Kapitel 1

Syntax versus Semantik

Text und seine Bedeutung

Druckfassung der Vorlesung *Logik für Informatiker* vom 27. Oktober 2006

Till Tantau, Institut für Theoretische Informatik, Universität zu Lübeck

1.1

Ziele und Inhalt

Die Lernziele der heutigen Vorlesung und der Übungen.

1. Die Begriffe Syntax und Semantik erklären können
2. Syntaktische und semantische Elemente natürlicher Sprachen und von Programmiersprachen benennen können
3. Die Begriffe Alphabet und Wort kennen
4. Objekte als Worte kodieren können

1.2

Inhaltsverzeichnis

1.3

1 Was ist Syntax?

Die zwei Hauptbegriffe der heutigen Vorlesung.

Grobe Definition (Syntax)
Unter einer *Syntax* verstehen wir *Regeln*, nach denen Texte *strukturiert* werden dürfen.

Grobe Definition (Semantik)
Unter einer *Semantik* verstehen wir die Zuordnung von *Bedeutung* zu Text.

1.4

1.1 Syntax natürlicher Sprachen

Beobachtungen zu einem ägyptischen Text.

Beobachtungen

- Wir haben keine Ahnung, was der Text bedeutet.
- Es gibt aber *Regeln*, die offenbar eingehalten wurden, wie »Hieroglyphen stehen in Zeilen«.
- Solche Regeln sind *syntaktische Regeln* – man kann sie überprüfen, ohne den Inhalt zu verstehen.

1.5

Beobachtungen zu einem kyrillischen Text.

Beobachtungen

- Wir haben keine Ahnung, was der Text bedeutet.
- Es gibt aber *Regeln*, die offenbar eingehalten wurden.
- Wir kennen mehr Regeln als bei den Hieroglyphen.

Zur Diskussion
Welche syntaktischen Regeln fallen Ihnen ein, die bei dem Text eingehalten wurden?

1.6

Beobachtungen zu einem deutschen Text.

> Informatiker lieben Logiker.

Beobachtungen

- Auch hier werden viele syntaktische Regeln eingehalten.
- Es fällt uns aber *schwerer*, diese zu erkennen.
- Der Grund ist, dass wir *sofort über die Bedeutung nachdenken*.

1.7

Zur Syntax von natürlichen Sprachen.

- Die *Syntax* einer natürlichen Sprache ist die Menge an *Regeln*, nach denen Sätze gebildet werden dürfen.
- Die *Bedeutung* oder der *Sinn* der gebildeten Sätze ist dabei unerheblich.
- Jede Sprache hat ihre eigene Syntax; die Syntax verschiedener Sprachen ähneln sich aber oft.
- Es ist nicht immer klar, ob eine Regel noch zur Syntax gehört oder ob es schon um den Sinn geht.
  Beispiel: Substantive werden groß geschrieben.

1.8

1.2 Syntax von Programmiersprachen

Beobachtungen zu einem Programmtext.

```
\def\pgfpointadd#1#2{%
  \pgf@process{#1}%
  \pgf@xa=\pgf@x%
  \pgf@ya=\pgf@y%
  \pgf@process{#2}%
  \advance\pgf@x by\pgf@xa%
  \advance\pgf@y by\pgf@ya}
```

Beobachtungen

- Der Programmtext sieht sehr kryptisch aus.
- Trotzdem gibt es offenbar wieder Regeln.
- So scheint einem Doppelkreuz eine Ziffer zu folgen und Zeilen muss man offenbar mit Prozentzeichen beenden.

1.9

Beobachtungen zu einem weiteren Programmtext.

```
for (int i = 0; i < 100; i++)
  a[i] = a[i];
```

Beobachtungen

- Wieder gibt es Regeln, die eingehalten werden.
- Wieder fällt es uns *schwerer*, diese zu erkennen, da wir *sofort über den Sinn nachdenken*.

1.10

Zur Syntax von Programmiersprachen

- Die *Syntax* einer Programmiersprache ist die *Menge von Regeln*, nach der Programmtexte gebildet werden dürfen.
- Die *Bedeutung* oder der *Sinn* der Programmtexte ist dabei egal.
- Jede Programmiersprache hat ihre eigene Syntax; die Syntax verschiedener Sprachen ähneln sich aber oft.

1.11

5-Minuten-Aufgabe

Welche der folgenden Regeln sind Syntax-Regeln?

1. Bezeichner dürfen nicht mit einer Ziffer anfangen.
2. Programme müssen in endlicher Zeit ein Ergebnis produzieren.
3. Öffnende und schließende geschweifte Klammern müssen »balanciert« sein.
4. Methoden von Null-Objekten dürfen nicht aufgerufen werden.
5. Variablen müssen vor ihrer ersten Benutzung deklariert werden.

1.12

1.3 Syntax logischer Sprachen

Beobachtungen zu einer logischen Formel.

$$p \to q \wedge \neg q$$

Beobachtungen

- Auch logische Formeln haben eine syntaktische Struktur.
- So wäre es *syntaktisch falsch*, statt einem Pfeil zwei Pfeile zu benutzen.
- Es wäre aber *syntaktisch richtig*, statt einem Negationszeichen zwei Negationszeichen zu verwenden.

1.13

Zur Syntax von logischen Sprachen

- Die *Syntax* einer logischen Sprache ist die *Menge von Regeln*, nach der Formeln gebildet werden dürfen.
- Die *Bedeutung* oder der *Sinn* der Formeln ist dabei egal.
- Jede logische Sprache hat ihre eigene Syntax; die Syntax verschiedener Sprachen ähneln sich aber oft.

1.14

2 Was ist Semantik?

2.1 Semantik natürlicher Sprachen

Was bedeutet ein Satz?

Der Hörsaal ist groß.

- Dieser Satz hat eine *Bedeutung*.
- Eine *Semantik* legt solche Bedeutungen fest.
- Syntaktisch falschen Sätzen wird im Allgemeinen keine Bedeutung zugewiesen.

1.15

Ein Satz, zwei Bedeutungen.

Steter Tropfen höhlt den Stein.

- Ein Satz kann *mehrere Bedeutungen haben*, welche durch *unterschiedliche Semantiken* gegeben sind.
- In der *wortwörtlichen Semantik* sagt der Satz aus, dass Steine ausgehöhlte werden, wenn man jahrelang Wasser auf sie tropft.
- In der *übertragenen Semantik* sagt der Satz aus, dass sich Beharrlichkeit auszahlt.

1.16

Die Semantik der Hieroglyphen

Unknown Author, Public Domain, Low Resolution

1.17

2.2 Semantik von Programmiersprachen

Was bedeutet ein Programm?

```
for (int i = 0; i < 100; i++)
  a[i] = a[i];
```

- Auch dieser Programmtext »bedeutet etwas«, wir »meinen etwas« mit diesem Text.
- Die *Semantik der Programmiersprache* legt fest, was mit dem Programmtext gemeint ist.

1.18

Ein Programm, zwei Bedeutungen.

```
for (int i = 0; i < 100; i++)
  a[i] = a[i];
```

- Ein Programmtext kann *mehrere Bedeutungen haben*, welche durch *unterschiedliche Semantiken* gegeben sind.
- In der *operationalen Semantik* bedeutet der Programmtext, dass die ersten einhundert Elemente eines Arrays `a` nacheinander ihren eigenen Wert zugewiesen bekommen.
- In der *denotationellen Semantik* bedeutet der Programmtext, dass nichts passiert.

1.19

2.3 Semantik logischer Sprachen

3 Grundlage der Syntax: Text

Eine mathematische Sicht auf Text.

- Viele (aber nicht alle!) syntaktische Systeme bauen auf *Text* auf.
- Auch solche Systeme, die nicht auf Text aufbauen, lassen sich trotzdem durch Text beschreiben.
- Es ist deshalb nützlich, auf Text Methoden der Mathematik anwenden zu können.
- Im Folgenden wird deshalb die *mathematische Sicht* auf Text eingeführt, die *in der gesamten Theoretischen Informatik* genutzt wird.

1.20

3.1 Alphabete

Formale Alphabete

Definition 1 (Alphabet)**.** Ein *Alphabet* ist eine nicht-leere, endliche Menge von *Symbolen* (auch *Buchstaben* genannt).

- Alphabete werden häufig mit griechischen Großbuchstaben bezeichnet, also Γ oder Σ. Manchmal auch mit lateinischen Großbuchstaben, also N oder T.
- Ein Symbol oder »Buchstabe« kann auch ein komplexes oder komisches »Ding« sein wie ein Pointer oder ein Leerzeichen.

Beispiele 2.
- Die Groß- und Kleinbuchstaben
- Die Menge $\{0,1\}$ (bei Informatikern beliebt)
- Die Menge $\{A,C,G,T\}$ (bei Biologen beliebt)
- Die Zeichenmenge des UNICODE.

1.21

3.2 Worte

Formale Worte

Definition 3 (Wort)**.** Ein *Wort* ist eine (endliche) Folge von Symbolen.

- »Worte« sind im Prinzip dasselbe wie Strings. Insbesondere können in Worten Leerzeichen als Symbole auftauchen.
- Die Menge aller Worte über einem Alphabet Σ hat einen besonderen Namen: Σ^*.
- Deshalb schreibt man oft: »Sei $w \in \Sigma^*, \ldots$ «
- Es gibt auch ein *leeres Wort*, abgekürzt ε oder λ, das dem String `""` entspricht.

Beispiele 4.
- `Hallo`
- `TATAAAATATTA`
- ε
- `Hallo Welt.`

1.22

5-Minuten-Aufgabe

Die folgenden Aufgaben sind nach Schwierigkeit sortiert. Lösen Sie *eine* der Aufgaben.

1. Schreiben Sie alle Worte der Länge höchstens 2 über dem Alphabet $\Sigma = \{0,1,*\}$ auf.
2. Wie viele Worte der Länge n über dem Alphabet $\Sigma = \{0,1,*\}$ gibt es?
3. Wie viele Worte der Länge höchstens n über einem Alphabet mit q Buchstaben gibt es?

1.23

3.3 Sprachen

Formale Sprachen Definition

- Natürlichen Sprachen sind komplexe Dinge, bestehend aus Wörtern, ihrer Aussprache, einer Grammatik, Ausnahmen, Dialekten, und vielem mehr.
- Bei *formalen Sprachen* vereinfacht man radikal.
- Formale Sprachen müssen weder sinnvoll noch interessant sein.

Definition 5 (Formale Sprache)**.** Eine *formale Sprache* ist eine (oft unendliche!) Menge von Worten für ein festes Alphabet.

- Statt »formale Sprache« sagt man einfach »Sprache«.
- Als Menge von Worten ist eine Sprache eine Teilmenge von Σ^*.
- Deshalb schreibt man oft: »Sei $L \subseteq \Sigma^*, \ldots$ «

1.24

Formale Sprachen Einfache Beispiele

Beispiele 6.

- Die Menge $\{AAA, AAC, AAT\}$ (endliche Sprache).
- Die Menge aller Java-Programmtexte (unendliche Sprache).
- Die Menge aller Basensequenzen, die `TATA` enthalten (unendliche Sprache).

Formale Sprachen in der Medieninformatik

- Ein Renderer produziert 3D-Bilder.
- Dazu erhält er eine *Szenerie* als Eingabe.
- Diese Szenerie ist als *Text*, also als ein *Wort* gegeben.
- Eine *Syntax* beschreibt die (formale) Sprache, die alle *syntaktisch korrekten Szenerien* enthält.
- Eine *Semantik* beschreibt, was diese Beschreibungen bedeuten.

Formale Sprachen in der Medieninformatik Das »Wort«, das eine Szenerie beschreibt...

```
global_settings␣{␣assumed_gamma␣1.0␣}

camera␣{
␣␣location␣␣<10.0,␣10,␣-10.0>
␣␣direction␣1.5*z
␣␣right␣␣␣␣␣x*image_width/image_height
␣␣look_at␣␣␣<0.0,␣0.0,␣␣0.0>
}

sky_sphere␣{␣pigment␣{␣color␣rgb␣<0.6,0.7,1.0>␣}␣}

light_source␣{
␣␣<0,␣0,␣0>␣␣␣␣␣␣␣␣␣␣//␣light's␣position␣(translated␣below)
␣␣color␣rgb␣<1,␣1,␣1>␣␣//␣light's␣color
␣␣translate␣<-30,␣30,␣-30>
␣␣shadowless
}

#declare␣i␣=␣0;
#declare␣Steps␣=␣30;
#declare␣Kugel␣=␣sphere{<0,0,0>,0.5␣pigment{color␣rgb<1,0,0>}};

#while(i<Steps)
␣␣␣␣object{Kugel␣␣translate<3,0,0>␣rotate␣<0,i␣*␣360␣/␣Steps,␣0>␣}
␣␣#declare␣i␣=␣i␣+␣1;
#end
```


Formale Sprachen in der Medieninformatik ... und was es bedeutet.

Formale Sprachen in der Medieninformatik Komplexeres Beispielbild, das ein Renderer produziert.

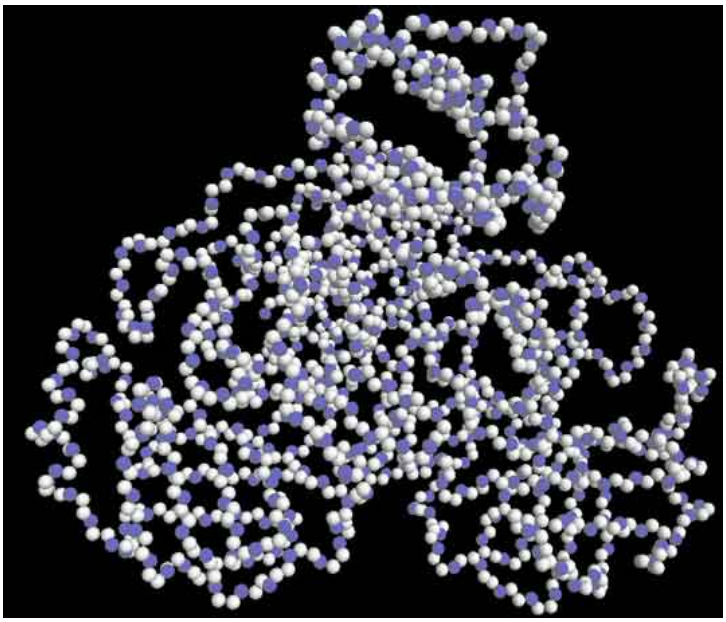
Formale Sprachen in der Bioinformatik

- In der Bioinformatik untersucht man unter anderem Proteine.
- Dazu erhält man *Molekülbeschreibungen* als Eingabe.
- Eine solche ist auch ein *Wort*.
- Eine *Syntax* beschreibt die (formale) Sprache, die alle *syntaktisch korrekten Molkülbeschreibungen* enthält.
- Eine *Semantik* beschreibt, was diese Beschreibungen bedeuten.

Formale Sprachen in der Bioinformatik Das »Wort«, das ein Protein beschreibt...

```
HEADER    HYDROLASE                               25-JUL-03   1UJ1
TITLE     CRYSTAL STRUCTURE OF SARS CORONAVIRUS MAIN PROTEINASE
TITLE    2 (3CLPRO)
COMPND    MOL_ID: 1;
COMPND   2 MOLECULE: 3C-LIKE PROTEINASE;
COMPND   3 CHAIN: A, B;
COMPND   4 SYNONYM: MAIN PROTEINASE, 3CLPRO;
COMPND   5 EC: 3.4.24.-;
COMPND   6 ENGINEERED: YES
SOURCE    MOL_ID: 1;
SOURCE   2 ORGANISM_SCIENTIFIC: SARS CORONAVIRUS;
SOURCE   3 ORGANISM_COMMON: VIRUSES;
SOURCE   4 STRAIN: SARS;
...
REVDAT   1   18-NOV-03 1UJ1    0
JRNL        AUTH   H.YANG,M.YANG,Y.DING,Y.LIU,Z.LOU,Z.ZHOU,L.SUN,L.MO,
JRNL        AUTH 2 S.YE,H.PANG,G.F.GAO,K.ANAND,M.BARTLAM,R.HILGENFELD,
JRNL        AUTH 3 Z.RAO
JRNL        TITL   THE CRYSTAL STRUCTURES OF SEVERE ACUTE RESPIRATORY
JRNL        TITL 2 SYNDROME VIRUS MAIN PROTEASE AND ITS COMPLEX WITH
JRNL        TITL 3 AN INHIBITOR
JRNL        REF    PROC.NAT.ACAD.SCI.USA         V. 100 13190 2003
JRNL        REFN   ASTM PNASA6  US ISSN 0027-8424
....
ATOM      1  N   PHE A   3      63.478 -27.806  23.971  1.00 44.82           N
ATOM      2  CA  PHE A   3      64.607 -26.997  24.516  1.00 42.13           C
ATOM      3  C   PHE A   3      64.674 -25.701  23.723  1.00 41.61           C
ATOM      4  O   PHE A   3      65.331 -25.633  22.673  1.00 40.73           O
ATOM      5  CB  PHE A   3      65.912 -27.763  24.358  1.00 44.33           C
ATOM      6  CG  PHE A   3      67.065 -27.162  25.108  1.00 44.20           C
ATOM      7  CD1 PHE A   3      67.083 -27.172  26.496  1.00 43.35           C
ATOM      8  CD2 PHE A   3      68.135 -26.595  24.422  1.00 43.49           C
ATOM      9  CE1 PHE A   3      68.140 -26.631  27.187  1.00 43.21           C
ATOM     10  CE2 PHE A   3      69.210 -26.046  25.108  1.00 42.91           C
ATOM     11  CZ  PHE A   3      69.216 -26.062  26.493  1.00 43.22           C
ATOM     12  N   ARG A   4      64.007 -24.666  24.228  1.00 34.90           N
ATOM     13  CA  ARG A   4      63.951 -23.376  23.543  1.00 37.71           C
...
```


Formale Sprachen in der Bioinformatik ... und das Protein, das beschrieben wird.



1.32

4 Zusammenfassung

Zusammenfassung

1. Ein *Wort* ist eine Folge von Symbolen aus einem *Alphabet*.
2. Eine *Syntax* besteht aus Regeln, nach denen Worte (Texte) gebaut werden dürfen.
3. Eine *Semantik* legt fest, was Worte *bedeuten*.
4. Eine *formale Sprache* ist eine Menge von Worten über einem Alphabet.

1.33