Tutorium: Computerlinguistik

Daniel Milne-Plückebaum, dmilne@uni-bielefeld.de

Die folgenden Übungen, Lösungen und Hilfen basieren auf der Vorlesung *Computerlinguistik und Sprachtechnologie*, gehalten von Marcus Kracht an der Universität Bielefeld (im Wintersemester 2014/15), und sind somit idealerweise nur zusammen mit dem zugehörigen Vorlesungsmanuskript zu verwenden. Über Verbesserungsvorschläge und Hinweise auf Fehler freue ich mich sehr.

Inhaltsverzeichnis

1 Typentheoretische Semantik

1.1 Typen in OCaml

1.1.1 Hilfe: Einige Typen

- Grundtypen:
  - Zeichen (character)
    - OCaml-String: `char`
    - Beispiele: `'A'`, `'q'`, `'5'`, `'\132'`, `'.'`
  - Zeichenkette (string)
    - OCaml-String: `string`
    - Beispiele: `"Aq5\."`, `"OCaml"`, `"1234"`, `"1.4"`, `"false"`
  - ganze Zahl (integer)
    - OCaml-String: `int`
    - Beispiele: `1`, `67`, `0`, `-405`, `-5999`
  - reelle Zahl (float)
    - OCaml-String: `float`
    - Beispiele: `1.`, `67.5`, `0.`, `-405.89853`, `-5999.5999`
  - Wahrheitswert (boolean)
    - OCaml-String: `bool`
    - `true`, `false`
- Typkonstruktoren:
  - Produkttyp: Wenn α und β Typen sind, dann ist auch $\alpha \bullet \beta$ ein Typ
    - OCaml-String: `'a * 'b`
    - Beispiele: `('A','A')`, `(1,"q")`, `('5',(1,2))`, `(('\132',7),(3.,("3",'3')))`

    * Aufgabe: Gib die genauen Typen der Beispiele an.
  - Listentyp: Wenn α ein Typ ist, dann ist auch α *list* ein Typ
    * OCaml-String: `'a list`
    * Beispiele: `[3;7;5]`, `['a';'q';'1';'x';';']`, `[(1.,''q'');(2.5,''maus'')]`, `[true]`, `[[true;false];[true;true;true];[false]]`
    * Aufgabe: Gib die genauen Typen der Beispiele an.
  - Funktionstyp: Wenn α und β Typen sind, dann ist auch $\alpha \to \beta$ ein Typ
    * OCaml-String: `'a -> 'b`

1.1.2 Übungen: Grundtypen, Paare und Listen

1. Welchen Typ haben die folgenden Ausdrücke? Gib eine abstrakte Charakterisierung sowie den OCaml-String dafür an. Gib außerdem ggf. das Ergebnis an.

   (a) `896`
   (b) `0.93`
   (c) `7*4`
   (d) `'r'`
   (e) `'' '5' ''`
   (f) `0.5+.0.5`
   (g) `false`
   (h) `''false''`
   (i) `6+3=3*3`
   (j) `'\132'`
   (k) `1+2*3+4`
   (l) `800/80/8`
   (m) `400>200`
   (n) `1<>1`
   (o) `true||false`
   (p) `true&&false`

2. Warum verwendet man nicht einfach z.B. die ganze Zahl `0`, um *falsch* zu repräsentieren, und `1`, um *wahr* zu repräsentieren? Warum hat man einen eigenen `bool`-Typ?

3. (a) Was ist der Effekt von Vergleichsoperationen wie `<` und `>` auf alphabetische Werte des Typs `char`? Was ist z.B. das Ergebnis von `'a'<'b'`?
   (b) Was ist der Effekt der Vergleichsoperatoren auf `true` und `false`?

4. Welchen Typ haben die folgenden Ausdrücke? Gib eine abstrakte Charakterisierung sowie den OCaml-String dafür an.

(a) `(1,2)`

(b) `(1,1.)`

(c) `((‘1’,‘‘maus’’),9.5)`

(d) `((false,‘‘false’’),(true,true))`

(e) `[1;2;3;4]`

(f) `[‘p’]`

(g) `([1;2],(1,2))`

(h) `[(true,false);(false,false)]`

(i) `[[1.;0.;7.];[89.;456.1;3.99];[11.11]]`

(j) `([[1.3;1.1]],[(false,false)])`

5. Gib (sinngemäß) an, was OCaml zu folgenden Eingaben ausgeben würde.

(a) `# 3;;`

(b) `# 3.0;;`

(c) `# ‘3.’;;`

(d) `# ‘‘ ‘3’ ’’;;`

(e) `# 3.0=3.;;`

(f) `# 3=3.;;`

(g) `# ‘3’=‘‘3.’’;;`

(h) `# ‘‘3.’’=‘‘3.0’’;;`

(i) `# ‘‘3’’=‘3’;;`

(j) `# (‘‘3.’’=‘‘3’’)=(3.0=3.00);;`

6. Gib (sinngemäß) an, was OCaml zu folgenden Eingaben ausgeben würde.

(a) `# 1+2;;`

(b) `# 1+2.;;`

(c) `# 1+.2;;`

(d) `# 1+.2.;;`

(e) `# 1.+2;;`

(f) `# 1.+2.;;`

(g) `# 1.+.2;;`

(h) `# 1.+.2.;;`

7. Was erwartet OCaml nach den folgenden Eingaben?

(a) `# ‘‘` ...

(b) `# ‘` ...

(c) `# 5.8<` ...

(d) `# false=` ...

(e) `# 5=8=` ...

(f) `# (5,` ...

8. Erläutere folgende OCaml-Ausgabe:

```
# 5=6 = 5=8;;

Error: This expression has type int but an expression was expected of
type bool
```

9. Erläutere, was es bedeutet, dass eine Funktion *polymorph* ist. Erläutere in dem Zusammenhang auch, was es mit `'a` auf sich hat. Z.B. ist die Gleichheit (=) polymorph.

```
# (=);;

val f : 'a -> 'a -> bool = <fun>
```

10. Erzeuge Objekte folgenden Typs und teste sie mit OCaml aus.

    (a) Eine Liste von ganzen Zahlen.

    (b) Ein Paar bestehend aus einem Paar bestehend aus zwei Zeichen und einem Zeichen.

    (c) Eine Liste von Listen von reellen Zahlen.

    (d) Ein Paar bestehend aus Paaren von Listen von Zeichenketten.

    (e) Eine Liste von Paaren von Listen von Wahrheitswerten.

1.1.3 Übungen: Funktionen

1. Welchen Typ haben die folgenden Ausdrücke? Welche Ausdrücke sind polymorph? Gib jeweils eine abstrakte Charakterisierung, den OCaml-String und ggf. das Ergebnis an.

    (a) `7 :: [] = [7]`

    (b) `(=)`

    (c) `(+)`

    (d) `(+.)`

    (e) `(fst)`

    (f) `f x = x + x`

    (g) `g x = x +. x`

    (h) `gleich x y = (x = y)`

    (i) `h x = (2 * x) + 3`

    (j) `[''7''] :: [''7.'']`

    (k) `''apple'' = ''apfel''`

    (l) `6 + 4`

    (m) `3.1 +. 1.3`

    (n) `fst (7, 4)`

    (o) `f 12`

    (p) `g 12.5`

    (q) `gleich 'a' 'b'`

    (r) `h 3`

2. Worin besteht in folgenden Eingaben jeweils der Fehler?

    (a) `# let x = (2 *. x) +. 3;;`

    (b) `# let rec listmap f l = match l with`

```
[] -> []
| h :: t -> f(h) :: (listmap t);;
```

3. (a) Programmiere die folgenden Funktionen.
   (b) Gib für jede der Funktionen ihren Typ an.
   (c) Berechne jeweils den Funktionswert für drei selbst gewählte Argumente.
      i. Eine Funktion, die für jede ganze Zahl ihren Kubikwert liefert.
      ii. Eine Funktion, die für jede ganze Zahl angibt, ob sie negativ ist.
      iii. Eine Funktion, die für jedes Zeichen angibt, ob es ein Vokal ist.
      iv. Eine Funktion, die für je zwei ganze Zahlen angibt, ob sie zusammen 10 ergeben.
      v. Eine Funktion, die für jeden Wahrheitswert seine Negation liefert.
      vi. Eine Funktion, die für jede Zahl ihr Zehnfaches liefert.
      vii. Eine Funktion, die für eine Liste angibt, ob sie leer ist.

4. Bestimme jeweils die ersten 6 Funktionswerte.

   (a) $f(0) := 1$
   $f(n+1) := (-1)f(n)$

   (b) $g(0) := 0$
   $g(1) := 1$
   $g(n+1) := g(n) + g(n-1)$

   (c) $h(0) := 0$
   $h(n+1) := (n+1) - h(h(n))$

5. Gib für die Funktion $s(n) = \frac{n(n+1)}{2}$. eine rekursive Funktion an.

6. *Die erste Zahl ist 0. Jede weitere Zahl ist der verdoppelte Vorgänger vermindert um 1.*

   (a) Bestimme eine rekursive Funktion für diese Zahlenfolge.
   (b) Bestimme die ersten 5 Funktionswerte.

7. (a) Definiere folgende Funktionen rekursiv.
   (b) Bestimme die ersten 5 Funktionswerte.
   (c) Programmiere die rekursiven Funktionen.
   (d) Gib für jede der rekursiven Funktionen ihren Typ an.

(e) Überprüfe, ob die ersten 5 Funktionswerte mit den in (b) bestimmten übereinstimmen.

i. Eine Funktion, die für jede Zahl ihre Fakultät liefert.

ii. Eine Funktion, die für jede Zahl n die Summe $1 + 2 + \cdots + n$ liefert.

iii. Eine Funktion, die für je zwei Zahlen n und x die Zahl n^x liefert.

iv. Eine Funktion, die für jede Liste ihre Länge liefert.

v. Eine Funktion, die für jede Liste ganzer Zahlen die Summe der in ihr enthaltenen Zahlen liefert.

vi. Eine Funktion, die für jede Liste eine Liste liefert, die nur die Elemente an ungerader Stelle in der Ausgangsliste enthält.

vii. Eine Funktion, die die Elemente zweier Listen in einer Liste hintereinander zusammenfasst.

viii. Eine Funktion, die die Elemente einer Liste umdreht.

ix. Eine Funktion, die die ersten beiden Elemente einer Liste streicht.

x. Eine Funktion, die zählt, wie oft `true` in einer Liste steht.

1.1.4 Hilfe: Ein paar Funktionen in OCaml

1. Eine Funktion, die für jede ganze Zahl ihren Kubikwert liefert.

```
# let kubik x = x * x * x;;
val cube : int -> int = <fun>
```

2. Eine Funktion, die für jede ganze Zahl angibt, ob sie negativ ist.

```
# let neg x =
    if x < 0 then true
    else false;;
val neg : int -> bool = <fun>
```

oder:

```
# let neg x = x < 0;;
val neg : int -> bool = <fun>
```

3. Eine Funktion, die für jedes Zeichen angibt, ob es ein Vokal ist.

```
# let vowel x = x = 'a' || x = 'e' || x = 'i' || x = 'o' || x = 'u';;
val neg : char -> bool = <fun>
```

oder

```
# let vowel x = match x with
    'a' | 'e' | 'i' | 'o' | 'u' -> true |
    _ -> false;;
val neg : char -> bool = <fun>
```

4. Eine Funktion, die für je zwei ganze Zahlen angibt, ob sie zusammen 10 ergeben.

```
# let zehn x y = x + y = 10;;
val neg : int -> int -> bool = <fun>
```

5. Eine Funktion, die für jeden Wahrheitswert seine Negation liefert.

```
# let not x =
    if x then false
    else true;;
val not : bool -> bool = <fun>
```

oder

```
# let not x = match x with
    true -> false |
    false -> true;;
val not : bool -> bool = <fun>
```

6. Eine Funktion, die für jede Zahl ihr Zehnfaches liefert.

```
# let zehnmal x = x * 10;;
val zehnmal : int -> int = <fun>
```

7. Eine Funktion, die für eine Liste angibt, ob sie leer ist.

```
# let istleer l = match l with
    [] -> true |
    _ -> false;;
val istleer : 'a list -> bool = <fun>
```

8. Eine Funktion, die für jede Zahl ihre Fakultät liefert.

```
# let rec fakultaet x =
```

```
    if x = 0 then 1
    else x * fakultaet (x - 1);;
val fakultaet : int -> int = <fun>
```

oder

```
# let rec fakultaet x = match x with
    0 -> 0 |
    __ -> x * fakultaet (x - 1);;
val fakultaet : int -> int = <fun>
```

9. Eine Funktion, die für jede Zahl n die Summe $0 + 1 + 2 + \cdots + n$ liefert.

```
# let rec summebis x =
    if x = 0 then 0
    else x + summebis (x - 1);;
val summebis : int -> int = <fun>
```

oder

```
# let rec summebis x = match x with
    0 -> 0 |
    __ -> x + summebis (x - 1);;
val summebis : int -> int = <fun>
```

10. Eine Funktion, die für je zwei Zahlen n und x die Zahl n^x liefert.

```
# let rec hoch x n =
    if n = 0 then 1
    else x * hoch x (n-1);;
val neg : int -> int -> int = <fun>
```

oder

```
# let rec hoch x n = match n with
    0 -> 1 |
    __ -> x * hoch x (n - 1);;
val hoch : int -> int = <fun>
```

11. Eine Funktion, die für jede Liste ihre Länge liefert.

```
# let rec laenge l = match l with
    [] -> 0 |
    | __ :: t -> 1 + laenge t;;
val laenge : 'a list -> int = <fun>
```

12. Eine Funktion, die für jede Liste ganzer Zahlen die Summe der in ihr enthaltenen Zahlen liefert.

```
# let rec listensumme l = match l with
    [] -> 0 |
    h :: t -> h + listensumme t;;
val listensumme : int list -> int = <fun>
```

13. Eine Funktion, die für jede Liste eine Liste liefert, die nur die Elemente an ungerader Stelle in der Ausgangsliste enthält.

```
# let rec ungerade l = match l with
    [] -> [] |
    [a] -> [a] |
    a :: __ :: t -> a :: ungerade t;;
val ungerade : 'a list -> 'a list = <fun>
```

oder

```
# let rec ungerade l = match l with
    __ -> l |
    a :: __ :: t -> a :: ungerade t;;
val ungerade : 'a list -> 'a list = <fun>
```

14. Eine Funktion, die die Elemente zweier Listen in einer Liste hintereinander zusammenfasst.

```
# let rec hintereinander l m = match l with
    [] -> m |
    h :: t -> h :: hintereinander t b;;
val hintereinander : 'a list -> 'a list -> 'a list = <fun>
```

15. Eine Funktion, die die Elemente einer Liste umdreht.

```
# let rec umdrehen l = match l with
    [] -> [] |
    h :: t -> umdrehen t @ [h];;
val umdrehen : 'a list -> 'a list = <fun>
```

16. Eine Funktion, die die ersten n Elemente einer Liste streicht.

```
# let rec streichen n l =
    if n = 0 then l
    else match l with
       h :: t -> streichen (n - 1) t;;
val streichen : int -> 'a list -> 'a list = <fun>
```

17. Eine Funktion, die zählt, wie oft `true` in einer Liste steht.

```
# let rec truezaehlen l = match l with
    [] -> 0 |
    true :: t -> 1 + truezaehlen t |
    false :: t -> truezaehlen t;;
val truezaehlen : bool list -> int = <fun>
```

1.2 Kombinatoren, semantische Typen und Grammatiken

1.2.1 Übungen: Kombinatoren

1. Betrachte die folgenden Terme.

   i. Füge die gedachten Klammern hinzu.

   ii. Welche Teilterme enthalten die Terme jeweils?

   iii. Vereinfache die Terme so weit wie möglich.

   iv. Welche Teilterme enthalten die Ergebnisse jeweils?

(a) **SI**xy

(b) **S**(**SKK**)xy

(c) **SII**x

(d) **S**(**SKK**)(**SKK**)x

(e) **K**(**II**)**S**xy

(f) **SKK**x(**SKK**y)

(g) **S**(**KS**)**K**xyz

(h) **S**(**K**(**S**(**KS**)**K**))wxyz

(i) **S**((**S**(**K**(**S**(**KS**)**K**)))**S**)(**KK**)xyz

(j) **S**(**K**(**SI**))(**S**(**KK**)**I**)xy

(k) **S**(**K**(**S**(**SKK**)))**K**xy

(l) **K**(**S**(**SKK**)(**SKK**))xy

2. Bestimme die Typen der folgenden Kombinatoren.

(a) **O**xy = y(xy)

(b) **B**xyz = x(yz)

(c) **D**wxyz = wx(yz)

(d) **C**xyz = xzy

3. Überprüfe, ob folgende Ausdrücke wohlgetypt sind.

(a) **SI**

(b) **SSK**

(c) **S**(**SKK**)

(d) **S**(**II**)**KS**

(e) **R**xy = y x y

(f) **R**xy = y (x y)

4. Zeige, dass man den Kombinator **I** mithilfe der Kombinatoren **S** und **K** definieren kann.

1.2.2 Hilfe: Ist S(II)KS wohlgetypt?

Wir wollen herausfinden, ob der Ausdruck **S**(**II**)**KS** wohlgetypt ist. Dazu ordnen wir zuerst jeder Konstituente des Ausdrucks einen abstrakten Typ zu. Für die Konstituenten **S** (zwei Vorkommen), **I** (zwei Vorkommen) und **K** können wir die abstrakten Typen etwas genauer angeben als für die komplexen Konstituenten:

S: $(\alpha \to \beta \to \gamma) \to (\alpha \to \beta) \to \alpha \to \gamma$

S(**II**): δ

S(**II**)**K**: ϵ

$\mathbf{S}(\mathbf{II})\mathbf{KS}$: ζ

$\mathbf{I}$: $\eta \to \eta$

$\mathbf{I}$: $\theta \to \theta$

$\mathbf{II}$: ι

$\mathbf{K}$: $\kappa \to \lambda \to \kappa$

$\mathbf{S}$: $(\mu \to \nu \to \xi) \to (\mu \to \nu) \to \mu \to \xi$

Da $\mathbf{S}(\mathbf{II})$ eine Funktion ist, die $\mathbf{K}$ (mit Typ $\kappa \to \lambda \to \kappa$) als Argument aufisst und $\mathbf{S}(\mathbf{II})\mathbf{K}$ (mit Typ ϵ) als Ergebnis ausspuckt, hat $\mathbf{S}(\mathbf{II})$ den Typ $(\kappa \to \lambda \to \kappa) \to \epsilon$. Und da das erste Vorkommen von $\mathbf{I}$ eine Funktion ist, die das zweite Vorkommen von $\mathbf{I}$ (mit Typ $\theta \to \theta$) als Argument aufisst und $\mathbf{II}$ (mit Typ ι) als Ergebnis ausspuckt, hat das erste Vorkommen von $\mathbf{I}$ den Typ $(\theta \to \theta) \to \iota$. Wir erhalten:

$\mathbf{S}$: $(\alpha \to \beta \to \gamma) \to (\alpha \to \beta) \to \alpha \to \gamma$

$\mathbf{S}(\mathbf{II})$: $\delta = (\kappa \to \lambda \to \kappa) \to \epsilon$

$\mathbf{S}(\mathbf{II})\mathbf{K}$: ϵ

$\mathbf{S}(\mathbf{II})\mathbf{KS}$: ζ

$\mathbf{I}$: $\eta \to \eta = (\theta \to \theta) \to \iota$

$\mathbf{I}$: $\theta \to \theta$

$\mathbf{II}$: ι

$\mathbf{K}$: $\kappa \to \lambda \to \kappa$

$\mathbf{S}$: $(\mu \to \nu \to \xi) \to (\mu \to \nu) \to \mu \to \xi$

Da das erste Vorkommen von $\mathbf{S}$ eine Funktion ist, die $\mathbf{II}$ (mit Typ ι) als Argument aufisst und $\mathbf{S}(\mathbf{II})$ (mit Typ $(\kappa \to \lambda \to \kappa) \to \epsilon$) als Ergebnis ausspuckt, hat das erste Vorkommen von $\mathbf{S}$ den Typ $\iota \to (\kappa \to \lambda \to \kappa) \to \epsilon$. Und da $\mathbf{S}(\mathbf{II})\mathbf{K}$ eine Funktion ist, die das zweite Vorkommen von $\mathbf{S}$ (mit Typ $(\mu \to \nu \to \xi) \to (\mu \to \nu) \to \mu \to \xi$) als Argument aufisst und $\mathbf{S}(\mathbf{II})\mathbf{KS}$ (mit Typ ζ) als Ergebnis ausspuckt, hat $\mathbf{S}(\mathbf{II})\mathbf{K}$ den Typ $((\mu \to \nu \to \xi) \to (\mu \to \nu) \to \mu \to \xi) \to \zeta$. Wir erhalten:

S: $(\alpha \to \beta \to \gamma) \to (\alpha \to \beta) \to \alpha \to \gamma = \iota \to (\kappa \to \lambda \to \kappa) \to \epsilon$

S(**II**): $\delta = (\kappa \to \lambda \to \kappa) \to \epsilon$

S(**II**)**K**: $\epsilon = ((\mu \to \nu \to \xi) \to (\mu \to \nu) \to \mu \to \xi) \to \zeta$

S(**II**)**KS**: ζ

I: $\eta \to \eta = (\theta \to \theta) \to \iota$

I: $\theta \to \theta$

II: ι

K: $\kappa \to \lambda \to \kappa$

S: $(\mu \to \nu \to \xi) \to (\mu \to \nu) \to \mu \to \xi$

δ kommt sonst nicht vor. Also können wir δ streichen. ϵ kommt in zwei weiteren Gleichungen vor. Wir können die Vorkommen von ϵ durch $((\mu \to \nu \to \xi) \to (\mu \to \nu) \to \mu \to \xi) \to \zeta$ ersetzen und ϵ streichen. Wir erhalten:

S: $(\alpha \to \beta \to \gamma) \to (\alpha \to \beta) \to \alpha \to \gamma = \iota \to (\kappa \to \lambda \to \kappa) \to ((\mu \to \nu \to \xi) \to (\mu \to \nu) \to \mu \to \xi) \to \zeta$

S(**II**): $(\kappa \to \lambda \to \kappa) \to ((\mu \to \nu \to \xi) \to (\mu \to \nu) \to \mu \to \xi) \to \zeta$

S(**II**)**K**: $((\mu \to \nu \to \xi) \to (\mu \to \nu) \to \mu \to \xi) \to \zeta$

S(**II**)**KS**: ζ

I: $\eta \to \eta = (\theta \to \theta) \to \iota$

I: $\theta \to \theta$

II: ι

K: $\kappa \to \lambda \to \kappa$

S: $(\mu \to \nu \to \xi) \to (\mu \to \nu) \to \mu \to \xi$

Nun haben wir zwei Gleichungen. Aus diesen folgen jeweils weitere Gleichungen, aus denen wiederum evtl. Gleichungen folgen. Die Gleichungen ergeben sich jeweils aus der Syntax der Ausdrucke für die abstrakten Typen. Es gilt: Wenn $\text{Typ}_1 \to \text{Typ}_2 = \text{Typ}_3 \to \text{Typ}_4$, dann $\text{Typ}_1 = \text{Typ}_3$ und $\text{Typ}_2 = \text{Typ}_4$.

- $(\alpha \to \beta \to \gamma) \to (\alpha \to \beta) \to \alpha \to \gamma = \iota \to (\kappa \to \lambda \to \kappa) \to ((\mu \to \nu \to \xi) \to (\mu \to \nu) \to \mu \to \xi) \to \zeta$

  Daraus folgt:

  - $\alpha \to \beta \to \gamma = \iota$
  - $(\alpha \to \beta) \to \alpha \to \gamma = (\kappa \to \lambda \to \kappa) \to ((\mu \to \nu \to \xi) \to (\mu \to \nu) \to \mu \to \xi) \to \zeta$

    Daraus folgt:

    * $\alpha \to \beta = \kappa \to \lambda \to \kappa$

      Daraus folgt:

      · $\alpha = \kappa$

      · $\beta = \lambda \to \kappa$

    * $\alpha \to \gamma = ((\mu \to \nu \to \xi) \to (\mu \to \nu) \to \mu \to \xi) \to \zeta$

      Daraus folgt

      · $\alpha = (\mu \to \nu \to \xi) \to (\mu \to \nu) \to \mu \to \xi$

      · $\gamma = \zeta$

- $\eta \to \eta = (\theta \to \theta) \to \iota$

  Daraus folgt:

  - $\eta = \theta \to \theta$
  - $\eta = \iota$

    Daraus folgt:

    * $\iota = \theta \to \theta$

Die Gleichungen $\iota = \theta \to \theta$ und $\alpha \to \beta \to \gamma = \iota$ ergeben zusammen die Gleichungen $\alpha \to \beta \to \gamma = \theta \to \theta$. Daraus folgt:

- $\alpha = \theta$
- $\beta \rightarrow \gamma = \theta$

  Daraus folgt:

  - $\alpha = \beta \rightarrow \gamma$

Nun haben wir schließlich:

- $\alpha = \kappa = \beta \rightarrow \gamma$
- $\beta = \lambda \rightarrow \kappa$

  Daraus folgt:

  - $\beta = \lambda \rightarrow \beta \rightarrow \gamma$

Einen solchen Typ kann es nicht geben. Damit ist das Gleichungssystem nicht lösbar. Der Ausdruck **S**(**II**)**KS** ist also nicht wohlgetypt.

1.2.3 Übungen: Typentheoretische Semantik

1. Betrachte folgende Sätze.
   (a) Bestimme sämtliche Konstituenten und ihre semantischen Typen.
   (b) Erstelle ein Lexikon, das sämtlichen Wörtern einen semantischen Typ und eine Bedeutung zuordnet (dabei soll "`Mindestens eine`" als ein Wort zählen).
   (c) Berechne die Wahrheitswerte auf der Basis deines Lexikons.
      i. `Hansi zwitschert.`
      ii. `Hansi zwitschert nicht.`
      iii. `Hansi ist musikalisch.`
      iv. `Hansi zwitschert oder Hansi ist musikalisch.`
      v. `Miezi frisst Hansi.`
      vi. `Mindestens eine Katze jagt Peppi.`

2. Betrachte folgende Sätze.

   (a) Wie lauten die Anordnungen der Wortbedeutungen, sodass die Wahrheitswerte berechnet werden können?

   (b) Berechne die Wahrheitswerte auf der Basis deines Lexikons aus Aufgabe 1.

      i. `Hansi ist nicht musikalisch.`

      ii. `Peppi zwitschert oder zwitschert nicht.`

      iii. `Hansi ist musikalisch oder nicht.`

      iv. `Miezi jagt Hansi oder Peppi.`

      v. `Mindestens eine Katze jagt Hansi oder frisst Peppi.`

1.2.4 Lösungen: Typentheoretische Semantik

1. Betrachte folgende Sätze.

   (a) Bestimme sämtliche Konstituenten und ihre semantischen Typen.

   (b) Erstelle ein Lexikon, das sämtlichen Wörtern einen semantischen Typ und eine Bedeutung zuordnet (dabei soll "`Mindestens eine`" als ein Wort zählen).

   (c) Berechne die Wahrheitswerte auf der Basis deines Lexikons.

      i. (a) $[[\texttt{Hansi}]_e[\texttt{zwitschert}]_{e\to t}]_t$

      (b) "`Hansi`" bezeichne die Zahl 4.

      "`zwitschert`" bezeichne die Funktion zwitschert', die für alle geraden Zahlen wahr ist.

      (c) R 4 zwitschert' =

      zwitschert' 4 =

      true

      ii. (a) $[[[\texttt{Hansi}]_e[\texttt{zwitschert}]_{e\to t}]_t[\texttt{nicht}]_{t\to t}]_t$

      (b) "`nicht`" bezeichne die Funktion nicht' mit nicht'x = true gdw. x = false.

      (c) R (R 4 zwitschert') nicht' =

      nicht' (R 4 zwitschert') =

      nicht' (zwitschert' 4) =

      nicht' true =

      false

      iii. (a) $[[\texttt{Hansi}]_e[[\texttt{ist}]_{(e\to t)\to e\to t}[\texttt{musikalisch}]_{e\to t}]_{e\to t}]_t$

      (b) "`ist`" bezeichne die Funktion ist', die für alle Funktionen $f_{e\to t}$ des Typs $e \to t$ wieder $f_{e\to t}$ als Wert annimmt.

"musikalisch" bezeichne die Funktion musikalisch', die für alle Primzahlen wahr ist.

(c) R 4 (A ist' musikalisch') =
A ist' musikalisch' 4 =
ist' musikalisch' 4 =
musikalisch' 4 =
false

iv. (a) $[[[\texttt{Hansi}]_e[\texttt{zwitschert}]_{e\to t}]_t$
$[[\texttt{oder}]_{t\to t\to t}[[\texttt{Hansi}]_e[[\texttt{ist}]_{(e\to t)\to e\to t}[\texttt{musikalisch}]_{e\to t}]_{e\to t}]_t]_{t\to t}]_t$

(b) "oder" bezeichne die Funktion oder' mit oder'xy = true gdw. x = true oder y = true.

(c) R (R 4 zwitschert') (A oder' (R 4 (A ist' musikalisch'))) =
A oder' (R 4 (A ist' musikalisch')) (R 4 zwitschert') =
oder' (R 4 (A ist' musikalisch')) (R 4 zwitschert') =
oder' (A ist' musikalisch' 4) (zwitschert' 4) =
oder' (ist' musikalisch' 4) (zwitschert' 4) =
oder' (musikalisch' 4) (zwitschert' 4) =
oder' false true =
true

v. (a) $[[\texttt{Miezi}]_e[[\texttt{frisst}]_{e\to e\to t}[\texttt{Hansi}]_e]_{e\to t}]_t$

(b) "Miezi" bezeichne die Zahl 81.
"frisst" bezeichne die Funktion frisst' mit frisst'xy = true gdw. $y \geq 2x$.

(c) R 83 (A frisst' 4) =
A frisst' 4 81 =
frisst' 4 81 =
true

vi. (a) $[[[\texttt{Mindestens eine}]_{(e\to t)\to(e\to t)\to t}[\texttt{Katze}]_{e\to t}]_{(e\to t)\to t}$
$[[\texttt{jagt}]_{e\to e\to t}[\texttt{Peppi}]_e]_{e\to t}]_t$

(b) "Peppi" bezeichne die Zahl 9.
"Katze" bezeichne die Funktion katze', die für alle Potenzzahlen wahr ist.
"jagt" bezeichne die Funktion jagt' mit jagt'xy = true gdw. $y = x^2$.
"Mindestens eine" bezeichne die Funktion mindeine' mit mindeine'PQ = true gdw. es mindestens ein x gibt mit Px =true und Qx =true.

(c) A (A mindeine' katze') (A jagt' 9) =
A mindeine' katze' (A jagt' 9) =
mindeine' katze' (jagt' 9) =
true

2. Betrachte folgende Sätze.

(a) Wie lauten die Anordnungen der Wortbedeutungen, sodass die Wahrheitswerte berechnet werden können?

(b) Berechne die Wahrheitswerte auf der Basis deines Lexikons aus Aufgabe 1.

i. `Hansi ist nicht musikalisch.`
(a) nicht' (ist' musikalisch' 4) =
(b) nicht' (musikalisch' 4) =
nicht' false =
true

ii. `Peppi zwitschert oder zwitschert nicht.`
(a) oder' (nicht' (zwitschert' 9)) (zwitschert' 9) =
(b) oder' (nicht' false) false =
oder' true false =
true

iii. `Hansi ist musikalisch oder nicht.`
(a) oder' (nicht' (ist' musikalisch' 4)) (ist' musikalisch' 4) =
(b) oder' (nicht' (musikalisch' 4)) (musikalisch' 4) =
oder' (nicht' false) false =
oder' true false =
true

iv. `Miezi jagt Hansi oder Peppi.`
(a) oder' (jagt' 9 81) (jagt' 4 81) =
(b) oder' true false =
true

v. `Mindestens eine Katze jagt Hansi oder frisst Peppi.`
(a) oder' (mindeine' katze' (frisst' 9)) (mindeine' katze' (jagt' 4)) =
(b) oder' true true =
true

1.2.5 Übungen: Grammatiken und Parser

1. Gegeben sei die Grammatik $G \coloneqq \langle A, N, S, R\rangle$ mit

   A $\coloneqq$ $\{\texttt{die}_\sqcup, \texttt{kleine}_\sqcup, \texttt{Maus}_\sqcup, \texttt{mag}_\sqcup, \texttt{süße}_\sqcup, \texttt{Katze}_\sqcup\}$

   N $\coloneqq$ $\{S, NP, VP, V, D, N, A\}$

   R $\coloneqq$ $\{\langle S,\ NP\ VP\rangle, \langle VP,\ V\ NP\rangle, \langle NP,\ D\ N\rangle, \langle N,\ A\ N\rangle, \langle D, \texttt{die}_\sqcup\rangle, \langle V, \texttt{mag}_\sqcup\rangle,$
   $\langle A, \texttt{süße}_\sqcup\rangle, \langle A, \texttt{kleine}_\sqcup\rangle, \langle N, \texttt{Katze}_\sqcup\rangle, \langle N, \texttt{Maus}_\sqcup\rangle\}$

   (a) Leite zwei verschiedene Sätze ab.

   (b) Leite einen Satz auf zwei verschiedene Arten ab.

   (c) Zeichne die Strukturbäume zu den Sätzen aus (a) und (b).

   (d) Ist die Grammatik ambig und/oder transparent?

2. Gegeben sei die Grammatik $H \coloneqq \langle\{S, A, B\}, \{\texttt{a}, \texttt{b}\}, S, \{\langle S, AB\rangle, \langle A, \texttt{a}A\texttt{b}\rangle, \langle A, \epsilon\rangle,$
   $\langle B, B\texttt{b}\rangle, \langle B, \texttt{b}\rangle\}\rangle$

   (a) Leite zwei verschiedene Sätze ab.

   (b) Leite einen Satz auf zwei verschiedene Arten ab.

   (c) Zeichne die Strukturbäume zu den Sätzen aus (a) und (b).

   (d) Ist die Grammatik ambig und/oder transparent?

3. Gegeben seien die Grammatiken

   (i) $I \coloneqq \{\{\texttt{x}, \{A\}, A, \{\langle A, AA\rangle, \langle A, \texttt{a}\rangle\}\rangle$

   (ii) $J \coloneqq \{\{\texttt{x}, \{A, B\}, A, \{\langle A, A\texttt{x}\rangle, \langle A, B\texttt{x}\rangle, \langle A, \texttt{x}\rangle, \langle B, \texttt{x}\rangle\}\rangle$

   (a) Finde für I und J jeweils möglichst viele Ableitungen für die Zeichenkette `xxx`.

   (b) Gibt es Ableitungen von `xxx` für (i) oder (ii) mit jeweils verschiedene Strukturbäume?

   (c) Sind die Grammatiken ambig und/oder transparent?

4. Gegeben seien

   (i) die Grammatik, die die Formeln der Aussagenlogik in polnischer Notation generiert (siehe Skript, Seite 135).

   (ii) die Grammatik, die die Formeln der Aussagenlogik in Klammer-Notation generiert (siehe Skript, Seite 135).

(iii) die Grammatik, die die Formeln der Aussagenlogik mit verschiedenen propositionalen Variablen generiert (siehe Skript, Seite 140).

(a) Leite für jede der Grammatiken zwei verschiedene Formeln auf jeweils zwei verschiedene Weisen ab.

(b) Gibt es für einer der Grammatiken verschiedene Strukturbäume für eine einzige Formel?

(c) Sind die Grammatiken ambig und/oder transparent.

(d) Finde für die Grammatik aus (i) mithilfe eines Shift-Reduce-Parsers eine Ableitung für folgende Formeln:

- `¬∧pp`
- `∨∧p¬p¬p`
- `→ ¬∨p∨¬ppp`

Gib außerdem zwei Formeln an, die nicht wohlgebildet sind, und zeige dies mithilfe eines Shift-Reduce-Parsers.

(e) Finde für die Grammatik aus (ii) mithilfe eines Shift-Reduce-Parsers eine Ableitung für folgende Formeln:

- `(¬(p∧p))`
- `(p∧(p→(¬ p)))`
- `((¬ p∨(¬p))∨(¬(p∧p)))`

Gib außerdem zwei Formeln an, die nicht wohlgebildet sind, und zeige dies mithilfe eines Shift-Reduce-Parsers.

(f) Finde für die Grammatik aus (iii) mithilfe eines Shift-Reduce-Parsers eine Ableitung für folgende Formeln:

- `(¬(p1∧p12))`
- `(p0∧(p→(¬p2)))`
- `((¬p2∨(¬p22))∨(¬(p1∧p21)))`

Gib außerdem zwei Formeln an, die nicht wohlgebildet sind, und zeige dies mithilfe eines Shift-Reduce-Parsers.

1.3 Kombinatorische Kategorialgrammatiken

1.3.1 Übungen: Kategorialgrammatiken 1

1. Gegeben seien die Wörter `Maus`$_\sqcup$, `Ratte`$_\sqcup$, `Katze`$_\sqcup$ `schnelle`$_\sqcup$, `fette`$_\sqcup$, `schnurrt`$_\sqcup$, `fängt`$_\sqcup$, `jagt`$_\sqcup$, `die`$_\sqcup$, `eine`$_\sqcup$, `jede`$_\sqcup$.

(a) Bilde drei grammatische Sätze. Verwende dabei jedes Wort.

(b) Bestimme auf der Grundlage deiner gebildeten Sätze die Kategorien der Wörter.

(c) Formuliere eine Kategorialgrammatik für die Wörter, sodass deine Sätze ableitbar sind.

(d) Schreibe für jeden deiner Sätze einen Strukturbaum auf, der zu deiner Kategorialgrammatik passt.

2. Gegeben seien deine Ergebnisse aus Aufgabe 1.

(a) Weise jedem Wort einen semantischen Typ zu.

(b) Denke dir für jedes Wort eine Bedeutung aus dem Universum über der Menge von semantischen Typen aus.

(c) Formuliere eine *interpretierte* Kategorialgrammatik für die Wörter.

(d) Berechne für jeden deiner Sätze den Wahrheitswert.

1.3.2 Hilfe: Ein Lexikon

Gegeben seien folgenden Wörter, Kategorien und semantische Typen:

Wort	Kategorie	semantischer Typ
`Garfield`$_\sqcup$	D	$\tau(\mathrm{D}) = \mathrm{e}$
`Tom`$_\sqcup$	D	$\tau(\mathrm{D}) = \mathrm{e}$
`Jerry`$_\sqcup$	D	$\tau(\mathrm{D}) = \mathrm{e}$
`Mickey`$_\sqcup$	D	$\tau(\mathrm{D}) = \mathrm{e}$
`Rémy`$_\sqcup$	D	$\tau(\mathrm{D}) = \mathrm{e}$
`Maus`$_\sqcup$	N	$\tau(\mathrm{N}) = \mathrm{e} \to \mathrm{t}$
`Ratte`$_\sqcup$	N	$\tau(\mathrm{N}) = \mathrm{e} \to \mathrm{t}$
`Katze`$_\sqcup$	N	$\tau(\mathrm{N}) = \mathrm{e} \to \mathrm{t}$
`schnelle`$_\sqcup$	N/N	$\tau(\mathrm{N}) \to \tau(\mathrm{N}) = (\mathrm{e} \to \mathrm{t}) \to \mathrm{e} \to \mathrm{t}$
`fette`$_\sqcup$	N/N	$\tau(\mathrm{N}) \to \tau(\mathrm{N}) = (\mathrm{e} \to \mathrm{t}) \to \mathrm{e} \to \mathrm{t}$
`schnurrt`$_\sqcup$	D\C	$\tau(\mathrm{D}) \to \tau(\mathrm{C}) = \mathrm{e} \to \mathrm{t}$
`fängt`$_\sqcup$	(D\C)/D	$\tau(\mathrm{D}) \to \tau(\mathrm{D\backslash C}) = \tau(\mathrm{D}) \to \tau(\mathrm{D}) \to \tau(\mathrm{C}) = \mathrm{e} \to \mathrm{e} \to \mathrm{t}$
`jagt`$_\sqcup$	(D\C)/D	$\tau(\mathrm{D}) \to \tau(\mathrm{D\backslash C}) = \tau(\mathrm{D}) \to \tau(\mathrm{D}) \to \tau(\mathrm{C}) = \mathrm{e} \to \mathrm{e} \to \mathrm{t}$
`die`$_\sqcup$	D/N	$\tau(\mathrm{N}) \to \tau(\mathrm{D}) = (\mathrm{e} \to \mathrm{t}) \to \mathrm{e}$
`eine`$_\sqcup$	(C/(D\C))/N	$\tau(\mathrm{N}) \to \tau(\mathrm{C/(D\backslash C)}) = \tau(\mathrm{N}) \to \tau(\mathrm{D\backslash C}) \to \tau(\mathrm{C}) =$ $\tau(\mathrm{N}) \to (\tau(\mathrm{D}) \to \tau(\mathrm{C})) \to \tau(\mathrm{C}) = (\mathrm{e} \to \mathrm{t}) \to (\mathrm{e} \to \mathrm{t}) \to \mathrm{t}$
`keine`$_\sqcup$	(C/(D\C))/N	$\tau(\mathrm{N}) \to \tau(\mathrm{C/(D\backslash C)}) = \tau(\mathrm{N}) \to \tau(\mathrm{D\backslash C}) \to \tau(\mathrm{C}) =$ $\tau(\mathrm{N}) \to (\tau(\mathrm{D}) \to \tau(\mathrm{C})) \to \tau(\mathrm{C}) = (\mathrm{e} \to \mathrm{t}) \to (\mathrm{e} \to \mathrm{t}) \to \mathrm{t}$
`jede`$_\sqcup$	(C/(D\C))/N	$\tau(\mathrm{N}) \to \tau(\mathrm{C/(D\backslash C)}) = \tau(\mathrm{N}) \to \tau(\mathrm{D\backslash C}) \to \tau(\mathrm{C}) =$ $\tau(\mathrm{N}) \to (\tau(\mathrm{D}) \to \tau(\mathrm{C})) \to \tau(\mathrm{C}) = (\mathrm{e} \to \mathrm{t}) \to (\mathrm{e} \to \mathrm{t}) \to \mathrm{t}$

Gegeben seien außerdem die folgenden Grundelemente des semantischen Universums:

$M_e \coloneqq \{g, t, j, k, m, r\}$
$M_t \coloneqq \{0, 1\}$

Dies sind nun die Bedeutungen der Wörter:

Wort	semantischer Typ	Bedeutung
$\texttt{Garfield}_\sqcup$	e	g
$\texttt{Tom}_\sqcup$	e	t
$\texttt{Jerry}_\sqcup$	e	j
$\texttt{Mickey}_\sqcup$	e	m
$\texttt{Rémy}_\sqcup$	e	r
$\texttt{Maus}_\sqcup$	e → t	$\{\langle g,0\rangle,\langle t,0\rangle,\langle j,1\rangle,\langle k,1\rangle,\langle m,1\rangle,\langle r,0\rangle\} = f_{\texttt{Maus}_\sqcup}$
$\texttt{Ratte}_\sqcup$	e → t	$\{\langle g,0\rangle,\langle t,0\rangle,\langle j,0\rangle,\langle k,0\rangle,\langle m,0\rangle,\langle r,1\rangle\} = f_{\texttt{Ratte}_\sqcup}$
$\texttt{Katze}_\sqcup$	e → t	$\{\langle g,1\rangle,\langle t,1\rangle,\langle j,0\rangle,\langle k,0\rangle,\langle m,0\rangle,\langle r,0\rangle\} = f_{\texttt{Katze}_\sqcup}$
$\texttt{schnelle}_\sqcup$	(e → t) → e → t	$\{\langle f_{\texttt{Maus}_\sqcup},\{\langle g,0\rangle,\langle t,0\rangle,\langle j,1\rangle,\langle k,1\rangle,\langle m,0\rangle,\langle r,0\rangle\}\rangle,$ $\langle f_{\texttt{Ratte}_\sqcup},\{\langle g,0\rangle,\langle t,0\rangle,\langle j,0\rangle,\langle k,0\rangle,\langle m,0\rangle,\langle r,0\rangle\}\rangle,$ $\langle f_{\texttt{Katze}_\sqcup},\{\langle g,0\rangle,\langle t,1\rangle,\langle j,0\rangle,\langle k,0\rangle,\langle m,0\rangle,\langle r,0\rangle\}\rangle,\dots\} = f_{\texttt{schnelle}_\sqcup}$
$\texttt{fette}_\sqcup$	(e → t) → e → t	$\{\langle f_{\texttt{Maus}_\sqcup},\{\langle g,0\rangle,\langle t,0\rangle,\langle j,0\rangle,\langle k,1\rangle,\langle m,0\rangle,\langle r,0\rangle\}\rangle,$ $\langle f_{\texttt{Ratte}_\sqcup},\{\langle g,0\rangle,\langle t,0\rangle,\langle j,0\rangle,\langle k,0\rangle,\langle m,0\rangle,\langle r,0\rangle\}\rangle,$ $\langle f_{\texttt{Katze}_\sqcup},\{\langle g,1\rangle,\langle t,0\rangle,\langle j,0\rangle,\langle k,0\rangle,\langle m,0\rangle,\langle r,0\rangle\}\rangle,\dots\} = f_{\texttt{fette}_\sqcup}$
$\texttt{schnurrt}_\sqcup$	e → t	$\{\langle g,1\rangle,\langle t,1\rangle,\langle j,0\rangle,\langle k,0\rangle,\langle m,0\rangle,\langle r,0\rangle\} = f_{\texttt{schnurrt}_\sqcup}$
$\texttt{fängt}_\sqcup$	e → e → t	$\{\langle g,\{\langle g,0\rangle,\langle t,0\rangle,\langle j,0\rangle,\langle k,0\rangle,\langle m,0\rangle,\langle r,1\rangle\}\rangle,$ $\langle t,\{\langle g,0\rangle,\langle t,0\rangle,\langle j,1\rangle,\langle k,0\rangle,\langle m,0\rangle,\langle r,0\rangle\}\rangle,$ $\langle j,\{\langle g,0\rangle,\langle t,0\rangle,\langle j,0\rangle,\langle k,0\rangle,\langle m,0\rangle,\langle r,0\rangle\}\rangle,$ $\langle k,\{\langle g,0\rangle,\langle t,0\rangle,\langle j,0\rangle,\langle k,0\rangle,\langle m,0\rangle,\langle r,0\rangle\}\rangle,$ $\langle m,\{\langle g,0\rangle,\langle t,0\rangle,\langle j,0\rangle,\langle k,0\rangle,\langle m,0\rangle,\langle r,0\rangle\}\rangle,$ $\langle r,\{\langle g,0\rangle,\langle t,0\rangle,\langle j,0\rangle,\langle k,0\rangle,\langle m,0\rangle,\langle r,0\rangle\}\rangle\} = f_{\texttt{fängt}_\sqcup}$
$\texttt{jagt}_\sqcup$	e → e → t	$\{\langle g,\{\langle g,0\rangle,\langle t,0\rangle,\langle j,0\rangle,\langle k,0\rangle,\langle m,0\rangle,\langle r,1\rangle\}\rangle,$ $\langle t,\{\langle g,0\rangle,\langle t,0\rangle,\langle j,1\rangle,\langle k,1\rangle,\langle m,1\rangle,\langle r,0\rangle\}\rangle,$ $\langle j,\{\langle g,0\rangle,\langle t,1\rangle,\langle j,0\rangle,\langle k,0\rangle,\langle m,0\rangle,\langle r,0\rangle\}\rangle,$ $\langle k,\{\langle g,0\rangle,\langle t,0\rangle,\langle j,0\rangle,\langle k,0\rangle,\langle m,0\rangle,\langle r,0\rangle\}\rangle,$ $\langle m,\{\langle g,0\rangle,\langle t,0\rangle,\langle j,0\rangle,\langle k,1\rangle,\langle m,1\rangle,\langle r,0\rangle\}\rangle,$ $\langle r,\{\langle g,0\rangle,\langle t,0\rangle,\langle j,0\rangle,\langle k,0\rangle,\langle m,0\rangle,\langle r,0\rangle\}\rangle\} = f_{\texttt{jagt}_\sqcup}$
$\texttt{die}_\sqcup$	(e → t) → e	$f_{\texttt{die}_\sqcup}(f_x) = \iota(f_x) = \begin{cases} a, & \text{falls } a \text{ das einzige } z \text{ ist mit} f_x(z) = 1 \\ \text{☠}, & \text{sonst} \end{cases}$
$\texttt{eine}_\sqcup$	(e → t) → (e → t) → t	$(f_{\texttt{eine}_\sqcup}(f_x))(f_y) =$ $(\exists(f_x))(f_y) = \begin{cases} 1, & \text{falls es ein } z \text{ gibt mit } \langle z,1\rangle \in f_x \cap f_y \\ 0, & \text{sonst} \end{cases}$
$\texttt{keine}_\sqcup$	(e → t) → (e → t) → t	$(f_{\texttt{keine}_\sqcup}(f_x))(f_y) =$ $(-\exists(f_x))(f_y) = \begin{cases} 0, & \text{falls es ein } z \text{ gibt mit } \langle z,1\rangle \in f_x \cap f_y \\ 1, & \text{sonst} \end{cases}$
$\texttt{jede}_\sqcup$	(e → t) → (e → t) → t	$(f_{\texttt{jede}_\sqcup}(f_x))(f_y) =$ $(\forall(f_x))(f_y) = \begin{cases} 1, & \text{falls für alle } z \text{ mit } \langle z,1\rangle \in f_x \text{ auch } \langle z,1\rangle \in f_y \\ 0, & \text{sonst} \end{cases}$

Nun haben wir die Zeichen des Lexikons.

Ein Zeichen ist ein Tripel $\langle \vec{x}, \alpha, m \rangle$, wobei $\vec{x}$ ein Symbol ist, α eine Kategorie und m eine Bedeutung.

Wort	Kategorie	semantischer Typ	Bedeutung	Zeichen
$\texttt{Garfield}_\sqcup$	D	e	g	$\langle \texttt{Garfield}_\sqcup, \text{D}, g \rangle$
$\texttt{Tom}_\sqcup$	D	e	t	$\langle \texttt{Tom}_\sqcup, \text{D}, t \rangle$
$\texttt{Jerry}_\sqcup$	D	e	j	$\langle \texttt{Jerry}_\sqcup, \text{D}, j \rangle$
$\texttt{Mickey}_\sqcup$	D	e	m	$\langle \texttt{Mickey}_\sqcup, \text{D}, m \rangle$
$\texttt{Rémy}_\sqcup$	D	e	r	$\langle \texttt{Rémy}_\sqcup, \text{D}, r \rangle$
$\texttt{Maus}_\sqcup$	N	$\text{e} \to \text{t}$	$f_{\texttt{Maus}_\sqcup}$	$\langle \texttt{Maus}_\sqcup, \text{N}, f_{\texttt{Maus}_\sqcup} \rangle$
$\texttt{Ratte}_\sqcup$	N	$\text{e} \to \text{t}$	$f_{\texttt{Ratte}_\sqcup}$	$\langle \texttt{Ratte}_\sqcup, \text{N}, f_{\texttt{Ratte}_\sqcup} \rangle$
$\texttt{Katze}_\sqcup$	N	$\text{e} \to \text{t}$	$f_{\texttt{Katze}_\sqcup}$	$\langle \texttt{Katze}_\sqcup, \text{N}, f_{\texttt{Katze}_\sqcup} \rangle$
$\texttt{schnelle}_\sqcup$	N/N	$(\text{e} \to \text{t}) \to \text{e} \to \text{t}$	$f_{\texttt{schnelle}_\sqcup}$	$\langle \texttt{schnelle}_\sqcup, \text{N/N}, f_{\texttt{schnelle}_\sqcup} \rangle$
$\texttt{fette}_\sqcup$	N/N	$(\text{e} \to \text{t}) \to \text{e} \to \text{t}$	$f_{\texttt{fette}_\sqcup}$	$\langle \texttt{fette}_\sqcup, \text{N/N}, f_{\texttt{fette}_\sqcup} \rangle$
$\texttt{schnurrt}_\sqcup$	D\C	$\text{e} \to \text{t}$	$f_{\texttt{schnurrt}_\sqcup}$	$\langle \texttt{schnurrt}_\sqcup, \text{D}\backslash\text{C}, f_{\texttt{schnurrt}_\sqcup} \rangle$
$\texttt{fängt}_\sqcup$	(D\C)/D	$\text{e} \to \text{e} \to \text{t}$	$f_{\texttt{fängt}_\sqcup}$	$\langle \texttt{fängt}_\sqcup, (\text{D}\backslash\text{C})/\text{D}, f_{\texttt{fängt}_\sqcup} \rangle$
$\texttt{jagt}_\sqcup$	(D\C)/D	$\text{e} \to \text{e} \to \text{t}$	$f_{\texttt{jagt}_\sqcup}$	$\langle \texttt{jagt}_\sqcup, (\text{D}\backslash\text{C})/\text{D}, f_{\texttt{jagt}_\sqcup} \rangle$
$\texttt{die}_\sqcup$	D/N	$(\text{e} \to \text{t}) \to \text{e}$	ι	$\langle \texttt{die}_\sqcup, \text{D/N}, \iota \rangle$
$\texttt{eine}_\sqcup$	(C/(D\C))/N	$(\text{e} \to \text{t}) \to (\text{e} \to \text{t}) \to \text{t}$	$\exists$	$\langle \texttt{eine}_\sqcup, (\text{C}/(\text{D}\backslash\text{C}))/\text{N}, \exists \rangle$
$\texttt{keine}_\sqcup$	(C/(D\C))/N	$(\text{e} \to \text{t}) \to (\text{e} \to \text{t}) \to \text{t}$	$-\exists$	$\langle \texttt{keine}_\sqcup, (\text{C}/(\text{D}\backslash\text{C}))/\text{N}, -\exists \rangle$
$\texttt{jede}_\sqcup$	(C/(D\C))/N	$(\text{e} \to \text{t}) \to (\text{e} \to \text{t}) \to \text{t}$	$\forall$	$\langle \texttt{jede}_\sqcup, (\text{C}/(\text{D}\backslash\text{C}))/\text{N}, \forall \rangle$

1.3.3 Übungen: Kategorialgrammatiken 2

1. Gegeben sei das Lexikon aus dem Dokument "Ein Lexikon".

   (a) Betrachte folgende Sätze. Gib an, welche grammatisch sind. Zeichne dazu die Strukturbäume.

      i. `Garfield␣schnurrt␣`
      ii. `Tom␣jagt␣Jerry␣`
      iii. `die␣fette␣Katze␣schnurrt␣`
      iv. `jede␣fette␣schnelle␣Katze␣schnurrt␣`
      v. `eine␣schnelle␣Katze␣fängt␣Rémy␣`
      vi. `jede␣schnelle␣Katze␣jagt␣die␣Ratte␣`
      vii. `Tom␣jagt␣die␣schnelle␣Maus␣`
      viii. `keine␣Katze␣fängt␣eine␣Maus␣`
      ix. `jede␣schnelle␣Katze␣fängt␣jede␣fette␣Maus␣`
      x. `Mickey␣fängt␣keine␣schnelle␣Katze␣`

   (b) Bestimme die Wahrheitswerte der grammatischen Sätze.

   (c) Wie müsste man das Lexikon erweitern, damit alle obigen Sätze grammatisch und interpretierbar sind?

   (d) Bestimme die Wahrheitswerte der Sätze, die (nur) auf der Grundlage deines neuen Lexikons grammatisch und interpretierbar sind.

   (e) Betrachte folgende Sätze. Wie müsste das Lexikon erweitert werden, damit sie grammatisch und interpretierbar sind?

      i. `Garfield␣schnurrt␣und␣Tom␣jagt␣Jerry␣`
      ii. `Tom␣jagt␣Jerry␣und␣schnurrt␣`
      iii. `Tom␣jagt␣und␣fängt␣Jerry␣`

   (f) Sind auf der Grundlage deines neuen Lexikons nun Sätze grammatisch, die intuitiv nicht grammatisch sind?

1.3.4 Hilfe: Ein paar Sätze

- $\texttt{Garfield}_{\sqcup}\texttt{schnurrt}_{\sqcup}$

  Struktur:

  $[[\texttt{Garfield}_{\sqcup}]_D[\texttt{schnurrt}_{\sqcup}]_{D\backslash C}]_C$

  Bedeutung:

  $$\frac{\langle \texttt{Garfield}_{\sqcup}, D, g\rangle \quad \langle \texttt{schnurrt}_{\sqcup}, D\backslash C, f_{\texttt{schnurrt}_{\sqcup}}\rangle}{\langle \texttt{Garfield}_{\sqcup}\texttt{schnurrt}_{\sqcup}, C, f_{\texttt{schnurrt}_{\sqcup}}(g) = 1\rangle}$$

- $\texttt{Tom}_{\sqcup}\texttt{jagt}_{\sqcup}\texttt{Jerry}_{\sqcup}$

  Struktur:

  $[[\texttt{Tom}_{\sqcup}]_D[[\texttt{jagt}_{\sqcup}]_{(D\backslash C)/D}[\texttt{Jerry}_{\sqcup}]_D]_{D\backslash C}]_C$

  Bedeutung:

  $$\frac{\langle \texttt{Tom}_{\sqcup}, D, g\rangle \qquad \dfrac{\langle \texttt{jagt}_{\sqcup}, (D\backslash C)/D, f_{\texttt{jagt}_{\sqcup}}\rangle \quad \langle \texttt{Jerry}_{\sqcup}, D, j\rangle}{\langle \texttt{jagt}_{\sqcup}\texttt{Jerry}_{\sqcup}, D\backslash C, f_{\texttt{jagt}_{\sqcup}}(j)\rangle}}{\langle \texttt{Tom}_{\sqcup}\texttt{jagt}_{\sqcup}\texttt{Jerry}_{\sqcup}, C, (f_{\texttt{jagt}_{\sqcup}}(j))(t) = 1\rangle}$$

- $\texttt{die}_{\sqcup}\texttt{fette}_{\sqcup}\texttt{Katze}_{\sqcup}\texttt{schnurrt}_{\sqcup}$

  Struktur:

  $[[[\texttt{die}_{\sqcup}]_{D/N}[[\texttt{fette}_{\sqcup}]_{N/N}[\texttt{Katze}_{\sqcup}]_N]_N]_D[\texttt{schnurrt}_{\sqcup}]_{D\backslash C}]_C$

  Bedeutung:

  $$\frac{\dfrac{\langle \texttt{die}_{\sqcup}, D/N, \iota\rangle \qquad \dfrac{\langle \texttt{fette}_{\sqcup}, N/N, f_{\texttt{fette}_{\sqcup}}\rangle \quad \langle \texttt{Katze}_{\sqcup}, N, f_{\texttt{Katze}_{\sqcup}}\rangle}{\langle \texttt{fette}_{\sqcup}\texttt{Katze}_{\sqcup}, N, f_{\texttt{fette}_{\sqcup}}(f_{\texttt{Katze}_{\sqcup}})\rangle}}{\langle \texttt{die}_{\sqcup}\texttt{fette}_{\sqcup}\texttt{Katze}_{\sqcup}, D, \iota(f_{\texttt{fette}_{\sqcup}}(f_{\texttt{Katze}_{\sqcup}})) = g\rangle} \qquad \langle \texttt{schnurrt}_{\sqcup}, D\backslash C, f_{\texttt{schnurrt}_{\sqcup}}\rangle}{\langle \texttt{die}_{\sqcup}\texttt{fette}_{\sqcup}\texttt{Katze}_{\sqcup}\texttt{schnurrt}_{\sqcup}, C, f_{\texttt{schnurrt}_{\sqcup}}(g) = 1\rangle}$$

- $\texttt{jede␣fette␣schnelle␣Katze␣schnurrt␣}$

  Struktur:

  $[[[\texttt{jede␣}]_{(C/(D\backslash C))/N}[[\texttt{fette␣}]_{N/N}[[\texttt{schnelle␣}]_{N/N}[\texttt{Katze␣}]_N]_N]_N]_{C/(D\backslash C)}[\texttt{schnurrt␣}]_{D\backslash C}]_C$

  Bedeutung:

$$\dfrac{\dfrac{\langle \texttt{jede␣}, (C/(D\backslash C))/N, \forall\rangle \quad \dfrac{\langle \texttt{fette␣}, N/N, f_{\texttt{fette␣}}\rangle \quad \dfrac{\langle \texttt{schnelle␣}, N/N, f_{\texttt{schnelle␣}}\rangle \quad \langle \texttt{Katze␣}, N, f_{\texttt{Katze␣}}\rangle}{\langle \texttt{schnelle␣Katze␣}, N, f_{\texttt{schnelle␣}}(f_{\texttt{Katze␣}})\rangle}}{\langle \texttt{fette␣schnelle␣Katze␣}, N, f_{\texttt{fette␣}}(f_{\texttt{schnelle}}(f_{\texttt{Katze␣}}))\rangle}}{\langle \texttt{jede␣fette␣schnelle␣Katze␣}, C/(D\backslash C), \forall(f_{\texttt{fette␣}}(f_{\texttt{schnelle␣}}(f_{\texttt{Katze␣}})))\rangle} \quad \langle \texttt{schnurrt␣}, D\backslash C, f_{\texttt{schnurrt␣}}\rangle}{\langle \texttt{jede␣fette␣schnelle␣Katze␣schnurrt␣}, C, (\forall(f_{\texttt{fette␣}}(f_{\texttt{schnelle␣}}(f_{\texttt{Katze␣}}))))(f_{\texttt{schnurrt␣}}) = 1\rangle}$$

- $\texttt{eine␣schnelle␣Katze␣fängt␣Rémy␣}$

  Struktur:

  $[[[\texttt{eine␣}]_{(C/(D\backslash C))/N}[[\texttt{schnelle␣}]_{N/N}[\texttt{Katze␣}]_N]_N]_{C/(D\backslash C)}[[\texttt{fängt␣}]_{(D\backslash C)/D}[\texttt{Rémy␣}]_D]_{D\backslash C}]_C$

  Bedeutung:

$$\dfrac{\dfrac{\langle \texttt{eine␣}, (C/(D\backslash C))/N, \exists\rangle \quad \dfrac{\langle \texttt{schnelle␣}, N/N, f_{\texttt{schnelle␣}}\rangle \quad \langle \texttt{Katze␣}, N, f_{\texttt{Katze␣}}\rangle}{\langle \texttt{schnelle␣Katze␣}, N, f_{\texttt{schnelle␣}}(f_{\texttt{Katze␣}})\rangle}}{\langle \texttt{eine␣schnelle␣Katze␣}, C/(D\backslash C), \exists(f_{\texttt{schnelle␣}}(f_{\texttt{Katze␣}}))\rangle} \quad \dfrac{\langle \texttt{fängt␣}, (D\backslash C)/D, f_{\texttt{fängt␣}}\rangle \quad \langle \texttt{Rémy␣}, D, r\rangle}{\langle \texttt{fängt␣Rémy␣}, D\backslash C, f_{\texttt{fängt␣}}(r)\rangle}}{\langle \texttt{eine␣schnelle␣Katze␣fängt␣Rémy␣}, C, (\exists(f_{\texttt{schnelle␣}}(f_{\texttt{Katze␣}})))(f_{\texttt{fängt␣}}(r)) = 1\rangle}$$

- $\texttt{jede␣schnelle␣Katze␣jagt␣die␣Maus␣}$

  Struktur:

  $[[[\texttt{jede␣}]_{(C/(D\backslash C))/N}[[\texttt{schnelle␣}]_{N/N}[\texttt{Katze␣}]_N]_N]_{C/(D\backslash C)}[[\texttt{jagt␣}]_{(D\backslash C)/D}[[\texttt{die␣}]_{D/N}[\texttt{Maus␣}]_N]_D]_{D\backslash C}]_C$

Bedeutung:

$$\dfrac{\dfrac{\langle \texttt{jede}_\sqcup, (C/(D\backslash C))/N, \forall\rangle \quad \dfrac{\langle \texttt{schnelle}_\sqcup, N/N, f_{\texttt{schnelle}_\sqcup}\rangle \quad \langle \texttt{Katze}_\sqcup, N, f_{\texttt{Katze}_\sqcup}\rangle}{\langle \texttt{schnelle}_\sqcup\texttt{Katze}_\sqcup, N, f_{\texttt{schnelle}_\sqcup}(f_{\texttt{Katze}_\sqcup})\rangle}}{\langle \texttt{jede}_\sqcup\texttt{schnelle}_\sqcup\texttt{Katze}_\sqcup, C/(D\backslash C), \forall(f_{\texttt{schnelle}_\sqcup}(f_{\texttt{Katze}_\sqcup}))\rangle} \quad \dfrac{\langle \texttt{jagt}_\sqcup, (D\backslash C)/D, f_{\texttt{jagt}_\sqcup}\rangle \quad \dfrac{\langle \texttt{die}_\sqcup, D/N, \iota\rangle \quad \langle \texttt{Maus}_\sqcup, N, f_{Maus}\rangle}{\langle \texttt{die}_\sqcup\texttt{Maus}_\sqcup, D\backslash C, \iota(f_{\texttt{Maus}_\sqcup} = \text{☠})\rangle}}{\langle \texttt{jagt}_\sqcup\texttt{die}_\sqcup\texttt{Maus}_\sqcup, D\backslash C, \text{☠}\rangle}}{\langle \texttt{jede}_\sqcup\texttt{schnelle}_\sqcup\texttt{Katze}_\sqcup\texttt{jagt}_\sqcup\texttt{die}_\sqcup\texttt{Maus}_\sqcup, C, \text{☠}\rangle}$$

- $\texttt{jede}_\sqcup\texttt{schnelle}_\sqcup\texttt{Katze}_\sqcup\texttt{jagt}_\sqcup\texttt{die}_\sqcup\texttt{fette}_\sqcup\texttt{Maus}_\sqcup$

  Struktur:

  $[[[\texttt{jede}_\sqcup]_{(C/(D\backslash C))/N}[[\texttt{schnelle}_\sqcup]_{N/N}[\texttt{Katze}_\sqcup]_N]_N]_{C/(D\backslash C)}[[\texttt{jagt}_\sqcup]_{(D\backslash C)/D}[[\texttt{die}_\sqcup]_{D/N}[[\texttt{fette}_\sqcup]_{N/N}[\texttt{Maus}_\sqcup]_N]_N]_D]_{D\backslash C}]_C$

  Bedeutung:

$$\dfrac{\dfrac{\langle \texttt{jede}_\sqcup, (C/(D\backslash C))/N, \forall\rangle \quad \dfrac{\langle \texttt{schnelle}_\sqcup, N/N, f_{\texttt{schnelle}_\sqcup}\rangle \quad \langle \texttt{Katze}_\sqcup, N, f_{\texttt{Katze}_\sqcup}\rangle}{\langle \texttt{schnelle}_\sqcup\texttt{Katze}_\sqcup, N, f_{\texttt{schnelle}_\sqcup}(f_{\texttt{Katze}_\sqcup})\rangle}}{\langle \texttt{jede}_\sqcup\texttt{schnelle}_\sqcup\texttt{Katze}_\sqcup, C/(D\backslash C), \forall(f_{\texttt{schnelle}_\sqcup}(f_{\texttt{Katze}_\sqcup}))\rangle} \quad \dfrac{\langle \texttt{jagt}_\sqcup, (D\backslash C)/D, f_{\texttt{jagt}_\sqcup}\rangle \quad \dfrac{\langle \texttt{die}_\sqcup, D/N, \iota\rangle \quad \dfrac{\langle \texttt{fette}_\sqcup, N/N, f_{\texttt{fette}_\sqcup}\rangle \quad \langle \texttt{Maus}_\sqcup, N, f_{\texttt{Maus}_\sqcup}\rangle}{\langle \texttt{fette}_\sqcup\texttt{Maus}_\sqcup, N, f_{\texttt{fette}_\sqcup}(f_{\texttt{Maus}_\sqcup})\rangle}}{\langle \texttt{die}_\sqcup\texttt{fette}_\sqcup\texttt{Maus}_\sqcup, D\backslash C, \iota(f_{\texttt{fette}_\sqcup}(f_{\texttt{Maus}_\sqcup})) = k\rangle}}{\langle \texttt{jagt}_\sqcup\texttt{die}_\sqcup\texttt{fette}_\sqcup\texttt{Maus}_\sqcup, D\backslash C, f_{\texttt{jagt}_\sqcup}(k)\rangle}}{\langle \texttt{jede}_\sqcup\texttt{schnelle}_\sqcup\texttt{Katze}_\sqcup\texttt{jagt}_\sqcup\texttt{die}_\sqcup\texttt{fette}_\sqcup\texttt{Maus}_\sqcup, C, (\forall(f_{\texttt{fette}_\sqcup}(f_{\texttt{Katze}_\sqcup}))(f_{\texttt{jagt}_\sqcup}(k)) = 1\rangle}$$

2 Formale Sprachen und Automaten

2.1 Formale Sprachen

2.1.1 Übungen: Buchstaben und Zeichenketten

1. Betrachte die Zeichenkette $\texttt{Weihnachtsmarkt}_\sqcup$.
   (a) Benenne die Funktion der Zeichenkette explizit.
   (b) Bestimme alle Präfixe, Postfixe und Teilwörter der Zeichenkette.
   (c) Verifiziere die Definition der Verkettung anhand der Zerlegung $\texttt{Weihnachtsmarkt}_\sqcup = \texttt{Weihnachts} \cdot \texttt{markt}_\sqcup$.
   (d) Gib ‘‘$\texttt{Weihnachtsmarkt}_\sqcup$’’.[0] und ‘‘$\texttt{Weihnachtsmarkt}_\sqcup$’’.[9] an.
   (e) Für welches n gilt ‘‘$\texttt{Weihnachtsmarkt}_\sqcup$’’.[n] = ‘$\sqcup$’? Wie kann man ‘$\sqcup$’ mit ‘‘$\texttt{Weihnachtsmarkt}_\sqcup$’’.[n] bekommen, ohne ein bestimmtes n anzugeben?
   (f) Finde zwei Vorkommen von Teilwörtern, die sich *überlappen*.
   (g) Versuche, den Begriff der Überlappung formal präzise zu definieren.

2. Es sei $P := \{\texttt{Bielefelder}_\sqcup, \texttt{Herforder}_\sqcup, \texttt{Paderborner}_\sqcup\}$, $Q := \{\texttt{Weihnachtsmarkt}_\sqcup, \texttt{Adventsbasar}_\sqcup\}$ und $R := \{\texttt{markt}_\sqcup, \texttt{basar}_\sqcup\}$.
   (a) Berechne $P \cdot Q$.
   (b) Berechne Q/R.
   (c) Berechne $(Q/R)\backslash R$.
   (d) Berechne $(Q/R) \cdot R$.

3. Es sei $S := \{\texttt{Frank}_\sqcup, \texttt{Susi}_\sqcup\}$, $T := \{\texttt{ist}_\sqcup\}$ und $U := \{\texttt{witzig}_\sqcup, \texttt{klug}_\sqcup\}$.
   (a) Berechne $V := S \cdot T \cdot U$.
   (b) Berechne $(V/T)/U$.
   (c) Berechne $V/(T \cdot U)$.
   (d) Berechne $(V/U)/T$.

4. Es sei die Zeichenkette $\vec{x}_0 = 1$ gegeben. Induktiv gehen die $\vec{x}_{n+1}$ aus $\vec{x}_n$ wie folgt hervor. $\vec{x}_{n+1}$ entspricht der Beschreibung des Vorgängers im Dezimalsystem. Man bestimme die maximale Länge der Blöcke gleicher Ziffern im Vorgänger und schreibe die Häufigkeit und Ziffer für jeden Block hintereinander. So ist z.B. $\vec{x}_0 = \texttt{1}$ *eine Eins*. Der Nachfolger ist damit die Zeichenkette $\vec{x}_1 = \texttt{11}$. Nun haben wir *zwei Einsen*. Der Nachfolger ist damit die Zeichenkette $\vec{x}_2 = \texttt{21}$. Nun haben wir *eine Zwei* und *eine Eins*. Der Nachfolger ist damit die Zeichenkette $\vec{x}_3 = \texttt{1211}$. Wie geht es weiter?

2.1.2 Hilfe: Eine Erläuterung zur Sprache L/M

Betrachten wir folgende Definition:

- $L/M := \{\vec{x} : \text{für alle } \vec{y} \in M \text{ gilt, dass } \vec{x} \cdot \vec{y} \in L\}$

Wie ist diese Definition zu verstehen? Die angegebene Menge sammelt alle Zeichenketten $\vec{x}$ für die gilt: Egal, welche Zeichenkette $\vec{y}$ aus M man an $\vec{x}$ hängt, die Verkettung $\vec{x} \cdot \vec{y}$ ist in L. Beispielsweise haben wir die Mengen

- $P := \{\texttt{Bielefelder}_\sqcup, \texttt{Herforder}_\sqcup, \texttt{Paderborner}_\sqcup\}$
- $Q := \{\texttt{Weihnachtsmarkt}_\sqcup, \texttt{Adventsbasar}_\sqcup\}$
- $R := \{\texttt{markt}_\sqcup, \texttt{basar}_\sqcup\}$

Dann ist beispielsweise Q/R die Menge, die alle Zeichenketten $\vec{x}$ sammelt, für die gilt: Egal, welche Zeichenkette $\vec{y}$ aus $R = \{\texttt{markt}_\sqcup, \texttt{basar}_\sqcup\}$ man an $\vec{x}$ hängt, die Verkettung $\vec{x} \cdot \vec{y}$ ist in $Q = \{\texttt{Weihnachtsmarkt}_\sqcup, \texttt{Adventsbasar}_\sqcup\}$. Welche Zeichenketten sind das? Es gibt einen naheliegenden Vorschlag:

- $\{\texttt{Weihnachts}, \texttt{Advents}\}$

Schließlich ergibt $\texttt{Weihnachts} \cdot \texttt{markt}_\sqcup = \texttt{Weihnachtsmarkt}_\sqcup$ und $\texttt{Advents} \cdot \texttt{basar}_\sqcup = \texttt{Adventsbasar}_\sqcup$. Doch erfüllen diese Zeichenketten auch wirklich die angegebene Bedingung? Nehmen wir die Zeichenkette $\texttt{Weihnachts}$. Wenn $\texttt{Weihnachts}$ wirklich in Q/R ist, dann muss gelten, dass, *egal welche* Zeichenkette $\vec{y}$ aus $R = \{\texttt{markt}_\sqcup, \texttt{basar}_\sqcup\}$ man an $\texttt{Weihnachts}$ hängt, die Verkettung $\texttt{Weihnachts} \cdot \vec{y}$ in $Q = \{\texttt{Weihnachtsmarkt}_\sqcup, \texttt{Adventsbasar}_\sqcup\}$ ist. Doch wir sehen, dass zwar $\texttt{Weihnachtsmarkt}_\sqcup$ in $Q = \{\texttt{Weihnachtsmarkt}_\sqcup, \texttt{Adventsbasar}_\sqcup\}$ ist, $\texttt{Weihnachtsbasar}_\sqcup$ aber nicht! $\texttt{Weihnachtsbasar}_\sqcup$ müsste aber in Q sein, damit die Bedingung für die Zeichenkette $\texttt{Weihnachts}$ erfüllt ist.

Der Vorschlag ist damit falsch. Doch für welche Zeichenketten $\vec{x}$ gilt dann, dass, egal welche Zeichenkette $\vec{y}$ aus $R = \{\texttt{markt}_\sqcup, \texttt{basar}_\sqcup\}$ man an $\vec{x}$ hängt, die Verkettung $\vec{x} \cdot \vec{y}$ in $Q = \{\texttt{Weihnachtsmarkt}_\sqcup, \texttt{Adventsbasar}_\sqcup\}$ ist? Offenbar für *gar keine*! Und damit ist $Q/R = \varnothing$.

2.1.3 Hilfe: L/M und M\L: Spezialfälle

Es sei A^* eine Sprache über A und $L, M \subseteq A^*$. Betrachten wir folgende Definitionen:

- $L/M := \{\vec{x} : \text{für alle } \vec{y} \in M \text{ gilt, dass } \vec{x} \cdot \vec{y} \in L\} = \{\vec{x} : \text{es gibt kein } \vec{y} \in M \text{ mit } \vec{x} \cdot \vec{y} \notin L\}$
- $M \backslash L := \{\vec{x} : \text{für alle } \vec{y} \in M \text{ gilt, dass } \vec{y} \cdot \vec{x} \in L\} = \{\vec{x} : \text{es gibt kein } \vec{y} \in M \text{ mit } \vec{y} \cdot \vec{x} \notin L\}$

Zu diesen Definitionen gibt es nun ein paar interessante Spezialfälle. Nur wenn man diese versteht, hat man die obigen Definitionen auch vollständig verstanden.

Spezialfall 1: $M = L$, $M \neq \varnothing$, $L \neq \varnothing$, M, L endlich

- $L/M = M/M = \{\epsilon\}$
- $M \backslash L = M \backslash M = \{\epsilon\}$

Spezialfall 2: $M = \varnothing$, $L \neq \varnothing$

- $L/M = L/\varnothing = A^*$

- $M \backslash L = \varnothing \backslash L = A^*$

Spezialfall 3: $L = \varnothing$, $M = \varnothing$

- $L/M = \varnothing/M = \varnothing$
- $M \backslash L = M \backslash \varnothing = \varnothing$

Spezialfall 4: $L = M = \varnothing$

- $L/M = \varnothing/\varnothing = A^*$
- $M \backslash L = \varnothing \backslash \varnothing = A^*$

2.1.4 Übungen: Reguläre Ausdrücke

1. Berechne.

   (a) $\{\mathtt{a}, \mathtt{b}, \mathtt{c}\}^0$

   (b) $\{\mathtt{a}, \mathtt{b}, \mathtt{c}\}^1$

   (c) $\{\mathtt{a}, \mathtt{b}, \mathtt{c}\}^3$

   (d) $\{\mathtt{a}, \mathtt{b}, \mathtt{c}\}^*$

2. Berechne.

   (a) $\mathrm{S}(\mathtt{(ho)^*(hi)^*})$

   (b) $\mathrm{S}(\mathtt{(ho)^*(hi)^*(ho)?})$

   (c) $\mathrm{S}(\mathtt{a^3bbc^*})$

   (d) $\mathrm{S}(\mathtt{((a^*b)^*c)^*})$

   (e) $\mathrm{S}(\mathtt{a^*(b \cup \epsilon)c?c^*})$

   (f) $\mathrm{S}(\mathtt{aa^*((b \cup b)c)^+})$

3. Finde jeweils einen regulären Ausdruck, der die Sprache beschreibt.

   (a) $\{\mathtt{a}\}^* - \{\mathtt{a}\}$

   (b) $\{\mathtt{a}, \mathtt{b}\}^* - \{\mathtt{a}, \mathtt{b}\}$

   (c) $\{\epsilon, \mathtt{en}, \mathtt{enen}, \mathtt{enenen}, \mathtt{enenenen}, \dots\}$

   (d) $\{\mathtt{Stern}, \mathtt{Sterne}\}$

   (e) $\{\mathtt{a}^{n+1}\mathtt{b}^m : n, m \in \mathbb{N} - \{0\}\}$

   (f) $\{\mathtt{(aa)}^n\mathtt{c} : n \in \mathbb{N} - \{0\}\}$

   (g) $\{\mathtt{(ho)}^l\mathtt{(hoo)}^m\mathtt{(hooo)}^n : l, m, n \in \mathbb{N}\}$

   (h) $\{\mathtt{Hu}^n\mathtt{nger!}^m : n, m \in \mathbb{N} - \{0, 1, 2\}\}$

4. Gib jeweils einen regulären Ausdruck X an. Welche Lösungen sind eindeutig?

   (a) $X = \mathtt{c} \cup \mathtt{ab}X$

   (b) $X = \mathtt{ac^*} \cup \mathtt{(a \cup d)}X$

   (c) $X = \mathtt{(ab)^+}X \cup \mathtt{(a \cup ca)}$

   (d) $X = \mathtt{(h \cup w^*c)}X \cup \mathtt{(e? \cup p)}$

5. Gib jeweils für alle X_i eine Lösung an, indem du das Gleichungssystem löst.

   (a)
   $$\begin{aligned} X_0 &= \mathtt{c} \cup \mathtt{a}X_0 \cup \mathtt{b}X_1 \\ X_1 &= \mathtt{b}X_0 \cup \mathtt{c}X_1 \end{aligned}$$

   (b)
   $$\begin{aligned} X_0 &= \mathtt{a}X_0 \cup (\mathtt{e} \cup \mathtt{f?})X_2 \\ X_1 &= \mathtt{b}X_0 \cup \epsilon \\ X_2 &= (\mathtt{bc})^+ \cup (\mathtt{cc})?X_1 \cup X_2 \end{aligned}$$

   (c)
   $$\begin{aligned} X_0 &= X_0 \cup \mathtt{q}\mathtt{r}X_1 \\ X_1 &= \mathtt{s}?X_0 \cup \mathtt{r}^3X_1 \cup \mathtt{s}X_2 \\ X_2 &= (\mathtt{s} \cup \mathtt{q})?X_0 \cup \mathtt{r}\mathtt{r}X_2 \end{aligned}$$

6. Finde für die Gleichungssysteme aus der vorigen Aufgabe jeweils andere reguläre Ausdrücke für die X_i. In welcher Beziehung stehen die verschiedenen Ausdrücke zueinander?

7. Es sei P gegeben. Welche Bedingung muss X erfüllen, damit die Gleichung $X = P \cup X$ gilt? Es sei Q eine weitere Sprache mit $\epsilon \in Q$. Welche Sprachen X erfüllen die Gleichung $X = P \cup QX$?

2.1.5 Hilfe: Ein paar Gleichungssysteme

LGS 1:

$$\begin{aligned} X_0 &= \mathtt{z}\mathtt{w}X_0 \cup \mathtt{y}X_1 \cup X_2 \\ X_1 &= \mathtt{y}X_0 \cup \mathtt{z}X_1 \\ X_2 &= \mathtt{w}^+X_1 \cup \mathtt{y}X_2 \end{aligned}$$

- Wir wenden Satz 11.5 auf die zweite Gleichung an:
  $$\begin{aligned} X_0 &= \mathtt{z}\mathtt{w}X_0 \cup \mathtt{y}X_1 \cup X_2 \\ X_1 &= \mathtt{z}^*\mathtt{y}X_0 \\ X_2 &= \mathtt{w}^+X_1 \cup \mathtt{y}X_2 \end{aligned}$$
- Wir wenden Satz 11.5 auf die dritte Gleichung an:
  $$\begin{aligned} X_0 &= \mathtt{z}\mathtt{w}X_0 \cup \mathtt{y}X_1 \cup X_2 \\ X_1 &= \mathtt{z}^*\mathtt{y}X_0 \\ X_2 &= \mathtt{y}^*\mathtt{w}^+X_1 \end{aligned}$$
- Wir setzen den rechten Term der zweiten und der dritten Gleichung in die erste Gleichung ein:
  $$\begin{aligned} X_0 &= \mathtt{z}\mathtt{w}X_0 \cup \mathtt{y}\mathtt{z}^*\mathtt{y}X_0 \cup \mathtt{y}^*\mathtt{w}^+X_1 \\ X_1 &= \mathtt{z}^*\mathtt{y}X_0 \\ X_2 &= \mathtt{y}^*\mathtt{w}^+X_1 \end{aligned}$$

- Wir setzen den rechten Term der zweiten Gleichung in die erste Gleichung ein:

$$\begin{aligned} X_0 &= \mathtt{zw}X_0 \cup \mathtt{yz^*y}X_0 \cup \mathtt{y^*w^+z^*y}X_0 \\ X_1 &= \mathtt{z^*y}X_0 \\ X_2 &= \mathtt{y^*w^+}X_1 \end{aligned}$$

- Wir formen die erste Gleichung um:

$$\begin{aligned} X_0 &= \mathtt{(zw \cup yz^*y \cup y^*w^+z^*y)}X_0 \\ X_1 &= \mathtt{z^*y}X_0 \\ X_2 &= \mathtt{y^*w^+}X_1 \end{aligned}$$

- Wir erweitern die erste Gleichung:

$$\begin{aligned} X_0 &= \mathtt{0} \cup \mathtt{(zw \cup yz^*y \cup y^*w^+z^*y)}X_0 \\ X_1 &= \mathtt{z^*y}X_0 \\ X_2 &= \mathtt{y^*w^+}X_1 \end{aligned}$$

- Wir wenden Satz 11.5 auf die erste Gleichung an:

$$\begin{aligned} X_0 &= \mathtt{(zw \cup yz^*y \cup y^*w^+z^*y)^*0} \\ X_1 &= \mathtt{z^*y}X_0 \\ X_2 &= \mathtt{y^*w^+}X_1 \end{aligned}$$

- Wir setzen den rechten Term der ersten Gleichung in die zweite Gleichung ein:

$$\begin{aligned} X_0 &= \mathtt{(zw \cup yz^*y \cup y^*w^+z^*y)^*0} \\ X_1 &= \mathtt{z^*y(zw \cup yz^*y \cup y^*w^+z^*y)^*0} \\ X_2 &= \mathtt{y^*w^+}X_1 \end{aligned}$$

- Wir setzen den rechten Term der zweiten Gleichung in die dritte Gleichung ein:

$$\begin{aligned} X_0 &= \mathtt{(zw \cup yz^*y \cup y^*w^+z^*y)^*0} \\ X_1 &= \mathtt{z^*y(zw \cup yz^*y \cup y^*w^+z^*y)^*0} \\ X_1 &= \mathtt{y^*w^+z^*y(zw \cup yz^*y \cup y^*w^+z^*y)^*0} \end{aligned}$$

LGS 2:

$$\begin{aligned} X_0 &= \mathtt{zw}X_0 \cup \mathtt{y}X_1 \cup X_2 \\ X_1 &= \mathtt{y}X_0 \cup \mathtt{z}X_1 \\ X_2 &= \mathtt{w^+}X_1 \cup \mathtt{y}X_2 \end{aligned}$$

- Wir wenden Satz 11.5 auf die zweite Gleichung an:

$$\begin{aligned} X_0 &= \mathtt{zw}X_0 \cup \mathtt{y}X_1 \cup X_2 \\ X_1 &= \mathtt{z^*y}X_0 \\ X_2 &= \mathtt{w^+}X_1 \cup \mathtt{y}X_2 \end{aligned}$$

- Wir setzen den rechten Term der zweiten Gleichung in die dritte Gleichung ein:

$$\begin{aligned} X_0 &= \quad \mathtt{zw}X_0 \quad \cup \quad \mathtt{y}X_1 \quad \cup \quad X_2 \\ X_1 &= \quad \mathtt{z}^*\mathtt{y}X_0 \\ X_2 &= \quad \mathtt{w}^+\mathtt{z}^*\mathtt{y}X_0 \qquad\qquad \cup \quad \mathtt{y}X_2 \end{aligned}$$

- Wir wenden Satz 11.5 auf die dritte Gleichung an:

$$\begin{aligned} X_0 &= \quad \mathtt{zw}X_0 \quad \cup \quad \mathtt{y}X_1 \quad \cup \quad X_2 \\ X_1 &= \quad \mathtt{z}^*\mathtt{y}X_0 \\ X_2 &= \quad \mathtt{y}^*\mathtt{w}^+\mathtt{z}^*\mathtt{y}X_0 \end{aligned}$$

- Wir wenden Satz 11.5 auf die erste Gleichung an:

$$\begin{aligned} X_0 &= \quad (\mathtt{zw})^* \qquad\qquad (\mathtt{y}X_1 \quad \cup \quad X_2) \\ X_1 &= \quad \mathtt{z}^*\mathtt{y}X_0 \\ X_2 &= \quad \mathtt{y}^*\mathtt{w}^+\mathtt{z}^*\mathtt{y}X_0 \end{aligned}$$

- Wir formen die erste Gleichung um:

$$\begin{aligned} X_0 &= \quad (\mathtt{zw})^*\mathtt{y}X_1 \quad \cup \quad (\mathtt{zw})^*X_2 \\ X_1 &= \quad \mathtt{z}^*\mathtt{y}X_0 \\ X_2 &= \quad \mathtt{y}^*\mathtt{w}^+\mathtt{z}^*\mathtt{y}X_0 \end{aligned}$$

- Wir setzen den rechten Term der zweiten und der dritten Gleichung in die erste Gleichung ein:

$$\begin{aligned} X_0 &= \quad (\mathtt{zw})^*\mathtt{yz}^*\mathtt{y}X_0 \quad \cup \quad (\mathtt{zw})^*\mathtt{y}^*\mathtt{w}^+\mathtt{z}^*\mathtt{y}X_0 \\ X_1 &= \quad \mathtt{z}^*\mathtt{y}X_0 \\ X_2 &= \quad \mathtt{y}^*\mathtt{w}^+\mathtt{z}^*\mathtt{y}X_0 \end{aligned}$$

- Wir formen die erste Gleichung um:

$$\begin{aligned} X_0 &= \quad ((\mathtt{zw})^*\mathtt{yz}^*\mathtt{y} \cup (\mathtt{zw})^*\mathtt{y}^*\mathtt{w}^+\mathtt{z}^*\mathtt{y})X_0 \\ X_1 &= \quad \mathtt{z}^*\mathtt{y}X_0 \\ X_2 &= \quad \mathtt{y}^*\mathtt{w}^+\mathtt{z}^*\mathtt{y}X_0 \end{aligned}$$

- Wir erweitern die erste Gleichung:

$$\begin{aligned} X_0 &= \quad \mathtt{0} \quad \cup \quad ((\mathtt{zw})^*\mathtt{yz}^*\mathtt{y} \cup (\mathtt{zw})^*\mathtt{y}^*\mathtt{w}^+\mathtt{z}^*\mathtt{y})X_0 \\ X_1 &= \quad \mathtt{z}^*\mathtt{y}X_0 \\ X_2 &= \quad \mathtt{y}^*\mathtt{w}^+\mathtt{z}^*\mathtt{y}X_0 \end{aligned}$$

- Wir wenden Satz 11.5 auf die erste Gleichung an:

$$\begin{aligned} X_0 &= \quad ((\mathtt{zw})^*\mathtt{yz}^*\mathtt{y} \cup (\mathtt{zw})^*\mathtt{y}^*\mathtt{w}^+\mathtt{z}^*\mathtt{y})^*\mathtt{0} \\ X_1 &= \quad \mathtt{z}^*\mathtt{y}X_0 \\ X_2 &= \quad \mathtt{y}^*\mathtt{w}^+\mathtt{z}^*\mathtt{y}X_0 \end{aligned}$$

- Wir setzen den rechten Term der ersten Gleichung in die zweite und dritte Gleichung ein:

$$\begin{aligned} X_0 &= ((\mathtt{zw})^*\mathtt{yz}^*\mathtt{y} \cup (\mathtt{zw})^*\mathtt{y}^*\mathtt{w}^+\mathtt{z}^*\mathtt{y})^*\mathtt{0} \\ X_1 &= \mathtt{z}^*\mathtt{y}((\mathtt{zw})^*\mathtt{yz}^*\mathtt{y} \cup (\mathtt{zw})^*\mathtt{y}^*\mathtt{w}^+\mathtt{z}^*\mathtt{y})^*\mathtt{0} \\ X_2 &= \mathtt{y}^*\mathtt{w}^+\mathtt{z}^*\mathtt{y}((\mathtt{zw})^*\mathtt{yz}^*\mathtt{y} \cup (\mathtt{zw})^*\mathtt{y}^*\mathtt{w}^+\mathtt{z}^*\mathtt{y})^*\mathtt{0} \end{aligned}$$

→ Für jedes X_i ($i \in \{0,1,2\}$) definiert der Term der zweiten Lösung dieselbe Sprache wie der Term der ersten Lösung. (Tatsächlich kann man *alle* Terme einfach zu $\mathtt{0}$ kürzen. Damit definieren alle Terme die leere Sprache.)

2.2 Endliche Automaten

2.2.1 Übungen: Endliche Automaten

1. Betrachte folgende Automaten.

   (a) Zeichne die Diagramme.

   (b) Welche sind deterministisch?

   (c) Welche sind total?

   (d) Totalisiere die partiellen Automaten.

   (e) Gib die akzeptierten Sprachen an.

   (f) Konstruiere $\mathfrak{D} \oplus \mathfrak{E}$ und bearbeite die Aufgaben (i) bis (v) für diesen Automaten.

   (g) Konstruiere $\mathfrak{D} \times \mathfrak{E}$ und bearbeite die Aufgaben (i) bis (v) für diesen Automaten.

   (h) Falls der Automat $\mathfrak{F}$ nicht deterministisch ist, determinisiere ihn.

   (i) Konstruiere einen Automaten, der die Sprache $A^* - \mathrm{L}(\mathfrak{F})$ akzeptiert.

$$\begin{aligned}
\mathfrak{A} &:= \langle \{\mathtt{0},\mathtt{1}\}, \{q_0,q_1,q_2\}, q_0, \{q_1\}, \\
&\quad \{\langle q_0,\mathtt{0},q_0\rangle, \langle q_0,\mathtt{0},q_1\rangle, \langle q_1,\mathtt{0},q_1\rangle, \langle q_1,\mathtt{1},q_2\rangle, \langle q_2,\mathtt{0},q_2\rangle, \langle q_2,\mathtt{1},q_0\rangle\}\rangle \\
\mathfrak{B} &:= \langle \{\mathtt{0},\mathtt{1}\}, \{q_0,q_1,q_2,q_3\}, q_0, \{q_2\}, \\
&\quad \{\langle q_0,\mathtt{0},q_0\rangle, \langle q_0,\mathtt{1},q_1\rangle, \langle q_1,\mathtt{0},q_1\rangle, \langle q_1,\mathtt{1},q_2\rangle, \langle q_2,\mathtt{1},q_3\rangle, \langle q_3,\mathtt{0},q_3\rangle, \langle q_3,\mathtt{1},q_3\rangle\}\rangle \\
\mathfrak{C} &:= \langle \{\mathtt{a},\mathtt{n},\mathtt{w}\}, \{q_0,q_1,q_2,q_3\}, q_2, \{q_2,q_3\}, \{\langle q_0,\mathtt{a},q_0\rangle, \langle q_0,\mathtt{w},q_3\rangle, \langle q_1,\mathtt{a},q_0\rangle, \langle q_1,\mathtt{u},q_1\rangle, \\
&\quad \langle q_1,\mathtt{u},q_2\rangle, \langle q_2,\mathtt{u},q_3\rangle, \langle q_2,\mathtt{w},q_0\rangle, \langle q_2,\mathtt{u},q_0\rangle, \langle q_2,\mathtt{w},q_1\rangle, \langle q_3,\mathtt{w},q_0\rangle, \langle q_3,\mathtt{w},q_1\rangle\}\rangle \\
\mathfrak{D} &:= \langle \{\mathtt{a},\mathtt{b}\}, \{q_0,q_1\}, q_0, \{q_1\}, \{\langle q_0,\mathtt{a},q_1\rangle, \langle q_0,\mathtt{b},q_0\rangle, \langle q_1,\mathtt{a},q_0\rangle, \langle q_1,\mathtt{b},q_1\rangle \langle q_1,\mathtt{a},q_1\rangle\}\rangle \\
\mathfrak{E} &:= \langle \{\mathtt{a},\mathtt{b}\}, \{q_0,q_1\}, q_0, \{,q_0\}, \{\langle q_0,\mathtt{b},q_1\rangle, \langle q_0,\mathtt{b},q_0\rangle, \langle q_1,\mathtt{a},q_0\rangle, \langle q_1,\mathtt{b},q_0\rangle\}\rangle \\
\mathfrak{F} &:= \langle \{\mathtt{a},\mathtt{b}\}, \{q_0,q_1\}, q_0, \{q_1\}, \{\langle q_0,\mathtt{b},q_1\rangle, \langle q_1,\mathtt{a},q_0\rangle, \langle q_1,\mathtt{b},q_1\rangle\}\rangle
\end{aligned}$$

2. Konstruiere für jeden regulären Term einen Automaten, der die bezeichnete Sprache akzeptiert. Versuche, die Automaten induktiv über den Aufbau der Terme zu

konstruieren.

(a) `(ur)*oma`

(b) `(ho)*(hi)*`

(c) `(ho)*(hi)*(ho)?`

(d) $\texttt{a}^3\texttt{bbc}^*$

(e) `((a*b)*c)*`

(f) $\texttt{a}^*(\texttt{b} \cup \epsilon)\texttt{c?c}^*$

(g) $\texttt{aa}^*((\texttt{b} \cup \texttt{b})\texttt{c})^+$

(h) $(\texttt{h} \cup \texttt{w}^*\texttt{c})^* \cup (\texttt{e?} \cup \texttt{p})$

2.2.2 Übungen: Farbterme

1. Welche der folgenden Ausdrücke sind Unifarbterme (und warum)?

   (a) $(\texttt{a|b})^*(\underline{\texttt{c}}^*\texttt{|a})$

   (b) $(\texttt{a|b})^*(\texttt{c}^*\texttt{|a})$

   (c) $(\texttt{a|b})^*(\underline{\texttt{c}^*}\texttt{|a})$

   (d) $(\texttt{a|}\underline{\texttt{b}})^*(\texttt{c}^*\texttt{|a})$

   (e) $(\texttt{a|}\underline{\texttt{b}})^*(\texttt{c}^*\texttt{|}\underline{\texttt{a}})$

   (f) $\underline{(\texttt{a|b})^*}(\texttt{c}^*\texttt{|a})$

2. Betrachte den Ausdruck $((\texttt{a|b})^*\texttt{|}(((\texttt{cb})^*\texttt{|ab})^*\texttt{|}\epsilon))^*$.

   (a) Gib alle Unifarbterme an.

   (b) Gib alle finalen Unifarbterme an.

   (c) Gib alle x-initialen Unifarbterme an (mit $x \in \{\texttt{a}, \texttt{b}, \texttt{c}\}$).

   (d) Gib alle x-Übergänge für die Unifarbterme an (mit $x \in \{\texttt{a}, \texttt{b}, \texttt{c}\}$).

   (e) Gib alle Multifarbterme an.

   (f) Gib alle akzeptierenden Multifarbterme an.

   (g) Gib alle x-Übergänge für die Multifarbterme (und $\heartsuit$) an (mit $x \in \{\texttt{a}, \texttt{b}, \texttt{c}\}$).

3. Gib für folgende reguläre Ausdrücke s jeweils an, ob $\epsilon \in L(s)$ ist (und warum).

   (a) `(ab*|0)*`

   (b) $(\texttt{ab}^*\texttt{|0})^*\texttt{|}(\texttt{cc})^+$

   (c) `(bb)?(ab*|0)*`

4. Gib für folgende reguläre Ausdrücke jeweils

   (i) alle Übergänge zwischen Unifarbtermen und

   (ii) alle Übergange zwischen Multifarbtermen an.

   Zeichne außerdem die resultierenden Automaten. Sind sie deterministisch? Total?

(a) `(a|b)*(c*|a)`

(b) `((a|b)*|(((cb)*|ab)*|`ϵ`))*`

(c) `(ab*|0)*`

(d) `(ab*|0)*|(cc)`$^+$

(e) `(bb)?(ab*|0)*`

(f) `(ho)*(hi)*(ho)?`

(g) `((a*b)*c)*`

(h) `a*(b|`ϵ`)c?c*`

(i) `aa*((b|b)c)`$^+$

(j) `(h|w*c)*(e?|p)`