Type1 и truetype шрифты в LaTeX.

mkondrin из hppi.troitsk.ru

7 апреля 2008 г.

Аннотация

В тексте рассматривается вопрос об использовании произвольных type1 и truetype шрифтов в pdf файлах, полученных из latex исходников с помощью системы tetex. При этом предполагается, что pdf-файлы используются для электронных презентаций, т.е. рассматриваемый метод позволяет вставлять шрифты именно как векторные (outlined), а не растеризованные (bitmapped) шрифты (качество последних при просмотре на устройствах с низкими dpi - дисплеях, например, - как правило, хуже, в связи с чем и возникают частые жалобы в форумах - "Написал курсовую в LaTeX, сгенерировал pdf, а все шрифты какие-то корявые").

Довольно большой набор русских type1 шрифтов распространяется в пакете PSCyr [1]. Установка этого пакета подробно (и по русски!) описана в файле README [1]. Однако ничто не мешает вам использовать любые понравившиеся вам шрифты в latex. Единственные проблемы - это 1) сделать эти шрифты доступными для latexa, и 2) отучить latex растеризовать их.

Как это сделать и будет описано в данном тексте.

1 Что нужно?

tetex версии >= 1.0.7, ghostscript версии >=6.5 и несколько фонтов. В качестве примера предлагаю использовать свободные фонты с сайта Vedi [2] - Quake (type1 версия) и Stonehenge, Nadejda, Coptic (ttf версии). Также нужен будет один файл из пакета pscyr t2a.enc (файл с вектором кодировки T2AAdobeEncoding), но вы можете просто установить у себя этот пакет целиком, тем более, что в нем содержатся очень симпатичные русские шрифты (мне лично очень понравились Лазурский и Академия - шрифт используемый в старых изданиях 50х годов прошлого века). Тем кто хочет поглубже разобраться в структурах ps, pdf файлов, а также в том, как использовать фонты в этих файлах могу порекомендовать толстые английские мануалы с сайта Adobe [3, 4, 5, 6].

2 В чем проблема?

"Корявые"фонты - это type3 фонты, т.е. фонты реализованные с помощью графических команд postscript. С одной стороны это делает фонт более гибким (можно нарисовать любой символ), но с другой - это все же графика, и хотя pdf умеет сглаживать графику, но результат оказывается хуже чем у outlined type1 и truetype шрифтов. Однако просто взять и вставить outlined шрифт не получится (например отредактировав ps-файл) - дело в том что эти шрифты должны быть правильно "реализованы"в postscript файле, иначе ps2pdf конвертор все равно будет растеризовать их и в результате для некоторых символов (как правило русских букв) будет использован type3 шрифт.

Что значит правильно реализовать шрифт? Дело в том, что хотя в самом файле шрифта может содержаться любое количество глифов, но одновременно адресовать и отобразить с помощью постскрипт команды show в постскрипт файле можно только 256 символов. Если вы заглянете внутрь файла с type1 шрифтом (лучше брать файлы с расширением pfa), то

[1] Описание, мягко говоря, не соответствует текущим реалиям. Подробности в тексте.

вы найдёте в начале список глифов реализованных по-умолчанию в этом фонте (строчка Encoding). Список может быть как стандартным, например, AdobeStandardEncoding, либо явно определяться как encoding-array, например, как в шрифте Quake:

```
/FontName /QuakeCyr def
/Encoding 256 array 0 1 255  \{ 1 index exch /.notdef put  \}  for
dup 32 /space put
dup 33 /exclam put
dup 34 /quotedbl put
```

и т.д. - ровно 256 символов. Если же просмотреть его с помощью ghostscript-команд (как это сделать, описано в "Почему Мозилла печатает пустые квадраты вместо русских букв") можно убедится, что он дополнительно содержит еще 3 не закодированных символа. В postscript-language любой символ имеет уникальное имя - /space - пробел и т.д., в частности для русских букв зарезервированы имена /afii10017–afii10049;/afii10065–/afii10097 (можно убедится, что в шрифте Quake они уже закодированы). Вывод текста в постскрипт файле осуществляется с помощью команды show, которая принимает в качестве аргумента последовательность байт - индексов букв в текущем фонте (т.е. в фонте предварительно реализованном в постскрипт-файле с помощью команды selectfont). Чтобы отобразить символы, которые не содержатся в предопределённом encoding-array фонта нужно в постскрипт файле "динамически"создать фонт (команда definefont), в котором бы содержались нужные символы. Причем сделать это можно двумя путями:

- Заменив кодировку type1 фонта. Это я бы назвал "правильно"реализованным фонтом. Замена кодировки заключается в том, что задаётся последовательность имён символов (которые уникальны и стандартны), которые будут включены в логический динамический фонт. Последовательность в которой имена указывается задаёт индексирование шрифта. В принципе из одного физического шрифта внутри одного постскрипт файл можно создать множество логических шрифтов, но в каждый момент активным (selected) будет только один. Когда шрифт подгружается в постскрипт файл, то автоматически создаётся логический фонт с кодировкой указанной в самом фонте.

- Создать фонт с "нуля". Нужные символы определяются в шрифте либо как битмапы, полученные растеризацией глифов исходного фонта, либо символы исходного фонта вставляются с помощью команд <имя символа> showglyph. В обеих этих случаях вы имеете type3 фонт.

Собственно, чтобы из latex файла получать красивые pdf - нужно научить dvips обрабатывать "нестандартные"символы согласно варианту 1, а не 2, что он склонен делать, так сказать, по умолчанию.

3 Фонты в LaTeX

Вообще говоря, LaTeX безразлично какие именно шрифты вы собираетесь использовать в своём документе (LaTeX не занимается отображением документа). Единственное что ему нужно знать о шрифте - это где-какие символы расположены и их метрику, чтобы правильно распределять текст на странице. Эту информацию latex извлекает из tfm (tex font metric) файла и его наличие - непременное условие, чтобы любой фонт можно было бы использовать в latex документах. Т.е. первый шаг к подключению какого-либо шрифта к latex - это преобразовании метрики шрифта. У type1 шрифтов информация о начертании букв и их размерах содержится в двух различных файлах - afm (может также иметь расширение pfm) и pfa (или pfb) соответственно, для ttf данные о глифах и метрика содержатся в одном и том-же файле. Если же вы хотите классифицировать шрифты по стилям, чтобы LaTeX автоматически подбирал шрифты для bold, italic, serifed, sans-serifed и т.д. текста, то для

этой цели вам понадобится еще fd-файл (font descriptor). Однако этот файл не является обязательным если вы собираетесь использовать фонта в своём документе "разово например для каких-нибудь декоративных надписей.

Прежде чем вставлять шрифты подготовьте временную директорию (`~/latex-fonts` к примеру), где будут собраны шрифты, копируйте в нее quake.pfa, quake.afm, coptic.ttf, stonhen.ttf и nadeb.ttf (предварительно скачав их с сайта Vedi), а также t2a.enc из пакета pscyr. Для генерации pdf файла будет использована команда pdflatex напрямую создающая pdf-файл из исходников latex. Поэтому создайте в этой временную директорию еще файл pdftex.cfg, скопировав системный pdftex.cfg файл (найти его можно командой

```
cd / ; kpsewhich -progname=pdftex pdftex.cfg
```

у меня это /usr/share/texmf/pdftex/config/pdftex.cfg). Где-нибудь в конце этого файла добавьте строчку

```
map +myfonts.map
```

и создайте в директории `~/latex-fonts` файл myfonts.map (в него будем записывать описания сгенерированных фонтов). Теперь, сделав `cd ~/latex-fonts`, убедитесь, что tetex видит локальный конфигурационный файл

```
cd ~/latex-fonts; kpsewhich --progname=pdftex pdftex.cfg
```

(ответ должен быть ./pdftex.cfg). Таким образом вы можете как угодно экспериментировать со шрифтами не затрагивая системную конфигурацию tetex.

Еще раз замечу, что метод который будет приведён ниже работает только в случае использования команды pdflatex, генерирующей pdf файл непосредственно из latex исходника. Если же вам для каких-то целей нужно сгенерировать постскрипт-файл (последовательностью команд latex...; dvips ...), то тут имеются проблемы связанные с тем, что ghostscript не умеет обрабатывать embedded truetype фонты (в конце я остановлюсь на этом).

4 type1 шрифты.

Поскольку технология type1 шрифтов, также как и формат pdf, разработаны Adobe, то с type1 шрифтами проблем не возникает. Вначале изготавливаем tfm из afm файла (например из шрифта QuakeCyr)

```
>afm2tfm quake.afm -T t2a.enc
quake QuakeCyr " T2AEncoding ReEncodeFont " < t2a.enc
```

Команда afm2tfm (входит в состав tetex) конвертирует afm (adobe font metric) файл в quake.tfm файл, попутно перекодируя его в T2AAdobeEncoding (-T t2a.enc). Вывод этой команды это примерно та строчка которую теперь нужно добавить в файл myfonts.map. Вам только нужно еще указать файл из которого нужно брать начертание шрифтов. Так что в окончательном виде вам нужно дописать в файл myfonts.map строку:

```
quake QuakeCyr " T2AEncoding ReEncodeFont " < quake.pfa <
t2a.enc
```

(означает она следующее - latex имя шрифта, постскрипт имя шрифта, команда указывающая перекодировку фонта, файл где находятся глифы шрифта и файл с кодировкой).

5 Лигатуры.

Как уже упоминалось выше, latex-у вообще говоря нужен всего один файл tfm с теховской метрикой шрифта. Однако, в этом файле содержится ещё одна важная информация, что нужный символ присутствует в шрифте и о лигатурах [2], которые поддерживает этот шрифт. Вообще-то, для латинского шрифта отутствие в нём 4-х f-лигатур (fi, ffi, fl и ffl) считается дурным тоном, но и в русском тексте лигатуры тоже крайне важны, поскольку TEXсильно полагается на поддержку шрифтом (точнее файлом tfm) нескольких специфических лигатур. Это подстановка вместо двух дефисов короткого тире (endash), вместо трёх - динного тире (emdash) и кавычек-"гусиных лапок" (guillemots) вместо сдвоенных знаков больше-меньше. Фактически это просто макросы, но выполненные не в виде TEX-овских подстановок, а в виде подстановок, реализованных в самом шрифте.

Так вот, к сожалению, способ приведенный выше этого не обеспечивает. Может быть для шрифта QuakeCyr это и не обязательно, но для нормального текстового шрифта это большой недостаток. Так что посмотрим, как этот пробел восполнить. Для этого нам придется перекодировать tfm ещё раз и, возможно, немного подкрутить в нем метрику. Так что запустим команду afm2tfm с дополнительной опцией:

```
afm2tfm quake.afm -v quake.vpl -T t2a.enc
```

где vpl-файл - это файл с текстовым описанием tfm метрики. Для LATEXон не нужен, и мы его используем только в качестве временного файла, который нужно будет открыть, при необходимости отдактировать и с помощью него получить уже "правильный" tfm файл. В частности нас интересует часть с лигатурами:

```
(LIGTABLE
   (LABEL O 25) (comment endash)
   (LIG O 55 O 26)
   (LIG O 177 O 26)
   (STOP)
   (LABEL O 47) (comment quoteright)
   (LIG O 47 O 21)
   (STOP)
   (LABEL O 55) (comment hyphen)
   (LABEL O 177) (comment hyphen)
   (LIG O 55 O 25)
   (LIG O 177 O 25)
   (STOP)
   (LABEL O 140) (comment quoteleft)
   (LIG O 140 O 20)
   (STOP)
   )
```

Интересно, что тут ничего править не надо - все нужные лигатуры присутствуют. В частности, в шрифте есть два дефиса - `(comment hyphen)`, под номером 55 и 177 (в этом файле все числа приводятся в восьмеричном коде). Лигатура знака дефиса с самим собой `(LIG O 55 O 25)` даёт символ с номером 25, т.е. endash. Можно проверить по файлу t2a.enc, что он занимает 21 позицию в кодировке, или же найти его в таблице символов в том же vpl файле, но несколько дальше:

```
(CHARACTER O 25 (comment endash)
   (CHARWD R 340)
   (CHARHT R 272)
   )
```

[2] т.е. связываниях, или иными словами операциях подстановки вместо группы символов другого символа.

Соответственно для endash определена лигатура с дефисом (LIG O 55 O 26), что даёт emdash - символ с порядковым номером 26. Точно также определены лигатуры одиночных кавычек в правые и левые двойные кавычки. Чтобы добавить лапки-guillemots (для шрифта Quake это особого смысла не имеет, поскольку он не содержит символов “больше-меньше”, из которых строятся guillemots), добавим в таблицу лигатур сразу же после команды LIGTABLE ещё пару определений:

```
(LABEL O 74) (comment less)
(LIG O 74 O 276)
(STOP)
(LABEL O 76) (comment greater)
(LIG O 76 O 277)
(STOP)
```

(т.е. лигатуры знаков больше и меньше самих с собой дают символы с порядковыми номерами 276 и 277, которые как раз и являются символами guillemotsleft и guillemotsright соответственно).

Как бы там ни было, в итоге с помощью команды:

```
vptovf quake.vpl quake.vf quake.tfm
```

из vpl файла можно сгенерировать улучшенный вариант tfm файла, а также ещё один файл с расширением vf. Вы будете смеяться, но это тоже шрифт, который используется при отображении dvi документов с помощью специализированных программ (xdvi, dvitk). На самом деле, можно обойтись и без vf-файлов, поскольку сейчас все эти просмотрщики довольно тесно интегрированны с ghostscript и используют вызовы ghostscript для генерирования растрового изображения из нужного шрифта. Решение, которое приводится в данном тексте, как раз позволяет выполнять такие преобразования на лету. Это занимает немного больше времени, но в принципе, заметных неудобств не создает. Так что в данном случае vf шрифты - это побочный продукт, который вы по своему усмотрению можете либо установить в каталог texmf, а можете отправить в /dev/null (я предпочитаю второй вариант).

6 ttf шрифты.

Расммотрим теперь TrueType шрифты. С ними ситуация хуже. Поскольку стандарт Adobe на true-type шрифты не распространяется, то кодировка символов в ttf шрифте может отличаться от стандартной адобовской и в основном проблемы возникают именно с тем, что в шрифте не удаётся найти глиф с именем (к примеру) /afii10017, а большая русская буква А именуется как Agrave. Поэтому прежде чем начать встраивать ttf шрифт в latex рекомендуется вначале просмотреть его с помощью ghostscript чтобы понять какой способ наименования глифов выбрал автор при создании шрифта. Проще всего это сделать добавив в ghostscript’овский Fontmap.GS файл строчку вида:

```
/Test-Font (/path/to/some/font.ttf);
```

и открыть в gv ps файл:

```
(prfont.ps) runlibfile
/Test-Font DoFont
```

В результате в ghostviewer’е вы увидите таблицу со всеми глифами содержащимися в шрифте вместе с их адобовскими названиями.

Способов кодирования символов может быть три (the good, the bad and the ugly):

- Автор шрифта следует адобовскому стандарту именования символов. Самый лучший вариант и (к счастью) самый распространённый - если вы имеете дело с профессионально сделанным шрифтом (например с Monotype'овским), то можно быть уверенным, что шрифт закодирован именно так.
- В шрифте сбита кодировка (случайно или умышленно). Некоторый глифы имеют странные названия (если вы смотрите шрифт с помощью DoFont команды) или не отображаются, когда вы пытаетесь использовать шрифт в latex-документе.
- Шрифт закодирован с помощью грубого хака - национальные символы отображаются на верхнюю половину ISOLatin1Charset.

Последние два случая лечению, тем не менее, поддаются.

Начнём с самого простого случая. В первую очередь вам нужно извлечь метрику шрифта из ttf файла (напомню, что в отличии от type1 шрифтов, где метрика и глифы хранятся в разных файлах, truetype шрифты состоят из одного файла). Для этого воспользуемся командой ttf2afm из пакета ghostscript:

```
ttf2afm -e t2a.enc -o dscoptic.afm dscoptic.ttf
```

Единственная не очевидная опция - это -e t2a которая указывает, что в dscoptic.afm файл будут помещена информация только о символах содержащихся в файле t2a.enc. Далее следуем уже известным путём:

```
afm2tfm dscoptic.afm -T t2a.enc
dscoptic DSCoptic " T2AEncoding ReEncodeFont " < t2a.enc
```

и добавляем эту строчку в файл myfonts.map так же как и раньше, опять же дописывая имя шрифта:

```
dscoptic DSCoptic " T2AEncoding ReEncodeFont " < dscoptic.ttf < t2a.enc
```

DSCoptic - это вариант 1, шрифт закодированный согласно адобовскому стандарту. Вариант 2 - шрифт Надежда, в котором латинская буква `/H` именуется как `/FL0048h`. Так что, если вы попытаетесь конвертировать его аналогично случаю приведённому выше, то большой латинской буквы H в текстах вы не увидите. Лечится это подкручиванием кодировки. Копируете файл `t2a.enc` в файл `t2a_nade.enc` и в последнем вместо строчки `/H` вставляете `/FL0048h`. После чего конвертируете файл nade.ttf

```
afm2tfm nade.afm -T t2a_nade.enc
echo 'nade Nadejda-Bold " T2AAdobeEncoding ReEncodeFont" < nade.ttf < nade.enc '
```

Шрифт Stonhenge - это третий случай. В принципе можно исправить кодировку в нем точно также как и в шрифте Надежда (исправлений вам, правда, придётся вносить гораздо больше), но можно просто попытаться найти кодировку, с помощью которой авторы шрифта делали подстановку символов. Для шрифта Stonehenge эта кодировка содержится в файле 8r.enc (TeXBase1Encoding), который входит в состав tetex.

```
ttf2afm -y 8r.enc -o stone.afm stone.ttf
afm2tfm stone.afm -T 8r.enc
echo 'stone Stonehenge " TeXBase1Encoding ReEncodeFont" < stone.ttf < 8r.enc ' >
```

Разумеется остаётся вариант поправить кодировку внутри шрифта с помощью программы fontforge [7] и привести кривой шрифт к варианту 1. Возможно с непривычки это будет более утомительно, но результат получается более прямым путём.

7 Тестируем шрифты.

Простой пример. В данном примере переопределяются стили для roman, sans-serif и typewriter текста. Четвёртый шрифт определяется in-place и используется разово.

```
\documentclass[a4]{article}
\usepackage[koi8-r]{inputenc}
\usepackage[russian]{babel}
\renewcommand{rmdefault}{stone}
\renewcommand{sfdefault}{nade}
\renewcommand{ttdefault}{dscoptic}
\title{Использование type1 и tt шрифтов в LaTeX.}
\author{Проект Vedi}
\sloppy
\begin{document}
\maketitle
Дурацкий тест включения true-type и type1 шрифтов в LaTeX. \sf Внимание!
\tt Специалистам в полиграфии и издательском деле просьба этот текст не компилир
Это может повлечь за собой \fontScary quake at20pt \Scary обострение
профессиональных заболеваний - аллергию, тошноту и рвоту !
\end{document}
```

В скомпилированном командой pdflatex test.tex (запускаем её в каталоге latex-fonts) документе, outlined окажется только шрифт Scary (т.е. QuakeCyr), вместо остальных tetex подставит растеризованные шрифты. Если вы посмотрите на ошибки, которые выдаст tetex, то вы заметите, что он не находит шрифты T2A/dscoptic/m/n и др. и честно сообщает, что вместо них он использует шрифты по-умолчанию.

Ошибка связана с тем обстоятельством, что LaTeX является WYSIWYM системой и кое-что пытается проделать за вас. А именно, в качестве /Scary фонта вы ему однозначно дали понять, что нужно использовать шрифт QuakeCyr, а с помощью \renewcommand{\rmdefault... вы ему указали, что для надписей, выполняемых roman шрифтами, он должен использовать семейство шрифтов stone, при этом самостоятельно (точнее в соответствии с используемым стилем) подбирая какой текст выделять italic-ом, bold-ом или normal. Таким образом нам понадобятся ещё файлы с описаниями семейств шрифтов, т.е. fd-файлы. Писать их придётся вручную, но все они достаточно однотипны.

t2adscoptic.fd

```
\ProvidesFile{t2adscoptic.fd}[DSCoptic Font]
\DeclareFontFamily{T2A}{dscoptic}{}
\DeclareFontShape{T2A}{dscoptic}{m}{n}{ <-> dscoptic}{}
```

t2anade.fd

```
\ProvidesFile{t2anade.fd}[Nadejda Font]
\DeclareFontFamily{T2A}{nade}{}
\DeclareFontShape{T2A}{nade}{m}{n}{ <-> nade}{}
```

t2astone.fd

```
\ProvidesFile{t2astone.fd}[Stonehenge Font]
\DeclareFontFamily{T2A}{stone}{}
\DeclareFontShape{T2A}{stone}{m}{n}{ <-> stone}{}
```

Ну и для полноты картины еще файл для QuakeCyr:

```
t2aquake.fd

\ProvidesFile{t2aquake.fd}[QuakeCyr Font]
\DeclareFontFamily{T2A}{quake}{}
\DeclareFontShape{T2A}{quake}{m}{n}{ <-> quake}{}
```

Т.е. мы в данном случае определили 4 семейства в кодировке T2A , каждое из которых содержит по одному medium normal (обычный прямой) шрифту. После того как все 4 fd-файла окажутся в директории latex-fonts, осталось только перезапустить команду `pdflatex test-koi.tex`.

В итоге, в сгенерированном pdf файле все шрифты окажутся или Type1 или TrueType, Embedded в том и другом случае - в чем можно убедиться командой :

```
$pdffonts test.pdf
name type emb sub uni object ID
------------------------------------ ------------ --- --- --- ---------
Stonehenge TrueType yes no no 6 0
Nadejda-Bold TrueType yes no no 9 0
DSCoptic TrueType yes no no 12 0
OXCCSV+QuakeCyr Type 1 yes yes no 15 0
```

8 Наводим порядок.

Если результат тестирования вас удовлетворил, то можно заняться раскладыванием шрифтов и конфигурационных файлов по их привычным местам.

```
cp *.afm /usr/share/texmf/fonts/afm/public/local
cp *.pfa *.pfb  /usr/share/texmf/fonts/type1/public/local
cp *.ttf /usr/share/texmf/fonts/truetype/public/local
cp  *.tfm  /usr/share/texmf/fonts/tfm/public/local
cp *.fd /usr/share/texmf/tex/latex/local
cp *.enc /usr/share/texmf/dvips/base
cat ./myfonts.map >> /usr/share/texmf/dvips/config/pdftex.map
```

Проверьте также, что файл pdftex.map указан в конфигурационном файле pdflatex'а /usr/share/texmf/pdftex/config/pdftex.cfg. Если нет допишите в него строчку map +pdftex.map (или map pdftex.map). Теперь остаётся только синхронизировать базу данных tetex (команда mktexlsr) и шрифты готовы к использованию.

9 tetex-3.0

Предыдущие советы работают в tetex-2.0. В следующих версиях с одной стороны поменялось расположение шрифтовых каталогов tetex, а с другой, произошло их разделение на общесистемные и пользовательские. Первые хранятся в директориях `/usr/share/texmf-config` `/usr/share/texmf-var`, а вторые в домашнем каталоге пользователя `${HOME}/.texmf-var`, `${HOME}/.texmf-config`. Для поддержания какой-то видимости порядка в этом случае приходится использовать shell-script `/usr/share/texmf/bin/updmap` и `/usr/share/texmf/bin/updmap-sys`. Последний из них применяется для обновления только общесистемных каталогов и использует внутри себя вызов updmap. По замыслу авторов updmap с его помощью можно иметь один конфигурационный файл и с его помощью генерировать конфиги, пригодные для dvips, pdflatx, pdfetex и т.д. [3]

[3] К сожалению их надежды не оправдались, по-крайней мере в части truetype шрифтов. Об этом будет сказано в следующей части.

Чтобы шрифты заработали в tetex-3.0 нужно скопировать их в системные каталоги, так же как и в предыдущем параграфе, кроме двух последних строчек:

```
...
cp *.enc /usr/share/texmf/fonts/enc/dvips
cp myfonts.map /usr/share/texmf/fonts/map/dvips/local/myfonts.map
```

После чего сгенерировать обновленный список конфигурационных файлов и список шрифтов:

```
mktexlsr
/usr/share/texmf/bin/updmap-sys --enable Map=myfonts.map
/usr/share/texmf/bin/updmap-sys
```

В тестовых целях можно так же как и в предыдущей версии tetex держать все шрифты и конфигурационные файлы в текушей директории или (более правильный вариант) в личном каталоге `~/texmf`. В этом случае нужно будет использовать вместо команды `updmap-sys` команду `updmap`.

Поскольку пакет PSCyr практически заморожен, то соответственно инструкции по установке, описанные в прилагаемом к нему файле README, также не работают. Однако его можно установить и в tetex-3.0 творчески переосмыслив сведения из этого параграфа (замените где надо каталоги local на pscyr). Всё должно работать. Можете воспользоваться моим пакетом soviet-fonts [8], он включает в себя все шрифты из PSCyr за исключением M$ шрифтов.

10 Почему же нельзя использовать ttf шрифты с dvips ?

На самом деле можно, но ;) ...

Зачем может понадобиться dvips, если pdflatex позволяет напрямую конвертировать latex в более современный pdf формат? Однако, pdflatex имеет одно серьёзное ограничение - он не умеет обрабатывать встроенную в latex-документ postscript графику [4]. Но, в то же время, вместо использования pdflatex напрямую можно пойти длинным путём - конвертировать latex в dvi-файл (`latex test.tex`), затем полученный dvi преобразовать в постскрипт (командой из пакета tetex - `dvips test.dvi`), и потом уже конвертировать его в pdf программой из пакета ghostscript `ps2pdf test.ps`. Как в таком случае программа dvips обработает truetype используемые в теле документа?

Программа dvips конвертирует latex шрифты в постскриптовские по правилам записанным в её конфигурационном файле /usr/share/texmf/dvips/config/psfonts.map. Его формат такой-же как и pdftex.map, поэтому строки соответствующие type1 шрифтам вы можете не задумываясь переносить в этот файл. С ttf шрифтами возникает следующая проблема: если в строке описания шрифта упоминается файл *.ttf, то dvips встраивает этот шрифт в постскрипт файл. А ghostscript не умеет работать со встроенными в постскрипт файл truetype шрифтами. Соответственно любые программы использующие при работе ghostscript (gv, gs и ps2pdf в том числе) вылетают с ошибкой (если у кого имеется интерпретатор постскрипта, отличный от gs, или постскрипт-принтер, то можете проверить как они реагируют на встроенные ttf).

Очевидное решение - не встраивать в постскрипт true-type шрифты - тоже не проходит, поскольку постскрипт без встроенных "экзотических"шрифтов - вещь крайне непортабельная и отсылать его своим сотрудникам или знакомым не стоит - все равно они там ничего не увидят. А именно, предположим вы добавляете в psfonts.map строчку

```
dscopt DSCoptic " T2AEncoding ReEncodeFont " < t2a.enc
```

[4] обойти это ограничение можно, если преобразовать рисунки в pdf-формат, а в latex вставлять уже сгенерированные pdf файлы

и генерируете постскрипт файл. Чтобы ghostscript смог его отобразить, нужно указать ghostscript'у, где именно нужно брать глифы для шрифта DSCoptic, для чего в файл /usr/share/ghostscript/<gs version>/lib/Fontmap.GS вставляете путь к файлу шрифта

```
/DSCoptic (/usr/share/texmf/fonts/type1/public/local/dscoptic.ttf);
```

Ghostviewer после этого должен правильно и без ошибок отобразить документ. Но нужно понимать, что в данном случае залогом правильного отображения документа является наличие шрифта DSCoptic в системе/ghostscripte. Если же этот ps-файл отослан на компьютер, где этот шрифт не установлен, то ghostviewer подставит шрифт по умолчанию (обычно какой-нибудь Courier) и вам здорово повезёт если в нем будут содержаться все используемые в тексте глифа и системная кодировка этого файла будет совпадать с кодировкой использованного в тексте шрифта.

В то же время из такого ps-файла можно собрать pdf с уже встроенными шрифтами, т.е. можнопереложитьзадачу по встраиванию шрифтов в документ с dvips на утилиты, входящие в состав ghostscript. Чтобы сгенерировать из ps-файла pdf запускаете команду ps2pdf14[5] и на выходе получаете файл со встроенными ttf шрифтами.

В tetex-3.0 (точнее в последних версиях ghostscript-а, вплоть до 8.5) ситуация не изменилась. Вставлять ttf шрифты в постскрипт по-прежнему невозможно. Поэтому с помощью updmap нельзя получить конфигурацию, которая одинаково хорошо работала бы как с dvips, так и pdflatex. Иными словами, если в myfonts.map оставить строчки с командой на встраивание шрифта (типа "... < dscoptic.ttf < t2a.enc"), то dvips будет создавать некорректный ps файл. Если же команду на встраивание шрифта убрать из map файла, то pdflatex будет создавать непереносимые pdf файлы (с не встроенными шрифтами). В качестве альтернативы, в уже упоминавшемся выше пакете soviet-fonts используется модифицированный updmap скрипт. Его отличие в том, что он по разному интерпретирует строки в map файле начинающиеся с ; и * (в оригинальном updmap эти строки считаются комментариями, также как и строки с %). В первом случае строка используется для генерации конфига для dvips, а во втором - для pdflatex. Поэтому можно иметь map файл вида:

```
;gentn Gentium " T2AAdobeEncoding ReEncodeFont " <t2a.enc
*gentn Gentium " T2AAdobeEncoding ReEncodeFont " <t2a.enc <GentiumR.ttf
```

так что после обновления конфигурации с помощью updmap (или updmap-sys) встроенные true-type шрифты будут генерироваться только в результате обработки исходного tex-файла с помощью команды pdflatex.

11 Один полезный скрипт

Большую часть описанных в статье действий можно автоматизировать. К примеру с помощью этого tcl-скрипта:

```
#!/usr/bin/tclsh
if { $argc > 0 } {
    set prefix [ lindex $argv 0 ]
} else {
    set prefix "Myfonts"
}
set h [ exec ls ]
set f [open "${prefix}.map" "w+"]
set f1 [open "${prefix}.GS" "w+"]
```

[5]Тут важно использовать именно комаду генерирующую pdf последней версии (в данном случае 1.4), поскольку ps2pdf, использующий вызов ps2pdf12, действует наиболее безопасным и универсальным способом. А именно – вставляет в pdf документ уже растеризованные символы.

```
foreach i $h {
    if { [regexp {\.ttf$} $i] } {
        set i1 [regsub \.ttf $i ""]
        catch {
            exec ttf2afm -e t2a.enc  -o ${prefix}${i1}.afm ${i1}.ttf
        }
        set l [exec afm2tfm ${prefix}${i1}.afm -v ${prefix}${i1}.vpl -T t2a.enc]
        set f2 [open ${prefix}${i1}.vpl RDWR ]
        set hh [split [read $f2] "\n"]
        seek $f2 0
        foreach j $hh {
            puts $f2 $j
            if { $j == "(LIGTABLE" } {
                puts "OK!"
                puts $f2 {   (LABEL O 25) (comment emdash)}
                puts $f2 {   (LIG C - O 26)}
                puts $f2 {   (STOP)}
                puts $f2 {   (LABEL C -) (comment endash)}
                puts $f2 {   (LIG O 177 O 177)}
                puts $f2 {   (LIG C - O 25)}
                puts $f2 {   (STOP)}
                puts $f2 {   (LABEL O 74) (comment less)}
                puts $f2 {   (LIG O 74 O 276)}
                puts $f2 {   (STOP)}
                puts $f2 {   (LABEL O 76) (comment greater)}
                puts $f2 {   (LIG O 76 O 277)}
                puts $f2 {   (STOP)}
                flush $f2
            }
        }
        close $f2
        exec vptovf ${prefix}${i1}.vpl ${prefix}${i1}.vf ${prefix}${i1}.tfm
        puts $f ";$l"
        puts $f "*${l} < ${i1}.ttf"
        set k [lindex $l 1]
        puts $f1 "/${k} ($i);"
    }
}
exec mkdir -p  texmf/fonts/afm/public/${prefix}
exec mkdir -p texmf/fonts/truetype/public/${prefix}
exec mkdir -p texmf/fonts/tfm/public/${prefix}
exec mkdir -p texmf/fonts/map/dvips/${prefix}
exec mkdir -p ghostscript/fonts
exec cp *.afm texmf/fonts/afm/public/${prefix}
exec cp *.ttf texmf/fonts/truetype/public/${prefix}
exec cp *.ttf ghostscript/fonts
exec cp  *.tfm  texmf/fonts/tfm/public/${prefix}
close $f
close $f1
set gver [exec gs --version]
exec mkdir -p ghostscript/${gver}/lib
exec cp ${prefix}.GS ghostscript/${gver}/lib
exec cp ${prefix}.map texmf/fonts/map/dvips/${prefix}
```

Скрипт запускается с единственным аргументом - названием шрифтового пакета, под которым этот набор шрифтов будет установлен в tetex (по умолчанию Myfonts). Скрипт создает для каждого шрифта afm, tfm и vpl файлы с исправленными лигатурами, а также создаёт списки шрифтов, пригодных для использования с dvips и ghostscript, и два готовых дерева каталогов, которые нужно переместить в директорию texmf (предварительно, разумеется, написав fd файлы - эту часть работы скрипт автоматизировать не в состоянии) и ghostscript. Полученный же на выходе GS файл проще всего импортировать в ghostscript с помощью скрипта:

```
PREFIX=Myfonts
k=$( echo $(gs --version) | sed -r -e 's/([^.]*\.[^\.]*).*/\1/g' )
echo "($PREFIX.GS) .runlibfile" >> /usr/share/ghostscript/$k/lib/Fontmap
```

12 Всякое-разное

- Как настроить поиск и копирование текста в pdf файлах сгенерированных с помощью pdflatex?

  Вообще-то это зависит от приложения, в котором вы просматриваете pdf-файлы. Вот как это можно сделать в acrobat reader версии 4.0 под Linux (не думаю что в 5.0 имеются какие-то отличия).

  Устанавливаете системную локаль cp1251 и генерируете truetype шрифты в кодировке cp1251 (как сделать это описано в статье Русский в X-ах). После этого правите шелл-скрипт acroread и меняете там первую строчку, где определяется переменная окружения `LC_ALL` на

  ```
  export LANG=ru_RU.cp1251
  export LC_CTYPE="ru_RU.cp1251"
  ```

  После этого вам нужно поменять шрифт интерфейса acrobat reader на любой шрифт в подходящей кодировке. Проще всего это сделать добавив строку в файл `~/.Xdefaults` вида:

  ```
  *FontList:-monotype-*-*-*-*-*-*-*-*-*-*-*-*-cp1251
  ```

  В таком случае текстовый поиск работает, а с копированием текста нужно иметь ввиду, что Xselection будет вам возвращать текст в кодировке cp1251, так что вклеивание текста будет правильно работать в приложениях запущенных в той-же локали.

- Я вставляю в latex файл postscript-графику, а после того как я делаю latex-dvips в получившемся постскрипт файле все надписи на рисунках куда-то пропадают?

  Значит вы используете одинаковые фонты как в рисунках, так и в latex документе, и кроме этого в рисунках фонты не embedded. Еще одна причина не использовать dvips. Проблема вот в чем: когда вы создаёте ps файл, все шрифты в нем получаются embedded с custom кодировкой, причем название шрифтов остаётся тем же самым. Если в графическом файле используется шрифт с тем же названием, то при отображении графики будет использован именно embedded шрифт, а не тот что определён в Fontmap.GS. Кодировка этих двух логических шрифтов (встроенного и ghostscript-овского) в общем случае не обязана совпадать, поэтому надписи на рисунках окажутся не теми, на что вы рассчитываете. Что делать? Самое правильное - конвертировать ps-графику в pdf и пользоваться pdflatex (как и было сказано ранее). Если по каким-то причинам этого сделать нельзя - используйте в графических файлах шрифты, постскриптовские имена которых не конфликтуют с именами шрифтов, применяемых в документе.

- Вопрос касается не совсем LaTeX, но тем не менее... Есть много графических программ (используемые для построения графиков, диаграмм, схем ...), которые при печати или выводе ps-графики, не вставляют шрифты в ps-файл (т.е. как в рассмотренном выше примере получается непортабельный ps-файл, с именами шрифтов, но без глифов). Можно ли как-нибудь вставить шрифты в такой постскрипт файл?

  Можно, конечно же, просто конвертировать его в pdf, но если этот вариант не устраивает, то можно вручную вклеить шрифты (которые, разумеется, должны быть type1) в этот файл. Для этого нужно все упоминаемые в файле шрифты конвертировать в pfa формат (это важно, поскольку непосредственно с pfb-шрифтами этот трюк не работает) и слить в один файл. К примеру, это можно сделать командой pfb2pfa, которая входит в состав tetex. Вот как это выглядит для набора шрифтов TextbookPSCyr:

```
touch textbookpackage
for i in textb*.pfb;
do
pfb2pfa $i zz
cat zz >> textbookpackage
done
```

  а затем полученный пакет вклеить в нужный ps файл где-нибудь поближе к началу, но после DSC комментариев. Я для этой цели использую sed-скрипт:

```
/^%(%|!)/H
/^%%EndComments$/  G; r /dev/stdin; d;
p
```

  Применяется он таким вот образом:

```
cat textbookpackage | sed -rnf dsc.comments.sed diagramm.ps \\
  > diagramm_with_embedded_fonts.ps
```

13 Шрифты. Где взять?

Ситуация сейчас такова, что проще найти иностранный шрифт с кириллическими глифами, чем шрифт изготовленный российскими разработчиками. Очень хороший набор шрифтов предлагает SIL [9] - Charis во всех начертаниях, Gentium и Doulos только в roman[10].

Также можно использовать вариант известного компьютерного пакета шрифтов BitstreamVera с кириллическими буквами (по условиям оригинальной лицензии разработчики должны были поменять название после добавления новых символов) - DejaVu , хотя, на мой взгляд, в печатном виде они выглядят несколько грубовато. Скачать последние версии можно с сайта [11], но вряд ли вам эта ссылка понадобится, поскольку эти шрифты включены в любой дистрибутив Линукса.

Ещё один набор компьютерных шрифтов LinuxLibertine [12], похожий на Times New Roman, разработан немецким автором Philipp H. Poll.

Кроме этих шрифтов время от времени в интернете можно найти интересные шрифты с русскими названиями, написанными в транслите (например, Svoboda или Vremya). Это продукты разорившейся компании Casady&Greene, тем не менее удостоившейся отдельной статьи в Wikipedia [13]. Их лицензионная чистота сомнительна, но против их использования в документах частного характера никто, я думаю, протестовать не будет (т.к. некому). Правда кодировка у этих шрифтов крайне экзотическая, что и не удивительно, поскольку большинство продуктов компании Casdy&Greene были для Macintosh.

Теперь про русские шрифты. Тут на рынке имеется единственный монополист, компания Paratype, которая состоит из числа работников НПО Полиграфмаш, отколовшаяся от него

в конце 80-х годов (под именем Параграфа) и получившая лицензии у НПО Полиграфмаш на распространение старых советских шрифтов. Непонятно, правда, почему на основании этой лицензии Паратайп объявляет все остальные версии советских шрифтов (в том числе PSCyr, Романа Павлова и Дмитриева) пиратскими.

Вот как выглядит эта история со слов очевидца [14].

> Литературная была первоначально оцифрована в 80-х годах XX века в отделе наборных шрифтов (ОНШ)Полиграфмаша в рамках программы переработки для фотонабора старых металлических шрифтов, причем оцифровка велась в формате Икарус. Поскольку Полиграфмаш выпускал только документацию, но не матрицы и шрифтоносители, документация на Литературную (рисунки знаков и таблицы ширин, а потом и цифровые данные) была передана на Ленинградский завод полиграфических машин (ЛЗПМ), который выпускал советские фотонаборные устройства и шрифтоносители к ним. Таким образом, рисунки и цифровые данные на Литературную (как и на многие другие шрифты советского периода) оказались одновременно в архиве ОНШ и в архиве ЛЗПМ. Естественно, цифровые данные в этих архивах хранились в формате Икарус.

Обычно, обосновывая права Паратайпа на советские шрифты, ссылаются на то, что они самостоятельно оцифровывали шрифты с бумаги в цифру. Как видно это не так. Т.е. советские шрифты были оцифрованы, когда Паратайпа еще и в проекте не было. Формат же Икарус позволяет автоматическую конвертацию из формата в формат [15].

> Когда единая отрасль полиграфического машиностроения развалилась в перестройку и каждый ее участник стал выживать самостоятельно как умеет, и в ОНШ и в ЛЗПМ попытались наладить выпуск цифровых шрифтов в формате ПостСкрипт Т1 на основе своих шрифтовых архивов. ОНШ быстро прекратил существование (в 1993 году), и шрифты из его архива (вернее, их постскриптовские интерпретации) разошлись в разные места – кое-что было лицензировано ПараГрафом, кое-что Интермикро, а что-то попытались утилизовать другие структуры Полиграфмаша (довольно безуспешно). Сейчас все шрифты ПараГрафа и Интермикро, в свое время оцифрованные в ОНШ, распространяются через библиотеку ПараТайпа (большинство в основной части, а остальные в дополнительной или непосредственно в библиотеке Интермикро).

Вот на сцене появляются первые падальщики и паратайп среди них (под именем ПараГрафа).

> Отдел шрифтов ЛЗПМ некоторое время распространял шрифты, разработанные ОНШ, в формате Т1, затем освоил формат ТруТайп и даже начал выпуск собственных оригинальных шрифтов. При этом в нем практически не было профессиональных шрифтовых дизайнеров, поэтому шрифты ЛЗПМ часто страдали некомплектом знаков и дефектами рисунка. Очевидно, в ЛЗПМ также не было менеджеров, способных наладить выпуск и распространение качественных цифровых шрифтов, поэтому в конечном итоге отдел шрифтов ЛЗПМ закрылся, как ранее ОНШ.

Как видно, в желающих распространять альтернативные версии советских шрифтов недостатка не было. Т.е. возникновение нескольких цифровых вариантов одних и тех же шрифтов произошло довольно давно, ещё до паратайпа. Поэтому крайне неубедительными выглядят утверждения (типа сделанного другим участником дискуссии):

> ”Академическая”и ”Обыкновенная Новая”Р.А. Павлова очевидным образом содраны с паратайповских: это видно невооружённым глазом, но я ещё специально совмещал контуры в редакторе и убедился, что они идентичны.”

Понятное дело, они похожи, так как имеют общего цифрового предка. А кто у кого сдирал - это ещё вопрос. Но тем не менее видно, что история компании несколько отличается от глянцевого образа, вроде этого [16]

> В 1989 году была основана фирма "ПараГраф занимавшаяся проектами в сфере компьютерных технологий, одним из отделов которой был отдел шрифтов. Он состоял из профессиональных шрифтовиков - бывших сотрудников НПО "Полиграфмаш"и нескольких молодых талантливых программистов. Последние разработали комплект программ - Fontain, с помощью которого дизайнеры могли проектировать шрифты непосредственно прямо на компьютере и переводить ранее сделанные с рисунков в цифровой вид. Советская типографика того времени не отличалась шрифтовым разнообразием - в арсенале дизайнеров было около десятка основных гарнитур ("Литературная "Обыкновенная новая "Журнальная рубленая "Школьная "Букварная"). "ПараГраф"перевёл эти шрифты в компьютерные форматы и занялся разработкой новых.
>
> Затем последовал контракт с американской шрифтовой фирмой ITC (International Typeface Corporation), по которому "ПараГраф"получал право разрабатывать кириллические знаки к наиболее популярным и распространённым западным шрифтам. Так образовывалась шрифтовая библиотека, получившая имя "ПараТайп". Тем временем у "ПараГрафа"сменился владелец и часть направлений, в том числе, шрифтовое, ему оказались не интересны. Таким образом, в 1998 году появилось ООО "ПараТайп"с уже "приличной"библиотекой шрифтов, квалифицированным штатом сотрудников и налаженными бизнес связями по миру.

Пакет шрифтов А.Дмитриева можно взять на его сайте [17]

> Пакет содержит основные отечественные книжные гарнитуры: Академическую, Елизаветинскую, Литературную и Обыкновенную новую, пригодные для набора текста в как в современной, так и в дореформенной русской орфографии, и всё необходимое для их использования с наборной системой LaTeX.

Пакет PSCyr постоянной дислокации не имеет, на момент последней редакции первая позиция в Гугле была [1].

Автор поддерживает в более-менее актуальном состоянии пакет шрифтов soviet-fonts [8], в который входят некоторые из упоминавшиеся в этом разделе шрифты, а также скрипты, позволяющие использовать эти шрифты (помимо tetex) в Ghostscript, freetype и в виде старых X-fonts.

Благодарности. Автор благодарит Evgeny Adishchev за полезные замечания.

Список литературы

[1] Последняя бета-версия пакета PSCyr.

[2] В своё время проект хостился на сайте `http://www.prodtp.ru/` и содержал большое количество относительно свободных декоративных шрифтов (автор большинства из них Н.Дубина). К сожалению, проект закрылся, а упоминаемые в статье шрифты я сейчас не могу найти в свободном доступе. Некоторые из шрифтов из этого проекта используемые в качестве примеров собраны в архиве: `http://post.hppi.troitsk.ru/~mike/fine-fonts-examples.tgz`.

[3] *Adobe Type 1 Font Format*.

[4] *PDF Reference, Sixth Edition, version 1.7*.

[5] *Supporting Fonts in the PostScript Language Environment*.

[6] Adobe. *PostScript Language Reference*, 3 edition.

[7] Домашняя страница Fontforge.

[8] Последнюю версию установочного пакета для Slackware можно найти в каталоге - .

[9] Summer Institute of Linguistics, христианская организация занимающаяся исследованиями и развитием малоизвестных языков.

[10] Каталог SIL.

[11] Шрифты DejaVu.

[12] Шрифт LinuxLibertine.

[13] Статья в Wikipedia о компании Casdy & Greene.

[14] Дискуссия в ЖЖ.

[15] Статья в Wikipedia о старом шрифтовом формате Ikarus.

[16] О компании.

[17] Сайт А.Дмитриева.