarrow$ (Kopiere VP_3 in VP_2)

c. $[_{NP}$ everyone Horace did $[_{VP}$ suspect(ed) t_4 $]_2$ $]_4$ Egbert $[_{VP}$ suspected t_4 $]_3$

Performanz und Kompetenz

- Unterscheidung
  1. **Kompetenz** bezeichnet die abstrakte Sprachfähigkeit des Menschen: er besitzt das sprachliche Wissen um alle und nur die grammatischen Sätze seiner Sprache zu bilden oder erkennen.
  2. **Performanz** bezeichnet den tatsächlichen Gebrauch dieses Wissens. Die Performanz wird dabei beiinflusst von Faktoren wie Arbeitsgedächtnis, Konzentration, etc.

- Konsequenz: **Mittenrekursive** Strukturen (19-a) werden tatsächlich oft zu **randrekursiven** Strukturen (wie die **endrekursive** in (19-b)) umstrukturiert, um leichter verarbeitbar zu sein.

(19) *Mitten- und randrekursive Strukturen:*

a. $[_\alpha \ldots [_\alpha \ldots [_\alpha \ldots] \ldots] \ldots]$
b. $[_\alpha \ldots [_\alpha \ldots [_\alpha \ldots]]]$

Performanz und Kompetenz 2

- Gängige Annahme: Trotzdem sind die Grammatiken natürlicher Sprachen im Prinzip fähig zur Mittenrekursion (dies wird durch die Abhängigkeiten gezeigt).

- Chomsky (1963, 326f): "In fact, such sentences [...] are quite incomprehensible on first hearing, but this has no bearing on the question whether those sentences are generated by the grammar that has been acquired, just as the inability of a person to multiply 18,674 times 26,521 in his head is no indication that he has failed to grasp the rules of multiplication.

Performanz und Kompetenz 3

(20) *3 × Extraposition in englischen Relativsätzen:*

a. The cat [$_{CP}$ that the dog [$_{CP}$ that the rat [$_{CP}$ that the elephant admired] bit] chased] died →

b. The cat died [$_{CP}$ that the dog [$_{CP}$ that the rat [$_{CP}$ that the elephant admired] bit] chased] →

c. The cat died [$_{CP}$ that the dog chased] [$_{CP}$ that the rat [$_{CP}$ that the elephant admired] bit] →

d. The cat died [$_{CP}$ that the dog chased] [$_{CP}$ that the rat bit] [$_{CP}$ that the elephant admired]

Performanz und Kompetenz 4

- "Es darf daher getrost, was auch von allen, deren Sinne, weil sie unter Sternen, die, wie der Dichter sagt, zu dörren, statt zu leuchten, geschaffen sind, geboren sind, vertrocknet sind, behauptet wird, enthauptet werden [...]"

  (Christian Morgenstern, Galgenlieder)

- Nach 4 × Extraposition:

- Es darf daher getrost enthauptet werden, was auch von allen behauptet wird, deren Sinne vertrocknet sind, weil sie unter Sternen geboren sind, die, wie der Dichter sagt, zu dörren, statt zu leuchten, geschaffen sind, [...]

Rekursion in der Phonologie

- Standardannahme früher: **phonologische** Strukturen sind nicht rekursiv.

- (Eine Silbe zum Beispiel kann nicht durch Hinzufügen eines Ansatzes oder Nukleus in eine andere Silbe verwandelt werden.)

- Dies wurde durch die **Strict Layer Hypothesis** sichergestellt ((Selkirk 1984), (Nespor & Vogel 1986))

(21) *Prosodische Hierarchie:*
Intonationsphrase $\succ$ phonologische Phrase $\succ$ phonologisches Wort $\succ$ Fuß $\succ$ Silbe $\succ$ Mora.

(22) *(Teil der) Strict Layer Hypothesis:*
Wenn K_1 und K_2 Kategorien der prosodischen Hierarchie sind, dann dominiert K_1 K_2 unmittelbar genau dann, wenn $K_1 \succ K_2$.

Rekursion in der Phonologie 2

- Mit Einführung der **Optimalitätstheorie** in die Phonologie ((Prince & Smolensky 1993, Prince & Smolensky 2004)) wurde die Strict Layer Hypothesis gelockert.

- Seither wird öfter theoretisch wie empirisch für das Auftreten von Rekursion (wie z.B. in (23-a,b)) in der Phonologie argumentiert.

(23) a. $[_{\Sigma} \ldots [_{\Sigma} \ldots]]$ rekursiver Fuß
b. $[_{\omega} \ldots [_{\omega} \ldots]]$ rekursives phon. Wort

Literatur

Abney, Steven (1987): The English Noun Phrase in Its Sentential Aspect. PhD thesis, MIT, Cambridge, Massachusetts.

Bußmann, Hadumod, ed. (1990): *Lexikon der Sprachwissenschaft*. Alfred-Kröner-Verlag, Stuttgart.

Chomsky, Noam (1963): Formal Properties of Grammars. *In:* R. D. Luce, R. Bush & E. Galanter, eds, *Handbook of Mathematical Psychology*. Wiley, New York, pp. 323–418.

Hauser, Marc, Noam Chomsky & Tecumseh Fitch (2002): 'The Faculty of Language: What Is It, Who Has It, and How did It Evolve?', *Science* **298**, 1569–1579.

Heck, Fabian (2004): A Theory of Pied-Piping. PhD thesis, Universität Tübingen.

Larson, Richard (1988): 'On the Double Object Construction', *Linguistic Inquiry* **19**, 335–391.

May, Robert (1985): *Logical Form: Its Structure and Derivation*. MIT Press, Cambridge, Massachusetts.

Nespor, Marina & Irene Vogel (1986): *Prosodic Phonology*. Foris, Dordrecht.

Prince, Alan & Paul Smolensky (1993): Optimality Theory. Constraint Interaction in Generative Grammar. Ms., Rutgers University and University of Colorado at Boulder.

Prince, Alan & Paul Smolensky (2004): Constraint Interaction in Generative Grammar. *In:* J. J. McCarthy, ed., *Optimality Theory in Phonology – A Reader*. Blackwell, Oxford, pp. 3–71.

Selkirk, Elisabeth (1984): *Phonology and Syntax. The Relation between Sound and Structure*. MIT Press, Cambridge Massachusetts.

Sells, Peter (1985): Pied Piping and the Feature [wh]. Ms., Stanford University, Ca.