Oprogramowanie

Wykresy w emTEX-u

Piotr Jachowicz, Przemysław Kowalik
i Wojciech Myszka

Problem „wstawiania” wykresów do tekstu przygotowanego w LATEX-u jest częścią szerszego problemu: integracji tekstu i grafiki. Jak w każdym przypadku problem składa sie z dwu zadań:

1. Wyboru narzędzia do przygotowania wykresów (grafik),
2. Wyboru metody zintegrowania uzyskanych „efektów” z tekstem.

W niniejszym artykule omawiane są:

- Różne sposoby wstawiania wykresów, omówienie wad i zalet.
- Zalety i wady użycia pakietu GLE.
- Szczegółowa instrukcja użycia GLE.

Nie omawiamy tutaj sposobu użycia ani wszystkich możliwych, ani też powszechnie znanych narzędzi (takich jak Matlab czy Mathematica), a dających podobne, czy może czasami lepsze możliwości przygotowania samych wykresów.

1 Możliwości

Pozornie, mamy dużo możliwości do wyboru:

(a) jeżeli do robienia wykresów użyjemy programu działającego pod Windows, to w tekście można zostawić wolne miejsce na wykres, tekst przekształcić za pomocą **dviwin** i pod Windows dołączyć wykres;

(b) jeżeli program do robienia wykresów, którego używamy ma możliwość eksportowania wykresów do postaci mapy bitowej (PCX, BMP), pliku HPGL[1] lub rysunku PostScript (ps, eps), to taki wykres możemy włączyć do tekstu:

  i. w przypadku mapy bitowej (czarno-białe pliki w formacie PCX lub BMP) przez `\special{em:graph plik.pcx}`;

  ii. w przypadku pliku „kolorowego” (TIFF, PCX, BMP, LBN, IFF, GIF, BMP, IMG, CUT) przez przekształcenie go w font przy pomocy programu `bm2font` (autor: F. Sowa) i włączenie do tekstu;

  iii. w przypadku PostScript-u przez użycie `epsf.sty` i przekształcenie tekstu do PostScript-u przez `dvips`, lub przekształcenie rysunku PostScript-owego do mapy bitowej przez GhostScript i postąpienie jak wyżej;

  iv. w przypadku pliku `hpgl` można użyć programu hp2xx pozwalającego dokonać jego konwersji do postaci METAFONT, eps, pliku `pcl`, komend `\special` emTEX-a, map bitowych;

(c) możemy też użyć programu, który wygeneruje wykres w postaci przyswajalnej przez emTEX-a;

(d) „narysować” wykres ręcznie przy pomocy różnych makrodefinicji (np. stylów `epic.sty`, `eepic.sty`, pakietu PICTEX); niekiedy użyteczny, jak sądzę, może być też `curves.sty`, chociaż w tym pakiecie występuje kolizja komendy `\arc` z identyczną komendą MEX-a.

(e) „narysować” wykres jako font dla METAFONT-a (z `bmp` jak w punkcie (b), lub przez pakiet `mfpic`);

(f) możemy „wydrukować” wykres na drukarkę laserową do pliku i przekształcić go przy pomocy programu `pcltomsp` (czasami ze względu na fonty download-owe trzeba użyć drukowania na DeskJet), lub na drukarkę PostScript-ową i wstawić jak rysunki w PostScripcie.

Jak to zwykle bywa, każdy sposób ma swoje wady. I tak:

(a) wymaga ręcznego wstawienia każdego wykresu do tekstu. Jeżeli włączamy jeden wykres, to można przeboleć, ale jeżeli ich jest np. 20 albo więcej, ten sposób jest bardzo pracochłonny.

(b) i. włączenie mapy bitowej każe nam pamiętać, jaka jest rodzielczość naszego tekstu. `\special` wstawia mapy bitowe w ten sposób, że na każdy punkt tekstu przypada jeden piksel obrazka. Na przykład wyeksportowany z QuattroPro for Windows „pełnokartkowy” wykres do mapy bitowej skurczył się po wstawieniu przez `\special` do rozmiarów o ile pamiętam coś około 5cm. Podobnie skromnie (co do rozmiarów) prezentuje się „zrzut” z ekranu. Jeżeli eksportujemy wykres do

[1] Język graficzny wykorzystywany do programowania ploterów.

mapy bitowej trzeba albo postarać się, żeby był odpowiedno duży, albo (w niektórych przypadkach możliwe) kazać wyeksportować w odpowiedniej rozdzielczości. Tak czy inaczej, taki wykres będzie zajmował sporo miejsca na dysku. Nieco miejsca można naturalnie zaoszczędzić wybierając odpowiedni format mapy bitowej. Na przykład plik .pcx jest na ogół mniejszy (czasami nawet sporo) niż plik .bmp z taką samą zawartością. Zaletą jest, że potem praca z nim jest dosyć szybka i łatwa. Wydruki do pliku z Quattro Pro 4.0 (wykres i arkusze) na drukarkę DeskJet 500 w rozdzielczości 300×300 dpi zostały przeze mnie (P. Kowalik) wykorzystane (co prawda w celach „pozaTEX-owych") do przygotowania artykułów do działu „Dla praktyków" w PCKurierze (1/95, 2/95, 17/95). Ilustracje przesłałem do redakcji jako pliki graficzne w formacie pcx. Były one otrzymane poprzez przetworzenie wydrukowanych do pliku fragmentów arkusza programem `pcltomsp.exe` i lekko poprawione `NeoPaintem`. Użyłem tych pcx-ów poźniej również w dokumencie LATEX-owym przygotowanym na własne potrzeby. W obu przypadkach efekt końcowy był, jak sądzę, zadowalający.

ii. ten sposób byłby najlepszy i najbardziej zgodny z „duchem" TEX-a, gdyby nie prowadził przez plik z bitmapą. Zdaje się, że to jest jego jedyna wada.

iii. przekształcenie całego tekstu do formatu PostScript jest jedynym znanym mi realnym sposobem „uratowania" niezależności pliku wyjściowego od urządzenia drukującego (patrz: Uwaga filozoficzna). Wydaje mi się, że każdy błąd popełniony przed przekształceniem do PostScript-u (np. zostawienie za dużo wolnego miejsca pod rysunkiem) każe nam powtarzać całą operację od początku, a błąd jest wykrywalny dopiero gdy mam gotowy `.ps`! Przekształcenie PostScript-u do mapy bitowej przez GhostView ma wiele zalet: nie wymaga interaktywnego udziału użytkownika (można napisać plik wsadowy `.bat` i przyjść jak skończy), umożliwia ustawienie rozdzielczości, łatwą obróbkę pliku `.ps` lub `.eps` (obroty, skalowania), umożliwia, jeżeli mamy `.eps` „narysowany" na całej kartce, wybranie tylko jego fragmentu (w pikselach). Wady: duży rozmiar pliku z mapą bitową, fonty generowane z bitmap są czasami bardzo brzydkie.

Podczas korzystania z rysunków (wykresów) w postaci EPS pamiętać należy, że najlepsze efekty osiągniemy gdy korzystamy z „prawdziwych" (to znaczy wektorowych) plików. Każdą mapę bitową można przekształcić do postaci EPS, ale mapy bitowe (w każdej postaci) bardzo źle się skalują – stosunkowo najlepiej można je powiększać stosując całkowite współczynniki powiększenia.

iv. program hp2xx używa bardzo nieładnych (ale wzorowanych na tych używanych przez plotery) fontów.

(c) z programów generujących kod przyswajalny przez emTEX-a spotkałem tylko GnuPlot-a. Generuje kod przy użyciu `\special` emTEX-a, lub kresek LATEX-owych. Niestety nie obywa się bez wad: w przypadku skomplikowanego wykresu kod obrazka jest bardzo długi (przetwarzanie jest wolne, albo brakuje pamięci do przetwarzania), trzeba ręcznie wyciąć wstawioną przez niego komendę ustawienia nieistniejącego fontu, i nigdy nie wiadomo, jakiej wielkości wykres dostaniemy (ewentualne zmiany trzeba robić skalując go). Dużą zaletą GnuPlot-a jest, że jeżeli robimy dużo podobnych wykresów (np. tylko ze zmiennymi danymi), to można kazać mu generować wykresy nieinteraktywnie. Drugim sposobem jest użycie makrodefinicji Eitan M. Gurari opisanych w książce „TEX & LATEX. Drawing & literate programming" i dostępnych przez ftp z `ftp.cis.ohio-state.edu` w katalogu `pub/tex/osu/gurari`. Tam też można znaleźć przykłady ich stosowania. Makra pozwalają generować różne wykresy (kołowe, liniowe, słupkowe, punktowe i wiele innych) w najróżniejszych konfiguracjach przy pomocy wpisywanych lub pobieranych z pliku wartości. Niestety, czas przetwarzania tych wykresów jest spory, a sam wygląd pozostawia nieco do życzenia (ukośne linie składają się z widocznych „schodków").

(d) jest sposobem dla ludzi, którzy albo mają do zrobienia tylko jeden wykres, albo duże zacięcie i BARDZO dużo czasu. Faktem jest jest jednak, że pracę tym sposobem można sobie nieco przyspieszyć. Na przykład PICTEX(dlaczego akurat PICTEX, patrz uwaga filozoficzna) może być włączony do formatu. Przygotowałem (P. Kowalik) dla próby format LATEX2.09 + LAMEX1.05 + PICTEX1.1 dla btex186.exe. Format ten wydawał się działać bez zarzutu, znacznie przyspieszając korzystanie z PICTEX-a. Przy okazji, polskie litery w Latin2 jak i w notacji „ciachowej" były dostępne bez najmniejszego problemu.

(e) j.w. Przekształcenie `.bmp` w font przez `bm2font` zostało opisane w (b)

Uwaga filozoficzna: osobiście uważam, że pracując z emTEX-em pod DOS-em nie mamy dobrego narzędzia do robienia wykresów: wygodnego i zgodnego z „duchem" TEX-a. Jedną z podstawowych idei TEX-a jest przekształcenie tekstu na plik, który nie jest zależny od urządzenia drukującego: ten sam plik `.dvi` możemy drukować na rozmaitych drukarkach, za każdym razem otrzymując najlepszy rezultat jaki jest do uzyskania. Wydaje się, że powyższe kryterium najlepiej spełnia PICTEX. Ma on jedną istotną zaletę: jest to zbiór makr TEX-a jako takiego, bez żadnych „podpórek" typu `\special{...}` czy PostScript. Zapewnia to rzeczywistą przenośność między różnymi implementacjami TEX-a zarówno na poziomie pliku źródłowego, jak i `.dvi`. Szkoda, że PICTEX jest tak powolny i wymaga wielkiej pamięci. Gdyby chociaż istniał program, coś jak krzyżówka Gnuplot-a z TEXCAD-em, dający na wyjściu kod dla PICTEX-a... Każdy wstawiony wykres w postaci mapy bitowej (lub np. przekształcony w font z mapy bitowej) wiąże nas z rozdzielczością, jaką przewidzieliśmy. Po prostu taka jest natura korzystania z rysunków rastrowych. Natomiast wstawienie rysunku wektorowego napotyka na inne opory: albo nie ma dobrych narzędzi do zrobienia go (jeżeli chcemy wstawić sposobem (c), (d), (e)), albo (jeżeli mamy wykres w PostScrip-cie) trzeba przekształcić cały dokument do formatu PostScript-u (niezależnego od urządzenia) i dopiero go drukować). Jeżeli nie mamy drukarki z wbudowanym PostScript-em, musimy używać emulatora lub interpretera i cała operacja jest mocno „okrężna" (dvi $\rightarrow^{\text{dvips}}$ ps $\rightarrow^{\text{emulator(interpreter)}}$ drukarka). Ponieważ na programy wykonujące wykresy w formacie DVI nie ma co liczyć (i nie jestem pewny, czy dałoby się je wstawiać), najlepiej byłoby gdyby TEX produkował plik od razu w formacie PS. Wtedy jedyną wadą byłby rozmiar pliku `.ps`.

2 Zalety pakietu GLE

Osobiście używam pakietu GLE. Zalety:

- istnieje 32-bitowa wersja wraz z pełną dokumentacją;
- opracowuje wykres na podstawie „programu" zapisanego w pliku tekstowym;
- umożliwia duże manipulowanie wykresem, używanie fontów PostScript-owych (czyli prawdopodobnie również polskich), TEX-owy zapis wyrażeń (np. `\Omega = 2^{14}`), rysowanie przy użyciu wielu narzędzi (pętle, zmienne, podprogramy, bezpośredni dostęp do plików);
- daje pomocnicze narzędzia do opracowywania wykresów:
  - najlepsze dopasowanie „dowolnej" funkcji elementarnej do danych,
  - manipulowanie danymi,
  - rysowanie wykresów trójwymiarowych,
  - generowanie wykresów trójwymiarowych dla „dowolnej" elementarnej funkcji dwu zmiennych,
  - „zaokrąglanie" linii na wykresach trójwymiarowych,
  - rysowanie poziomic z wykresów trójwymiarowych,
- ma przyjazdne środowisko;
- tworzy nieinteraktywnie plik `.eps` lub `.ps`;
- jest dostępny na kilku platformach sprzętowych: DOS, UNIX (Linux), VMS, OS2.

3 Szczegółowe użycie pakietu GLE

Komputer `tui.marc.cri.nz` jest serwerem „rodzinnym" dla GLE. W podkatalogach katalogu `/pub/gle` znajdują się jego wersje na różne platformy sprzętowe, dokumentacja itd. Jeżeli chodzi o GLE dla DOS-a, to istnieją wersje 16 i 32 bitowe. Na polskich serwerach pakiet GLE można zdobyć np. przez ftp z `ftp.cyf-kr.edu.pl` w katalogu `/pub/mirror/SimTel/msdos/graphics`, lub `ftp.gust.org.pl` w katalogu `/pub/TeX/support/` `/graphics/msdos`. 32-bitową wersję tworzą (w Polsce, na `tui.marc.cri.nz` podział na pliki jest inny):

gle32h_1.zip	1134788
gle32h_2.zip	590609
gleread.zip	6766

Do przekształcania pliku .eps na .pcx używam GhostScript-a (wersja 2.6.1, ale nie ma powodu żeby nie używać nowszej). Najczęściej robię kilka lub kilkanaście podobnych wykresów – różniących się między sobą tylko danymi. Robię to tak:

1. Przygotowuję przy użyciu GLE jeden wykres. Dobieram wszystkie parametry. Najczęściej używam rozmiaru 18×12 (GLE używa centymetrów). Nazwę pliku z danymi przechowuję jako zmienną. Przypuśćmy, że mam dane w plikach: dane1.dat, dane2.dat, dane3.dat. Najprostszy wykres dla GLE będzie wyglądał tak (po szczegółowy opis odsyłam do dokumentacji):

```
size 18 12
dane$ = "dane1.dat"
begin graph
     nobox
     data dane$
     d1 line marker fcircle
end graph
```

2. Następnie przygotowuję tyle plików dla GLE ile mam wykresów do zrobienia, w każdym zmieniając tylko parametr dane$. Można przygotować mały programik, który będzie takie rzeczy szybko i sprytnie robił (zmieniał w drugiej linijce napis i zapisywał pod inną nazwą). Niech te pliki się nazywają dane1.gle, dane2.gle, dane3.gle

3. Teraz piszę plik .bat, który wygląda następująco:

```
gle_ps dane1.gle
gle_ps dane2.gle
gle_ps dane3.gle
```

   Po uruchomieniu dostaję pliki dane1.ps, dane2.ps, dane3.ps

4. ponieważ GLE przy zamianie na .ps ustawia wykresy dłuższym bokiem pionowo, muszę teraz przy konwersji na .pcx obrócić rysunek zachowując lewy dolny róg. W tym celu przygotowałem z pliku landscap.ps swój plik rotate.ps, który wygląda tak:

```
%!
gsave clippath pathbbox grestore
4 dict begin
/ury exch def /urx exch def
/lly exch def /llx exch def
-90 rotate llx lly add
ury sub lly neg translate % llx,lly
end
```

   Teraz przygotowuję taki plik .bat (oczywiście każda instrukcja w jednej linijce):

```
gs386 -sDEVICE=pcxmono
 -sOutputFile=dane1.pcx -r300
 -dNOPAUSE rotate.ps dane1.ps quit.ps

gs386 -sDEVICE=pcxmono
 -sOutputFile=dane2.pcx -r300
 -dNOPAUSE rotate.ps dane2.ps quit.ps

gs386 -sDEVICE=pcxmono
 -sOutputFile=dane3.pcx -r300
 -dNOPAUSE rotate.ps dane3.ps quit.ps
```

   i go uruchamiam. Dostaję pliki dane1.pcx, dane2.pcx, dane3.pcx, które

5. włączam do tekstu w taki sposób:

```
\begin{figure}
\begin{picture}(400,280)
\put(-50,360)%
{\special{em:graph dane1.pcx}}
\end{picture}
\end{figure}
```

6. ostatnią czynnością może być (choć polecam dopiero po wydrukowaniu pracy) skasowanie plików *.gle, *.ps.

Wykonanie w praktyce punktów 2–4 + 6 zajmuje dużo mniej czasu, niż mi teraz ich napisanie. Poza tym, większość rzeczy jest wykonywana automatycznie, bez udziału użytkownika. Można po wykonaniu punktu (a) napisać plik .bat, który wykona czynności 2–4 + 6 sam lub posłużyć się programem make.

Piotr Jachowicz
jachow@krysia.uni.lodz.pl
Przemysław Kowalik
Przemko@antenor.pol.lublin.pl
Wojciech Myszka
W.Myszka@immt.pwr.wroc.pl