✓ MEX

Polerowanie TEX-a: od systemu gotowego do użycia do systemu wygodnego w użyciu

Bogusław Jackowski i Marek Ryćko*

Abstract

W artykule opisana została polska adaptacja systemu składu TEX (dla wersji $\geqslant$3.0). Autorzy starali się pokazać, że z profesjonalnego punktu widzenia TEX *nie jest systemem wielojęzycznym* i że przystosowanie go do konkretnego języka nie jest bynajmniej zadaniem trywialnym. W artykule omówione są subtelne aspekty takiego przystosowania, dotyczące zarówno TEX-owej, jak i METAFONT-owej części zadania. Pokrótce omówione jest również łącze TEX–PostScript.

Słowa kluczowe: wersje narodowe formatów `plain` i LATEX, znaki narodowe, podcięcia (*kerns*), ligatury, METAFONT, PostScript.

1 Wstęp

Podejmując pierwszą próbę przystosowania TEX-a do języka polskiego [Jackowski et al. 88] byliśmy — jak to widzimy z dzisiejszej perspektywy — bardzo naiwni. Rok poźniej prezentując pracę o szumnym tytule "Polish TEX is ready for use" (*Polski TEX jest gotowy do użycia*) nie spodziewaliśmy się, że czeka nas jeszcze tak długa droga do uzyskania adaptacji zadowalającej z profesjonalnego punktu widzenia. Czym się różni skład „profesjonalny" od „nieprofesjonalnego"? Naszym zdaniem skład „nieprofesjonalny" po prostu nie istnieje.

Po kilku latach pracy TEX wreszcie został przystosowany do składania tekstów w języku polskim. Staraliśmy się nie uronić żadnej z TEX-owych subtelności wierząc, że właśnie subtelności stanowią o wysokiej jakości systemu składu. Trzeba przyznać, że mieliśmy sporo szczęścia: nasz język okazał się przystosowywalny, co wcale nie było takie oczywiste.

* Jest to tłumaczenie pracy „Polishing TEX: from ready to use to handy in use", która została wyróżniona nagrodą im. Cathy Booth na konferencji EuroTEX'92 w Pradze

Opracowane zostały — rzecz jasna — zasady dzielenia wyrazów w języku polskim, opracowane (i zaprogramowane w języku METAFONT) zostały też kształty polskich znaków diakrytycznych, ale to jeszcze nie wszystko. Użytkownik ma możliwość definiowania instrukcji zawierających w nazwie polskie litery, napisy pojawiające się na ekranie i tworzone w trakcie obróbki tekstu pliki mogą zawierać polskie znaki, o ile polskie znaki są dostępne na danej instalacji, a dana implementacja TEX-a ma konfigurowalną tabelę kodów (*code page*). Dopuszczalna jest również dwuznakowa notacja dla polskich znaków diakrytycznych. Polskie odpowiedniki formatów `plain` i LATEX są z tymi formatami w pełni zgodne w tym sensie, że polskie formaty stanowią rozszerzenie formatów oryginalnych. I tak dalej.

Tym niemniej nie jest nadal oczywiste, czy TEX daje się przystosować do dowolnego języka, a tym bardziej czy można utworzyć wielojęzyczną adaptację dla dowolnego zestawu języków. Nasze doświadczenia zdają się sugerować negatywną odpowiedź w obu przypadkach. O tych doświadczeniach traktuje właśnie nasza praca, a czytelnikowi pozostawiamy wyciągnięcie ostatecznych wniosków co do wielojęzyczności TEX-a.

Niezależnie od osądu czytelnika TEX jest dokładnie taki jaki jest. Niewątpliwie posiada pewne cechy wielojęzyczności, które można wykorzystywać w różny sposób. Chcielibyśmy wskazać czytelnikowi niebezpieczeństwa kryjące się za niektórymi rozwiązaniami.

Ograniczymy się do rozważenia jedynie dwóch zagadnień, mianowicie opiszemy polskie rozszerzenie fontów Computer Modern i przeanalizujemy problem zastosowania ligatur jako metody dostępu do znaków narodowych w foncie. Powody po temu są następujące: (1) w przeciwnym wypadku artykuł byłby stanowczo za długi; (2) szczegółowy opis tych dwóch zagadnień pozwala wyrobić sobie zdanie o całym procesie przystosowywania TEX-a do pracy w danym języku.

2 Fonty

Fonty Computer Modern (CM) nie nadają się do składania polskich tekstów. Głównym powodem jest brak w tych fontach liter 'ą', 'ę', 'Ą' i 'Ę'. Brak nawet znaku „ogonek", chociaż — jak okaże się dalej — nawet gdyby fonty CM taki znak zawierały, wiele by to nie pomogło. Tym samym byliśmy zmuszeni zastosować podejście odmienne od podejścia zastosowanego przez Holendrów w projekcie Babel czy Niemców w pakiecie LaTeX. Nie mogliśmy też użyć standardu ECM (*Extended Computer Modern*) zaakceptowanego podczas 5-ej Europejskiej Konferencji TeX-owej (Cork, Irlandia, 3–15 września 1990), gdyż prace nad tym standardem wciąż pozostają niezakończone. Przykładowo, nie został opublikowany zestaw niezbędnych podcięć i ligatur, a kształty znaków są niedopracowane. Ponadto TeX-owy format `plain` nie nadaje się do pracy z fontami ECM. Być może zabrakło nam odwagi, a na pewno motywacji, by pisać od początku „plainopodobny" format dla fontów ECM.

Mniejszym złem wydawało się nam rozszerzenie tabeli znaków fontów CM o polskie znaki diakrytyczne. Przedrostek CM w nazwach fontów zastąpiliśmy przedrostkiem PL, gdyż zgodnie z decyzją Knutha przedrostek CM jest zastrzeżony dla fontów oryginalnych.

Fonty tekstowe rodziny PL zawierają osiemnaście polskich liter: 'ą', 'ć', 'ę', 'ł', 'ń', 'ó', 'ś', 'ź', 'ż', 'Ą', 'Ć', 'Ę', 'Ł', 'Ń', 'Ó', 'Ś', 'Ź', 'Ż' oraz dodatkowe cudzysłowy: '„' (polski cudzysłów otwierający), '«' (francuski cudzysłów otwierający), i '»' (francuski cudzysłów zamykający). Jednakże nie dołączyliśmy tych znaków do fontu PLTEX10. Jego odpowiednik, font CMTEX10 odzwierciedla układ klawiatury, na której pracował Knuth (p. [Knuth86b], str. 568). Wydaje się zupełnie nieprawdopodobne, że klawiatura ta zawierała polskie znaki. Tym samym fonty CMTEX10 i PLTEX10 są identyczne.

Układ dolnej połówki fontów PL (znaki o kodach <128) jest identyczny z układem fontów CM. Zachowane zostały nawet takie anomalie jak rozbieżność między układem CMR10 a CMR5. Polskim znakom diakrytycznym przypisane zostały kody zgodne ze standardem ECM, zaś cudzysłowom — dość dowolnie wybrane kody >127 (mamy nadzieję, że nie okaże się któregoś dnia, że kody te zostały wybrane błędnie).

Fonty matematyczne PL nie zawierają polskich znaków diakrytycznych. Każdy z pewnością się zgodzi, że formuła „$ą^2 = ę^3$" wygląda nieco zabawnie. Jedynie font PLSY został poszerzony o znaki '$\leqslant$' i '$\geqslant$' używane w polskiej typografii zamiast '$\leq$' i '$\geq$'.

2.1 Zgodność z fontami CM

Mimo nieuniknionych różnic między fontami PL i CM, dążyliśmy do uzyskania możliwie najwyższej zgodności między obiema rodzinami, tzn. staraliśmy się by każdy font PL zawierał odpowiadający mu font CM.

Wiele razy walczyliśmy z pokusą zmieniania oryginalnych fontów. Na przykład chętnie zwiększylibyśmy kropki nad literami 'i' oraz 'j', zwłaszcza w fontach italikowych i imitujących maszynę do pisania fontach CMTT. Nie zrobiliśmy tego, respektując ważniejszą niż poczucie estetyki ideę zgodności. Inny przykład: w fontach CM nie jest wstawiane automatycznie podcięcie ani między 'A' i 'w', ani między 'A' i 'v', podczas gdy istnieją słowa angielskie w których takie pary liter mogą się pojawić, np. (*nomen omen*) „Awkward" lub „Average". Stosowne podcięcia zostałyby wygenerowane, gdyby w programach METAFONT-owych opisujących fonty PL zmiennej boole'owskiej *improve_kerns* nadać wartość *true*. Wartością domyślną tej zmiennej jest oczywiście *false* i nie przewiduje się, by w normalnych zastosowaniach wartość ta miała ulec zmianie.

Jedyne niezgodności, jakich nie udało się nam uniknąć, spowodowane są ograniczeniami systemu TeX/METAFONT: plik metryczny każdego z fontów (plik TFM) nie może zawierać więcej niż szesnaście różnych wysokości. Gdy program generuje większą liczbę różnych wysokości, METAFONT przed utworzeniem pliku metrycznego redukuje ich liczbę zastępując wymiary oryginalne wymiarami przybliżonymi.

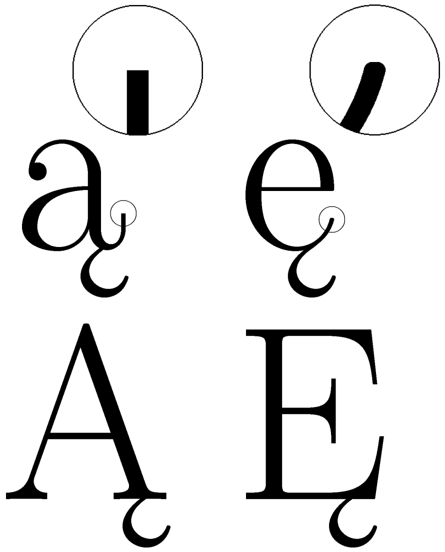
Tymczasem prawie wszystkie fonty CM zawierają już znaki mające łącznie szesnaście różnych wysokości (np. CMR, CMTI, CMSS, CMCSC, CMTT). Z drugiej strony wprowadzenie dodatkowych wysokości jest w przypadku polskiego języka niezbędne ze względu na majuskuły 'Ć', 'Ń', 'Ó', 'Ś', 'Ź' oraz 'Ż'. Nie wydaje się sensowne sztuczne zmniejszanie wysokości tych znaków, a dodanie choćby jednej nowej wysokości musi spowodować, że wysokości niektórych znaków w fontach PL będą się różnić od wysokości takich samych znaków w fontach CM. Oznacza to, że w pewnych — miejmy nadzieję niezwykle rzadkich — przypadkach TeX inaczej przełamie na strony tekst złożony fontami CM niż tekst złożony fontami PL. Łamanie wierszy będzie jednakże w obu wypadkach identyczne[1].

[1] Nie tylko liczba różnych wysokości w foncie jest ograniczona. Również liczba tzw. poprawek italikowych (*italic corrections*) nie może przekroczyć liczby 64. Otarliśmy się prawie o tę granicę, gdyż w fontach PLSL10, PLSL12, PLSSI17, PLTI8 oraz

Mimo iż niezgodności nie są wielkie, czasem może się okazać, że jedynie użycie fontów CM wchodzi w rachubę. Nie oznacza to bynajmniej konieczności generowania odpowiednich fontów za pomocą METAFONT-a. Wystarczy zastosowanie fontów wirtualnych opartych o pliki metryczne fontów CM i mapy bitowe fontów PL (patrz [Knuth90b]).

2.2 Kształty znaków diakrytycznych

Najbardziej charakterystycznymi znakami dla polskiego języka są litery z ogonkiem, tzn. ‘ą’, ‘ę’, ‘Ą’ oraz ‘Ę’. Należy podkreślić, że w istocie co najmniej trzy różne ogonki są potrzebne: inny ogonek jest potrzebny dla ‘ą’, inny dla ‘ę’, a dla ‘Ą’ i ‘Ę’ jeszcze inny. Powodem jest różny sposób połączenia litery z ogonkiem w każdym z tych przypadków[2]. Proszę także zwrócić uwagę na różne zakończenia łuków w literach ‘a’ oraz ‘e’ — sprawiło nam sporo kłopotu takie zaprojektowanie ogonka, by harmonijnie łączył się z obiema literami:



Co do rozmiarów ogonka, to zdecydowaliśmy, że duże i małe litery będą miały ogonki tej samej wielkości (z wyjątkiem fontu PLCSC). Dzięki temu w foncie bezszeryfowym wystarczają szczęśliwie dwa ogonki, gdyż ogonek przy literze ‘ą’ jest dokładnie taki sam jak ogonek przy ‘Ą’ i ‘Ę’:

ąęĄĘ

Niektórzy typografowie wolą jednak, by rozmiary ogonków przy majuskułach były nieco większe niż przy minuskułach. Jeśli dodatkowo ogonki przy ‘Ą’ i ‘Ę’ byłyby różne (np. z powodu ekscentrycznych szeryfów — takich jak w fontach CMFF10 czy CMFI10), mielibyśmy do czynienia z czterema różnymi ogonkami. A jeśli, na przykład, chcielibyśmy uwzględnić litewskie ogonki przy ‘i’ i ‘u’, to liczba ogonków jeszcze by wzrosła.

Typowymi dla języka polskiego są także litery ‘ł’ i ‘Ł’. Fonty CM zawierają wprawdzie znak przekreślenia dla liter ‘l’ i ‘L’, ale naszym zdaniem litery uzyskane za jego pomocą są dalekie od doskonałości. Zdecydowaliśmy się na wyraźne rozróżnienie pomiędzy przekreśleniem minuskuły i majuskuły (lewa para znaków to efekt działania makr \l i \L, prawa — to znaki z fontów PL):

łŁłŁ

Oryginalne (CM-owe) litery ‘ł’ i ‘Ł’ prowadzą z reguły do katastrofalnych efektów. Na przykład w słowie „Jagiełło” litery ‘ł’ są stanowczo zbyt blisko siebie:

Jagiełło Jagiełło

Fonty PL zdają się okazywać nieco więcej szacunku królewskiemu nazwisku:

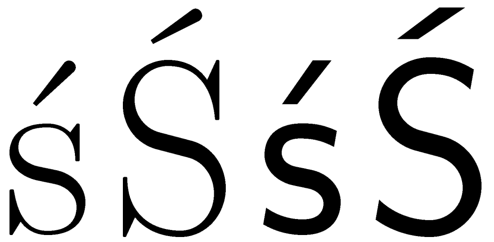
Jagiełło Jagiełło

Akcenty nad minuskułami (litery ‘ć’, ‘ó’, ‘ń’, ‘ś’ oraz ‘ź’) w fontach PL są niemal identyczne

PLTI12 „wyszły” nam aż 63 różne poprawki italikowe. Gdybyśmy mieli mniej szczęścia i przekroczyli „zakazaną granicę” 64, również akapity mogłyby być łamane na wiersze inaczej w obu przypadkach.

[2] Ogonek przy ‘ą’ powinien w zasadzie łączyć się z literą łagodnie, podobnie jak w literze ‘ę’. Jednakże w fontach CM prowadziłoby to do zbytniego przesunięcia ogonka przy ‘ą’ w prawo, dlatego zastosowaliśmy inne podejście.

z akcentami uzyskiwanymi za pomocą TEX-owej instrukcji `\'`. Natomiast akcenty nad majuskułami są nieco „spłaszczone”, gdyż w przeciwnym razie litery byłyby zbyt wysokie, co utrudniałoby zachowanie odpowiednich odstępów międzywierszowych. Proszę zwrócić uwagę, że dla fontów bezszeryfowych zwykły obrót lub pochylenie (ang. *slant*) akcentu nie wystarcza — potrzebny jest nieco subtelniejszy zabieg:



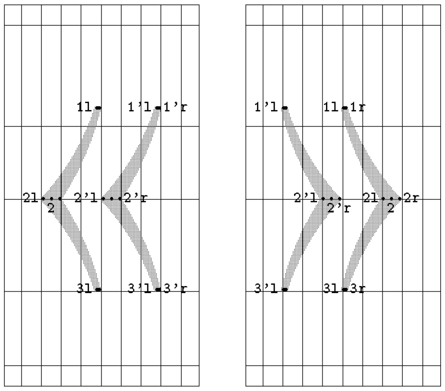
Pozostały nam jeszcze litery ‘ż’ i ‘Ż’. W fontach CM kropka akcentowa uzyskiwana za pomocą instrukcji `\.` nie zawsze jest taka sama jak kropka nad ‘i’ czy ‘j’. Jest to ewidentny błąd, typowy zresztą dla fontów zagranicznego pochodzenia pojawiających się na polskim rynku. Proszę porównać powiększone słowo „niż” złożone fontem `CMTI` (z lewej) i fontem `PLTI` (z prawej):

Warto podkreślić, że w ogólnym przypadku dodanie znaków diakrytycznych może oznaczać konieczność zmiany kształtu niektórych innych znaków w foncie. Innymi słowy, zabieg typu „dorobienia akcentów” jest z reguły niewystarczający dla uzyskania fontu zawierającego znaki diakrytyczne dobrze pasujące do pozostałych znaków.

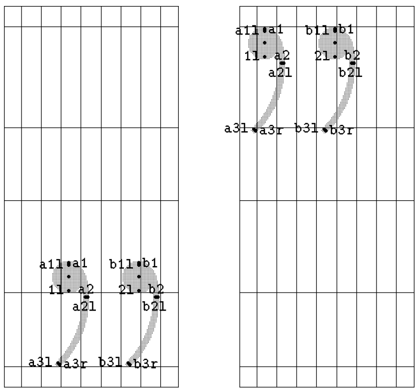
2.3 Cudzysłowy

Polski cudzysłów otwierający jest podobny do angielskiego cudzysłowu zamykającego, tyle że przesunięty w dół do podstawy znaku i umieszczony w polu znaku nieco bardziej na prawo; polski cudzysłów zamykający jest dokładnie taki sam jak cudzysłów angielski:

Cudzysłowy francuskie (używane w języku polskim wewnątrz cytatów) zostały dołożone do zestawu znaków, gdyż w fontach CM nie istnieją znaki, które do tego celu można by wykorzystać.



Nie należy mylić cudzysłowu otwierającego ‘«’ z zamykającym ‘»’ (por. pkt 3.2).



Nawiasem mówiąc kolejny raz mieliśmy tu szczęście. Gdyby polski cudzysłów zamykający był nie taki jak angielski lecz taki jak niemiecki, tj. ‘“’, to byśmy się znaleźli w kłopotliwej sytuacji. Otóż CM-owa ligatura ‘“’ nie może zostać po prostu użyta jako cudzysłów zamykający: powinna być przesunięta trochę w lewo, inne współczynniki odstępu (*space factors*) powinny zostać przypisane znakom tworzącym ligaturę (p. pkt 3.2) i tylko jeden ze znaków — albo *angielski cudzysłów otwierający* ‘“’, albo *niemiecki zamykający* ‘“’ — mógłby otrzymać uprzywilejowaną pozycję znaku będącego ligaturą dwóch apostrofów.

2.4 Automatycznie wstawiane podcięcia

Dodatkowe odstępy, wstawiane automatycznie przez system składu między niektórymi znakami (*implicit kerns*), są sprawą kluczową dla uzyskania druku wysokiej jakości. Nawet najpiękniej zaprojektowane litery nie pomogą, gdy odstępy te są dobrane nieprawidłowo. Z przykrością należy stwierdzić, że podcięcia fontów CM nie są dobrane najlepiej (p. pkt 2.1). Po części wynika to ze śmiałego zamierzenia Knutha, by całą rodzinę fontów opisać jednym programem METAFONT-owym. Wydaje się nam, że jest rzeczą niemal niemożliwą odpowiednie wyważenie podcięć we wszystkich siedemdziesięciu pięciu fontach rodziny CM za pomocą wspólnej dla całej rodziny instrukcji `ligtable` (zawierającej tylko jedną instrukcję warunkową, sprawdzającą czy dany font jest fontem szeryfowym, czy też bezszeryfowym)!

Polskie znaki diakrytyczne niewiele się różnią od odpowiadających im liter łacińskich, dlatego też i odpowiednie podcięcia niewiele się różnią. Jak zawsze, tak i w tym wypadku jest trochę wyjątków. Np. bez dodatkowego odstępu między 'ą' i 'g' litery te prawie by się dotykały (sytuację bez podcięcia przedstawia para znaków z prawej strony):

ąg ąg

Wiele podcięć zostało wprowadzonych w związku z polskim cudzysłowem otwierającym '„', gdyż musi on być odsunięty od dolnych szeryfów i dosunięty do takich liter jak 'W' czy 'T'. Tym niemniej staraliśmy się zachować styl CM-owy i zawsze posłusznie przestrzegaliśmy pouczenia Knutha [Knuth86a, str. 317]:

> *„Nowicjusze często przesadzają z podcięciami. Najlepsze efekty uzyskuje się stosując podcięcia dwukrotnie mniejsze od tych, które wydają się dobre na pierwszy rzut oka, jako że podcięcia nie powinny być zauważalne (zauważalny powinien być ich brak). Podcięcia, które dobrze wyglądają w znaku firmowym lub tytule, na ogół zakłócają rytm czytania w zwykłym tekście."*

2.5 Składanie polskich tekstów z użyciem PostScript-u

Łącze TEX-PostScript jest bardzo ważnym elementem w procesie otrzymywania druków wysokiej jakości. Dlatego skupimy się przez chwilę na zagadnieniach związanych z tym łączem.

Ogólnie rzecz biorąc PostScript pozwala na użycie trzech rodzajów fontów:

- fontów reprezentowanych za pomocą map bitowych
- ładowalnych fontów obwiedniowych
- wbudowanych fontów obwiedniowych

Pierwsze dwa rodzaje fontów nie sprawiają żadnych kłopotów, tyle że — póki co — obwiedniowe PostScript-owe fonty PL nie istnieją. Na szczęście na TEX-u świat się nie kończy. Wiele firm produkuje obwiedniowe fonty PostScript-owe zawierające polskie znaki. Jest tylko jeden problem z takimi fontami, ten mianowicie, że polskie znaki diakrytyczne są na ogół niedopracowane. Jedynie fonty z firmy BitStream zdają się być wyjątkiem, przynajmniej jeśli chodzi o znaki polskie, chociaż akcenty nad majuskułami i minuskułami są identyczne (p. pkt 2.2), a podcięcia zostawiają wiele do życzenia.

Wadliwie zaprojektowane znaki narodowe stanowią wyzwanie dla lokalnych twórców fontów, skutkiem czego na rynku pojawia się mnóstwo rozmaitych odmian i przeróbek fontów oryginalnych, co z kolei rodzi kłopoty ze standaryzacją. Ale to już zupełnie inna historia.

Efekty uzyskiwane za pomocą fontów trzeciego rodzaju (wbudowanych fontów firmy Adobe) są zdecydowanie najgorsze. Jak to już zostało wyjaśnione w punkcie 2.2, są potrzebne co najmniej trzy różne ogonki, podczas gdy firma Adobe oferuje w swoich fontach wbudowanych tylko jeden (oczywiście nie ma w tych fontach gotowych polskich znaków diakrytycznych). W efekcie co najmniej jedna litera z ogonkiem musi mieć kształt „nieprofesjonalny".

Podobne zarzuty można postawić znakom akcentowanym: w fontach wbudowanych jest tylko jeden akcent *acute*, podczas gdy — jak to wyjaśnialiśmy — przydałyby się dwa. Operację „spłaszczania" akcentu można oczywiście próbować zaprogramować w języku PostScript, choć nie jest bynajmniej proste zadanie. Co więcej, PostScript (podobnie jak TEX) traktuje znaki złożone inaczej niż zwykłe litery. Na przykład wydrukowanie obwiedni znaku z doczepionym ogonkiem jest praktycznie niemożliwe, PostScript wydrukuje nałożone na siebie obwiednie litery i ogonka.

I wreszcie należy podkreślić, że *nie istnieje* jeden PostScript (inaczej niż TEX): staranne zaprogramowanie umieszczania akcentów okazać się może daremnym trudem, gdyż przy zmianie maszyny PostScript-owej ten sam program może umieścić akcenty w naprawdę zadziwiających miejscach.

Tak więc opinię Knutha [Knuth90b, str. 13]:

> *„Używając tych fontów[3] łatwiej składam teksty w językach europejskich za pomocą fontów Times-Roman, Helvetica czy Palatino niż za pomocą fontów Computer Modern!”*

wypada uznać za zbyt entuzjastyczną. Naszym zdaniem chcąc składać polskie teksty z wykorzystaniem PostScript-u należy unikać używania wbudowanych fontów PostScript-owych. Znacznie lepsze wyniki uzyskuje się używając fontów obwiedniowych lub bitmapowych. Dodać jeszcze tylko można, że przydałoby się łącze TEX-PostScript uzupełnić łączem METAFONT-PostScript.

Na zakończenie części dotyczącej fontów chcielibyśmy podkreślić, że gdyby nie życzliwa i wszechstronna pomoc ze strony zmarłego niedawno pana Romana Tomaszewskiego, wybitnego polskiego typografa, prezesa polskiego oddziału A-Typ-I (*Association Typographique Internationale*) nie uniknęlibyśmy mnóstwa pułapek i kto wie czy udałoby się nam doprowadzić prace do końca.

3 Ligatury uznane za niepożądane

W podstawowym podręczniku systemu TEX, zatytułowanym TEXbook, Knuth sugeruje, że mechanizm tworzenia ligatur można wykorzystać do uzyskiwania w TEX-u europejskich znaków narodowych [Knuth90a, str. 45–46]. Rozważając fonty wirtualne Knuth idzie nawet dalej [Knuth90b, str. 13–14]:

> *„Zgrabny schemat ligatur dla wielu języków europejskich został dopiero co opisany przez Haralambousa w listopadowym numerze TUGboat-a. Haralambous używa wyłącznie znaków ASCII uzyskując Aacute za pomocą kombinacji* `<A`*. Z łatwością mógłbym dodać ten schemat do mojego dopisując kilka linii w moich plikach* `vpl`*. W istocie kilka różnych konwencji można by zastosować równocześnie (choć nie polecałbym takiego podejścia).”*

Pozwalamy sobie nie zgodzić się z naszym Wielkim Szamanem: *nie polecamy żadnej konwencji ligaturowej* jako metody dostępu do znaków diakrytycznych. Pokażemy dalej, że niektóre mechanizmy TEX-a nie funkcjonują jak należy, jeśli użyć ligatur w ten właśnie sposób.

Weźmy dla przykładu język polski. Aby móc wprowadzać polskie znaki jako ligatury TEX powinien być przygotowany w następujący sposób:

- Należy wygenerować font zawierający wszystkie żądane znaki diakrytyczne.
- Pliki metryczne (pliki TFM) tych fontów należy tak skonstruować, by pewne ciągi znaków przekształcane były za pomocą mechanizmu ligaturowania w pojedyncze znaki, w naszym przypadku w znaki diakrytyczne; na przykład pary: `/a`, `/c`, `/e`, `/l`, `/n`, `/o`, `/s`, `/x` oraz `/z` powinny się ligaturować odpowiednio do: ‘ą’, ‘ć’, ‘ę’, ‘ł’, ‘ń’, ‘ó’, ‘ś’, ‘ź’ oraz ‘ż’. Można też użyć notacji *postfiksowej*: `a/`, `c/`, `e/`, `l/`, `n/`, `o/`, `s/`, `x/`, `z/`. Analogiczne ligatury należy przygotować dla majuskuł. W grę mogą wchodzić także inne rodzaje ligatur, np. do pomyślenia jest notacja *infiksowa* w rodzaju `a/a` (co mogłoby oznaczać np. ‘ą’). Haralambous we wspomnianej publikacji [Haralambous89] zaproponował ligatury potrójne. Jeszcze bardziej skomplikowany schemat został zaprojektowany przez twórców rodziny fontów cyrylicowych `WNCYR`[4].

Przyjrzyjmy się teraz konsekwencjom takiego zastosowania ligatur. Rozważymy kolejno: podcięcia, współczynniki odstępu (*space factors*) oraz parametry `\lefthyphenmin` i `\righthyphenmin`.

3.1 Ligatury kontra podcięcia

TEX automatycznie wstawia podcięcia wokół ligatur zgodnie z następującymi zasadami:

- podcięcie wstawiane między pojedynczym znakiem — powiedzmy — `v`, a ligaturą zestawioną ze znaków — powiedzmy — `xyz` jest takie samo jak podcięcie między `v` a `x` (tj. między `v` a pierwszym znakiem ligatury);
- podcięcie wstawiane między ligaturą `xyz` a pojedynczym znakiem `w` zależy od danej ligatury i znaku, i na ogół różni się od podcięcia między `z` a `w`.

W przypadku tradycyjnych ligatur (takich jak cudzysłowy czy ligatury zawierające literę ‘f’,

[3] Tzn. wbudowanych fontów PostScript-owych.

[4] Na przykład literę ‘Щ’ można uzyskać w fontach `WNCYR` na jeden z następujących sposobów: ШЧ, Шч, 6Х, 6х, ШЦХ, ШЦх, ШцХ, Шцх, СХЦХ, СХЦх, СХцХ, СХцх, СхЦХ, СхЦх, СхцХ, bądź Схцх. Tajemniczą szóstkę można otrzymać jako ligaturę ШЦ bądź Шц.

tj. 'ff', 'fi', 'fl', 'ffi', 'ffl') zasady te prowadzą do pożądanych efektów, gdyż ligatura przypomina kształtem znaki, z których została zestawiona. Jeśli chodzi o znaki diakrytyczne, rzecz się ma zupełnie inaczej: kształt ligatury na ogół *nie ma wiele wspólnego* z kształtem znaków tworzących tę ligaturę.

Załóżmy dla przykładu, że polskie litery mają być reprezentowane jako ligatury zawierające znak / („ciach") oraz literę: /a, /c, /e, /l, /n, /o, /s, /x, /z. Skutek będzie taki, że podcięcie przed polską literą będzie niezależne od litery, gdyż będzie związane ze znakiem /. Gdyby zastosować jakąkolwiek inną notację *prefiksową*, np. taką w której każda litera miałaby inny prefiks, nie poprawi to w istotny sposób sytuacji.

Knuth z pewnością był świadom tego rodzaju kłopotów proponując o/ i O/ (a nie /o i /O) jako przykładowe ligatury dla norweskich znaków narodowych 'ø' i 'Ø' [Knuth90a, str. 45–46].

Niestety, notacja *postfiksowa* także nie rozwiązuje problemu.

Gdyby polskie litery były kodowane z użyciem notacji *postfiksowej*, np. jako ligatury liter i znaku / (a/, c/, e/, l/, n/, o/, s/, x/, z/), podcięcie poprzedzające ligaturę byłoby takie samo jak podcięcie związane z literą rozpoczynającą ligaturę.

Dla *niektórych* języków i dla *niektórych* fontów notacja tego typu może dać poprawne podcięcia. Niestety, w ogólnym przypadku podcięcie przed ligaturą *nie musi być* takie samo jak przed literą rozpoczynającą ligaturę. Proszę na przykład porównać słowa „kat" i „kąt" złożone tzw. fontem kancelaryjnym:

kat kąt

Kolizja liter 'k' i 'ą' jest nie do uniknięcia, jeśli podcięcie przed 'a' i 'ą' jest takie samo, choć w tym szczególnym przypadku zalecamy zmianę kształtu ogonka lub użycie innego wariantu litery 'k' (lub i jedno, i drugie)[5].

A zatem używanie ligatur jako notacji dla znaków diakrytycznych koliduje z podcięciami wstawianymi automatycznie przez TEX-a. Szczególnie niekorzystna jest notacja *prefiksowa*.

[5] O bardziej przekonujące przykłady łatwiej w innych językach, proszę np. pomyśleć o niemieckich słowach „Tasche" i „Täglich".

3.2 Ligatury kontra współczynniki odstępu

TEX przypisuje każdemu znakowi „współczynnik odstępu" (\sfcode), dzięki czemu odstęp po dowolnym znaku może być regulowany. Patrząc pod tym kątem można wyróżnić cztery kategorie znaków:

- Wszystkie minuskuły oraz większość pozostałych znaków mają współczynnik odstępu równy 1000, co powoduje, że jeśli słowo jest zakończone takim znakiem, to odstęp po tym znaku pozostaje bez zmian.
- Znaki przestankowe w formacie plain mają współczynnik odstępu większy niż 1000, a to oznacza, że odstęp po takim znaku ma być nieco większy od normalnego odstępu.
- Majuskuły mają współczynnik odstępu równy 999; powoduje to, że odstęp po takim znaku praktycznie się nie zmieni jeżeli między majuskułą a odstępem pojawi się znak przestankowy. Za tym nieco dziwnym prawidłem kryje się obserwacja, że jeśli po dużej literze pojawia się kropka, to najprawdopodobniej jest to koniec skrótu (np. imienia) a nie koniec zdania.
- Są również znaki o współczynniku odstępu równym 0; taki współczynnik odstępu powoduje, że znak jest „przezroczysty" dla algorytmu wyznaczającego wielkość odstępu, tzn. wielkość odstępu zależy od pierwszego licząc od końca słowa znaku o niezerowym współczynniku odstępu. Format plain współczynnik równy 0 przypisuje nawiasom zamykającym oraz apostrofowi (a tym samym i ligaturze powstającej z dwóch apostrofów).

Co to ma wspólnego z wersjami narodowymi TEX-a i z dostępem do znaków diakrytycznych *via* ligatury? Z tego co zostało powiedziane powyżej wynika, że współczynnik odstępu wszystkich znaków narodowych powinien być równy 1000 dla małych liter, a 999 dla dużych.

TEX oblicza współczynnik odstępu dla ligatury na podstawie współczynników znaków tworzących ligaturę. W przypadku notacji *prefiksowej* (/a, /c, /e, /l, /n, /o, /s, /x, /z, /A, /C, /E, /L, /N, /O, /S, /X, /Z) wszystko jest w porządku. Dla notacji *postfiksowej* należałoby użyć innego znaku dla ligatur małych liter, a innego dla dużych, np.: a/, c/, e/, l/, n/, o/, s/, x/, z/, A', C', E', L', N', O', S', X', Z'.

Ale wówczas trzeba by nadać jednemu ze znaków (w tym wypadku „lewemu" apostrofowi) nienaturalny i niezgodny z formatem `plain` współczynnik odstępu równy 0 lub 999. W sposób oczywisty te znaki, które w formacie `plain` mają współczynnik odstępu równy 0, nie nadają się jako składniki ligatur (p. wyżej).

Jeśli chodzi o sugestię Knutha, by norweskie znaki 'ø' i 'Ø' zdefiniować jako ligatury o/ i O/, należy stwierdzić, że zawsze pojawią się kłopoty związane ze współczynnikami odstępu: albo współczynnik odstępu litery 'Ø' będzie równy 1000 zamiast 999, albo współczynnik odstępu znaku / będzie równy 0 lub 999 zamiast 1000.

Wróćmy jeszcze na moment do wspomnianej rodziny fontów `WNCYR`. Znak 'Ш' można otrzymać jako jedną z dwóch ligatur: `SH` lub `Sh`. Każda z nich prowadzi oczywiście do innego współczynnika odstępu. Ten sam współczynnik odstępu (999) otrzymało by się zastępując drugą z ligatur przez `sH`.

A zatem używanie ligatur jako notacji dla znaków diakrytycznych prowadzi na ogół do nieprawidłowych odstępów między słowami. Szczególnie niekorzystna jest notacja *postfiksowa*.

3.3 Ligatury kontra parametry `\lefthyphenmin` i `\righthyphenmin`

Ligaturę tworzą co najmniej dwa znaki. Okazuje się, że ma to nieoczekiwane i na ogół niepożądane konsekwencje jeśli chodzi o współdziałanie z parametrami `\lefthyphenmin` oraz `\righthyphenmin`.

Parametry te (dokładniej rejestry licznikowe, *count registers*) zostały wprowadzone do TEX-a 3.0. Bodźcem niewątpliwie był tu powstały wcześniej wielojęzyczny TEX M. Fergusona, w którym podobne parametry zostały wykorzystane. Załóżmy, że rejestry te zawierają liczby $\mathfrak{l}$ (`\lefthyphenmin`) oraz $\mathfrak{r}$ (`\righthyphenmin`). Dla algorytmu dzielenia wyrazów oznacza to, że jedynie pozycje między znakami $a_l a_{l+1} \ldots a_{n-r+1}$ słowa $a_1 a_2 \ldots a_n$ będą brane pod uwagę jako ewentualne miejsca podziału. W szczególności algorytm nie będzie próbował dzielić słów o długości $< l + r$.

Załóżmy, że słów pięcioliterowych i krótszych nie chcemy dzielić i dlatego przyjmujemy $l = r = 3$. Gdyby słowo „żądać" zapisać za pomocą ligatur jako `/z/ada/c`, TEX mimo wszystko mógłby podzielić to słowo po pierwszej sylabie („żą-dać"). Powodem jest sposób interpretowania słów przez algorytm przenoszenia. Otóż algorytm ten dzieli wyrazy *przed* utworzeniem ligatur. Z punktu widzenia algorytmu przenoszenia należy podzielić *ośmioznakowe* słowo `/z/ada/c`, i może ono zostać podzielone po *czwartym* znaku, tj. po fragmencie `/z/a`, gdyż ustawienie parametrów na taki podział zezwala.

A zatem używanie ligatur jako sposobu dostępu do znaków diakrytycznych prowadzi do nieprawidłowego i niejednorodnego traktowania długości słów przez algorytm przenoszenia. Parametry `\lefthyphenmin` i `\righthyphenmin` tracą swoje naturalne znaczenie. Oba rodzaje notacji, tj. *prefiksowa* i *postfiksowa* z tego punktu widzenia są nieprzydatne.

3.4 I co począć z ligaturami?

Argumenty wysunięte w punktach 3.1–3.3 wystarczyły nam, by zrezygnować z reprezentowania polskich znaków diakrytycznych za pomocą ligatur, choć listę zarzutów przeciw ligaturom można by jeszcze wydłużyć.

Na przykład wzorce dzielenia wyrazów (*hyphenation patterns*) w istotny sposób zależą od użytej notacji. Powoduje to sporo kłopotów, zwłaszcza w kontekście automatycznego przekształcania wzorców przy zmianie notacji.

Morałem z tej historii może być parafraza sformułowania Knutha [Knuth91]: *ligatury jako sposób dostępu do znaków narodowych są dopuszczalne przy następujących zastrzeżeniach: (1) albo użytkownik jest D. E. Knuthem, (2) albo nie czyni z tych ligatur żadnego użytku.*

Rzecz jasna nic nie stoi na przeszkodzie, by używać ligatur w ich naturalnym znaczeniu, tzn. do uzyskiwania pauz (myślników), cudzysłowów, ligatur zawierających literę 'f', itp.

4 Podsumowanie

Tak więc zdecydowaliśmy się na własne, inne od istniejących, podejście do narodowej adaptacji TEX-a. Problemy na jakie w trakcie pracy natrafiliśmy skutecznie nas zniechęciły do myślenia o uniwersalnej adaptacji wielojęzycznej.

Jest rzeczą znaną, że TEX został początkowo zaprojektowany jako system do pracy w jednym języku, przede wszystkim angielskim. Wersja 3.0 stanowi duży krok w stronę wielojęzyczności. Tym niemniej oryginalny TEX nie ma możliwości składania tekstów pionowo czy też od prawej do lewej; znaki akcentowane mają inny status niż znaki pojedyncze; instrukcja `\hyphenation` nie jest zbyt użyteczna dla języków fleksyjnych; sztywna granica 255 dla liczby tzw. operacji (*ops*)[6] jest niebezpiecznie niska; o ograniczeniach związanych z liczbą parametrów charakteryzujących font już mówiliśmy...

Argumenty można by mnożyć, ale nie w tym rzecz. Jest oczywiste, że pojęcie języka po prostu nie zostało wbudowane w system. Nieco myląca nazwa `\language` została wybrana dla instrukcji zmieniającej jedynie bieżący zestaw wzorców dzielenia wyrazów. Peter Breitenlohner zwraca uwagę, że języki austriacki i niemiecki mogą korzystać z tych samych wzorców, chociaż są między nimi istotne różnice [Breitenlohner90, str. 10]. Z drugiej strony jeśli zachodzi potrzeba użycia kilku fontów o różnym rozmieszczeniu znaków narodowych, pojawia się odwrotna sytuacja: kilka różnych zestawów wzorców TEX musi przechowywać w pamięci dla jednego języka (właśnie z taką sytuacją przyszło nam się zetknąć przy tworzeniu polskiej adaptacji TEX-a).

Podzielamy opinię F. Mittelbacha, że możliwe jest *ukierunkowanie przyszłych prac w taki sposób, by kiedyś się nie okazało, że wokoło jest mnóstwo różnych, niezgodnych ze sobą systemów „opartych na TEX-u”* [Mittelbach90, str. 337]. Listę rozszerzeń zaproponowanych przez Mittelbacha poszerzylibyśmy tylko o jeden punkt — o wielojęzyczność. W przeciwnym wypadku pojawienie się, i to raczej wcześniej niż później, owych wielojęzycznych „plus-minus-TEX-ów” wydaje się nieunikniONE.

Na koniec chcielibyśmy wyrazić nasze ogromne uznanie dla niezwykłej pracy wykonanej przez Knutha. Wydawać by się mogło, że stworzenie tak niezawodnego i uniwersalnego systemu jest wręcz niemożliwe. A jednak TEX istnieje i pomimo pewnych jego cech faworyzujących język angielski, dał się szczęśliwie przystosować do języka polskiego.

References

[Breitenlohner90] Peter Breitenlohner, “Using TEX 3 in a multilingual environement”, *TUGboat*, Vol. 11, No. 1, str. 9–12, 1990.

[Haralambous89] Yannis Haralambous, “TEX and latin alphabet languages”, *TUGboat*, Vol. 10, No. 3, str. 342–345, 1990.

[Jackowski et al. 88] Bogusław Jackowski, Tomasz Hołdys, Marek Ryćko, “With TEX to the Poles”, *Proceedings of the 3rd European TEX Conference*, Exeter 1988.

[Jackowski et al. 89] Bogusław Jackowski, Marek Ryćko, “Polish TEX is ready for use”, *Proceedings of the 4th European TEX Conference*, Karlsruhe 1989, (nie wydane drukiem).

[Knuth86a] Donald E. Knuth, “The METAFONTbook”, *Computers & Typesetting*, Vol. C, Addison Wesley, 1986.

[Knuth86b] Donald E. Knuth, “Computer Modern Typefaces”, *Computers & Typesetting*, Vol. E, Addison Wesley, 1986.

[Knuth90a] Donald E. Knuth, “The TEXbook”, *Computers & Typesetting*, Vol. A, Addison Wesley, 1990, eighteenth printing.

[Knuth90b] Donald E. Knuth, “Virtual Fonts: More Fun for Grand Wizards”, *TUGboat*, Vol. 11, No. 1, str. 13–23 1990.

[Knuth91] Donald E. Knuth, “`TEX.WEB` file, ver. 3.14”, marzec 1991.

[Mittelbach90] Frank Mittelbach, “E-TEX: Guidelines for future TEX”, *TUGboat*, Vol. 11, No. 3, str. 337-345, 1990.

[6] Jest to pojęcie związane ze sposobem zapamiętywania wzorców dzielenia wyrazów dla danego języka; informację w rodzaju

```
   Hyphenation trie of length 11344 has
375 ops out of 750
      194 for language 1
      181 for language 0
```

TEX podaje w trakcie tworzenia plików `FMT`.