

Proseminar \LaTeX
Ausarbeitung zum Vortrag
” \LaTeX in der Informatik”

Thilo Ohlemueller

Juni 2006

Inhaltsverzeichnis

1	Einführung	2
2	Bäume	2
2.1	Einleitung	2
2.2	Das Paket <code>qtree</code>	2
2.3	<code>qtree</code> - Befehle	2
2.4	<code>qtree</code> - Beispiel	3
3	Graphen und Automaten	3
3.1	Einleitung	3
3.2	Das Paket <code>GasTeX</code>	4
3.3	<code>GasTeX</code> - Befehle (kleiner Ausschnitt)	4
3.4	<code>GasTeX</code> - Beispiel	5
3.5	<code>GasTeX</code> - weitere Beispiele	6
4	Quellcodedarstellung	6
4.1	Einleitung	6
4.2	Das Paket <code>listings</code>	6
4.3	<code>listings</code> - Befehle	6
4.4	<code>listings</code> - Beispiel 1	7
4.5	<code>listings</code> - Eigene Sprache definieren	8
4.6	<code>listings</code> - Beispiel 2	8
5	Sonstiges(UML, Message Sequence Charts, ...	10
6	Quellen	10

1 Einführung

Informatiker möchten ihre Dokumente (Publikationen, Spezifikationen) auch mit \LaTeX erstellen. Dabei ist es einerseits erforderlich, Programmquellcode schön und einfach (Syntaxhighlighting, Einrückung), und andererseits auch Grafiken, wie Bäume und Graphen, darzustellen.

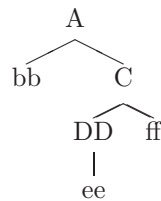
Ferner gibt es die Möglichkeit der automatischen Erstellung von Bäumen und Automaten. Man hat also bspw. bestimmte Daten in einer Datenbank und generiert daraus automatisch ein \LaTeX -Dokument, welches die Daten als Baum oder Graph darstellt.

Ab einem gewissen Grad ist ein grafischen Tool oder ein WYSIWYG-Editor wesentlich komfortabler und einfacher, z.B. UML Klassendiagramme.

2 Bäume

2.1 Einleitung

Zunächst ein Beispiel für einen einfachen Baum:



Bäume werden z.B. im Compilerbau für Syntaxanalyse in Form von Abstrakten-Syntax-Bäumen verwendet.

2.2 Das Paket `qtree`

Um das Paket `qtree` benutzen zu können, wird die Datei `qtree.sty` benötigt. Diese kann z.B. von der in den Quellen angegebenen Webseite heruntergeladen werden. Man kopiert die Datei dann entweder in das \LaTeX -Paketverzeichnis oder in das Verzeichnis, in dem auch das zu erstellende Dokument liegen wird. Mit `\usepackage{qtree}` wird das Pakete in das \LaTeX -Dokument eingebunden. In dieses wird auch automatisch das Paket 'eepic' eingebunden. Dies verschönert die Ausgabe für Postscript. Wenn man allerdings bspw. 'pdfLaTeX' einsetzen möchte, muß mit `\usepackage[noeepic]{qtree}` dies Paket explizit ausgeschlossen werden.

`qtree` kann Bäume mit einer maximalen Tiefe von 20 und einer maximalen Verzweigung pro Knoten von 5 darstellen.

2.3 `qtree` - Befehle

Ein Baum wird mit folgendem Befehl erstellt:

```
\Tree [.Wurzel [.Knoten Blatt] ].Wurzel
```

Dabei wird die Wurzel immer als oberster Knoten zentriert dargestellt. Ein Knoten wird mit `[. eingefügt`. Weitere Knoten bzw. Blätter werden hinter den entsprechenden Knoten, durch ein Leerzeichen getrennt, geschrieben.

Das Nennen der Wurzel am Ende ist optional \Rightarrow es funktioniert auch:

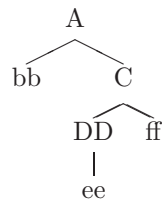
```
\Tree [.Wurzel [.Knoten Blatt] ].
```

2.4 qtree - Beispiel

Der in der Einleitung abgebildete Baum wird mit folgendem Befehl erstellt:

```
\Tree [.A bb [.C [.DD ee ] ff ] ]
```

Wenn alles funktioniert, ist das Ergebnis:



Bei Blättern sollte man unbedingt beachten, dass nicht ein einzelnes Zeichen als Blatt verwendet wird. Ansonsten kann es nämlich vorkommen, daß die Verbindungslinien zwischen den Knoten/Blättern nicht gezeichnet werden - wie folgendes Beispiel zeigt.

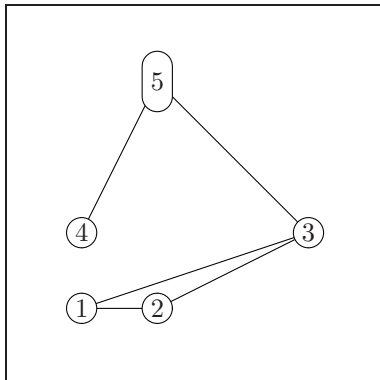
```
\Tree [.A B C ]   \Tree [.A BB CC ]
```

```
graph TD
  A1 --- B
  A1 --- C
  A2 --- BB
  A2 --- CC
```

3 Graphen und Automaten

3.1 Einleitung

Zunächst wieder ein einfaches Beispiel für einen Graphen zum Einstieg:



Graphen werden hauptsächlich in der Theoretischen Informatik benötigt \Rightarrow Graphen- und Automatentheorie.

3.2 Das Paket GasTeX

GasTeX steht für: Graphs and Automata Simplified in TeX.

Die Verwendung von GasTeX ist nur mit PostScript möglich. Es wird die Paktdatei `gastex.sty` benötigt. Außerdem muß eine sogenannte Headerdatei `gastex.pro` vorhanden sein. Diese enthält spezielle PostScript-Prozeduren, die für das Zeichnen der Knoten und Kanten etc. gebraucht werden. Beide Dateien müssen sich wie bei `qtree` im \LaTeX -Paketverzeichnis oder im Verzeichnis des aktuellen Dokumentes befinden. Die pdf-Erstellung ist über einen kleinen Umweg möglich: $\LaTeX \Rightarrow \text{PS} \Rightarrow \text{PDF}$ (`ps2pdf`). Es sollte auch folgendes funktionieren: $\LaTeX \Rightarrow \text{DVI} \Rightarrow \text{PS} \Rightarrow \text{PDF}$

Zu GasTeX gibt es leider keine wirkliche Dokumentation. Die Autoren von GasTeX verweisen auf die umfangreichen Beispiele um GasTeX zu 'lernen'.

Ferner gibt es ein grafisches Interface `JasTeX`. Dies ein in Java geschriebener WYSIWYG-Editor, der zum erstellten Graphen den \LaTeX -Quelltext ausgibt.

3.3 GasTeX - Befehle (kleiner Ausschnitt)

Der folgende Abschnitt bietet nur einen sehr kleinen Ausschnitt aus den Befehlen, die GasTeX zur Verfügung stellt. Es wird sich auf die Befehle beschränkt, die erforderlich sind, um das o.g. Beispiel zu erstellen.

GasTeX arbeitet in der `picture`-Umgebung. Diese wird mit

```
\begin{picture}(breite,hoehe) ... \end{picture}
```

eingeleitet. Mit `\node[Opt.]{Bezeichner}{PosX,PosY}{Bezeichnung}` wird ein Knoten definiert, wobei die Position (X und Y) und der Bezeichner Pflichtparameter sind. Der Bezeichner wird benötigt um später z.B. beim Kantenziehen den Knoten anzusprechen. Die Bezeichnung ist optional – diese wird im Graphen als Knotenbezeichnung verwendet. In den Optionen wird unter anderem die Größe und das Aussehen des Knoten angegeben.

Eine Kante kann mit `\drawedge[Opt.](Knoten1,Knoten2){Bezeichnung}` eingefügt werden. Dabei stehen `Knoten1` und `Knoten2` für die Bezeichner der bereits definierten Knoten. Optional ist wieder auch eine Bezeichnung der Kante im Graphen. In den Optionen kann wiederum Art und Aussehen der Kante definiert werden.

3.4 Gas \TeX - Beispiel

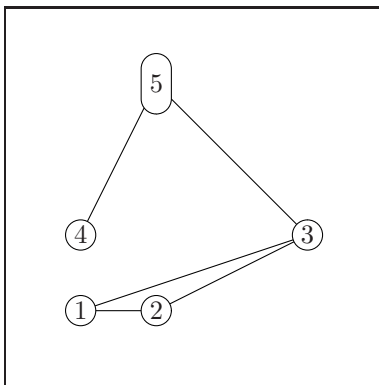
Der folgende \LaTeX -Quelltext beschreibt den in der Einleitung gezeigten Graphen.

```
\begin{picture}(50,50)(0,0)
\put(0,0){\framebox(50,50){}}
\node[Nw=4.0,Nh=4.0](n1)(10.0,10.0){1}
\node[Nw=4.0,Nh=4.0](n2)(20.0,10.0){2}
\node[Nw=4.0,Nh=4.0](n3)(40.0,20.0){3}
\node[Nw=4.0,Nh=4.0](n4)(10.0,20.0){4}
\node[Nw=4.0,Nh=8.0](n5)(20.0,40.0){5}

\drawedge[AHnb=0](n1,n2){}
\drawedge[AHnb=0](n3,n2){}
\drawedge[AHnb=0](n1,n3){}
\drawedge[AHnb=0](n4,n5){}
\drawedge[AHnb=0](n3,n5){}
\end{picture}
```

Es wird hier eine `picture`-Umgebung mit der Größe $50\text{mm} \cdot 50\text{mm}$ geöffnet. Die zweite Zeile sorgt lediglich dafür, daß ein Rahmen um die geöffnete `picture`-Umgebung gezeichnet wird. Als Optionen treten hier für die Knoten die Breite (`Nw`) und die Höhe (`Nh`) auf und für die Kanten, ob ein Pfeil als Kante verwendet werden soll (`AHnb=0` \rightarrow 0 Pfeile).

Das Ergebnis ist:



3.5 GasTeX - weitere Beispiele

Auf der Webseite von GasTeX (siehe Quellen) kann eine Beispieldatei heruntergeladen werden. Diese zeigt sehr schön, was alles mit GasTeX möglich ist, und ist gleichzeitig als Dokumentation zu verstehen.

4 Quellcodedarstellung

4.1 Einleitung

Es gibt viele verschiedene Möglichkeiten Quellcode darzustellen. Die einfachste aber auch nicht wirklich schöne ist wohl mit

```
\begin{verbatim} ... \end{verbatim} bzw. \verb.
```

Hier wird das Paket *listings* vorgestellt. Dies scheint als Standard für die Quellcodedarstellung durchgesetzt zu haben. Es gibt aber auch diverse andere Pakete. Diese werden freundlicherweise in der Dokumentation zu *listings* genannt.

4.2 Das Paket listings

Das Paket *listings* bietet eine sehr schöne und einfache Art und Weise Quellcode in einem L^AT_EX-Dokument darzustellen. Es enthält bereits eine sehr große Auswahl an definierten Sprachen. Das Paket bietet auch die Möglichkeit eigene Sprachen zu definieren.

Wenn man *listings* zusammen mit *beamer* verwenden möchte, muß beachtet werden, daß ein *listing* nicht direkt in einen *Frame* eingebunden werden kann. Mit `\defverbatim\name{...}` kann aber vor dem *Frame* das *listing* definiert werden und ist dann mit `\name` im *Frame* verwendbar.

4.3 listings - Befehle

Das Paket wird mit `\usepackage{listings}` eingebunden. Mit dem Befehl `\lstset{Option1=Wert,Option2=Wert}` wird unter anderem die Sprache und das Aussehen des *listing* bestimmt. Nach dieser Definition kann man mit drei verschiedenen Arten ein *listing* einfügen:

1. *listings*-Umgebung: `\begin{lstlisting} ... \end{lstlisting}`
2. einzelne Zeile: `\lstinline/ ... /`, wobei das Anfang- und Endzeichen beliebig sein kann (hier `/`)
3. ganze Quellcodedatei: `\lstinputlisting{datei.xyz}`

Besondere Bedeutung gilt dem Befehl `lstset`. Folgende Optionen können mit `lstset` definiert werden:

- `language=[Dialekt]Sprache` (z.B.: `language=[LaTeX]TeX`)

- `keywordstyle=style` (z.B.: `keywordstyle=\color{red}\bfseries`)
- `commentstyle=style` (z.B.: `commentstyle=\color{gray}\bfseries`)
- `keywords={keyword1,keyword2,...}`
- `emph={word1,word2,...}`
- `emphstyle=style`
- `numbers=position` (z.B.: `numbers=left`)
- ...

Es gibt noch eine Reihe weiterer Optionen, die im Handbuch zu listings nachgelesen werden können.

4.4 listings - Beispiel 1

```
\lstset{language=Java,
        keywordstyle=\color{blue}\bfseries,
        emph={HelloWorld},
        emphstyle=\color{green}}
\begin{lstlisting}
// author: Thilo Ohlemueller
// Beispiel 1 listings

public class HelloWorld {

    public static void main(String[] args) {
        System.out.println("Hello world!");
    }
}
end{lstlisting}
```

Die Ausgabe im L^AT_EX-Dokument ist dann folgende:

```
// author: Thilo Ohlemueller
// Beispiel listings

public class HelloWorld {

    public static void main(String[] args) {
        System.out.println("Hello world!");
    }
}
```

Wenn der o.g. Java-Quellcode in einer Datei HelloWorld.java gespeichert ist, würde

```
\lstset {language=Java ,
         keywordstyle=\color {blue}\bfseries ,
         emph={HelloWorld} ,
         emphstyle=\color {green}}
```

```
\lstinputlisting {HelloWorld.java}
```

die gleiche Ausgabe wie oben ergeben.

4.5 listings - Eigene Sprache definieren

Eine eigene Sprache kann mit `\lstdefinlanguage{name}{Optionen}` definiert werden. Die Optionen sind dabei unter anderen:

- `keywords={word1, word2, ...}`: Schlüsselwörter benennen
- `sensitive=(true | false)`: Groß-/Kleinschreibung beachten
- `comment=[1]{Zeichen}`: einzeilige Kommentare
- `comment=[s]{Anfangsz.}{Endez.}`: durch Zeichen abgetrennte Kommentare
- `string=[b]Zeichen`: Zeichenketten mit 'Zeichen' als Anfang bzw. Ende

4.6 listings - Beispiel 2

Hier wird gezeigt, wie man eine eigene Sprache definiert und diese verwendet. Bei der Beispielsprache handelt es sich um Assembler mit besonderen Befehlen, die innerhalb des Projekts zur Technischen Informatik 2 entwickelt wurden.

```
\lstdefinlanguage {ti2projekt} {
    keywords={blabla1, GetDist, GetVelocity,
             SetBrakeLight, ReadTable,
             cmp, jmpz, jmp, SetVelocity,
             DEC, INC, blabla2 },
    sensitive=false,
    comment=[1]{//}
}
```

```
\lstset {language=ti2projekt,
         keywordstyle=\color {blue}\bfseries,
         commentstyle=\color {red}\itshape,
         numbers=left
}
```

```
\begin {lstlisting}
```

```
GetDist D
```

```

GetVelocity V
ReadTable V D2
cmp D D2
jmpz 5 // springe 5 Befehle weiter falls cmp true
cmp D2 D
jmpz 7
SetVelocity V
SetBrakeLight
jmp 1
INC V
SetVelocity V
SetBrakeLight 0
jmp 1
DEC V
SetVelocity V
SetBrakeLight 1
jmp 1

```

`end{lstlisting}`

Etwas irreführend sind sicherlich die Schlüsselwörter 'blabla1' und 'blabla2' in der keywords-Option. Dies rührt daher, daß bei allen meinen Versuchen, das erste und das letzte Schlüsselwort nicht als solches erkannt wurde. Daher habe ich einfach zwei Dummy-Schlüsselwörter definiert. Die Ausgabe ist:

```

1 GetDist D
2 GetVelocity V
3 ReadTable V D2
4 cmp D D2
5 jmpz 5 // springe 5 Befehle weiter falls cmp true
6 cmp D2 D
7 jmpz 7
8 SetVelocity V
9 SetBrakeLight
10 jmp 1
11 INC V
12 SetVelocity V
13 SetBrakeLight 0
14 jmp 1
15 DEC V
16 SetVelocity V
17 SetBrakeLight 1
18 jmp 1

```

5 Sonstiges(UML, Message Sequence Charts, ...)

Es gibt natürlich auch \LaTeX -Pakete um UML (Unified Modelling Language) oder MSC (Message Sequence Chart) darzustellen. Es gibt bspw. Pakete unter:

- UML Paket (basiert auf Postscript):
<http://humbert.in.hagen.de/iffase/Artikel/latex5.html>
- MSC Paket:
<http://www.win.tue.nl/~sjouke/misc/mscpackage/index.html>

Wie in der Einführung bereits erwähnt, sind diese Elemente sicherlich mit grafischen Editoren komfortabler zu erstellen.

6 Quellen

- Bäume:
qtree: <http://www.essex.ac.uk/linguistics/clmt/latex4ling/trees/qtree/>
- Graphen:
GasTeX:
<http://www.lsv.ens-cachan.fr/~gastin/gastex/gastex.html>
JasTeX:
<http://www.lsv.ens-cachan.fr/~gastin/JasTeX/JasTeX.html>
- Quellcode:
listings:
www.ctan.org/tex-archive/macros/latex/contrib/listings/listings-1.3.pdf