TEX 与汉字处理：过去，现在，将来

马起园

2015 年 5 月

pTEX-ng 是什么

pTEX-ng 的设计目标是成为是下一代汉字处理的标准 pTEX。在底层引擎上支持：汉字处理、禁则处理、间距处理、直排、OpenType 处理，多语言处理、UTF-8 编码支持。对于排版输出的基本规则，pTEX-ng 遵循日本 JIS 标准 X 4051 及中国大陆、中国台湾的行业惯用法。

pTEX 是日本 ASCII 公司开发的汉字处理引擎。在 2008 年和 2010 年经历合并及收购之后，pTEX 的开发已经停止，后续维护由 TEX Users Group 开发团队进行，但最近几年并无新功能加入。

pTEX-ng 的开发最早可以追溯到 2012 年春节假期，最早在 LuaTEX 上进行试验性质的修改。在 2013 年一度中断开发，转向读代码和读文档，2014 年年初最先使用 Common TEX 进行扩展，后转向 Y&Y TEX 进行二次开发。pTEX-ng 第一版在 2014 年 10 月发布，使用 GPL 第二版许可。

pTEX-ng 的开发概要

pTEX-ng 的前身是 Y&Y 公司的一个用 C 语言实现的带内存管理的 TEX。早期源码是使用 `web2c` 进行转换得到，不具可读性和可扩展性。pTEX-ng 的源码建立于 2014 年重写代码的基础之上。

pTEX-ng 的的汉字处理相关代码来源于 pTEX；Unicode 编码处理代码来源于 upTEX。对于 ε-TEX 的支持来源于 eupTEX。对于 OpenType 的处理，来源于 `libm17n`。

pTEX-ng 只支持 PDF 文件输出，相关代码来自 `dvipdfmx`，没有使用任何来自 pdfTEX 的代码，所以大量 PDF 输出相关的 primitive 并不能在 pTEX-ng 中使用。

TEX 扩展：面向汉字处理

- Fujita TEX，由 Hiroshi Fujita 开发，基于 TEX78（SAIL 语言）。由于现在没有更多的材料，所以这个扩展已经无从考证了。
- NTT JTEX，由斋藤康己（Yasuki Saito）开发，起于 1988 年，在 1994 年得到 NTT 的 ASTEM Software Culture Award。Unix 版本由千叶大学的樱井贵文（Takafumi Sakurai）移植。该扩展在 Debian 以及 W32TEX 中均可安装，但未进入 TEX Live。JTEX 的开发思路，是在字体处理上做了扩展，这样在处理起来需要的补丁量不大。JTEX 支持类似 magin kerning 的 burasage。
- ASCII pTEX，ASCII 公司的大野俊治（Shunji Ohno）和仓泽良一（Ryoichi Kurasawa）等开发，起于 1987 年（昭和 62 年）。pTEX 中的 p 代表的是 publishing 的意思。pTEX 的设计相对于 JTEX 来说，逻辑更为复杂，比如支持直行排版。
- PUTEX，台湾静宜大学资管系蔡奇伟开发，起于 1997 年，止于 2004 年。最后版本为 PUTEX4。PUTEX 提供了很多 primitive 来实现了 Virtual Font 功能，需要使用 `cidpdfmx` 生成 PDF。

扩展的难点

- 字符集编码形式
- 字体操作

扫描器的处理

TEX82 的设定中，一般来说，行尾的 carriage return 做一个空格插入文本中处理。但在汉字排版实践中，会造成多余的空隙，所以 JTEX 和 pTEX 都做了相应的额外处理。PUTEX 以及 XETEX 都没有对这些行尾进行操作。而在 LuaTEX 中可以使用 Lua 的 callbacks 进行干预。

- JTEX 引入了`\jendlinetype` 进行处理，该命令为整数赋值型，可选值为：*jend_ascii* (0)，*jend_comment_bit* (1)，*jend_ignore_bit* (2)。
- pTEX 则没有如 JTEX 这种处理方式，“汉字+CR”中的 CR 不会处理为空格，但“ASCII+CR”中的 CR 会被处理为空格。但，pTEX 将 TEX82 的状态处理扩展为：*mid_line*，*mid_kanji*，*skip_blanks*，*new_line*。这样使 pTEX 多了数十个状态来控制汉字与非汉字的处理。

字体的处理

TEX82 的设定中，字体，亦即 font 指的实际上是 metrics，也可以说是通常认知上的 font 的抽象版本。这其实和 TEX 的算法相关，即只需要宽度高度深度可以进行断行算法等的处理。

- JTEX 中使用 subfont 技术，也就是将一个外部的大字体分割为多个小字体。这样做的好处是每个码位都会有非常详细的信息。
- pTEX 中使用的是 char class 技术，是用了扩展的 TFM 格式，即 JFM 格式。JFM 技术的优点是可以将外部字体的所有信息归类分为几个大的 class。但，JFM 格式实际上可以覆盖整个 Unicode 区域，在后期将 JFM 映射到外部字体的时候，需要比较复杂的 VF 配置。
- PUTEX 中使用的是 fake font 技术，实现上最为简单，但其通过一些额外的 primitive 进行了对于一部分外部 Virtual Font 的处理。PUTEX 会将字体信息最终存储到扩展了的 DVI 文件 CDI 文件中。实际上这部分处理和 XETEX 的比较相似。

TEX 中的 format 或者数据序列化

在 TEX 中，为了快速加载宏以及字体信息，通常需要读取 format 文件，扩展名为.fmt。比如我们常见的 latex.fmt 文件就就是含有整套的 LATEX 宏和相关字体的。TEX 中将内部数据结构存到文件的过程叫做 dump，用现在的话来说，叫做 serialization。

在现代操作系统以及现代的 TEX 扩展当中，format 文件已经有了不少的变化。比如：

- 压缩，主要通过 zlib 进行压缩，其主要的目的是减少 C 底层的 IO 操作
- sparse 化，涉及到内部的一些 sparse array 处理，主要是为了减少文件大小
- endian 控制，现在的计算机中的 endian 基本上有 Big-Endian 和 Little-Endian 两种，这涉及到生成的 format 文件的跨平台行为

pTEX-ng 的 catcode

catcode 是 TEX 及其扩展的一个核心概念，即应用于一套编码中所有码点（code point）的一个属性值。亦即，可构造出一个带有附加属性的编码表。TEX82、XETEX 和 LuaTEX 的 catcode 有 16 种，而 pTEX-ng 则有 20 种，多出的 4 种主要用来处理汉字。在具体使用中，catcode 通常用来动态改变码点的语义，最终会影响到 TEX 内部的状态机处理。

0	escape character	10	space
1	beginning of group	11	letter
2	end of group	12	other character
3	math shift	13	active characative
4	alignment tab	14	comment character
5	end of line	15	invalid character
6	parameter	16	kanji
7	superscript	17	hiragana, katakana, alphabet
8	subscript	18	cjk symbol codes
9	ignored charcacter	19	hangul codes

pTEX-ng 的 node/noad

node/noad 是 TEX 在读取文本后，根据文本内容生成的链表元素。TEX 核心的功能就是生成链表，并根据链表最后输出 DVI 或者 PDF。pTEX-ng 中新添加了两种新 node，一个是 dir，用来控制输出内容链表的方向；另一个是 disp，用来控制汉字字串基线的偏移。

0	hlist	10	whatsit	20	bin	30	left
1	vlist	11	math	21	rel	31	right
2	dir	12	glue	22	open	32	—
3	rule	13	kern	23	close	33	—
4	ins	14	penalty	24	punct	34	—
5	disp	15	unset	25	inner	35	—
6	mark	16	style	26	radical	36	—
7	adjust	17	choice	27	under	37	—
8	ligature	18	ord	28	over	38	—
9	disc	19	op	29	accent	39	—

pTEX-ng 的 Font Layout Engine

pTEX-ng 使用的 Font Layout Engine 是日本产业技术综合研究所的半田剑一、锦见美贵子、高桥直人和户村哲等合作开发的 `m17n` 库提供。`m17n` 即 multilingualization，其开发思想最早可以追溯到 Emacs 上的 mule 多语言处理模块。

`m17n` 的字体呈现主要是通过 `libotf` 及 `m17n-flt` 实现的。和其它的字体处理引擎（ICU、HarfBuzz、Graphite2）相比，`m17n-flt` 最便利的一点是可以通过类 lisp 的 FLT 脚本（即 **F**ont **L**ayout **T**able）进行配置。此外还有数个预先定义好的 fontset 可供使用，用来处理单个字体无法覆盖整个 Unicode 全区域的情况，即相当于某些操作系统下的 font fallbacks。

`libm17n` 所提供的字体相关功能包括：

- 管　理（font cache 中查找，通过 fontconfig）
- 呈　现（rendering，即渲染成二维字串，字即 glyph）
- 字体集（fontset，源于 Emacs 及 X11）

TEX 中的 font 描述形式

无论是在 XeTeX 还是在 LuaTeX 中都可以使用 OpenType 字体。但在语法使用上需要使用下列形式（X 为 XeTeX，L 为 LuaTeX）：

- `X \font\Hoefler = "Hoefler Text/BI"`
- `L \font\pagella = {name:TeX Gyre Pagella} at 9pt`
- `L \font\LMR      = LatinModernRoman:clig=true;kern=false`

实际上，无论是 XeTeX 还是 LuaTeX 的`\font` 语法都比较复杂，其本质是 OpenType 的使用较为复杂。OpenType 的使用涉及到三大方面：script，language，features。而另一方面，OpenType 的调用是另一个复杂的问题。按照 TEX 最初的设计思路，我们可以首先使用文件名来调用，其次，我们可以使用字体内部的名字进行调用（见 Uniscribe、GDI 和 fontconfig）。

X11 下的 font 描述形式

在 m17n 中，font 的使用方式相对固定，比如有预定义的字体操作，例如 FLT。通常而言，只需要在 m17n 的代码中使用字体的唯一标识名称即可。在描述上，m17n 支持 XLFD（**X** **L**ogical **F**ont **D**escription）。其形式如下：

- `-microsoft-sylfaen-normal-r-normal-*-unicode-bmp`
- `-adobe-source code pro-light-r-normal-*-unicode-bmp`
- `-monotype-shonar bangla-bold-r-normal-*-unicode-bmp`
- `-itc-pristina-normal-r-normal-*-unicode-bmp`

对于 TEX 来说，使用这种格式可能会使`\font` 的命令变得冗长。但对 m17n 来说，这种格式便于解析和便于抛出错误。且由于格式是唯一的，对于使用 dvipdfmx 生成 PDF 之时的字体映射来说，更为方便。

m17n 中对于 OpenType 的配置

m17n 支持的 OpenType 配置有下面的语法（另有兼容语法，此处不讲）：

- `:otf=guru=blwf`, `:otf=deva=half+`
- `:otf=deva=abvs,blws,psts,haln+abvm,blwm,dist`

即为（类 BNF 语法）：

- OTF-spec ::= `':otf='` *script* *langsys* `?` *GSUB-features* `?` *GPOS-features* `?`
- script ::= *symbol*
- langsys ::= `'/'` *symbol*
- GSUB-features ::= `'='` *feature-list* `?`
- GPOS-features ::= `'+'` *feature-list* `?`
- feature-list ::= `(` *symbol* `','` `)` `*` `[` *symbol* `|` `'*'` `]`

pTEX-ng 的 primitive

- \jfont 和\tfont，这两个和\font 语义一致。pTEX 中的 TFM 有两种，一种是 TEX82 定义了的 TFM 格式；另一种是 pTEX 定义的专门用于重载字符宽度的 JFM（后文会专门介绍 JFM）。在 JFM 中，已经定义了对应的文字方向，所以无论使用上述三种那个都不会产生问题。另外，还支持级数和齿数，即 Q 和 H，在大小上为 0.25cm，在传统意义上只有前者可用于定义字体大小，后者用来指配其他非字体元素的尺寸。
- ```
  \font\tenmin=upjisr-h % mincho(KANJI)
  \font\sevenmin=upjisr-h at 7pt
  \font\fivemin=upjisr-h at 5pt
  ```
  ```

pTEX-ng 的 primitive

- `\kanjiskip` 和`\xkanjiskip`，前者会被自动插入到汉字之间，后者会自动插入到非汉字与汉字之间。在 TEX82 上，可以使用 `ex` 和 `em`。而在 pTEX 中，可以使用 `zw` 和 `zh`，z 表示 *zenkaku*（全角，ぜんかく），前者为全角宽度，后者为全角高度。
- ```
  \kanjiskip=0pt plus .4pt minus .4pt
  %\xkanjiskip=2.5pt plus 1pt minus 1pt
  \xkanjiskip=.25zw plus 1pt minus 1pt
  ```
  ```

pTEX-ng 的 primitive

- `\tbaselineshift` 和`\ybaselineshift`，控制基线的偏移值。前者为直行文本的偏移量，请注意直行的汉字的基线位垂直方向的中心；后者为横行文本的基线偏移量。
- `\tbaselineshift=3pt`
  `\ybaselineshift=3pt`
- 林 hayasi 林, 林 hayasi 林

pTEX-ng 的 primitive

- \pdfstrcmp，比较字符串，LATEX3 必须
- \quitvmode，来自 PDFTEX
- \ngbanner，输出 banner: This is Asian pTeX, Version 3.14159265
- \ngostype，输出当前系统名称，当前为：Win64
- \tracingfontloaders，显示字体载入信息
- \pdfcompresslevel，设定 PDF 文件压缩比率（0—9，当前为 9）
- \pdfminorversion，设定 PDF 文件版本号（0—7，当前为 5）
- \synctex，SyncTEX 支持（当前为 1）