关于新版 CCT 的说明

张林波
(zlb@lsec.cc.ac.cn)
中国科学院数学与系统科学研究院

2006 年 2 月 3 日

摘　要

本文档对新版 CCT 系统的功能、用法进行说明。我们假设阅读本文档的读者已经对老版 CCT 系统有一个基本的了解，否则请先阅读老版 CCT 手册：*ftp://ftp.cc.ac.cn/pub/cct/old-cct/cctman.ps.gz*。本文档的编译需要 CJK 字库。如果没安装 CJK 字库的话请将 “`\documentclass`” 中的 “`[CJK]`” 可选项去掉后再编译。

致谢：`cxterm@CTeX` 提供了书签处理程序 `gbk2uni`，`hooklee@CTeX` 对其进行了重写与完善，`hooklee@CTeX` 和 `nsii@CTeX` 帮助完成了 Windows 环境下的移植与调试，`mytex@CTeX` 提供了 NSIS 安装脚本及 `CCTfntef` 和 `ccmap` 宏包，特在此表示感谢。对其他所有对 CCT 提出过建议、参与过讨论的各方 TEX 高手们在此一并致谢。特别要感谢 `aloft@CTeX` 建立的 CTEX 论坛，使得大家有了一个共同讨论、开发的平台。

目　录

§1 构成与功能

新版 CCT 中主要增加了一个宏文件 CCT.sty，同时对老版本中的一些程序和宏文件做了少量修改，并增加了一些辅助程序和宏文件。新版 CCT 使用新的版本号系列，目前的版本是 0.6180-3 (老版 CCT 的最终版本号为 5.14)。

根据 ifuleyou 的建议从 0.6.0 版开始 CCT 对汉字标点符号进行了特殊处理，以便当它们出现在行首或行尾时能够与其它行对齐，同时，还对某些连续标点符号 (如“。)”) 间的空进行了调整。由于这项改动可能导致排版结果与以前的版本不一样，新版本中将前一个版本的 CCT.sty 更名为 CCT047.sty (相应的配置文件更名为 CCT047.cfg)，并在 cctbase.sty 中增加了一个可选项 “CCT047”，可用在 \documentclass 命令中，例如 “\documentclass[CJK,CCT047]{cctart}”，来调用前一个版本的 CCT.sty。

0.6.0 版的另一个改动是禁止在汉字和 TEX 字符间断行，除非它们之间原来就有弹性空间 (如空格等)。这是为了解决以前版本中存在的可能破坏行禁则的一个 bug (例如，在以前版本中可能会在全角右括号和半角逗号之间换行)。必要时可以在它们中间插入 \allowbreak 命令来解决由于无法断行而导致的 overfull 问题，例如，可将“标点，word”写成“标点，\allowbreak word”来允许在逗号和 “word” 之间换行。CCT.sty (≥0.618-1) 中修改了 \allowbreak 命令，使得当它用在汉字标点后面时，不但允许换行，还会消除行尾可能多出来的空。

此外，0.6.0 版中还增加了两个条件变量 \ifCCTpunct 和 \ifCCTbreak。\ifCCTpunct 用于控制是否对标点符号进行处理，用户可用命令 \CCTpunctfalse 关闭对标点符号的特殊处理，使所有全角标点与普通汉字占有一样的宽度，并且可以任意断行，该命令可以满足要求汉字严格对齐的排版 (例如将汉字排在格子中)，命令 \CCTpuncttrue 恢复对标点符号的处理。\ifCCTbreak 用于控制是否允许在汉字间断行，\CCTbreaktrue (默认) 允许在汉字间断行，而 \CCTbreakfalse 则禁止在汉字间断行 (用在如 verbatim 等环境中)。这些命令的作用域符合 TEX 的分组 (grouping) 规则。

老版 CCT 的 DOS 版是捆绑 emTEX 系统发行的。由于 emTEX 不支持高位为 1 的字符，因此新版 CCT 中所提供的宏文件不能直接在 emTEX 中使用，而需要先进行转换，参看 §4.3 最后一段中的说明。

新版 CCT 主要增加了下面一些功能:

(1) 排版源文件可以不经过预处理程序而直接用 TEX/LATEX 排版，解决了老版本在 `verbatim` 等环境中不能使用汉字的问题，也可以方便包含多个源文件的文档的处理。

(2) 新版本提供了使用 CJK 字库的可选项，用这种方式可以充分利用许多软件对 CJK 系统的支持，如使用 Type 1 字库生成矢量字库的中文 PS 文件，利用 `dvipdfmx`，`pdflatex` 等程序生成高质量的 PDF 文件，等等。几乎所有支持 CJK 系统的工具或宏包都可以以某种方式在 CCT 中用。

新版 CCT 支持两种排版流程。第一种排版流程是先用 CCT 的预处理程序 `cct` 对排版源文件进行预处理，然后再调用 TEX 进行排版，这种排版处理过程与老版 CCT 中完全一样，得到的排版结果也应该一样。第二种排版流程是不对排版源文件做预处理而直接用 TEX 进行排版，此时可以使用 CCT 的所有新功能，但排版结果与使用老版 CCT 系统相比会略有差别。

使用上述第二种排版流程时，CCT 支持生成两种 dvi 文件格式。第一种格式与老版 CCT 中使用的 dvi 文件格式一样，简称 CCT 格式，它需要用 `patchdvi` 程序进行转换后才能用其它处理 dvi 文件的程序 (如 `dvips`，`xdvi`，`yap` 等等) 处理。第二种格式与 CJK 系统的 dvi 文件兼容，简称 CJK 格式，它直接调用 CJK 系统的中文字库，这种格式的 dvi 文件不需要用 `patchdvi` 进行转换便可以直接用其它处理 dvi 文件的程序做进一步处理。至于如何选择 CCT 输出的 dvi 文件格式请参看 §4.1 节中的说明。当使用 CJK 格式时，还可以直接用 PDFTEX 或 PDFLATEX 生成 PDF 文件。

建议新的 CCT 排版源文件的命名遵循下述规则：如果希望用 CCT 的预处理程序 `cct` 处理，则将源文件的扩展名取为 "`.ctx`"，否则将源文件的扩展名取为 "`.tex`"。我们鼓励 CCT 的用户使用 "`.tex`" 做为排版文件的扩展名，逐渐淘汰老版 CCT。关于如何将老版 CCT 文件转换为新版 CCT 文件请参看 *ftp://ftp.cc.ac.cn/pub/cct/NewCCT-HOWTO.txt*。

我们为 CTEX 中文套装用户提供补丁用于更新 CTEX 中文套装中的 CCT 系统。补丁的文件名为 `cct-x.xx-x-CTeX-update.exe` 的形式，下载地址为 *ftp://ftp.cc.ac.cn/pub/cct/*，其中 "`x.xx-x`" 对应于 CCT 发布版的版本号。补丁中包含最新 CCT 的宏文件、WIN32 可执行文件、文档和源程序。

CTEX 中文套装的下载地址是 *ftp://ftp.ctex.org/pub/tex/systems/ctex*。

§2 CCT 源码的编译

通常情况下不推荐直接从 CCT 的源码编译、安装。Linux 用户可以使用 CCT 提供的 RPM 或 Debian 包进行安装、升级，而 Windows 用户可以先安装中文 TEX 套装，然后再下载运行 CCT 的 CTeX 升级程序进行升级。

新版 CCT 的源码可从 *ftp://ftp.cc.ac.cn/pub/cct/src* 中下载。展开源码包后，所有文件位于目录 `cct-dist` 中。

CCT 最新开发版本可以通过匿名 CVS 从 `:pserver:anonymous@lsec.cc.ac.cn:/cvsroot/cct` 获取。例如：

```
cvs -d :pserver:anonymous@lsec.cc.ac.cn:/cvsroot/cct login
(要求输入密码时直接敲回车即可)
cvs -d :pserver:anonymous@lsec.cc.ac.cn:/cvsroot/cct co .
```

或是用 Web 浏览器访问 *http://lsec.cc.ac.cn/cgi-bin/viewcvs.cgi/cct* 由于源码包中的部分内容没有在 CVS 中，因此建议首先下载 CCT 的源码包，然后从用 CVS 下载的目录替换源码包中的相应目录。

§2.1 Linux 下的编译、安装

Linux 下编译 CCT 源码除需要 gcc 编译器外，还需要 KPATHSEA 的头文件与库 (RedHat/FC 系统中它们包含在 tetex-fonts 包，而 Debian 系统中它们包含在 libkpathsea-dev 包)。

(1) 编译：

```
cd cct-dist
make all
```

如果使用 gcc4 则应该用下面的命令编译：

```
cd cct-dist
make EXTRA_CFLAGS=-Wno-pointer-sign all
```

(2) 安装：

```
make prefix=/usr install
```

或：

```
make prefix=/usr/local install
```

安装完毕后，如果你的系统上安装了 Fontforge 程序，则建议运行一遍 "`cctgentbl`" 命令，它会自动为所有已安装的 CJK GBK/GB 字体生成相应的 `.tbl` 文件。每次添加新的 CJK 字体后 (GBK 或 GB) 都应该运行一遍这个脚本 (重复运行不会有负作用)。

§2.2 Windows 版本的编译

Windows 版本的 CTeX 升级程序是在 Linux 下用下述命令产生的：

```
cd cct-dist
make install_win32
```

上述编译命令需要安装 crossmingw32、wine 以及在 wine 下安装的 Windows 程序 NSIS。如果只想编译生成部分可执行程序，则可以分别进入到相关的子目录中 (如 `cct`、`ctex` 等)，然后键入命令 "`make win32`" 即可。

CCT 的大部分程序在 Windows 下用 VC 应该也能编译，但没有试过。

§2.3 制作 RPM 包

只需运行命令 (以 `root` 身份)

```
sh cct-dist/Linux/makerpm.sh
```

便可生成 RPM 源码和 `i386` 包。生成的源码包在目录 `/usr/src/redhat/SRPMS/` 中，`i386` 包在目录 `/usr/src/redhat/RPMS/i386/` 中。在制作 RPM 包之前必须安装其它一些软件包，否则会报错，此时可按照错误提示安装所需的包，然后重新运行上述命令来制作 RPM 包。

§2.4 制作 Debian 包

只需运行命令

```
sh cct-dist/Linux/Debian/makedeb.sh cct
```

便可在目录 `cct-dist/Linux/Debian/cct` 中生成 Debian 包。

注意，开始制作 CCT Debian 包之前需要先安装 `libkpathsea-dev` 包。安装 CCT 包之前必须先安装 `fontforge`, `tetex-base` 和 `tetex-bin` 包，另外，建议安装 `cjk-latex` 和 `dvipdfm-cjk` 包 (制作 Debian 包的相关文件由 `weigela@CTeX` 提供)。

安装完 `cct` 包后，建议安装 `CJK-GBKfonts` 包 (用 `alien` 命令将 RPM 包转换成 Debian 包)，然后用命令 "`gbkfont-inst`" 来添加 GBK 字体。

Debian 下安装 CCT 的具体步骤参阅文件 *ftp://ftp.cc.ac.cn/pub/cct/README.Debian*。

§3 对老版 CCT 的改动部分

(1) 预处理程序 `cct.c`

主要改动是当中英文间已经有空格或 "~" 时不再加入空格。因为 CJK 系统的用户习惯在中英文间加 "~"，如 "`中文~\TeX~`..."，新的预处理程序可以避免在这种情况下插入多余的空格。其它一些小改动包括增加 '`-q`' 命令行可选项 (用于关闭多余的输出信息)，允许对输入/输出文件用 '`-`' 来表示标准输入或标准输出以便可以用在管道中，等。

(2) 初始化程序 `cctinit.c`

由于 `CCT.sty` 中重新定义了 '`~`' (相当于使用了 CJK 中的 `\CJKtilde` 命令)，`cctinit` 中相应修改了 `cchead.sty` 中的一些定义以避免破坏中文行禁则。升级或安装新版 CCT 系统时必须运行一次 `cctinit` 程序生成新的 `cchead.sty` 文件。

(3) 宏文件 `cctbase.sty`

自动包含 `CCT.sty`，并做了一些相关的修改。增加了环境 `CCTdot`，用于在汉字下面加点 (将来拟根据需要逐步引进其它类似环境，如用于在汉字下面加波浪线的 `CCTwave` 环境等)。另外，增加了一个 '`zihaoAny`' 可选项，它使用 `\zihaoAny` 命令匹配中英文大小，使得中英文的大小更为一致，缺点是中文的大小不符合中文字号标准。

(4) 文档类 `cctart.cls` 和 `cctbook.cls`

增加了 `CJK` 可选项，用于选择生成 CJK 格式的 dvi 文件。如果用了 `CJK` 可选项，则不能再用 CCT 的预处理程序进行处理 (这种情况下排版时会产生警告信息)。

其它改动请参看 *ftp://ftp.cc.ac.cn/pub/cct/Changelog*。CCT 宏包 (`cctbase.sty`) 和文档类的说明可参看 *ftp://ftp.cc.ac.cn/pub/cct/CCTLaTeX.pdf*。

§4 新增功能

§4.1 宏文件 `CCT.sty`

这是一个 TEX 宏包，用于替代老版 CCT 中的预处理程序，适用于 Plain TEX，LATEX，PDFTEX，PDFLATEX 等。它的主要作用是用 TEX 语言实现了 CCT 预处理程序的主要功能，从而使得 CCT 排

版文件可以不经预处理而直接用 TEX 排版。就排版结果而言，CCT.sty 与 CCT 预处理程序间的主要区别在于：CCT 的预处理程序会自动在中英文间加入空格，而 CCT.sty 则不会自动插入空格，并且 (默认情况下) 会吃掉汉字后面的空格，因此需要手工在中英文间加空格，并且在汉字后面需要使用“控制空格”(即 “\␣” 或 “~”) 以免空格被汉字吃掉。建议用户借鉴 CJK 排版文件的写法，即在中文－英文间加一个 ‘~’，而在英文－中文间加一个空格或 ‘~’ (可参看本文档中的写法)。另一个方法是利用 §4.2 中介绍的辅助程序 cctspace 自动在中英文间加空格。

老版 CCT 的预处理程序自动在中英文间插入空格。由于一些处理不尽合理，有时会在标点符号的前后插入多余的空格，导致较难看的排版结果。为了保持排版结果的一致性，我们不可能在这方面对老版 CCT 再做任何改进。新版 CCT 在空格及标点符号的处理上有所改善，排版质量应该优于老版 CCT。

CCT.sty 使用一个配置文件 CCT.cfg 来定义一些参数，包括各个字号的大小和 CCT 字体到 CJK 字体的映射。CCT.cfg 中定义的字号与字体只有在用户选择了输出 CJK 格式的 dvi 文件时 (即在 \documentclass 中使用了 [CJK] 可选项) 才起作用。当输出 CCT 格式的 dvi 文件时，字号与字体依然由文件 cct.dat 和 ccfonts.def 定义。

CCT.sty 中定义了一些用户命令，这些命令的作用与 CJK 中的对应命令类似，包括 \CCTspace，\CCTnospace，\CCTtilde，\standardtilde，\nbs 和 \CCTchar。其中命令 \CCTchar 允许用户直接输入汉字编码，如可用 ‘\CCTchar{176}{161}’ (或是 16 进制形式 ‘\CCTchar{"B0}{"A1}’) 来输入汉字“啊”，该命令通常用于输入一些不易输入的字符。

CCT.sty 中还提供了一条命令 \zihaoAny 用来将当前汉字的大小设成任意用户指定的尺寸，如 \zihaoAny{15pt} 将当前汉字的大小设成 15pt。\zihaoAny 命令参数中的长度单位可以省略，当省略长度单位时以 pt 为单位。

其余命令普通用户一般不用，因而不在此介绍。

CCT.sty 的使用方法如下：

(1) LATEX 2_ε 系统

如果使用新的中文文档类 cctart.cls，cctbook.cls (或宏包 cctbase.sty)，则 CCT.sty 已经被自动包含，因此可以直接使用相关的功能，并且可以用可选项 “[CJK]”来选择生成 CJK 格式的 dvi 文件 (没有该可选项时生成 CCT 格式的 dvi 文件)。如果使用其它文档类，则需在源文件中加入：

```
\usepackage{cctbase}
```

或：

```
\usepackage{CCT}
```

(“\usepackage{cctbase}” 或 “\usepackage{CCT}” 中可以使用可选项 [CJK] 来选择生成 CJK 格式的 dvi 文件)。

在 \documentclass{cctxxxx} 和 \usepackage{cctbase} 命令中还可使用可选项 zihaoAny，它允许 LATEX 使用中文非标准字号与英文进行大小匹配。

(2) 其它系统

需在文档开头加入：

```
\input CCT.sty
```

或：

```
\let\CCTCJKfonts=1
\input CCT.sty
```

(“\let\CCTCJKfonts=1” 选择生成 CJK 格式的 dvi 文件)。

CCT.sty 默认使用 GBK 编码。可以通过可选项 [enc=GB] 或命令 \let\CCTGBKencoding=0 来选择使用 GB 编码 (如果用 CJK 字库的话还必须配置好 GB 编码的字库)。

需要注意的是，如果用 PDFTEX 或 PDFLATEX 编译，则必须使用 CJK 字库 (即加上 [CJK] 可选项或 \let\CCTCJKfonts=1 命令)。

§4.2 辅助程序 cctspace

这是一个辅助程序，用于自动调整中英文间的空。其用法如下：

```
cctspace [可选项] [输入文件名] [输出文件名]
```

或：

```
cctspace <输入文件名 >输出文件名
```

cctspace 可以作为一个辅助工具，用于对老的 .ctx 文件进行转换以便不用预处理程序处理。也可以看作一个新的“预处理程序”，但它只在输出文件中加入一些空格和 “~”。

cctspace 默认在中文－英文间加 ‘~’，在英文－中文间加空格。如果使用了 “-t” 可选项，则在英文－中文间也加 “~” 并且删除源文件中原有的空格 (以免引进多余的空)。cctspace 运行时会读入一个名为 cctspace.cfg 的配置文件，其中列出不应该加空格或 “~” 的地方。Unix 版本的 CCT 中，cctspace 按照 TEX 搜索其它输入文件的规则搜索 cctspace.cfg 文件，因此后者可以放在当前目录或 TEX 的某个 input 路径中，而在 Windows 版本的 CCT 中，可以将 cctspace.cfg 放在当前目录或是 cctspace.exe 所在的子目录。

cctspace.cfg 文件中分五个表列出不应该加空格或 ‘~’ 的字符串或 TEX 命令：

LEFT
: 当其中列出的字符串位于汉字左边时不要插入空格或 ‘~’。

RIGHT
: 当其中列出的字符串位于汉字右边时不要插入空格或 ‘~’。

COMMON
: 当其中列出的字符串位于汉字左边或右边时都不要插入空格或 ‘~’ (相当于同时属于 LEFT 和 RIGHT 表)。

VERBATIM
: 不要在所列出的命令的参数中插入空格或 ‘~’。

TRANSPARENT
: 根据命令参数的内容来判断是否应该插空格 (即将这些命令看成是“透明的”)。

VERBATIM_ENV
: 不要在所列出的环境中插入空格 (verbatim 类环境)

VERB
: 所列出的命令将与 \verb 命令一样处理，即不要在由跟在它们后面的单个字符所界定的参数中插入空格。

除了 cctspace.cfg 文件中列出的不应该加空格或 “~” 的地方外，cctspace 在下列情况下也不插入空格或 “~”：

(1) 中英文或英中文间有两个以上换行的。

(2) 该插入 “~” 的地方已经有 “~”，或是该插入空格的地方已经有空格或 “~” (包括控制空格 “\␣”)。

(3) LaTeX 的 verbatim 和 verbatim* 环境，\verb 和 \verb* 命令，comment 环境中等。

(4) 括在一对花括号间、花括号前不是命令的内容 (当花括号前是命令时，花括号间的内容会被当做命令参数对待)。

cctspace 偶而会引入不应该有的空格。用户可以通过加入一些列在 cctspace.cfg 中的特殊命令来防止 cctspace 在某些地方加入空格。如 “啊ABC” 经处理后会变成 “啊~ABC”，如果不希望在 “ABC” 前面加入空格，则可写成 “啊\relax ABC” 的形式，因为 cctspace 不会在 “\relax” 的前后加空格或 “~”，而 “\relax” 命令本身不起任何作用，也可以使用一对花括号，如写成 “啊{ABC}” 形式。

cctspace 程序另外一个有趣的用法是当用户在中英文间手工插入空格时，用来检查是否有漏加空格的地方。方法是用 cctspace 程序处理排版文件，然后比较处理前后文件间的差异 (Unix 下可用 “diff” 命令比较，DOS/Windows 下可用 “fc” 或 “comp” 命令比较)。

注：cctspace 0.6.0.2 版 (随 CCT 0.6.0-4 版发布) 中增加了对老版本 CCT 造字功能的支持。它将输入文件中 ‘#[...]’ 形式的内容做为对用户自造字的引用进行适当转换。

§4.3 辅助程序 cctconv

GB2312 编码中汉字的两个字节都在 161–254 之间，用 TeX 处理这些汉字不会出问题。GBK 编码是 GB2312 的超集，在 GBK 编码中，第一个字节的取值范围为 129–254，第二个字节的取值范围为 64–254 (不能取 127)。GBK 汉字的第二个字节的取值范围中包含了 TeX 特殊字符 ‘\’, ‘{’, ‘}’, ‘^’, ‘_’, ‘~’ 和 ‘^^80’，前四个字符用 TeX 处理时会出问题，而后三个字符用在 PDF 书签文件中 (.out 文件) 时可能出问题，因此用 TeX 排版包含这样的汉字的文件时需要做一些特殊处理。辅助程序 cctconv 对汉字中的特殊字符进行转换，以便 TeX 系统能正确地处理它们。cctconv 程序或者将汉字中的字符 ‘\’, ‘{’, ‘}’, ‘^’, ‘_’, ‘~’, 和 ‘^^80’ 分别转换成 ‘0’, ‘1’, ‘2’, ‘3’, ‘4’, ‘5’ 和 ‘6’, CCT.sty 排版时再将它们转换回来 (这种形式只对 CCT 有效)，或者将这些汉字转换成 ‘^^7f...^^7f...^^7f’ 的形式 (该形式同时适用于 CCT 和 CJK)。具体转换成何种形式取决于 cctconv 编译时用的选项，目前发布版中的 cctconv 程序采用后一种形式，即 ‘^^7f ...’ 的形式，因此同时适用于 CCT 和 CJK 文件的处理。

cctconv 程序的用法如下：

```
cctconv [输入文件名] [输出文件名]
```

或：

```
cctconv <输入文件名 >输出文件名
```

注意，如果用户使用 CCT 的预处理程序，则不能用 cctconv 程序对排版文件做转换，否则包含这些字符的汉字会不对。

cctconv 程序有一个命令行可选项 “-f”，它强制将所有第 8 位为 1 的字符 (即 128–255 范围内的字符) 转换成 “^^xx” 的形式，其中 “xx” 为字符的 16 进制编码。如，组成汉字 “啊” 的两个字节的 1，进制编码为 b0 和 a1，因此 “cctconv -f” 会将 “啊” 转换， “^^b0^^a1”。有些老的 TeX 系统 (如 emTeX) 不允许输入文件中直接使用大于 127 的字符，此时需用 “cctconv -f” 对所有含汉字的文件进行转换 (包括用户排版文件、CCT.cfg、cctbase.sty、cctart.cls 等)，然后才能进行处理。

由于简体中文中很少用到 GB2312 编码之外的汉字，而且所有汉字中只有数百个含有 TEX 的特殊字符，因此 cctconv 程序只在极少数特殊情况下有必要。

§4.4 书签文件转换程序 gbk2uni

这个程序由 hooklee 在 cxterm 编制的程序基础上完善而成。由于 Acrobat Reader 程序只能正确处理 PDF 文件中使用 Unicode 编码的书签，而 CCT 或 (使用 GB/GBK 编码的) CJK 排版文件中通常使用 GBK 编码，因而生成的 PDF 文件可能会在书签中出现乱码[1]。

如果用 dvipdfmx 程序生成 PDF 文件，可以在排版文件中加入命令

```
\AtBeginDvi{\special{pdf:tounicode GBK-EUC-UCS2}}
```

来告诉 dvipdfmx 将书签中的汉字转换成 Unicode 编码从而在 AcroBat Reader 中得到正确的书签。而 pdflatex 程序则没有提供类似机制，此时可用 gbk2uni 程序对书签文件进行处理。使用 dvipdfmx 时也可以用 gbk2uni 程序处理书签文件 (这样就不用在排版文件中使用上面的命令)。

gbk2uni 程序的使用方法如下：

(1) hyperref + pdflatex:

```
pdflatex myfile.tex
pdflatex myfile.tex
gbk2uni myfile.out
pdflatex myfile.tex
```

(2) hyperref + dvipdfmx:

```
latex myfile.tex
latex myfile.tex
gbk2uni myfile.out
latex myfile.tex
dvipdfmx myfile
```

汉字中含有 TEX 特殊字符时经 cctconv 程序处理后也会在书签文件中产生不正确的汉字 (参看 §4.3)，gbk2uni 程序也负责对这些字符进行转换。

目前 CCT.sty 在处理用 \CCTchar 命令或 '^^7f' 定义的汉字时似乎与 hyperref 之间不兼容，如果使用了 dvipdfm 可选项或用 pdflatex 处理，在生成书签文件时可能会出错，另外包含有特殊字符的汉字经 cctconv 处理后也不能生成正确的 PDF 书签。由于这些情况在排版简体中文时极其少见，因此我们不打算立即着手去解决这些问题。用 CCT 排版时请尽量避免在 \chapter，\section 等的参数中使用 \CCTchar 命令。

§4.5 索引处理程序 cctmkind

CCT 提供了一个处理索引、术语文件的程序 'cctmkind'。它的功能与用法与标准 makeindex 程序类似，但增加了汉字排序的支持。用户可以用可选项 '-C' 来选择汉字排序的方式，目前可用 '-C pinyin' 选择按拼音首字母排序、用 '-C stroke' 选择按笔画排序、用 '-C mixed' 选择按拼音排序并且中西文混排。

[1]注：为正确处理含汉字的书签文件，应该在 hyperref 包中使用 CJKbookmarks 可选项

与 makeindex 程序类似，cctmkind 程序允许通过可选项 '-s' 指定一个 index style 文件 (.ist 文件)，用于定义索引的处理方式与排版格式。cctmkind 按查找 TEXMF 文件的标准方式搜索 .ist 文件，默认搜索顺序是当前目录、LOCALTEXMF 下的 makeindex 目录 (及子目录)、TEXMF 下的 makeindex 目录 (及子目录)。

我们随 cctmkind 提供了分别用于中文索引和术语排版的两个简单样例 'cct.ist' 和 'cctglo.ist' 供用户参考。

关于 cctmkind 的使用细节可参考 makeindex 的手册，因为这两个程序除了排序结果不同外，用法和功能基本上一样。

§4.6 通用前端程序 ctex

ctex 是适用于排版 CCT 和 CJK 文件的前端程序，它可一次完成排版文件的预处理、排版和后处理。ctex 程序能够递归地对所有 \input 或 \include 文件进行预处理。对于 CJK 排版文件而言，ctex 程序的主要作用是调用 cctspace (可选) 调整中英文间的，调用 cctconv 对汉字进行转换以确保用到 TEX 特殊字符的汉字 (参看 §4.3) 能在 CCT/CJK 系统中被正确处理，调用 gbk2uni 程序处理 PDF 书签文件以便能够通过 pdflatex 或 dvipdfmx 生成正确的中文书签，以及调用 cctmkind 或 makeindex 程序处理书签 (.idx) 和术语 (.glo) 文件。

ctex 的命令行参数形式为：

```
ctex [可选项] 排版文件名
```

ctex 将顺序搜索 "排版文件名.ctx"，"排版文件名.tex" 和 "排版文件名"。(注：在 MikTEX 中，如果 "排版文件名" 本身不含扩展，则 ctex 将只搜索 "排版文件名.ctx" 和 "排版文件名.tex")。

除非使用了 "-quiet" 可选项，ctex 运行时会将它所执行的命令 (或等价形式) 显示在 stderr。因此 Windows 下可用 "ctex 排版文件名 >nul"，Unix/Linux 下可用 "ctex 排版文件名 >/dev/null" 来屏蔽掉其它程序的输出信息以便观察 ctex 的处理过程.

ctex 会将所有它不认识的命令行可选项，以及 "&fmt" 形式的参数传递给 TEX 程序。因此，用户可以在 ctex 的命令行上使用 tex 或 latex 程序的可选项。需要注意的是，如果这些可选项中含有空格，则必须将它们括在 (单或双) 引号中，否则 ctex 会错把空格后面的部分当成排版文件名。例如，可以采用下面的形式传递带有空格的参数：

```
ctex "--translate-file cp8bit.tcx" mypaper.tex
```

(上例中如果将空格换成 '=' 就不必用引号了)。

用户还可以用 '%&' 的形式在排版源文件中给出 ctex 的可选项。如：

```
%& -no-cctspace
%& -latex-runs=1
\documentclass{cctart}
... ...
```

上述形式的可选项必须位于排版文件的最前面，每个可选项必须占一行，以 '%&' 开始，'%&' 前面不能有空格或其它字符，并且 (除 -no-cctspace 外) 只有主排版文件 (即在命令行上给出的文件) 中的可选项起作用。可选项行的前面可以有其它说明行，但这些行的第一个字母必须是 '%'。可选项的参数可以用单引号或双引号括起来。命令行上给出的可选项的优先级要高于排版源文件中给出的可选项。

当不希望 ctex 用 cctspace 程序处理某个文件时，可在该文件第一行写上 "%& -no-cctspace"。主排版文件中给出的 "%& -no-cctspace" 作用域是全局的，即它表示所有输入文件都不用 cctspace

处理，而其它输入文件开头的 “%& -no-cctspace” 则只对该文件起作用 (注：ctex 扫描源文件中的可选项时，对主排版文件会一直读到第一个非说明行，而对其它文件则只读第一行)。

ctex 根据命令行中给出的输入文件 (即主输入文件) 的扩展名决定是否调用 cct 对输入文件进行预处理。如果主输入文件的扩展名为 .ctx，则调用 cct 进行预处理，否则则调用 cctspace 和 cctconv 进行转换。因此，对于 CCT 排版文件，如果希望使用老版 CCT 的处理方式，则应该将源文件的扩展名取为 .ctx，而如果希望使用新版 CCT 的处理方式，则应该将源文件的扩展名取为 .tex。我们希望 CCT 用户在新文档中尽量使用新版 CCT 的处理方式，逐渐淘汰老版 CCT。

ctex 在原文件名前加上前缀 ‘ctextemp_’ 做为预处理或转换后的文件名。如果输入文件的扩展名为 .ctx，则将输出文件的扩展名改为 .tex。

当用 cct 进行预处理时，如果一个 \input 或 \include 文件的扩展名为 .tex，或者该文件不存在，则 ctex 自动寻找扩展名为 .ctx 的文件作为输入文件。而当用 cctspace 和 cctconv 进行转换时，只有当 \input 或 \include 文件的扩展名为 .tex 并且该文件不存在时，ctex 才会去找对应的 .ctx 文件。

递归搜索 \input 或 \include 文件时，ctex 扫描排版文件中的下述命令：

```
\include{文件名}，\input{文件名}，\input 文件名
\documentclass[可选项]{文件名}，
\usepackage[可选项]{文件名}，\RequirePackage[可选项]{文件名}
\LoadClass[可选项]{文件名}
```

并递归地对所涉及的文件进行处理。

由于 ctex 将所有预处理或转换后的结果文件放在当前目录，因此 \input 或 \include 文件不能同名，即使它们的源文件位于不同的子目录中。

此外，ctex 还对下述命令中的文件名进行转换 (在文件名前面加上 ctextemp_ 前缀)：

```
\includeonly，\ProvidesClass，\ProvidesPackage，\ProvidesFile
```

为提高处理效率，ctex 在处理每个输入文件前先比较该文件与对应的加了 ‘ctextemp_’ 前缀的文件的修改时间，如果后者比前者新则跳过该文件的处理。必要时可以用 ‘-force-pre’ 可选项强制 ctex 重新处理所有输入文件，或用 ‘ctex -clean’ 命令删除以 ‘ctextemp_’ 为前缀的所有文件。

排版 LaTeX 文档时 (用 latex，pdflatex 或 amslatex 等命令排版时)，ctex 会检查 .aux，.out，.idx 等文件的变化，以决定是否需要再运行一次 latex 以保证交叉引用的正确性。为避免死循环，命令行可选项 ‘-latex-runs’ 用来控制最大重复运行的次数 (默认值可参看 ctex 程序的帮助信息)。排版结束后，如果生成了新的 .aux 并且使用了 “-bibtex” 命令行可选项，则 ctex 会 (在第一次排版结束时) 运行 bibtex 程序以刷新 .bbl 文件。如果生成了新的 .out 文件 (PDF 书签文件)，则 ctex 会运行 gbk2uni 程序对其进行转换以保证在 PDF 文件中生成正确的书签。如果生成了新的 .idx 文件 (索引文件)，则 ctex 会运行 makeindex 程序生成 .ind 文件。如果生成了新的 .glo (glossary，术语) 文件，则 ctex 会运行 makeindex 程序生成 .gls 文件。

Windows 版本的 ctex 程序在排版完毕后如果 dvi 文件是 CCT 格式，则运行 patchdvi 程序，因此最终产生的 dvi 文件可以直接用其它程序 (如 yap 或 dvips) 处理。而 UNIX 版本的 ctex 程序则不运行 patchdvi 程序，所产生的 dvi 文件如果是 CCT 格式的话不能直接用其它程序处理，而必须用 CCT 提供的 xdvic 和 dvipsc 脚本处理 (参看 §6)。

默认情况下，ctex 只进行排版。如果用户指定了 -dvips 可选项，则 ctex 会在排版完毕后运行 dvips 生成 PS 文件。如果用户指定了 -dvipdfmx 可选项，则 ctex 会在排版完毕后运行 dvipdfmx 生成 PDF 文件。

ctex 中可以用 '-tex'、'-amstex'、'-pdftex'、'-latex'、'-amslatex' 和 '-pdflatex' 可选项来选择 TEX 的排版命令，默认时用 latex 命令排版。也可以通过改变 ctex 的可执行文件名来选择排版命令。例如，在 Windows 下如果将 ctex.exe 改名为 ccttex.exe，则默认调用 tex 排版，而如果将 ctex.exe 改名为 cctpdflatex.exe，则默认用 pdflatex 排版。UNIX 下可以通过给 ctex 程序创建不同的符号链接，如 ccttex、cctlatex、cctpdflatex 等等，来分别完成不同的工作。

为了便于控制索引文件的处理，ctex 提供了以下三个可选项：

-makeindex_prog，-makeindex_opts，-glossary_opts

第一个可选项指定处理索引和术语文件的程序名，默认值为 'cctmkind' (参看 §4.5)。后两个可选项给出传递给 Makeindex 程序的参数，其中 -makeindex_opts 用于处理 .idx 文件，默认值为空，而 -glossary_opts 则用于处理 .glo 文件，默认值为 '-s gglo.ist'。由于这些参数比较长，建议将它们用 '%&' 的形式放在主排版文件的开头，如：

```
%& -makeindex_prog=cctmkind
%& -makeindex_opts="-s cct.ist"
%& -glossary_opts="-s cctglo.ist"
```

ctex 还提供了一其它 xxxxx_opts 形式的命令行可选项，用来传递参数给 ctex 调用的程序，其中 'xxxxx' 代表程序名。这些可选项有：

-gbk2uni_opts，-bibtex_opts，-dvips_opts，-dvipdfmx_opts

如：

```
ctex -dvipdfmx -dvipdfmx_opts="-p b5" -gbk2uni_opts="-npe" myfile.tex
```

将在运行 dvipdfmx 时使用参数 '-p b5' (设置 B5 幅面)，在运行 gbk2uni 时使用参数 '-npe'。建议用户将这些参数以 '%& -xxxxx_opts="...."' 的形式放在主排版文档的开头。

其它命令行可选项请参看 ctex 自带的帮助信息。

§4.7 辅助工具的局限与使用时的注意事项

辅助程序 ctex 和 cctspace 对用户的排版源文件进行转换。经 ctex 转换后排版源文件中一些文件名会发生变化，如 '\include{chap1}' 可能会被变成 '\include{ctextemp_chap1}'。而经 cctspace 转换后的排版源文件会在汉字与英文间插入一些空格和 '~'，如 '中文\TeX' 会被转换成 '中文~\TeX'。由于 ctex 和 cctspace 不可能对复杂的 TEX 命令进行完全处理，不可避免地会偶尔改变一些不应该改变的排版内容。ctex 和 cctspace 在设计时考虑了一些常见情况，例如，ctex 和 cctspace 会避免修改抄录环境 (verbatim 环境，\verb 命令等) 中的内容，但它们只能识别标准的抄录环境或命令，对于其它形式，如通过 shortvrb 包引入的"短"形式或用户自定义的宏则可能对其中的内容进行转换，导致错误的排版结果。

因此，使用 cctspace 和 ctex 时，排版源文件中尽量不要用非标准的抄录环境或命令以及通过 shortvrb 包定义的"短"抄录形式，特别是当这些环境中包含中文时。也不要用自定义宏代替 \include，\includeonly 和 \input 等命令来插入文件。此外，还应尽量避免在文件名中使用汉字。只要做到这几条，基本上就可以避免经 ctex 或 cctspace 转换后产生不正确的排版结果。

§5 CCT.sty 的处理流程及配置文件 CCT.cfg

图 1 给出了 CCT.sty 的处理流程。

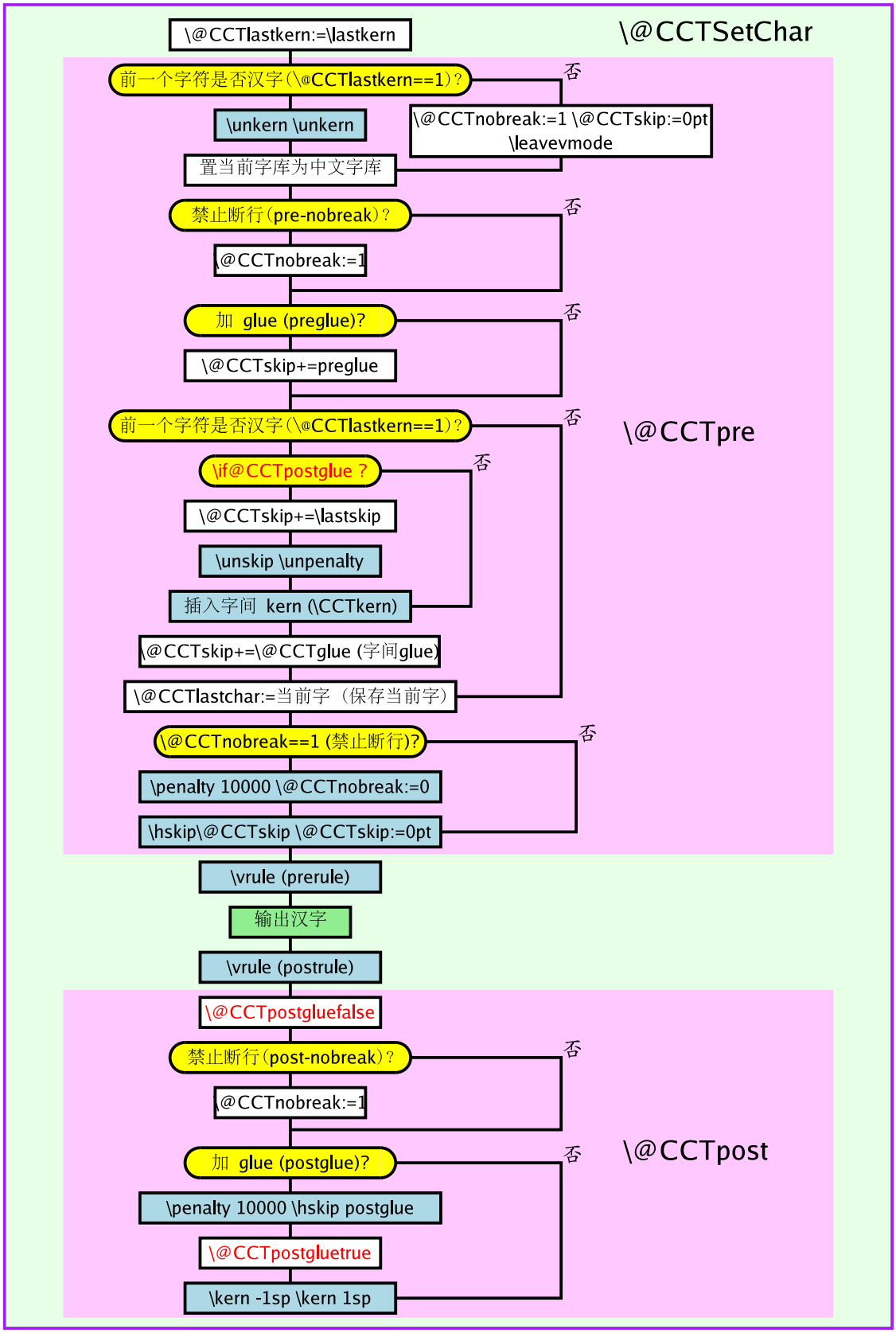


图 1 CCT.sty 排版流程图

新版 CCT 提供了四条命令用来定义 CCT 的字体、字号及标点符号处理。这些命令的参数之间用空格隔开，并且最后一个参数后面也必须至少有一个空格 (或换行)。

(1) `\CCTdefzihao` *CCT字号 磅数*

该命令将给定 CCT 字号的大小设成指定的磅数 (pt)，它只对 CJK 字库有效，如果使用 CCT 字库的话字号大小由文件 `cct.dat` 定义。新设定的参数在下一次执行 `\zihao` 命令后才起作用。

(2) `\CCTdefziti` *字母 字库名 替代字库名*

该命令定义 CCT 字体到 CJK 字库名的映射，它只有当使用 CJK 字库时才有效，如果使用 CCT 字库的话字体映射由文件 `hzfonts.def` 定义。第一个参数为 `\ziti` 命令中使用的字母。第二个参数为 CJK 字体名，如 "`song`"，"`hei`" 等。第三个参数为替换字库名，当由第二个参数指定的字库不存在时将用该参数指定的字库代替。新设定的参数在下一次执行 `\ziti` 命令后才起作用。

(3) `\GBKpunct` *全角字符 前断行 前*`rule` *前*`glue` *后断行 后*`rule` *后*`glue`

该命令用于调整一些特殊的全角字符 (标点符号) 前后的空及设定与之相关的行禁则。命令中各参数的含义如下：

前断行，后断行

这两个参数分别用于说明是否禁止在指定字符的前面 (pre-break) 或后面 (post-break) 断行，参数值取数字 0 或 1，0 表示允许断行，1 表示禁止断行。

前`rule`*，后*`rule`

CCT 可以在一个字符的前面 (pre-rule) 或后面 (post-rule) 附加一个高度为 0 的 `\vrule`，用于调整该字符前后的空，这两个参数分别给出附加在指定字符前面和后面的 `\vrule` 的宽度，以 em 为单位 (通常取负值来减少字符前后的空)。

前`glue`*，后*`glue`

这两个参数分别给出附加在指定字符的前面 (pre-glue) 和后面 (post-glue) 的弹性空间 (glue) 的大小。按照 TEX 的排版规则，这些弹性空间只在一行中间起作用，在换行时会自动消失。这两个参数均以 "`flag,space,stretch,shrink`" 的形式给出，其中 `flag` 取 0 或 1，当 `flag` 为 0 时表示忽略该 glue， `space`，`stretch` 和 `shrink` 给出 glue 的值，以 em 为单位。例如 "`1,0.3,0.2,0.2`" 中 `flag` 为 1，相应的 glue 值为 `0.3em plus 0.2em minus 0.2em`。

用 `\GBKpunct` 定义的全角字符经 `CCT.sty` 处理后变成以下形式:

`[penalty] [glue] [vrule]` 字符 `[vrule] [penalty] [glue]`

其中 penalty (罚分) 用于禁止断行。连续两个全角字符间的 penalty 和 glue 会被自动合并起来。

例如，假如对 "。" 的参数设定如下:

```
\GBKpunct 。  1  .0  0,0,0,0  0  -.510  1,.510,.0,.4
```

则 TEX 排版 "...啊。啊..." 时会产生下述结果:

```
...
啊
\penalty 10000
\hskip 0pt plus 0.29999pt minus 0.20004pt
。
\vrule width -5.35493pt height 0pt depth 0pt
\hskip 5.35493pt plus 0.29999pt minus 4.39998pt
啊
```

...

其中第一个 glue (\hskip) 是汉字的字间空，第二个 glue (\hskip) 是字间空与 “后glue” 值之和 (这里 “后glue” 值为 0.510em plus 0em minus 0.4em)，\penalty 10000 阻止 TEX 在句号前面断行。有兴趣的读者可以使用下面的排版命令:

```
\showboxbreadth=20 \showboxdepth=2 \setbox0\hbox{啊。啊} \showbox0
```

然后在 log 文件中观察 CCT.sty 的排版结果和 CCT.cfg 中的参数的作用。

参数 “前rule” 和 “后rule” 的设定应该依据字体的度量信息计算，使它们正好去掉标点符号前后多余的空白，而标点符号与前后字符间应该留出的空则由参数 “前glue” 和 “后glue” 设定，这些 glue 在换行时会消失，从而使得位于行首或行尾的标点符号正好顶格。

(4) \GBKkern 全角字符1 全角字符2 长度

(参数 “全角字符1” 和 “全角字符2” 间的空格可以省略。) 该命令用于调整两个连续全角字符间的空 (TEX 术语中称为 kerning)。参数 “长度” 给出附加在两个给定字符间的空白的宽度 (\kern)，以 em 为单位，可以取负值。该命令同时取消通过 \GBKpunct 命令附加在 “全角字符1” 后面和 “全角字符2” 前面的 glue，因此 “长度” 参数通常可取值为 0。

由于命令 \CCTdefzihao、\CCTdefziti 和 \GBKkern 定义的参数与具体使用的字库无关，因此 CCT 将它们分别放在与字库无关的配置文件 CCT.cfg 和 GBKkern.cfg 中。而命令 \GBKpunct 中的参数与标点符号的度量信息有关，它们是根据具体字库计算出来的，不同字库的参数可能会不一样。例如，在 Windows 的四种基本中文 TrueType 字体中，simsun 和 simhei 的参数是一样的，simkai 和 simfang 的参数是一样的，但这两组字体的参数却不同。为了便于对不同字体的标点符号进行不同的调整，CCT 将 \GBKpunct 命令单独放在一个与字体名挂钩的参数文件中，这些文件的文件名与字体文件名相同，扩展名为 .tbl，如 gbksong.tbl，gbkhei.tbl 等等。CCT 处理标点符号时会对不同字体自动使用相应参数文件中定义的参数以达到最佳效果。当一个 CJK 字体的参数文件 (.tbl 文件) 不存在时，CCT 使用文件 gbkdef.tbl 中的默认参数。由于所有 CCT 字库共用一个标点符号库 (clib256e.pps)，因此当使用 CCT 字库时标点符号参数从文件 clib256e.tbl 中获得 (该文件将随 CCT 系统发布)。

.tbl 文件中只能包含 \GBKpunct 命令及参数、说明 (%) 及空格、制表符 (<tab>) 和空行，除此之外不能有其它任何命令 (包括 \endinput 命令)，否则可能出现意想不到的结果。由于 .tbl 文件中的参数是根据字库的度量信息确定的，很难直接手工制作。为此，CCT 提供了一个 bash 脚本，它会自动从 Type 1 字库中读取度量信息来生成相应的 .tbl 文件。该脚本的文件名为 GBKpunct，包含在 Linux 版的 CCT 中，可以在 Linux 或 Cygwin 下运行。GBKpunct 运行时需要 FontForge 程序 (*http://fontforge.sourceforge.net*)，或它的前身 PfaEdit。Windows 下的 CTEX 用户只要将 Linux 中生成的 .tbl 文件拷贝到目录 \ctex\localtexmf\tex\latex\cct 下，再刷新一次 MikTEX 的文件名数据库即可 (注意 Windows 下使用的中文字库可能与 Linux 不同，CTeX 用户请使用随 CTeX 发布的 .tbl 文件，如果 .tbl 文件与所使用的字库不配套则会影响排版的效果)。GBKpunct 中有一组参数，放在文件结尾处，必要时可以修改。随 CCT 发布的 .tbl 文件是在 Linux 下用下述命令产生的：

```
% GBKpunct clib256e > clib256e.tbl
% cp clib256e.tbl gbkdef.tbl
% GBKpunct gbksong  > gbksong.tbl
% GBKpunct gbkhei   > gbkhei.tbl
% GBKpunct gbkkai   > gbkkai.tbl
% GBKpunct gbkfs    > gbkfs.tbl
```

```
% GBKpunct gbkbs    > gbkbs.tbl
```

用户如果需要改变字体映射或字号大小，可以直接修改配置文 CCT.cfg。如果希望改变标点符号的处理则需修改 GBKkern.cfg 文件。此外，也可以使用本节介绍的命令将要修改的参数放在自己的排版文件中。但我们不推荐直接手工修改 .tbl 文件，如确有必要，应该修改 GBKpunct 中的参数，然后重新生成 .tbl 文件。

§6 Linux 版本

我们为 Linux 用户提供一组新版 CCT 的 RPM 包，包括 src 和 i386 包，它们可以从网址:

ftp://ftp.cc.ac.cn/pub/cct/Linux

处下载。

CCT 的 RPM 包中包含一组命令 ccttex，cctlatex，cctamstex，cctpdftex 和 cctpdflatex，它们是 CCT 的前端程序 ctex 的符号链接，可分别代替 tex，latex，amstex，pdftex 和 pdflatex 命令用于 CCT 或 CJK 文档的排版，参看 §4.6。CCT 的 Linux 版本中还提供了三个脚本文件 xdvic，dvipsc，和 eps2pdf。脚本 xdvic 和 dvipsc 分别用于显示和打印 CCT 格式的 dvi 文件，它们在调用 xdvi 或 dvips 前先运行 patchdvi 程序对 dvi 文件进行转换，在退出时删除所生成的临时 dvi 文件和 PK 字库；脚本 eps2pdf 将 EPS 文件转换为 PDF 文件 (与 Ghostscript 的 ps2pdf 脚本的区别在于它保持图像原有的 BoundingBox 不变)。

我们建议在安装 CCT 系统的同时安装 CJK 系统，以便能够使用新 CCT 系统的所有功能。网址：

ftp://ftp.cc.ac.cn/pub/cct/CJK

处提供了一组 CJK 的 RPM 包，自行安装 CJK 系统有困难时可以试试它们 (如果已经安装了 CJK 系统及字库，则在安装这些包前应先备份好重要的配置文件)。这些包都是 src 包，可用命令：

```
rpmbuild --rebuild {src 包名}
```

编译生成相应的 i386 包。

CJK-GBKfonts 包提供了 gbkfonts 程序和一个 bash 脚本 gbkfont-inst，后者可用来方便地安装、卸载供 CCT 和 CJK 用的 GBK 字体，例如：

```
gbkfont-inst /usr/share/fonts/zh_CN/TrueType/gbsn00lp.ttf song
```

(最好以 root 身份运行) 将安装字体 gbksong，而：

```
gbkfont-inst remove song
```

则将卸载 gbksong。重复安装同一个字体不会有副作用。不带任何参数运行 gbkfont-inst 会显示一个简要使用说明。

另外，随 RedHat–9 发布的 tetex-xdvi 包不支持 Type 1 字库及彩色，建议下载安装 xdvik 的最新 beta 版，网址为：

http://xdvi.sourceforge.net/cvs-upgrade.html

或：

http://sourceforge.net/project/showfiles.php?group_id=23164

或是用：

ftp://ftp.cc.ac.cn/pub/cct/xdvik

中的 xdvik 包替换 tetex-xdvi 包，以便在预览时直接使用 Type 1 字库 (如果使用下面介绍的 Kile 与 Gvim 的集成环境的话则必须用该处提供的 RPM 包才能实现 CCT 文档的双向搜索功能)。使用 teTEX 1.0 的用户安装 xdvik 包后可能需要修改 /usr/share/texmf/xdvi/xdvi.cfg 文件，将倒数第二行上的说明符去掉，详见该文件中的说明。

网址:

ftp://ftp.cc.ac.cn/pub/cct/kile

处有经过修改的 Kile 版本，其中加入了对 CCT 及前端程序 ctex 的支持。有关细节可参看该目录中的 README-kile.txt 文件。

网址:

ftp://ftp.cc.ac.cn/pub/cct/ite

处提供了一个名为 ite 的 RPM 包 (意为 Integrated TeX Environment)，其中包含一些脚本及支持文件，它们与 GVIM 或 Kile 结合构成编排 LaTeX 文档的集成环境。命令 "rpm -qip ite-*-*.i386.rpm" 可显示一些相关说明。该目录下还有一套点阵 X11 字库 (gb20st.pcf.gz 和 gb20st-iso10646.pcf.gz)，可用于改善汉字显示效果 (安装时只需将两个文件拷贝到目录 /usr/X11R6/lib/X11/fonts/misc 下然后运行：

```
mkfontdir /usr/X11R6/lib/X11/fonts/misc
/etc/init.d/xfs reload
```

即可)。安装完 ite 包后，用命令 "vtex 主排版文件名" 即可进入基于 GVIM 的 TEX 集成环境。

ite 中有一个拼写错误检查按钮，用于提供简单的英文拼写检查功能，用户可以创建一个文件 $HOME/.ite/spell.words，在其中加入自定义的单词 (单词间用空格或换行隔开)。由于 ite 包依赖于 aspell 包，如果升级了 aspell 包的话最好重新安装一次 ite 包并删除文件 $HOME/.ite/spell.dic。

ite 包要求 Vim 6.0 以上的版本，因此 RedHat 7.2 及更早的 Linux 版本需要升级 Vim (RedHat updates 中有相应的升级包)。对于 RedHat 9 的用户，推荐从下述网址下载安装 Vim 6.2:

ftp://ftp.cc.ac.cn/pub/cct/vim-6.2,

该版本基于 GTK2，使用了 anti-aliase 技术，字符显示效果好得多，并且不需要前面所提到的中文 X11 字库 (gbst20*.pcf.gz)。它与官方发布的 Vim 6.2 的区别在于改正了几个对汉字支持的问题 (参看 *ftp://ftp.cc.ac.cn/pub/cct/vim-6.2/README.txt*)。

另外，如果希望使用双向搜索功能的话，则最好安装或升级到 teTEX 2 (如果使用 teTEX 1.x 的话需要在排版源文件中加入 "\usepackage{srcltx}" 才能使用搜索功能)。建议安装 ite 包之前安装或升级下述包：

```
vim-{common,enhanced,X11} >= 6.1
tetex >= 2.0.2 (不要装 tetex-xdvi 包)
cct >= 0.5.0-3
```

```
xdvik >= 22.74.2
aspell >= 0.33.7
```

该目录中还有一个 ite-latexSuite 包 (*http://vim-latex.sourceforge.net*)，其中提供了许多非常有用的用于编排 LaTeX 2ε 文档的功能。安装该包后，可在 vtex 环境下用命令 :help latex-suite (或在命令行上键入 vim /usr/share/ite/doc/latex-suite.txt) 来阅读它的用户手册。

附录 A

附录 A.1 如何在同一文档中同时使用 CCT 和 CJK 进行排版

这里介绍的方法仅供参考。它们不一定总有效，也许有些地方不对，欢迎大家指正。

在一个 CCT 文档 (\documentclass{cctxxx}) 的导言区加入下述命令：

```
\standardtilde \let\standardtilde=\relax
\usepackage{CJK}
```

便可在文档中使用 CJK 或 CJK* 环境。需要注意的是应该在命令 \end{CJK} 或 \end{CJK*} 之后加上 "\input CCT.sty" 才能再用 CCT 的汉字。该方法可用于在一个 CCT 文档中利用 CJK 排出一些 GBK 编码之外的文字。

如果需要在一个 CJK 文档中插入 CCT 的排版内容，可以在导言区加入 "\usepackage{cchead}"，然后用下面的命令进入 CCT 环境：

```
{
\input CCT.sty
CCT 排版内容 ...
}
```

注意将 CCT 排版内容局限在一个 group 中，以免影响 CJK 排版。CCT 排版内容可以插在 CJK 环境之内，也可以插在 CJK 环境之外。(将 "\usepackage{cchead}" 加在导言区是为了避免重复调入 cchead.sty 而无谓地占用 TeX 寄存器。CCT.sty 设计时已经做了考虑，可以重复载入而不会多占资源。)

附录 A.2 其它一些宏包

本节介绍的宏包还都是实验性的，它们尚有待进一步测试、完善。欢迎大家共同对它们进行改进。

附录 A.2.1 CCTty

这是一个支持天元中文 TeX 系统的宏包。使用时只需将天元 TeX 排版源文件中的

```
\input tyinput
```

改成

```
\input CCTty.sty
```

(LaTeX 2ε 中可以用 "\usepackage{CCTty}")，即可用 CCT 来排版。与 CCT 排版文件类似，可以在 "\input CCTty.sty" 前面加上 "\let\CCTCJKfonts=1" 或在 "\usepackage{CCTty}" 中使用 "[CJK]" 可选项来选择使用 CJK 字库。用 CCTty 宏包排版天元 TeX 文件时与天元系统的排版结果会略有不同，在排版命令的处理上也有一些差异，主要体现在以下几方面：

- 用 CCTty 处理天元排版命令 (如 "\宋"，"\大" 等) 时，命令的作用域严格遵循 TeX 的分组规则。

而用天元系统排版时它们仅受限于用 “{” 和 “}” 界定的组。

- 目前在 CCTty 宏包中 “\瘦”，“\扁” 和 “\阔” 命令是利用 PS 和 PDF 的缩放功能实现的，只有在将排版结果转成 PS 或 PDF 文件后才能看到最终排版效果。

此外，CCTty 宏包中只定义了 \宋、\楷、\仿、\黑 四条字体命令，以及 \半、\五、\六、\七、\八、\小、\九、\标、\中、\大、\特、\双、\巨、\叁、\肆 十五条字号命令。如果用到其它字体、字号命令的话可参照 CCTty.sty 中的格式自行定义。例如命令：

```
\CCTtycmd 题 {\biaosong}
```

将天元字体“\题”定义为 CCT 的 “\biaosong” (标宋)。可将新的字体、字号定义放在排版源文件的开头，但必须位于命令 “\input CCTty.sty” (或 “\usepackage{CCTty}”) 之后。

附录 A.2.2 hzcmd

宏包 hzcmd 引入了对以汉字命名的 TEX 命令的支持。包含汉字的命令名中可以有英文字母，但命令名的第一个字符必须是汉字。该宏包修改了 LATEX 2_ε 的下述命令:

\newcommand，\renewcommand，\DeclareRobustCommand，\ProvidesCommand

可以用它们定义以汉字命名的 TEX 命令。此外，还提供命令 \cdef，\cgdef，\cedef，\cxdef 和 \clet，用于在 PLAIN TEX 或 $\mathcal{AMS}$-TEX 中定义汉字命令。宏包 hzcmd 除了可以在 CCT 中使用外，也可在 CJK 排版文件中使用。

以汉字命名的命令目前有以下局限：

- 包含汉字的命令名第一个字符必须是汉字，不能以英文字母开头。
- 与普通由英文字母构成的命令不同，包含汉字的命令后面的空格不会被“吃掉”。必要时可以在定义汉字命令时在最后加上 \ignorespaces 命令来自动去掉跟随在命令后面的空格。
- 汉字命令不能用在其它命令的参数中 (有一个方法可以去掉该限制，但由于会显著降低处理速度而没有采用，参看宏文件中用 “%???” 注释掉的部分)。
- 汉字命令不能用在浮动内容中，如 \section 命令的参数中。
- 出现在命令名中的汉字的两个字节必须都在 129–254 范围，或者是英文字母，因此少数不常用的汉字不能用在命令名中。
- 由于宏包 hzcmd 修改了 \newcommand 等一些 LATEX 2_ε 命令，可能会影响到其它一些宏包 (如 hyperref) 的使用。一个解决办法是在调入这些宏包之后再调入 hzcmd。

除汉字命令外，宏包 hzcmd 还允许用户在环境名中使用汉字，如：

```
\usepackage{hzcmd}
... ...
\newenvironment{居中}{\begin{center}}{\end{center}}
\begin{居中}
  ... ...
\end{居中}
```

目前定义汉字命名的环境时有一个限制：只有 \newenvironment 或 \renewenvironment 命令的第一个参数 (即新环境名) 中允许含有汉字，其余参数 (即定义体) 中不允许含有以汉字命名的命令或环境 (但可以含有汉字)。

注意宏包 CCTty 和 hzcmd 不相容，它们不能在一起使用。

附录 A.2.3 everb

宏包 everb 扩展了 LaTeX 2_ε 的 verbatim 环境。它具有如下特点：

- 可以有选择地排印 (直角或圆角) 边框、背景色、行号等。
- 允许在源文件的一行中间开始或结束环境。
- 自动将 tab 展开成正确数目的空格。
- 通过逃逸字符 (escape char) 可以在环境中插入 LaTeX 2_ε 命令。
- 保证 CCT 和 CJK 中英文字符间的对齐 (一个汉字与两个英文字符同宽)。
- 结合逃逸字符可以在环境中用 \label 命令产生对行号的引用。该功能支持 hyperref 宏包。

宏包 everb 提供下述命令和环境：

\everbinput[可选参数]{文件名}

类似于 \verbatiminput 命令，区别在于可以通过可选参数控制排版样式，并且文件名中可以使用除 “{” 和 “}” 之外的任何 ASCII 字符 “\”。\everbinput* 命令将空格排成可见的。该命令会自动分页。

\begin{everbatim}[可选参数] ... \end{everbatim}

类似于 verbatim 环境，区别在于可以通过可选参数控制排版样式。everbatim* 环境将空格排成可见的。注意 everbatim 和 everbatim* 环境不能用在浮动体，如 figure 环境中，必要时可以在这些地方用 \everbinput 命令。该环境会自动分页。

\everb

用法类似于 \verb 命令 (\everb* 命令将空格排成可见的)。

\begin{colorboxed}[可选参数] ... \end{colorboxed}

可用于给一段排版内容加上边框和背景色，并且允许中间换页。例如：

```
\fboxrule=1pt \fboxsep=3pt \fboxradius=20pt
\begin{colorboxed}[oval=true,fgcolor=black,bgcolor={[gray]{0.9}},width=6cm]
    $$\int_0^14/(1+x^2)\,dx = \pi$$
\end{colorboxed}
```

的排版结果为：

$$\int_0^1 4/(1+x^2)\,dx = \pi$$

colorboxed 环境必须在纵向模式下使用，其排版内容总高度不能超过 16383.99998pt (约 576cm，这是 TeX 允许的最大长度)。需要注意的是，colorboxed 环境与其它一些涉及分页的环境，如 multicols，longtable，supertabular 等可能冲突，应该避免它们之间相互嵌套使用。

\colorovalbox[模式]{边框色}{背景色}{...}

用法与标准宏包 color 中的 \fcolorbox 命令类似，但可以排出任意半径和宽度的圆角框。当指定的圆角半径太大时，该命令自动将半径设成盒子的宽度或高度的一半 (该命令的 * 形式禁止对半径做调整)。该命令中不允许断页。

\newverbatim[可选参数]{环境名}

定义新抄录环境。可选参数给出该环境使用的参数。例如，命令:

```
\newverbatim[oval=true,number=true]{verbatim}
```

重新定义 LaTeX 2ε 的 verbatim 环境。

\everbsetup{参数列表}

设定默认参数。

除 \colorovalbox 外，上述命令和环境通过一组参数来控制排版式样。这些参数的公共默认值可以用命令 \everbsetup 设定，除 \everb 命令外，其它命令还可以在每次调用时以可选参数的方式指定不同的值。参数采用 "关键字=值" 的形式，不同参数用逗号隔开，详细格式可参看 keyval 宏包的说明。特别要注意的是当参数值中含有 "[" 或 "]" 时一定要用 "{...}" 将它们括起来，否则排版会出错，例如，必须将 "bgcolor=[gray]{0.8}" 写成 "bgcolor={[gray]{0.8}}" 的形式。everb 提供的参数及默认值如下 (一些命令会忽略某些参数)。

- width=<长度> — 设定排版宽度 (默认或 0 表示排版宽度取当前行宽)
- bg=true|false — 是否填充背景色 (默认值 true)
- bgcolor=<颜色> — 背景颜色 (默认值 white)
- fgcolor=<颜色> — 前景颜色 (默认值 black)
- box=true|false|[lrtb]* — 是否画框 (默认值 true)
- boxcolor=<颜色> — 框颜色 (默认值 black)
- oval=true|false|[abcd]* — 是否用圆角 (默认值 false)
- number=true|false — 是否要行号 (默认值 true)
- firstnumber=auto|last|<整数> — 首行行号 (last 表示继续上一次的行号，默认值 auto)
- nbrsep=<长度> — 行号到盒子的距离 (默认值 .5em)
- nbrcolor=<颜色> — 行号颜色 (默认值 black)
- tab=<整数> — 制表符宽度 (默认值 8)
- escape=<整数> — 逃逸字符 (ASCII 码，默认值 256，表示无逃逸字符)
- first=<整数> — 起始行 (默认值 1)
- last=<整数> — 终止行 (默认值 $2^{31}-1$)
- prologue=<token list> — 前导命令 (默认值 "{}")
- epilogue=<token list> — 后继命令 (默认值 "{}")
- pagebreak=true|false — 是否允许换页 (默认值 true)
- quoteshack=true|false — 是否将 "‚" 和 "‘" 换成 "'" 和 "`" (默认值 false)
- toptitle=<token list> — 指定盒子顶部标题
- bottomtitle=<token list> — 指定盒子底部标题
- title=<token list> — 同时指定盒子顶部和底部标题
- titlesep=<长度> — 设定标题两侧的空白大小 (默认值 .5em)
- titleskip=<长度> — 标题左右两侧保留的最小横线长度 (默认值 .5em)
- vscale=<浮点数> — 字符纵向缩放比例 (执行 graphicx 包的 \scalebox{}{vscale}{...}，默认值为空)

这些命令和环境与 LaTeX 2ε 的 \fbox 命令一样，通过 \fboxrule 和 \fboxsep 来指定框的线宽以及排版内容到边框的距离。everb 宏包定义了一个新的长度变量 \fboxradius 用来指定圆角半径 (当它的值小于或等于 100sp 时，everb 将其当做百分比处理，例如，\fboxradius=25sp 表示圆角半径取

为盒子宽度和高度中较小者的 25%)。everb 宏包利用一个特殊字库 “corners” 来排出任意大小和宽度的圆角框，该字库随 CCT ($\geq$ 0.618 版) 发行。

逃逸字符用来在抄录环境中插入 TEX 命令。界定在两个逃逸字符之间的内容会被当成普通 TEX 命令处理。一个比较常用的作法是利用逃逸字符在抄录环境中插入数学公式，或是插入 \label 命令用于引用抄录环境中的行号。

注意 prologue 和 epilogue 参数对 \everb 命令不起作用。用户可以通过宏 \everbprehook 和 \everbposthook 来将特定的内容插入 \everb 命令，其中 \everbprehook 被插在 \everb 的排版内容之前，\everbposthook 被插在 \everb 的排版内容之后。由于插入的内容被包含在一个组内，因此其作用域局仅限于 \everb 命令内部。做为应用的例子，这里给出用 \everb 取代 LATEX 标准命令 \verb 的两个方法：第一个方法利用 \everbprehook

```
\renewcommand\everbprehook{\everbsetup{box=false,bg=false,oval=false}%
  \fboxsep=0pt \fboxrule=0pt }
\let\verb\everb
```

第二个方法利用 \everbposthook

```
\renewcommand\verb{\bgroup
  \everbsetup{box=false,bg=false,oval=false}\fboxsep=0pt \fboxrule=0pt %
  \renewcommand\everbposthook{\egroup}\everb}
```

上述第二个方法的好处是允许 \verb 和 \everb 采用不同的格式排版。

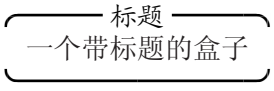
参数 title、toptitle 和 bottomtitle 用于定义盒子的标题，参数值是标题的内容。参数值前面可以用两个括在方括号中的字母来控制标题的位置，第一个字母给出水平方向的位置，可以是 “r” (靠右)、“c” (居中) 和 “l” (靠左)，第二个字母给出垂直方向的位置 (与框线对齐的部位)，可以是 “t” (顶部)、“c” (中部) 和 “b” (底部)，当省略位置参数时默认取 “[cc]”。

此外，参数 box 的取值除了可以是 “true” 和 “false” 外，还可以是一个由字符 “l” (left，左)，“r” (right，右)，“t” (top，上) 和 “b” (bottom，下) 构成的字符串，用于控制具体画出盒子的哪条边。类似地，参数 oval 的值可以是一个由字符 “a” (左上角)，“b” (右上角)，“c” (左下角) 和 “d” (右下角) 构成的字符串，用于控制盒子的哪个角画成圆角。注意，“box=false,” 和 “box=,” (空字符串) 的含义是不同的，前者取消盒子四周、包括角上圆角部分的线条，相当于将 fboxrule 设为 0pt，而后者仅取消盒子四周的直线。

例如命令

```
\fboxrule=1pt \fboxsep=5pt \fboxradius=5pt
\everbsetup{ oval=true, bg=true, box=tb, toptitle={[cc]{\textit{标题}}},
    titlesep=0.25em, titleskip=1em }
\colorovalbox{black}{white}{一个带标题的盒子}
```

产生的输出是：



附录 A.2.4 CCTfntef

这是由南开大学孙文昌老师设计的一个宏包，用来给汉字加上各种特效 (分散对齐、加点、加下划线、加波浪线等)，其使用可参看它的示例文件 CCTfntef-sample.tex。

附录 A.2.5 ccmap

这是由南开大学孙文昌老师在 cmap 基础上修改的一个宏包，调用它后通过 PDFLaTeX 编译可以生成支持查找/复制/粘贴的 PDF 文件。

索 引