Bretter, die die Welt bedeuten

BibTool – **Manipulation von** BibTeX**-Dateien**

Gerd Neugebauer

Zusammenfassung

Es war einmal, da verspürte ich das dringende Bedürfnis, BibTeX-Dateien aus verschiedenen Quellen zusammen weiterzuverwenden. Da es sich dabei nicht nur um einige wenige Einträge in den BibTeX-Dateien handelte, ergaben sich daraus diverse Aufgaben, die gelöst werden mußten. Das einfachste Problem war, daß das Erscheinungsbild der Dateien vereinheitlicht werden sollte. Das bedeutet, daß die Dateien neu formatiert werden müssen. Das nächste Problem war die Vereinheitlichung der Schlüssel, unter denen die BibTeX-Einträge angesprochen werden. Schließlich und endlich ergab sich das Problem, die Dateien zu sortieren.

Da es damals keine entsprechenden Werkzeuge gab – insbesondere nicht für den Rechner, den ich damals verwendete – wurde das BibTool-Programm geboren. Als dann einmal die grundlegenden Routinen geschrieben waren, kamen noch weitere Funktionen hinzu, sodaß man BibTool heute als das Schweizer-Armee-Messer zur BibTeX-Vorverarbeitung bezeichnen könnte.

BibTeX – **Eine Würdigung**

An Anfang aller Bemühungen steht BibTeX[1] [1, 2, 3]. BibTeX wird meistens als Programm bezeichnet, das Literaturzitate mit einem TeX-Dokument kombiniert. In Wirklichkeit ist BibTeX viel allgemeiner. BibTeX bietet eine einfache „Datenbank“ mit der Möglichkeit, Datensätze daraus in ein TeX-Dokument zu übernehmen. Dabei sind einige Manipulationen möglich, die hauptsächlich von der Anwendung als Literaturdatenbank inspiriert sind.

Um die Konzepte von BibTeX aufzuzeigen, betrachten wir kurz ein Beispiel. Als erstes brauchen wir eine BibTeX-Datei, die die Literaturzitate enthält. Wir betrachten den folgenden Datensatz aus der BibTeX-Datei `lit.bib`.

[1] Zur Zeit ist BibTeX 0.99c aktuell. Der Aufbau der Literaturdatenbankdateien für die schon lange angedrohte Version 1.0 sollte sich aber nicht allzu sehr unterscheiden – sollte sie jemals veröffentlicht werden.

```
@Book{gms:companion,
  author = "Michel Goossens and Frank Mittelbach and Alexander Samarin",
  title = "The {\LaTeX} Companion",
  publisher = "Addison-Wesley",
  year = 1994
}
```

Um nun ein Zitat dieser Literaturstelle in unser LaTeX-Dokument aufzunehmen, geben wir dort den Schlüssel in einem Befehl der Form `\cite{gms:companion}` an. Weiterhin müssen wir festlegen, welchen Stil wir für die Referenzen bevorzugen. Dies geschieht z. B. durch die Anweisung `\bibliographystyle{alpha}`. Schließlich muß noch angegeben werden, welchen Dateien die Referenzen zu entnehmen sind. Dies geschieht mit dem Befehl `\bibliography{lit}`. Dieser Befehl legt auch fest, an welcher Stelle die Literaturliste eingefügt werden soll.

Damit haben wir die Vorbereitungen in unserem Dokument abgeschlossen. Jetzt wollen wir uns das Ergebnis unserer Bemühungen ansehen. Als erstes wird LaTeX aufgerufen. Damit werden die Informationen aus den Befehlen `\cite`, `\bibliographystyle` und `\bibliography` in die `.aux`-Datei übernommen. Nun rufen wir BibTeX auf. BibTeX liest die `.aux`-Datei und erzeugt die `.bbl`-Datei. Ein weiterer LaTeX-Lauf befördert die Literaturliste in die `.dvi`-Datei und die Zitat-Schlüssel in die neue `.aux`-Datei, von wo aus sie durch einen anschließenden LaTeX-Lauf in die Ausgabe übernommen werden.

Soweit so gut. Einige Dinge mußten bisher beachtet werden. In der BibTeX-Datei sind Fehler verheerend. BibTeX hat eine sehr grobe Methode der Fehlerbehandlung: es ignoriert einfach alles, was noch in dem aktuellen Datensatz stehen mag. Dies geschieht insbesondere dann, wenn die Schlüssel, in unserem Beispiel ist das `gms:companion`, nicht für alle Datensätze verschieden sind. Eindeutige Schlüssel sind insbesondere dann schwierig zu garantieren, wenn man BibTeX-Dateien anderer mitbenutzen will. Hier setzt BibTool an.

BibTool – **Analyse und Neuformatierung**

Die erste Aufgabe, die anfällt, wenn man eine fremde BibTeX-Datei in die Hände bekommt, ist es, diese auf Fehler zu untersuchen und möglicherweise auch in eine „schönere“ Form zu bringen. Das ist die einfachste Aufgabe für BibTool. Ein Aufruf der Form

`bibtool` *infile* `-o` *outfile*

analysiert die Datei *infile* und legt das Ergebnis neuformatiert in der Datei *outfile* ab. Dabei gehen wir, wie im Rest dieser Beschreibung, von einem Aufruf über einen Kommando-Interpreter aus.

Das Format läßt sich dabei von einer Vielzahl von Parametern bestimmen. Diese Parameter sind mit „sinnvollen“ Vorgaben versehen. Danach sieht unser Beispiel folgendermaßen aus:

```
@Book{            gms:companion,
  author        = "Michel Goossens and Frank Mittelbach and Alexander
                  Samarin",
  title         = "The {\LaTeX} Companion",
  publisher     = "Addison-Wesley",
  year          = 1994
}
```

Auf jeden Fall werden wir aber bei diesem Durchgang auch auf Fehler aufmerksam gemacht. In einigen Fällen wird versucht, diese Fehler zu korrigieren. So werden zum Beispiel fehlende oder überzählige Kommata zwischen den Feldern ergänzt bzw. gelöscht.

BibTool – **Sortieren**

Als nächstes wollen wir eine BibTeX-Datei sortieren. Das ist auch bei unseren eigenen Dateien gelegentlich ganz nützlich. Dies geschieht mit dem folgenden Kommando:

`bibtool -s` *infile* `-o` *outfile*

Die Ausgabe-Datei *outfile* enthält die Eingabe-Datei *infile*, wobei die Datensätze entsprechend den Schlüsseln sortiert erscheinen. Diese Sortierreihenfolge – nach den Schlüsseln – ist nicht immer wünschenswert. Deshalb kann es sich als sinnvoll erweisen, vor dem Sortieren neue Schlüssel zu erzeugen. Darauf wird im nächsten Abschnitt näher eingegangen. Alternativ kann mit denselben Mitteln, die es erlauben, einen Schlüssel zu generieren, auch ein Sortierschlüssel spezifiziert werden.

BibTool – **Schlüssel**

Ein großes Problem beim Umgang mit BibTeX stellen die Schlüssel dar. Als Schlüssel könnte, mit Ausnahme einiger Zeichen, jeder beliebiger Text verwendet werden. Üblicherweise wird man jedoch seine Schlüssel so wählen, daß sie

einfach zu merken sind. Würde man einfach sinnlose Wörter als Schlüssel verwenden, müßte man sonst bei jedem Zitat den Schlüssel aus der BibTeX-Datei heraussuchen. Folglich bietet es sich an, die Schlüssel nach einem einheitlichen Verfahren zu generieren. Damit kann man einfach anhand der gewünschten Literaturstelle den zugehörigen Schlüssel ableiten und muß ihn nicht nachsehen. Dadurch entfällt ebenfalls die Aufgabe, sich für jede Literaturstelle einen neuen Schlüssel überlegen zu müssen.

BibTool enthält einen mächtigen Apparat, der es erlaubt, eindeutige Schlüssel für einen Datensatz zu generieren. Dieser Apparat ist sehr flexibel und in einem hohen Grade konfigurierbar. Im einfachsten Fall jedoch kann eine (sinnvolle) Vorgabe benutzt werden. Dies geschieht mit dem Aufruf

`bibtool -k` *infile* `-o` *outfile*

Der Parameter `-k` bewirkt die Generierung von neuen Schlüsseln. Unser Beispiel-Datensatz würde danach den Schlüssel `goossens.mittelbach.ea:companion` erhalten. Dieser setzt sich aus den Namen der Autoren (höchstens zwei werden explizit gemacht)[2] und dem ersten „relevanten“ Wort zusammen. Die Anzahl der expliziten Autoren und der relevanten Worte sind natürlich konfigurierbar. Die Worte, die nicht als relevant angesehen werden sollen, sind in einer Liste enthalten, die beliebig erweitert werden kann.

Diese Art der Schlüsselgenerierung spiegelt eine sehr persönliche Vorliebe des Autors wieder. Deshalb ist es in BibTool möglich, einen Format-String für die Generierung der Schlüssel anzugeben. Diese Bezeichnung ist eigentlich reichlich untertrieben, da die Möglichkeiten fast zu einer kleinen Programmiersprache ausgeweitet wurden.

Um einen Eindruck von den Möglichkeiten zu geben, betrachten wir einige Beispiele. Am Anfang war der Versuch, die Format-Strings der Programmiersprache C zu imitieren. Als erstes werden alle Zeichen, die keine besondere Bedeutung haben, identisch in den zu generierenden Schlüssel übernommen. Da das `%` kein erlaubtes Zeichen für einen Schlüssel darstellt, kann es zur Einleitung einer Format-Angabe benutzt werden. Das sieht etwa so aus: `%s`. Damit werden alle erlaubten Zeichen in den Schlüssel übernommen. Um anzugeben, welches Feld gemeint ist, wird dieses einfach in Klammern dahinter angegeben: `%s(title)`. Damit haben wir einen gültigen Format-String.

Wie in C ist es auch in BibTool möglich, Parameter zwischen dem `%` und dem Format-Buchstaben – hier `s` – einzufügen. In diesem Fall kann es eine Zahl sein, die angibt, wieviele Zeichen maximal übernommen werden sollen. Also ergibt

[2] `ea` steht für *et alii* (und andere).

`%16s(title)` die ersten erlaubten Zeichen des Feldes `title`, aber höchstens 16 solche Zeichen.

Andere Format-Angaben erlauben die Formatierung eines Namens (`%n`), eines Titels (`%T`) – es werden nur relevante Worte übernommen – und einer Zahl (`%d`) – es werden nur Ziffern übernommen.

Bisher wurde nicht gesagt, was passiert, wenn das angegebene Feld nicht in dem Datensatz enthalten ist. In diesem Fall scheitert hier der Versuch, einen Schlüssel zu generieren. Falls eine Alternative gegeben wurde, wird die nächste Alternative versucht. Alternative Teile eines Schlüssels lassen sich durch das Zeichen `#` trennen und in geschweifte Klammern `{ }` einschließen.

Betrachten wir den folgenden Format-String:

```
{ %n(author) # %n(editor) # nobody }
```

Er spezifiziert drei Alternativen. Diese sind durch `#` getrennt. Zuerst wird versucht, das Feld `author` zu benutzen. Ist es nicht vorhanden, dann wird das Feld `editor` versucht. War dies auch erfolglos, so wird der konstante Wert `nobody` als Schlüssel eingetragen.

An diesem Beispiel sehen wir ebenfalls, daß Leerzeichen (fast) beliebig eingestreut werden können. Da Leerzeichen nicht in einem Schlüssel vorkommen können, werden sie nicht in den automatisch erzeugten Schlüssel übernommen. Sie können jedoch die Lesbarkeit eines Format-Strings erhöhen.

Zusätzlich zu den Alternativen gibt es noch ein *if-then-else*-Konstrukt, bei dem explizit angegeben werden kann, was im Falle der Existenz oder Nichtexistenz eines Feldes zu geschehen hat.

Ein Punkt sei noch zum Abschluß zu erwähnen. Dies betrifft die Formatierung von Namen und Titeln. Hier kann es geschehen, daß die entsprechenden Felder TeX-Makros enthalten. Deshalb ist in BibTool ein kleines Modul enthalten, das TeX-Makros expandiert. Dies kann z. B. dazu benutzt werden, die vordefinierten Makros, wie `\TeX` oder `\LaTeX` in die entsprechenden Strings umzuwandeln – ansonsten werden alle unbekannten Makros einfach zum leeren String expandiert. Weiterhin ergibt sich hierdurch die Möglichkeit, die Umlaute entsprechend umzuwandeln.

Ohne vordefinierte Makros wird z. B. aus dem Autor `M"uller` der Teil eines Schlüssels `mller`. Werden die entsprechenden Makros aktiviert, so ergibt sich `mueller`, was sicher die bessere Alternative darstellt.

BibTool – **Konfigurieren**

Wir haben von der Konfigurierbarkeit von BibTool gehört. Diese kann über die Kommandozeile erfolgen, was aber ziemlich umständlich ist, insbesondere, wenn wir an längere Spezifikationen von Schlüsseln denken. Deshalb kann die Konfiguration in einer Konfigurationsdatei, der sogenannten Resource-Datei, abgelegt werden. Diese Resource-Dateien werden entweder automatisch oder durch einen speziellen Befehl von BibTool ausgewertet.

In den Quellen von BibTool sind einige nützliche Beispiele für Resource-Dateien enthalten, die entweder für eigene Entwicklungen hinzugeladen oder modifiziert werden können. Darunter ist z. B. die bereits erwähnte Definition von wichtigen TeX-Makros.

BibTool – **Extrahieren**

Nun haben wir also erfolgreich megabyte-weise BibTeX-Dateien zusammengesammelt und benutzen diese fleißig. Aber nun wollen wir ein Dokument zusammen mit den Referenzen in Form einer BibTeX-Datei an jemanden weitergeben. Also müssen wir die Referenzen heraussuchen, die in diesem Dokument benutzt werden, und sie in eine eigene BibTeX-Datei schreiben. Aber halt – wir haben ja noch BibTool. BibTool kann uns nämlich diese Aufgabe abnehmen.

Dazu müssen wir einfach die entsprechende `.aux`-Datei erzeugen. Wir haben im ersten Abschnitt gesehen, daß diese Datei die Information enthält, welche Zitate in einem Dokument vorkommen. Weiterhin ist dort die Information enthalten, welche BibTeX-Dateien benutzt werden sollen. Diese Informationen kann BibTool auswerten. Durch einen Aufruf der Form

`bibtool -x` *auxfile* `-o` *outfile*

werden die benutzten Datensätze in die Ausgabedatei *outfile* geschrieben, die genügen sollte, um alle Referenzen des ursprünglichen Dokumentes aufzulösen.

Weitere Möglichkeiten

Die Möglichkeiten von BibTool sind mit den beschriebenen Anwendungen noch nicht ausgeschöpft. Weitere Möglichkeiten sollen hier nur noch stichwortartig vorgestellt werden:

Hinzufügen und Löschen von Feldern kann z. B. dazu benutzt werden, um automatisch die Herkunft zu markieren oder eine Zeitmarke anzubringen.

Extraktion anhand von regulären Ausdrücken erlaubt eine weitere Zerlegung von BibTeX-Dateien, z. B. anhand des ersten Buchstabens des Schlüssels.

Feld-Umformung ermöglicht das konsistente Ändern aller Felder anhand von vorzugebenden Regeln.

Semantische Überprüfungen ermöglichen es, über die syntaktischen Eigenschaften hinaus Überprüfungen durchführen zu lassen, die die Plausibilität der Werte überprüfen. Das kann beispielsweise ein Test sein, ob das `year`-Feld einen sinnvollen Wert enthält.

Expansion von BibTeX-*Strings* nimmt den Mechanismus von BibTeX vorweg, Makros in Strings umzuwandeln.

Statistiken über die analysierten BibTeX-Dateien können ausgegeben werden.

Bisher unerwähnt blieb auch noch, daß die vorgestellten Möglichkeiten nicht nur einzeln genutzt, sondern (fast) beliebig kombiniert werden können.

Quellen

BibTool ist in C geschrieben. Jeder ANSI-C-Compiler sollte in der Lage sein, ein ausführbares Programm zu erzeugen. Bisher ist es zumindest schon auf UNIX-Workstations (Sun, HP, Linux), Atari und unter MS-DOS erfolgreich eingesetzt worden. BibTool wird in Source-Form verteilt. Binär-Versionen sind beim Autoren *nicht* erhältlich.

Wer nach dieser Kurzbeschreibung Geschmack auf BibTool bekommen hat, kann es sich aus verschiedenen Quellen besorgen. An erster Stelle sei hier natürlich der Server von DANTE e.V. (`ftp.dante.de`) genannt. Dort ist BibTool in

```
tex-archive/biblio/bibtex/utils/bibtool/BibTool
```

Selbstverständlich ist es damit auch auf allen CD-ROM-Abzügen der CTAN-Server (z. B. PrimeTimeFreeware TeXcetera oder die CD von DANTE e.V. und Addison-Wesley) zu finden.

Literatur

[1] Michel Goossens, Frank Mittelbach und Alexander Samarin: The LaTeX Companion, Addison-Wesley, Reading, Mass., 1994.

[2] Leslie Lamport: LaTeX: A Document Preparation System, Addison-Wesley, Reading, Mass., 2. Auflage, 1994.

[3] Oren Patashnik: BibTeXing, 1988.

Anmerkung der Redaktion: `bibclean`, ein mit BibTool vergleichbares Werkzeug, stammt aus der Feder Nelson Beebes und ist in dessen Artikel „Bibliography Prettyprinting and Syntax Checking“ beschrieben, der in TUGboat 14(1993), No. 4, S. 395–419 veröffentlicht wurde. Dieses Werkzeug kann man auch auf dem CTAN-Server von DANTE e.V. (`ftp.dante.de`) im Verzeichnis

```
tex-archive/biblio/bibtex/utils/bibclean/
```

als C-Quelle finden. Im Gegensatz zu BibTool erledigt `bibclean` nur den „Syntaxcheck“ und das „Prettyprinting“ von Datenbankdateien, wobei man es durch Konfigurationsdateien an seinen eigenen Stil anpassen kann. Als weitere Option kann es als Frontend benutzt werden, das eine Datenbankdatei in lexikalische Tokens wandelt, die ein nachgeschalteter Parser weiterverarbeiten kann. Neben der Beschreibung von `bibclean` stellt Nelson Beebe eine Analyse des Formats der Literaturdatenbankdateien vor, die BibTeX verwendet. Aus dieser Analyse entwickelt er eine Grammatik, die auch in diesem Artikel veröffentlich ist, und die man für eigene Programmentwicklungen, die diese Datenbankdateien verwenden, nutzen sollte.

Bernd Raichle

Computer Modern Bright

Walter Schmidt

Die Endstrichlose oder Grotesk ist eine
zur Zeit fast unerträglich häufige Schriftart.
Es wäre gut, wenn man ihren Gebrauch auf Schlagzeilen
und wichtige öffentliche Aufschriften ... beschr 01tt0..d u 90.1J