

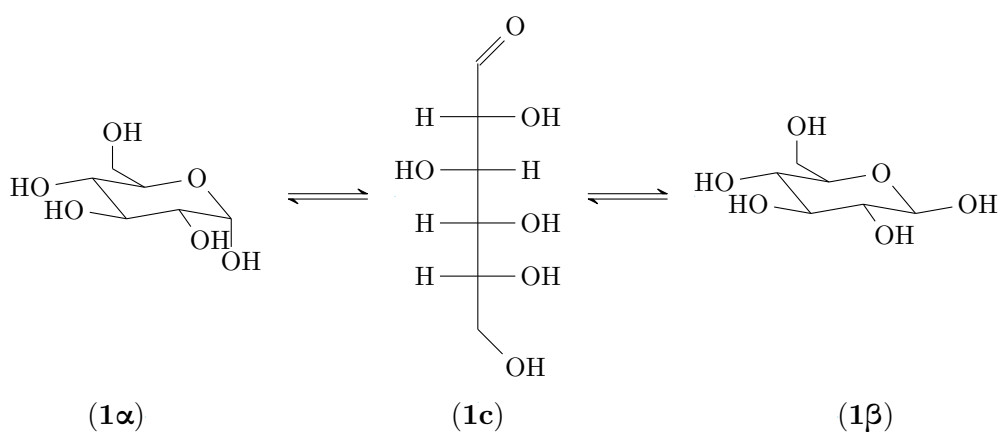
chemnum v0.3

2011/08/02

Clemens NIEDERBERGER

<http://www.mychemistry.eu/>
contact@mychemistry.eu

chemnum ist ein neuer, möglichst umfassender Ansatz zum Nummerieren chemischer Verbindungen.



Schema 1: Das α - und das β -Anomer von D-Glucose (1 α und 1 β) stehen in Lösung über die offenkettige Form (1c) im Gleichgewicht.

Inhaltsverzeichnis

1	Lizenz, Voraussetzungen	3
2	Neu in chemnum v0.3	3
3	Paket-Optionen	3
4	Motivation	4
5	Verbindungen Nummerieren	5
5.1	Basisbefehl	5
5.2	Sublabel	6
5.3	Listen	7
6	Label zentral initialisieren	7
6.1	Verwendung	7
6.1.1	Haupt-Label	7
6.1.2	Sublabel	8
6.2	Fehler statt Warnung	9
6.3	Warum initialisieren?	9
7	Zurücksetzen des Zählers	10
8	Besondere Verwendung	10
8.1	Abschnitts-Titel	10
8.1.1	Test-Überschrift mit Verbindungslabel (7a)	11
8.1.2	Test-Überschrift mit Verbindungslabel (7b)	11
8.2	Gleitumgebungen	11
8.3	Schemata externer Programme	12
9	Personalisieren	15
9.1	Das Prinzip	15
9.2	Schriftstil	16
9.3	Eigene Labels	16
9.4	Eigene Marker in Bildern	16
9.5	Zähler	17
9.6	Begrenzer	17
9.7	Suffix und Präfix	19
9.8	Trenner	19
9.8.1	Bei Labeln	19
9.8.2	Bei Listen – allgemein	20
9.8.3	Bei Listen – letzter Trenner	20
9.9	Sublabel-Marker	21
9.10	Listen und Bereiche von Sublabeln	22
9.11	Übersicht	24

1 Lizenz, Voraussetzungen

chemnum v0.3 steht unter der L^AT_EX Project Public License Version 1.3 oder später.
(<http://www.latex-project.org/lppl.txt>)

chemnum ruft intern die Pakete expl3 und xparse¹ auf. xparse ist Teil des l3packages²-Bundles, expl3 ist Teil des l3kernel³-Bundles. Das heißt, chemnum benötigt L^AT_EX3-Unterstützung.

chemnum benötigt außerdem die Pakete etoolbox⁴ und psfrag⁵ und (manchmal) das Paket textgreek⁶, siehe Abschnitt 3.

2 Neu in chemnum v0.3

Neu seit v0.3 ist folgendes:

- der Key sub-cmpd-sep heißt jetzt sub-marker, siehe Seite 21
- der Key cmpd-sub-counter heißt jetzt sub-counter, siehe Seite 17

Diese Umbenennung folgt dem Wunsch, die Keys ihrer Verwendung nach logisch in verschiedene Klassen zu unterteilen, siehe Seite 15.

- mehrere Keys zum Erstellen von Labels der Art **1a-c,e**, siehe Abschnitt 9.10 ab Seite 22.
- der Befehl `\cmpdref` zum Esetzen von Text-Strings in eps-Dateien, siehe Abschnitt 8.3.
- der Key ref-tag, siehe Seite 16.

3 Paket-Optionen

chemnum hat eine Option.

```
\usepackage[textgreek = <value>]{chemnum}
```

¹<http://www.ctan.org/pkg/xparse>

²<http://www.ctan.org/pkg/l3packages>

³<http://www.ctan.org/pkg/l3kernel>

⁴<http://www.ctan.org/pkg/etoolbox>

⁵<http://www.ctan.org/pkg/psfrag>

⁶<http://www.ctan.org/pkg/textgreek>

Damit können Sie sowohl einen der drei Stile des `textgreek`-Pakets wählen⁷ oder `chemnum` ohne `textgreek` laden. Folgende Werte stehen zur Auswahl: `artemis`, `cbgreek`, `euler` und `false`. Wenn Sie `chemnum` ohne Option verwenden, wird `textgreek` mit dem `cbgreek`-Stil geladen. Die Verwendung von `\usepackage[textgreek]{chemnum}` entspricht `\usepackage[textgreek = cbgreek]{chemnum}` oder `\usepackage{chemnum}`.

Wenn Sie das `textgreek`-Paket nicht installiert haben und keine griechischen Label benötigen, können Sie `\usepackage[textgreek = false]{chemnum}` verwenden. Damit resultieren die Counter-Optionen `greek` und `Greek` in den Optionen `alph` und `Alph` (siehe Abschnitt 9.5).

4 Motivation

Soweit mir bekannt ist, gab es bislang drei Pakete, die das Nummerieren von Verbindungen erleichtern sollen. Sie alle haben ihre Schwächen.

Das erste, `chemcono`⁸, hat für diesen Zweck Bibliographie-Befehle umdefiniert. Dort müssen Verbindungen in einer Art Literaturliste definiert werden. Dadurch kann dann im Text auf sie referenziert werden. Da es die „Verbindungs-Bibliothek“ immer mit ausgibt, schlägt der Paket-Autor folgendes vor:

After compilation and printout, discard the last page.

Stefan Schulz

Ganz offensichtlich ist das keine perfekte Lösung.

Das zweite, `chemcompounds`⁹, ist entstanden, weil der Paket-Autor ebenfalls mit den Schwächen von `chemcono` nicht mehr arbeiten wollte. Bei der Erstellung des Pakets hielt er sich stark an den gleichen prinzipiellen Mechanismus wie `chemcono`.

When taking a closer look at the chemcono package, I realised that the only thing one has to do is to get rid of everything which produces text. Thus, as a basis I used the mechanism of `\bibitem` and `\cite` in pretty much the same way as chemcono does by extracting the corresponding code from `article.cls` and `latex.ltx` but deleting any unnecessary commands producing output. I also introduced several lines of code to make the printing of the compound names more customisable.

Stephan Schenk

Manche Schwächen hat auch dieses Paket:

1. Verbindungen müssen in der Regel mit `\declarecompound` deklariert werden. Sie müssen es auf jeden Fall, wenn man Label der Art **1a** haben möchte. Dann muss man die Label sogar von Hand vergeben, was den Sinn einer automatischen Nummerierung untergräbt.
2. Das Layout lässt sich nicht einzeln ändern, sondern nur für alle Label.
3. Die Nummerierung kann nicht zurückgesetzt werden. Obwohl das in der allermeisten Fällen *weder nötig noch zu empfehlen ist*, kann es Einzelfälle geben, in denen solches Verhalten gewünscht ist.

⁷Zu Details siehe Dokumentation des `textgreek`-Pakets.

⁸<http://www.ctan.org/pkg/chemcono>

⁹<http://www.ctan.org/pkg/chemcompounds>

4. Eine Liste mehrerer Verbindungen `\compound{a,b,c}` kann nicht oder nur umständlich eigenen Vorstellungen angepasst werden.

Dann gibt es noch `bpchem`¹⁰, das Befehle ähnlich den Befehlen `\label` und `\ref` zur Verfügung stellt: `\CNlabel{}`, `\CNlabelnoref{}` und `\CNref{}`. Außerdem hat es Befehle für Sublabel: `\CNlabelsub{ }{ }`, `\CNlabelsubnoref{ }{ }` und `\CNrefsub{ }{ }`. Damit ist es flexibler als die anderen, stellt allerdings kaum Möglichkeiten zur Personalisierung bereit, Listen sind nicht möglich und die Tatsache, dass es verschiedene Befehle für Label und Label mit Sublabeln gibt, ist ebenfalls nicht die beste Lösung.

`chemnum` soll diese Schwächen beheben. Dafür wurden alle Befehle vollständig neu geschrieben. Allerdings wurden Ideen von `chemcompounds` aufgegriffen was Darstellungsmöglichkeiten wie Begrenzer usw. betrifft.

Wenn Ihnen irgendwelche fehlenden Features auffallen, geben Sie mir bitte Bescheid, indem Sie mir eine E-Mail senden.

5 Verbindungen Nummerieren

5.1 Basisbefehl

Der eigentliche Befehl des Pakets ist

`\cmpd{<label name>}`

Beim ersten Aufruf mit `<label name>` wird das Label erzeugt und gespeichert (= deklariert) und ausgegeben. Bei jedem weiteren Aufruf wird nun dieses Label verwendet.

```
1 Verbindungen \cmpd{a} und \cmpd{b} werden deklariert und k"onnen nun
   jederzeit wieder aufgerufen werden: \cmpd{a}. Es ist keine vorherige
   Deklaration n"otig. Die Substanzen wie \cmpd{c} werden in der
   Reihenfolge ihres ersten Erscheinens im Text nummeriert.\par
2 Noch einmal: \cmpd{b}, \cmpd{a}, \cmpd{c}.
```

Verbindungen **1** und **2** werden deklariert und können nun jederzeit wieder aufgerufen werden: **1**. Es ist keine vorherige Deklaration nötig. Die Substanzen wie **3** werden in der Reihenfolge ihres ersten Erscheinens im Text nummeriert.
Noch einmal: **2**, **1**, **3**.

Falls es nötig sein sollte, eine Verbindung zu deklarieren, ohne ein Label auszugeben, kann man an entsprechender Stelle

`\cmpd*{<label name>}`

verwenden. Dadurch wird ein entsprechendes Label deklariert aber nicht ausgegeben. **Dieser Befehl wird nicht benötigt, wenn man `\cmpdinit{}` verwendet, siehe Abschnitt 6 (S. 7ff).**

¹⁰<http://www.ctan.org/pkg/bpchem>

1 Die versteckte Version `\cmpd*{d}` erzeugt zwar das Label, schreibt es aber nicht aus. Der `n` "achste `\cmpd{e}` f"ührt die Zählung einfach weiter. Mit `\cmpd{d}` kann `nat` natürlich auf das erzeugte Label zurückgegriffen werden.

Die versteckte Version erzeugt zwar das Label, schreibt es aber nicht aus. Der nächste **5** führt die Zählung einfach weiter. Mit **4** kann natürlich auf das erzeugte Label zurückgegriffen werden.

Als Label-Namen können Sie im Prinzip einsetzen, was Sie wollen¹¹. Sie sollten allerdings darauf achten, dass der Name keine Leerzeichen enthält. Das führt zwar nicht unbedingt zu einem Fehler, könnte aber verschiedenen Labels die gleiche Nummer zuweisen.

1 `\cmpd{aa}`, `\cmpd{a_a}`, `\cmpd{a_aa}`, `\cmpd{_a_a}`, `\cmpd{_aa}` und `\cmpd{aa_}` erzeugen alle das gleiche Label. So, wie auch `\cmpd{aa}`, `\cmpd{a_a}`, `\cmpd{_aa}` und `\cmpd{aa_}`.

6, **6**, **6**, **6**, **6** und **6** erzeugen alle das gleiche Label. So, wie auch **6**, **6**, **6** und **6**.

5.2 Sublabel

Möchte man Label der Form **1a** erzeugen, geschieht das nach folgender Syntax:

```
\cmpd{<label name>.<subname>}
```

Dabei muss `<label name>` der gleiche bleiben, `<subname>` variiert. Diese Syntax bedeutet, dass der Punkt `.` weder Teil von `<label name>` noch von `<subname>` sein darf.

1 `\cmpd{f.one}` und `\cmpd{f.two}` sind eng verwandt, genauso `\cmpd{g.one}` und `\cmpd{g.two}`. `nat` natürlich bleibt auch hier bei `sp` "aterem Aufruf das Label gleich: `\cmpd{g.two}` und `\cmpd{f.one}`.

7a und **7b** sind eng verwandt, genauso **8a** und **8b**. Natürlich bleibt auch hier bei späterem Aufruf das Label gleich: **8b** und **7a**.

Diese Syntax funktioniert auch, wenn der Hauptname bereits vergeben wurde.

1 <code>\cmpd{a}</code> und seine Varianten	1 und seine Varianten 1a und 1b
2 <code>\cmpd{a.one}</code> und <code>\cmpd{a.two}</code>	

Genauso kann der Hauptname von kombiniert vergebenen Labels einzeln aufgerufen werden.

¹¹Es gibt ein Einschränkungen, siehe Abschnitt 5.2 (S. 6) und Abschnitt 9.9 (S. 21).

Wie Sie ein Label der Art **7a,b** erzeugen können, wird in Abschnitt 9.10 erklärt.

5.3 Listen

Tatsächlich wurde bislang nur die halbe Wahrheit über den Befehl erzählt. Er kann auch für mehrere Label auf einmal verwendet werden.

Die richtige Beschreibung des Befehls lautet also eher:

`\cmpd{<(possibly comma separated list of)label name(s)>}`

1 Mehrere Label können einfach durch Kommata getrennt eingegeben werden. Dann wird eine Liste wie `\cmpd{a, b, c, e, g.two}` ausgegeben.

Mehrere Label können einfach durch Kommata getrennt eingegeben werden. Dann wird eine Liste wie **1, 2, 3, 5, and 8b** ausgegeben.

Wie das nicht-US-Amerikanern merkwürdig anmutende **, and** zwischen **5** und **8b** schon vermuten lässt, kann man die Ausgabe der Liste noch verändern, siehe Abschnitt 9.8 (S. 19).

6 Label zentral initialisieren

Die in diesem Abschnitt beschriebenen Befehle sind für den Einsatz von `chemnum` nicht nötig, stellen aber Funktionen bereit, die der ein oder andere Anwender nützlich finden könnte.

6.1 Verwendung

6.1.1 Haupt-Label

Vielleicht vermissen Sie ja den `\declarecompound`-Befehl von `chemcompounds`. Schließlich hat man dadurch am Beginn einen Überblick, welche Namen man bereits vergeben hat. Und man wird durch eine Fehlermeldung unter Umständen gewarnt, wenn man ein falsches Label setzt¹². Dieses Verhalten lässt sich mit den folgenden Befehlen realisieren.

`\cmpdinit{<comma separated list of label names>}`
`\cmpdinit*{<comma separated list of label names>}`

¹²z. B. aufgrund eines Schreibfehlers

Alle Verbindungen, die innerhalb dieses Befehls gesetzt werden, werden *in der verwendeten Reihenfolge* initialisiert. Verwendet man ein Label, das nicht in der Liste enthalten ist, wird eine Warnung ausgegeben.

```
1 \cmpdinit{A, B, C} \cmpd{B}           2 1a 3 4
2 \cmpd{A.a} \cmpd{C} \cmpd{D}
```

```
*****
* chemnum warning: "cmpd-init"
*
* You used \cmpdinit but didn't initiate compound "D" on line 1.
*****
```

Beachten Sie, dass Sie bei Labeln mit Sublabeln *nur den Hauptnamen* initialisieren können und sollten. Dadurch sind dann *alle* Verbindungen mit gleichem Hauptnamen initialisiert.

`\cmpdinit*{}` gibt die gleiche Warnung aus, allerdings werden die Nummern noch nicht vergeben, sondern erst bei Verwendung im Text und in der im Text verwendeten Reihenfolge.

```
1 \cmpdinit*{X, Y, Z} \cmpd{Y}         1 2a 3 4
2 \cmpd{X.x} \cmpd{Z} \cmpd{W}
```

```
*****
* chemnum warning: "cmpd-init"
*
* You used \cmpdinit but didn't initiate compound "W" on line 1.
*****
```

Wenn Sie also eigene Label vergeben wollen oder müssen (siehe Seite 16) und die Label geschlossen initialisieren wollen, sollten Sie `\cmpdinit*{}` verwenden, da dort bei der Initialisierung *keine* Label vergeben werden.

Der Befehl kann mehrfach verwendet werden, mit oder ohne *. Sinnvoll ist aber wohl eine *einmalige* Verwendung, vielleicht in der Dokumentenpräambel nach dem chemnum-Setup (siehe Abschnitt 9).

6.1.2 Sublabel

Wenn Sie zusätzlich auch für jede Label-Sublabel-Kombination gewarnt werden wollen oder die Reihenfolge im Voraus festlegen wollen, dann können Sie

```
\cmpdinit[sub-init = true]{<comma separated list of label names>}
```

verwenden. So werden Sie auch in diesen Fällen gewarnt.


```

1 \cmpdinit[sub-init = true]{E.e}          1a and 1b
2 \cmpd{E.e, E.f}

```

```

*****
* chemnum warning: "cmpd-sub-init"
*
* You used \cmpdinit and "sub-init = true" but didn't initiate sub-compound
* "E.f" on line 1.
*****

```

6.2 Fehler statt Warnung

Wenn Sie lieber eine Fehler-Meldung anstelle einer Warnung mögen, können Sie das realisieren, indem Sie `\cmpdinit` folgendermaßen aufrufen:

```
\cmpdinit[strict = true]{<comma separated list of label names>}
```

Die Verwendung `strict=false` entspricht der Verwendung ohne optionalem Argument. Die Option kann natürlich auch bei der Stern-Variante eingesetzt werden.

Durch `strict=true` wird der L^AT_EX-Lauf mit einer entsprechenden Fehlermeldung abgebrochen, wenn man einen Label-Namen verwendet, der nicht initialisiert wurde.

Natürlich können Sie `\cmpdinit` auch mit beiden Keys aufrufen:

```
\cmpdinit[strict = true, sub-init = true]{<comma separated list of label
names>}
```

Wenn Sie einen anderen Sub-Label-Marker ausgewählt haben (siehe Seite 21), müssen Sie mit `\cmpdinit[sub-init = true]` ebenfalls verwenden.

6.3 Warum initialisieren?

Wie zu Beginn dieses Abschnitts erwähnt, ist die Initialisierung von Labeln nicht nötig. Es gibt jedoch Gründe, warum man `\cmpdinit` vielleicht trotzdem verwenden möchte.

1. Der Einsatz von `\cmpdinit*` ist eine Möglichkeit, den Überblick darüber zu behalten, welche Label man verwendet hat. Man bekommt eine Warnung/Fehlermeldung, wenn man ein bestehendes falsch geschrieben hat. Abhängig davon, wieviele Labels man verwendet (oder verwenden muss) kann das aber viel zusätzliche Arbeit bedeuten.
2. Der Einsatz von `\cmpdinit` bewirkt erst einmal dasselbe wie `\cmpdinit*`, deklariert aber *zusätzlich* die Label in der vorgegebenen Reihenfolge. Damit wird die Verwendung von `\cmpd*` (Abschnitt 5.1) und `\cmpd+` (Abschnitt 8.1) überflüssig. Man hat damit direkte Kontrolle, welche Verbindung welche Nummer erhält.

7 Zurücksetzen des Zählers

Es ist möglich, mit folgendem Befehl

`\cmpdreset[<number>]`

den Zähler wieder zurückzusetzen. Ohne Argument wird er auf 1 gesetzt und entspricht damit `\cmpdreset[1]`. Dieser Befehl wirkt sich global aus, nicht nur lokal innerhalb einer Gruppe – im Gegensatz zu den meisten anderen vorgestellten Befehlen.

```
1 \cmpdreset
2 Die Verbindungen werden jetzt wieder von 1 gez\"ahlt: \cmpd{h, i, j}
```

Die Verbindungen werden jetzt wieder von 1 gezählt: **1**, **2**, and **3**

Sie sollten mit diesem Befehl sehr vorsichtig umgehen! In der Regel ist es nicht zu empfehlen, den Zähler zurückzusetzen. Dadurch können ein und dieselbe Nummer an verschiedene Verbindungen vergeben werden!

```
1 Gleiche Nummern: \cmpd{a}, \cmpd{h} Gleiche Nummern: 1, 1
```

8 Besondere Verwendung

8.1 Abschnitts-Titel

Die Verwendung von `\cmpd` in einer Überschrift scheint auf den ersten Blick kein Problem zu sein. Es gibt keine Fehler und der Output sieht wie erwartet aus:

```
1 \section{Compound \cmpd[cmpd-delim]{b}}
2 \ldots
```

Wenn man allerdings `\tableofcontents` verwendet – was ziemlich wahrscheinlich ist, wie ich vermute –, geschieht etwas unerwünschtes: da das Inhaltsverzeichnis am Anfang des Dokuments geschrieben wird, wird das Label beim Erscheinen im Inhaltsverzeichnis deklariert, anstatt an der Stelle im Text, an der es zuerst erscheint. Es gibt zwei Möglichkeiten, das zu umgehen:

1. Sie deklarieren die Label in der Präambel mit `\cmpdinit{}` (siehe Seite 7).
2. Sie verwenden `\cmpd+`, um das Label zu setzen. Damit wird das Label von der Datei `<jobname>.cmpd` ausgelesen (vorausgesetzt ihre Hauptdatei heißt `<jobname>.tex`). Das bedeutet, dass *mindestens zwei*, unter Umständen mehr L^AT_EX Durchläufe nötig sind, bis alle Labels richtig gesetzt sind.

Das gleiche gilt für jede andere Liste von beweglichen Argumenten, die *vor* dem Erscheinen des Labels im Text eingesetzt wird.

Der Befehl

`\cmpd+{<label name>}`

liest den Labelnamen, der ausgegeben werden soll, aus der Datei <jobname>.cmpd. Damit ist er sozusagen das Gegenteil von `\cmpd*`, da das Label nicht deklariert sondern nur referenziert und ausgegeben wird. Damit man also `\cmpd+{<label>}` einsetzen kann, muss <label> entweder mit `\cmpd{<label>}` oder `\cmpd*{<label>}` (davor oder danach) deklariert worden sein. `\cmpd*` ist nützlich, wenn ein Label in einem beweglichen Argument wie in `\section{}` eingesetzt werden soll. **Er wird nicht benötigt, wenn `\cmpdinit{}` verwendet wird, siehe Abschnitt 6 (S. 7ff).**

```
1 \subsubsection{Test-\text{Überschrift mit Verbindungslabel \cmpd+[cmpd-delim]{f
   .one}}
2 % mit hyperref:
3 \subsubsection{Test-\text{Überschrift mit Verbindungslabel \texorpdfstring{\
   cmpd+[cmpd-delim]{f.two}}{(7b)}}
```

8.1.1 Test-Überschrift mit Verbindungslabel (7a)

8.1.2 Test-Überschrift mit Verbindungslabel (7b)

8.2 Gleitumgebungen

Die Verwendung von `\cmpd` in Gleitumgebungen sollte kein Problem darstellen: die Labels werden in der Reihenfolge deklariert, wie Sie im Code auftauchen, eingeschlossen die Gleitumgebungen. Um auf der sicheren Seite können Sie natürlich dennoch `\cmpd+` verwenden.

```
1 % preamble:
2 % \usepackage{chemscheme}
3 % \renewcommand*\schemename{Schema}
4 % document:
5 \cmpdreset\cmpd{float1, float2}. Und jetzt ein Schema:
6 \begin{scheme}[ht]
7 \centering
8 \caption{Dieses Label sollte eine 2 sein: \cmpd{float2}}
9 Wenn 2 = \cmpd{float2} wahr ist, ist alles ok.
10 \end{scheme}
```

1 and 2. Und jetzt ein Schema:

Es funktioniert auch alles wie gewünscht, wenn die Gleitumgebung vor den aktuellen Code rutscht, etwa mit der Option [t].

Wenn $2 = 2$ wahr ist, ist alles ok.

Schema 2: Dieses Label sollte eine 2 sein: **2**

8.3 Schemata externer Programme

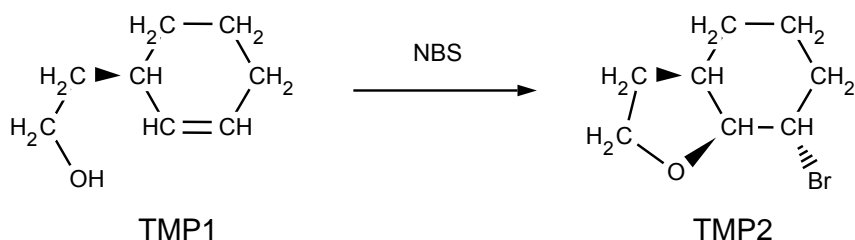
Wenn Sie Ihre Schemata mit externen Programmen wie z. B. CHEMDRAW erstellen und dort ebenfalls chemnum einsetzen wollen, dann können Sie das mit

`\cmpdref[<keyval>][<tag>]{<label name>}`

tun. Dieser Befehl ist vom Befehl `\schemeref` des Pakets `chemscheme`¹³ inspiriert und funktioniert letztlich gleich. Sie erstellen das Schema und speichern es mit temporären Labeln als `eps`-Datei.

```
1 \begin{scheme}[ht]
2 \centering
3 \includegraphics{scheme-tmp}
4 \caption{Schema mit temporären Markern.}
5 \end{scheme}
6 \begin{scheme}[ht]
7 \centering
8 \cmpdref{Alc} % ersetzt TMP1
9 \cmpdref{EtherBr} % ersetzt TMP2
10 % \cmpdref{drittes} wuerde TMP3 ersetzen
11 \includegraphics{scheme-tmp}
12 \caption{Schema mit automatisierten Labeln.}
13 \end{scheme}
```

Sie nummerieren also im Schema die Substanzen mit TMP1, TMP2 usw. durch. Diese

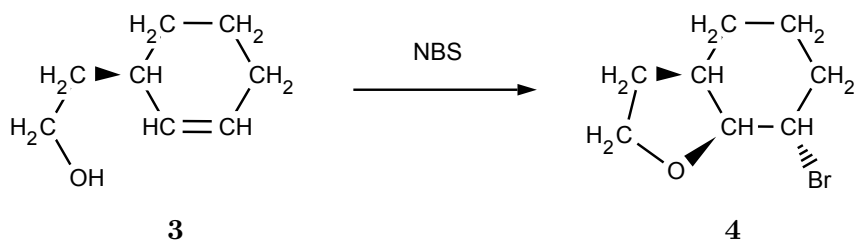


Schema 3: Schema mit temporären Markern.

Marker werden dann durch die entsprechenden Label ersetzt. Dafür verwendet `chemnum` genau wie `chemscheme` den Befehl `\psfrag`. Damit er funktioniert, müssen Sie also entweder über LATEX, DVIPS, PS2PDF kompilieren, oder z. B. das Paket `auto-pst-pdf`¹⁴ verwenden, um trotzdem PDFLATEX einsetzen zu können.

¹³<http://www.ctan.org/pkg/chemscheme>

¹⁴<http://www.ctan.org/pkg/auto-pst-pdf>



Schema 4: Schema mit automatisierten Labeln.

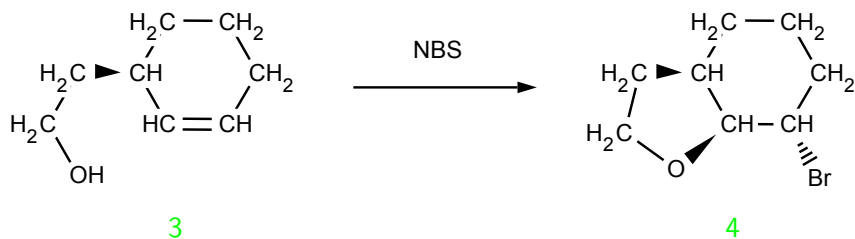
In jedem Fall ist es wichtig, dass der Marker als *Text* im **eps** gespeichert ist.

Mit den Keys, die in Abschnitt 9 vorgestellt werden, können Sie die Label hier auch personalisieren.

```

1  \begin{scheme}[ht]
2  \centering
3  \cmpdref[cmpd-style=\bf\sf\textcolor{green}]{Alc} % ersetzt TMP1
4  \cmpdref[cmpd-style=\bf\sf\textcolor{green}]{EtherBr} % ersetzt TMP2
5  \includegraphics{scheme-tmp}
6  \caption{Schema mit automatisierten Labeln.}
7  \end{scheme}

```



Schema 5: Schema mit automatisierten Labeln.

Sie können auch beliebigen Text ersetzen. Dafür gibt es das zweite optionale Argument.

```

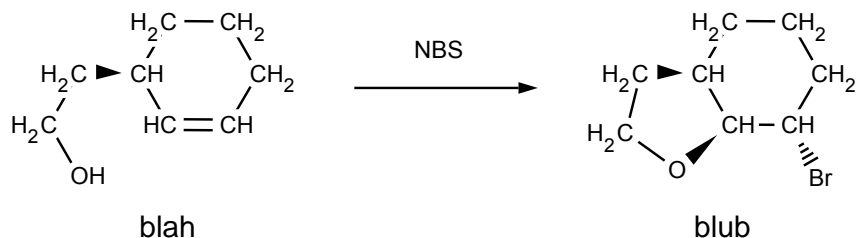
1  \begin{scheme}[ht]
2  \centering
3  \includegraphics{scheme-bla}
4  \caption{Schema mit beliebigen Markern.}
5  \end{scheme}
6  \begin{scheme}[ht]
7  \centering
8  \cmpdref[][blah]{Alc}
9  \cmpdref[][blub]{EtherBr}
10 % \cmpdref{drittes} wurde TMP1 ersetzen
11 \includegraphics{scheme-bla}

```

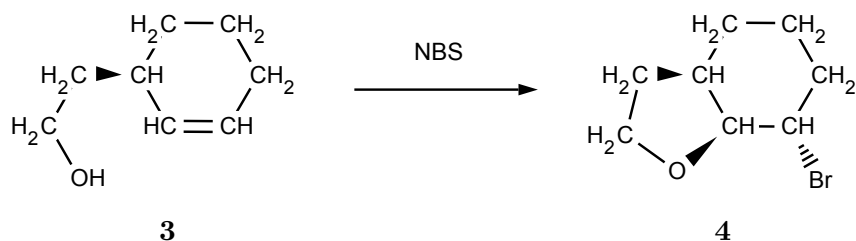
```

12 \caption{Schema mit explizit gesetzten Labeln.}
13 \end{scheme}

```



Schema 6: Schema mit beliebigen Markern.



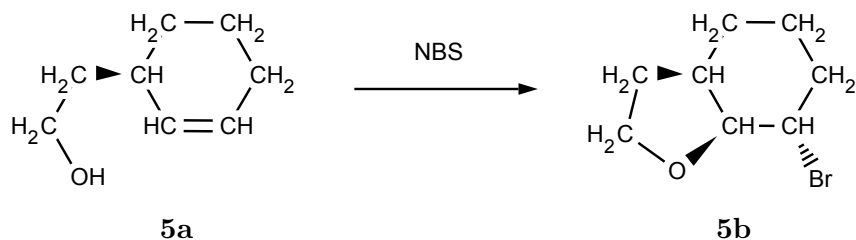
Schema 7: Schema mit explizit gesetzten Labeln.

Natürlich funktioniert das mit Sublabeln nach dem gewohnten Muster:

```

1 \begin{scheme}[ht]
2 \centering
3 \cmpdref{cpd.red}
4 \cmpdref{cpd.ox}
5 \includegraphics{scheme-tmp}
6 \caption{Schema mit automatisierten Labeln und Sublabeln.}
7 \end{scheme}

```



Schema 8: Schema mit automatisierten Labeln und Sublabeln.

9 Personalisieren

9.1 Das Prinzip

Das Personalisieren geschieht mit einem Schlüssel-Wert-System. Einstellungen können dabei entweder zentral über

```
\cmpdsetup{key1 = value1, key2 = value2}
```

oder lokal direkt beim Befehl

```
\cmpd[key1 = value1, key2 = value2]{<label name>}
```

vorgenommen werden.

Dabei gibt es verschiedene Arten von Schlüsseln. Je nach Art können sie andere Werte bekommen.

macro Diese Schlüssel erwarten als Eingabe ein Makro, das auch ein obligatorisches Argument besitzen kann, z. B. `\bfseries` oder `\textbf`. `chemnum` hat nur einen einzigen Schlüssel dieser Art, siehe Abschnitt 9.2.

literal Diese Sorte Schlüssel verwendet das eingegebene direkt, also buchstäblich. Die meisten Schlüssel von `chemnum` entsprechen dieser Art.

choice Auswahl-Schlüssel, für den es vorgegebene Optionen gibt, aus denen gewählt werden kann.

Wenn Sie Schlüssel ohne Wertzuweisung verwenden `\cmpdsetup{key1, key2}`, werden Defaultwerte verwendet. Diese sind *nicht* unbedingt leer und stimmen auch *nicht* unbedingt mit der Voreinstellung des Pakets überein. *Alle* Schlüssel von `chemnum` haben Defaultwerte. Sie werden in Abschnitt 9.11 auf Seite 24 aufgelistet.

Die Schlüssel sind in verschiedene Klassen unterteilt, die auf ihren Verwendungsbereich hindeuten. Sie folgen alle der Form `<class>-key = <value>`:

cmpd Schlüssel, die damit beginnen, wirken sich auch das Label insgesamt oder speziell auf das Hauptlabel aus.

sub Diese Schlüssel haben mit den Sublabeln direkt zu tun.

list Schlüssel, die die Ausgabe der Listen ändern.

ref Schlüssel, die speziell mit `\cmpdref` zu tun haben.

9.2 Schriftstil

Um den Stil zu ändern, mit dem die Nummern geschrieben werden, verwenden Sie

```
cmpd-style = <style>
```

Typ: macro

```
1 % preamble:
2 % \usepackage[normalem]{ulem}
3 % document:
4 \cmpd{a, b, f.two, k}\\ % default
5 \cmpd[cmpd-style = \uline]{a, b, f.two, k}\\ % unterstrichen
6 \cmpd{a, b, f.two, k}\\ % wieder default
7 \cmpdsetup{cmpd-style = \itshape\uline}
8 \cmpd{a, b, f.two, k} % kursiv und unterstrichen
```

1, 2, 7b, and 9
1, 2, 7b, and 9
1, 2, 7b, and 9
1, 2, 7b, and 9

9.3 Eigene Labels

Mit dem Schlüssel

```
cmpd-label = <name>
```

Typ: literal

kann man eigene Labels vergeben.

```
1 Eigenes Label: \cmpd[cmpd-label = XYZ]{l}, wird bei einer Liste an \emph{
  alle neuen} vergeben! \cmpd[cmpd-label = XYZ]{l, a, m}
```

Eigenes Label: **XYZ**, wird bei einer Liste an *alle neuen* vergeben! **XYZ**, **1**, and **XYZ**

9.4 Eigene Marker in Bildern

Wenn Sie für die temporären Label, die durch `\cmpdref` (siehe Seite 12) nicht TMP verwenden wollen, können Sie das ändern, z. B. in `tmp`:

```
1 \cmpdsetup{ref-marker = tmp}
```


9.5 Zähler

Wenn Sie mögen, können Sie auch das Zählerformat **1a** ändern.

<code>cmpd-counter = <counter></code>	Typ: choice
<code>sub-counter = <counter></code>	Typ: choice

Zur Auswahl stehen `arabic`, `alph`, `Alph`, `greek`, `Greek`, `roman`, `Roman` und `Symbol`.

Bitte beachten Sie: ein Ändern des Zählerformats wirkt sich nur auf *noch nicht erzeugte* Label aus!

```
1 \cmpdsetup{cmpd-counter = Alph, sub-counter = arabic}
2 \cmpd{a, b, f.two, k}\\ % keine Wirkung bei bereits erzeugten Labeln!
3 \cmpd{n.one,n.two,o.one}\\
4 \cmpd{f.three,f.four}\\ % Vorsicht: Hauptlabel wurde bereits erzeugt und
   wird nicht geändert!
5 \cmpd[cmpd-counter,sub-counter = greek]{p.one,p.two}
```

1, 2, 7b, and 9

J1, J2, and K1

73 and 74

12 α and 12 β

9.6 Begrenzer

Sie können sowohl den einzelnen Nummern wie auch eine ganze Liste mit Begrenzern versehen. Dafür gibt es die Schlüssel:

<code>cmpd-delim = <odelim><cdelim></code>	Typ: literal
<code>list-delim = <odelim><cdelim></code>	Typ: literal
Benötigen <i>zwei</i> Token als Wert!	

```
1 \cmpd{a, b, f.two, k}\\ % default
2 \cmpd[cmpd-delim = ()]{a, b, f.two, k}\\ % eingeklammert
3 \cmpd{a, b, f.two, k}\\ % wieder default
4 \cmpdsetup{cmpd-delim = ()}
5 \cmpd{a, b, f.two, k} % eingeklammert
```

1, 2, 7b, and 9

(1), (2), (7b), and (9)

1, 2, 7b, and 9

(1), (2), (7b), and (9)

Bitte Beachten Sie, dass die Default-Werte (siehe S. 15 und 24) von `cmpd-delim` und `list-delim` und die Voreinstellung von `chemnum` *nicht* übereinstimmen. Per Default verwendet

chemnum keine Begrenzer. Beachten Sie außerdem, dass die Listenbegrenzer erst ab *zwei* Labeln verwendet werden.

```

1 \cmpd{a, b, f.two, k}\\ % default
2 \cmpd[list-delim = {[}{]}]{a, b, f.two, k}\\ % eingeklammert
3 \cmpd{a, b, f.two, k}\\ % wieder default
4 \cmpdsetup{list-delim = []}
5 \cmpd{a, b, f.two, k}\\ % eingeklammert
6 \cmpd{a} % KEINE Liste!

1, 2, 7b, and 9
[1, 2, 7b, and 9]
1, 2, 7b, and 9
[1, 2, 7b, and 9]
1

```

Wenn Sie generell die Label eingeklammert verwenden, aber ein einzelnes einmal ohne Klammern verwenden wollen, haben Sie mehrere Möglichkeiten:

```

1 \cmpdsetup{cmpd-delim = ()}
2 Normal \cmpd{b}, \cmpd{c}, \cmpd{d.one}, aber manchmal so \cmpd[cmpd-delim
   = ]{e} oder so \cmpd-{e}.

Normal (2), (3), (4a), aber manchmal so 5 oder so 5.

```

Mit

$$\begin{array}{l} \backslash\text{cmpd}-\{\langle\text{label name}\rangle\} \\ \backslash\text{cmpdref}-\{\langle\text{label name}\rangle\} \end{array}$$

kann man die Klammern eines Labels schnell entfernen (nicht jedoch die der Liste).

```

1 \cmpd{a, b, f.two, k}\\ % default
2 \cmpdsetup{cmpd-delim = (), list-delim = []}
3 \cmpd{a, b, f.two, k}\\ % doppelt eingeklammert
4 \cmpd-{a, b, f.two, k} % einfach eingeklammert

1, 2, 7b, and 9
[(1), (2), (7b), and (9)]
[1, 2, 7b, and 9]

```

Durch leere Argumente, $\backslash\text{cmpdsetup}\{\text{cmpd-delim} = \}$ und $\backslash\text{cmpdsetup}\{\text{list-delim} = \}$, können Sie die Voreinstellung wiederherstellen.

9.7 Suffix und Präfix

Falls Sie wollen, können sie sowohl einzelnen Labeln als auch den Listen Prä- und Suffixe geben.

<code>cmpd-prefix = <prefix></code>	Typ: literal
<code>cmpd-suffix = <prefix></code>	Typ: literal
<code>list-prefix = <prefix></code>	Typ: literal
<code>list-suffix = <prefix></code>	Typ: literal

Wie bei den Begrenzern gilt: Listen-Attribute werden erst bei einer Liste – also zwei oder mehr Labeln – verwendet.

Beispiel für die Label-Attribute:

```
1 \cmpd{a, b, f.two, k}\\ % default
2 \cmpd[cmpd-prefix = Nr.]{a, b, f.two, k}\\
3 \cmpdsetup{cmpd-prefix = \textnumero}
4 \cmpd{a, b, f.two, k}\\
5 \cmpd{a} % KEINE Liste!
```

1, 2, 7b, and 9

Nr. **1**, Nr. **2**, Nr. **7b**, and Nr. **9**

Nº **1**, Nº **2**, Nº **7b**, and Nº **9**

Nº **1**

Beispiel für die Listen-Attribute:

```
1 \cmpd{a, b, f.two, k}\\ % default
2 \cmpd[list-prefix = Liste:]{a, b, f.two, k}\\
3 \cmpdsetup{list-prefix = Sammlung:}
4 \cmpd{a, b, f.two, k}\\
5 \cmpd{a} % KEINE Liste!
```

1, 2, 7b, and 9

Liste: **1, 2, 7b, and 9**

Sammlung: **1, 2, 7b, and 9**

1

9.8 Trenner

9.8.1 Bei Labeln

Haben Sie Label mit Sublabeln, können Sie dort auch ein Trenn-Symbol verwenden.

<code>cmpd-sep = <separator></code>	Typ: literal
---	--------------

```

1 \cmpd{a, b, f.two, f.three, k}\\ % default
2 \cmpd[cmpd-sep = -]{a, b, f.two, f.three, k}\\
3 \cmpdsetup{cmpd-sep = $\cdot$}
4 \cmpd{a, b, f.two, f.three, k}\\
5 \cmpd[cmpd-sep = :]{a, b, f.two, f.three, k}

```

1, 2, 7b, 73, and 9
 1, 2, 7-b, 7-3, and 9
 1, 2, 7·b, 7·3, and 9
 1, 2, 7:b, 7:3, and 9

9.8.2 Bei Listen – allgemein

Auch bei Listen lässt sich ein Trennsymbol festlegen, das bestimmt, wie einzelne Label voneinander getrennt werden. Per Default ist das das Komma.

`list-sep = <separator>`

Typ: literal

```

1 \cmpd{a, b, f.two, k}\\ % default
2 \cmpd[list-sep = ]{a, b, f.two, k}\\
3 \cmpd{a, b, f.two, k}\\
4 \cmpdsetup{list-sep = ;}
5 \cmpd{a, b, f.two, k}

```

1, 2, 7b, and 9
 1 2 7b and 9
 1, 2, 7b, and 9
 1; 2; 7b; and 9

9.8.3 Bei Listen – letzter Trenner

Der letzte Trenner bei Listen, derjenige zwischen dem vorletzten und letzten aufgelisteten Label, ist ein besonderer Fall. Hierfür gibt es zwei Schlüssel, mit dem er angepasst werden kann.

`list-last-sep = <separator>`

Typ: literal

`list-lang = <lang>`

Typ: choice

Zum einen gibt es den choice-Schlüssel `list-lang`, mit dem Sprachspezifische Einstellungen geändert werden können. Zur Auswahl stehen US (default), GB, DE, FR, ES und IT¹⁵.

¹⁵Wenn Ihnen eine Sprache fehlt oder Sie einen Fehler bei den Übersetzungen finden, senden Sie mir doch bitte eine E-Mail.

```

1 \cmpd[list-lang = US]{a, b, f.two, k} \cmpd[list-lang = US]{a, b}\\
2 \cmpd[list-lang = GB]{a, b, f.two, k} \cmpd[list-lang = GB]{a, b}\\
3 \cmpd[list-lang = DE]{a, b, f.two, k} \cmpd[list-lang = DE]{a, b}\\
4 \cmpd[list-lang = FR]{a, b, f.two, k} \cmpd[list-lang = FR]{a, b}\\
5 \cmpd[list-lang = ES]{a, b, f.two, k} \cmpd[list-lang = ES]{a, b}\\
6 \cmpd[list-lang = IT]{a, b, f.two, k} \cmpd[list-lang = IT]{a, b}

1, 2, 7b, and 9 1 and 2
1, 2, 7b and 9 1 and 2
1, 2, 7b und 9 1 und 2
1, 2, 7b et 9 1 et 2
1, 2, 7b y 9 1 y 2
1, 2, 7b e 9 1 e 2

```

Zum anderen können Sie den letzten Trenner mit `list-last-sep` auch individuell festlegen.

```

1 \cmpdsetup{list-lang = DE}%
2 \cmpd[list-last-sep = {und auch}]{a, b, f.two, k}\\
3 \cmpd[list-last-sep = ]{a, b, f.two, k}\\
4 \cmpd[list-last-sep = sowie]{a, b, f.two, k}\\
5 \cmpd[list-last-sep = empty]{a, b, f.two, k}

1, 2, 7b und auch 9
1, 2, 7b 9
1, 2, 7b sowie 9
1, 2, 7b, 9

```

Ein spezieller Wert ist `empty`. Wird er verwendet, wird überall der Trenner eingesetzt, der mit `list-sep` festgelegt wurde.

```

1 \cmpdsetup{list-last-sep = empty}%
2 \cmpd{a, b, f.two, k}\\
3 \cmpd[list-sep = ;]{a, b, f.two, k}\\
4 \cmpd[list-sep = {\ und}]{a, b, f.two, k}

1, 2, 7b, 9
1; 2; 7b; 9
1 und 2 und 7b und 9

```

9.9 Sublabel-Marker

Als Voreinstellung verwendet `chemnum` den Punkt `.` als Zeichen, um Hauptlabelnamen von Sublabelnamen zu unterscheiden. Sie können das nach Belieben ändern. Die Zeichen `,` `%` `#` können Sie nicht verwenden und `@` sollten Sie nicht verwenden.

sub-marker = <separator>

Typ: literal

```
1 \cmpdsetup{sub-marker = !}%
2 \cmpd{f!one, g!two}\\
3 \cmpd[sub-marker= +]{f+one, g+two}\\
4 \cmpd[sub-marker= ~]{f~one, g~two}\\
5 \cmpd[sub-marker= &]{f&one, g&two}\\
6 \cmpd[sub-marker= *]{f*one, g*two}
```

7a and 8b

7a and 8b

7a and 8b

7a and 8b

7a and 8b

Sie sollten – nicht nur aus Gründen der Konsistenz – nur *einmal* am Anfang ihres Dokuments entscheiden, welches Zeichen Sie verwenden. Und zwar *bevor* Sie `\tableofcontents` einsetzen. Am besten in der Präambel. Ansonsten kann `\tableofcontents` nicht wissen, welchen Marker Sie verwenden und Label im Inhaltsverzeichnis können falsch dargestellt werden.

9.10 Listen und Bereiche von Sublabeln

Manchmal kann es nützlich oder erwünscht sein, ein Label mit einem Bereich von Sublabeln anzuzeigen. Nehmen wir an, Sie habe die Verbindungen **13a**, **13b**, **13c**, **13d**, and **13e**, die sich z. B. in ihren Substituenten unterscheiden. Unter Umständen möchte man auf alle gleichzeitig verweisen: **13a-e**.

`chemnum` stellt zwei Keys zur Verfügung, mit denen solche Label erzeugt werden können, und zwei, mit denen sie angepasst werden können.

cmpd-all = <bool>

Typ: boolean

sub-list = {<list of sublabel names>}

Typ: other

sub-range-sep = <separator>

Typ: literal

sub-range-marker = <marker>

Typ: literal

Der Typ des Keys `sub-list` unterscheidet sich von dem anderer Keys. Er ähnelt `literal`. Am besten schauen Sie Sich das Beispiel unten an.

```
1 Liste von Labeln: \cmpd{q.one, q.two, q.three, q.four, q.five} \\
2 Alle auf einmal: \cmpd[cmpd-all]{q}
```

Liste von Labeln: **13a**, **13b**, **13c**, **13d**, and **13e**

Alle auf einmal: **13a-e**

Wenn Sie nicht auf alle Varianten auf einmal verweisen wollen, können Sie den Key `sub-list` verwenden. Als Eingabe benötigt er eine Komma-separierte Liste der Sublabel-Namen, die zu dem Hauptlabel gehören, das Sie verwenden.

```
1 \cmpd[sub-list={one,three,four}]{q} 13a,c,d
```

Sie können mit diesem Key auch Bereiche von Sublabeln darstellen. Als Marker zwischen den Namen der beiden Sublabel werden zwei Punkte `..` verwendet.

```
1 \cmpd[sub-list={two..four}]{q}\
2 \cmpd[sub-list={one,three..five}]{q}\
3 \cmpd[sub-list={one..three,five}]{q}

13b-d
13a,c-e
13a-c,e
```

Mit den Keys `sub-range-sep` und `sub-range-marker` kann man sowohl das Symbol, mit dem der Bereich angezeigt wird als auch den Marker für die Eingabe ändern.

```
1 \cmpdsetup{sub-range-sep = {--}, sub-range-marker = : }
2 \cmpd[sub-list={two:four}]{q}

13b--d
```

Damit die Keys die richtigen Sublabel verwenden können, müssen diese deklariert worden sein. Daher werden sie aus der Datei `<jobname>.cmpd` (angenommen Ihre Quelldatei heißt `<jobname>.tex`) gelesen. Daher müssen Sie zweimal (oder öfter) kompilieren, bis alle Labels richtig dargestellt werden.

In der Zwischenzeit benötigt der Key `sub-list` etwas Aufmerksamkeit. Leider ist automatisches Sortieren nicht so trivial, wie man sich wünschte. Daher müssen Sie die Sublabel-Namen in der richtigen Reihenfolge einsetzen, um Label wie diese zu vermeiden:

```
1 \cmpd[sub-list={five..three}]{q} or \cmpd[sub-list={three,one,four}]{q}.

13e-c or 13c,a,d.
```

9.11 Übersicht

Im folgenden sind alle verfügbaren Schlüssel aufgelistet. Bitte beachten Sie, dass die Default-Werte von `cmpd-delim` und `list-delim` *nicht* mit der Voreinstellung von `chemnum` übereinstimmen. `chemnum` setzt in der Voreinstellung *keine* Begrenzer.

key	default	type
<code>cmpd-style</code>	<code>\textbf</code>	macro
<code>cmpd-label</code>		literal
<code>cmpd-delim^a</code>	<code>()</code>	literal
<code>cmpd-odelim</code>		literal
<code>cmpd-cdelim</code>		literal
<code>cmpd-prefix</code>		literal
<code>cmpd-suffix</code>		literal
<code>cmpd-sep</code>		literal
<code>cmpd-counter</code>	arabic	choice
<code>cmpd-all</code>	true	boolean
<hr/>		
<code>sub-marker</code>	<code>.</code>	literal
<code>sub-counter</code>	alph	choice
<code>sub-list</code>		other ^b
<code>sub-range-sep</code>	<code>-</code>	literal
<code>sub-range-marker</code>	<code>..</code>	literal
<hr/>		
<code>list-delim^a</code>	<code>()</code>	literal
<code>list-odelim</code>		literal
<code>list-cdelim</code>		literal
<code>list-prefix</code>		literal
<code>list-suffix</code>		literal
<code>list-sep</code>	<code>,</code>	literal
<code>list-lang</code>	US	choice
<code>list-last-sep</code>	and ^c	literal
<hr/>		
<code>ref-tag</code>	TMP	literal

a Benötigt *zwei* Token als Wert.

b Eingabe ist eine
Komma-separierte Liste, siehe
Seite 23.

c Hängt vom Wert von
`list-lang` ab.

10 Liste der Befehle

Im folgenden werden alle von chemnum definierten Befehle und deren Varianten aufgelistet.

Befehl	Beschreibung
<code>\cmpd[]{}{}</code>	Hauptbefehl, erzeugt und schreibt Label oder Liste von Labeln, siehe Seite 5ff.
<code>\cmpd*{}{}</code>	unsichtbar, erzeugt Label, siehe Seite 5
<code>\cmpd-[]{}{}</code>	ohne Begrenzer, siehe Seite 18
<code>\cmpd+-[]{}{}</code>	Label wird aus Hilfsdatei gelesen, siehe Seite 11
<code>\cmpdinit{}{}</code>	Label initialisieren und erzeugen, siehe Seite 7f.
<code>\cmpdinit*{}{}</code>	Label initialisieren, siehe Seite 7f.
<code>\cmpdref[] []{}{}</code>	temporäre Label in eps -Dateien durch Label ersetzen, siehe Seite 12
<code>\cmpdref-[] []{}{}</code>	temporäre Label in eps -Dateien durch Label ohne Begrenzer ersetzen, siehe Seite 18
<code>\cmpdreset[]</code>	Zähler zurücksetzen, siehe Seite 10
<code>\cmpdsetup{}{}</code>	chemnum einstellen, siehe Seite 15ff.

11 Dank

Ich möchte Joseph WRIGHT und Russell HEWITT danken, die mir wertvolle Vorschläge und Feedback gaben, um chemnum zu verbessern.