Trochę za luźne \looseness

Marcin Woliński

Ostatnio łamałem tekst, w którym jedyną możliwością zapobiegnięcia powstawaniu szewców i bękartów było przeskładywanie akapitów w inną liczbę wierszy (gubienie lub dorabianie wierszy). Postanowiłem napisać program, który, uzyskawszy od TEXa informację o możliwych sposobach złożenia akapitów, ustali dla każdego z nich liczbę wierszy optymalną dla łamania i przekaże te dane z powrotem TEXowi (o takim automatycznym łamaniu opowiem dokładniej innym razem). Potrzebowałem zatem niezawodnego sposobu zadawania liczby wierszy dla akapitów formowanych przez TEXa.

Jak wiadomo, TEX ma stosowny mechanizm przeskładywania akapitów, steruje się nim za pomocą parametru o nazwie `\looseness`. Jak pisze Knuth w *The TEXbook*:

> Przy ustawieniu `\looseness=1` TEX będzie się starał wydłużyć akapit o jeden wiersz w stosunku do optymalnej liczby wierszy, o ile tylko da się to osiągnąć nie przekraczając zadanej tolerancji dla brzydoty poszczególnych wierszy. Podobnie jeśli `\looseness` wynosi 2, TEX postara się uzyskać o dwa wiersze więcej. Ustawienie `\looseness=-1` spowoduje próbę skrócenia akapitu. W ogólności, TEX najpierw składa akapit jak zwykle, a potem, jeśli liczba wierszy przy wyborze optymalnych punktów złamania wynosi n, a wartość parametru `\looseness` jest l, TEX wybiera punkty podziału tak, aby ostateczna liczba wierszy była tak bliska $n + l$, jak to tylko możliwe bez przekraczania tolerancji. Co więcej ostatecznie wybrane punkty łamania wierszy dają najmniejszą sumaryczną brzydotę, przy ustalonej liczbie wierszy.

Przeszkodą przy wykorzystaniu `\looseness` do łamania automatycznego jest to, że nie każdy akapit da się skrócić lub wydłużyć. Jednak nawet jeżeli umiemy przewidzieć, na ile wierszy da się podzielić każdy akapit, trafiają się pewne anomalie.

Na przykład poniżej przytoczono akapit pochodzący z *Anny Kareniny* Lwa Tołstoja, angielskie tłumaczenie C. Garnett, pobrane ze strony Projektu Gutenberg (`http://promo.net/pg/`). Stosujemy ustawienia właściwe dla Plain TEXa z następującymi zmianami:

```
\hsize=225pt
\parindent=10pt
\tolerance=300
```

Jest to szerokość łamu właściwa dla Biuletynu, wielkość wcięcia akapitowego wydaje się stosowna do niej. Wartość tolerancji jest większa niż w Plain TEXu, gdzie wynosi ona 200. Jest to uzasadnione przy dość wąskim łamie. W sumie nie dzieje się tu nic niezwykłego i konkretne wartości parametrów nie mają zasadniczego znaczenia w eksperymencie — dla trochę innych ustawień trzeba by trochę innego akapitu. Oto akapit testowy:

The first person he saw was Mademoiselle Linon. She walked across the room, and her ringlets and her face were beaming. He had only just spoken to her, when suddenly he heard the rustle of a skirt at the door, and Mademoiselle Linon vanished from Levin's eyes, and a joyful terror came over him at the nearness of his happiness. Mademoiselle Linon was in great haste, and leaving him, went out at the other door. Directly she had gone out, swift, swift light steps sounded on the parquet, and his bliss, his life, himself–what was best in himself, what he had so long sought and longed for–was quickly, so quickly approaching him. She did not walk, but seemed, by some unseen force, to float to him. He saw nothing but her clear, truthful eyes, frightened by the same bliss of love that flooded his heart. Those eyes were shining nearer and nearer, blinding him with their light of love. She stopped still close to him, touching him. Her hands rose and dropped onto his shoulders.

Jak widać, przy `\looseness=0` TEX składa ten akapit w 19 wierszy. Spróbujmy powtórzyć eksperyment ustawiając `\looseness=1`:

The first person he saw was Mademoiselle Linon. She walked across the room, and her ringlets and her face were beaming. He had only just spoken to her, when suddenly he heard the rustle of a skirt at the door, and Mademoiselle Linon vanished from Levin's eyes, and a joyful terror came over him at the nearness of his happiness. Mademoiselle Linon was in great haste, and leaving him, went out at the other door. Directly she had gone out, swift, swift light steps sounded on the parquet, and his bliss, his life, himself–what was best in himself, what he had so long sought and longed for–was quickly, so quickly approaching him. She did not walk, but seemed, by some unseen force, to float to him. He saw nothing but her clear, truthful eyes, frightened by the same bliss of love that flooded his heart. Those eyes were shining nearer and nearer, blinding him with their light of love. She stopped still close to him, touching him. Her hands rose and dropped onto his shoulders.

Staranne przeliczenie wierszy daje wynik 21. Wykonujemy w pamięci odejmowanie i stwierdzamy, że akapit został wydłużony o 2 wiersze. Coś się jakby nie zgadza.

Można wyrazić przypuszczenie, że ten akapit nie da się złamać w 20 wierszy i że TEX w związku z tym uznał, że 21 jest lepszym przybliżeniem liczby 20 niż 19. Tak jednak nie jest:

The first person he saw was Mademoiselle Linon. She walked across the room, and her ringlets and her face were beaming. He had only just spoken to her, when suddenly he heard the rustle of a skirt at the door, and Mademoiselle Linon vanished from Levin's eyes, and a joyful terror came over him at the nearness of his happiness. Mademoiselle Linon was in great haste, and leaving him, went out at the other door. Directly she had gone out, swift, swift light steps sounded on the parquet, and his bliss, his life, himself–what was best in himself, what he had so long sought and longed for–was quickly, so quickly approaching him. She did not walk, but seemed, by some unseen force, to float to him. He saw nothing but her clear, truthful eyes, frightened by the same bliss of love that flooded his heart. Those eyes were shining nearer and nearer, blinding him with their light of love. She stopped still close to him, touching him. Her hands rose and dropped onto his shoulders.

Żeby wyjaśnić parametry użyte w trzeciej próbie składu oraz pokazać, co się tu właściwie dzieje, musimy przyjrzeć się bliżej procesowi łamania akapitów.

Algorytm dzielenia akapitu na wiersze ma trzy fazy. TEX wykonuje je kolejno. Najpierw stara się połamać akapit bez dzielenia wyrazów. Jeżeli da się to zrobić tak, aby brzydota (*badness*) żadnego wiersza nie przekraczała wartości parametru `\pretolerance` (w Plain TEXu 100), proces się kończy. W fazie drugiej TEX uwzględnia wśród możliwych miejsc podziału akapitu dopuszczalne miejsca przeniesienia słów i ponawia próbę składu. Jeżeli da się osiągnąć brzydotę nieprzekraczającą `\tolerance`, proces się kończy. W przeciwnym wypadku TEX przechodzi do trzeciej fazy, która tym różni się od drugiej, że do sumarycznej rozciągliwości odstępów w każdym tworzonym wierszu jest dodawana wartość parametru `\emergencystretch`.

Pierwsza faza algorytmu jest pomijana, jeżeli `\pretolerance=-1`. Druga faza jest zawsze wykonywana. Trzecia faza jest pomijana, jeżeli `\emergencystretch=0pt`.

Jeżeli TEX utknie w ostatniej z wykonywanych faz (czyli będzie zmuszony odrzucić wszystkie możliwości uformowania wiersza), awaryjnie utworzy wiersz przepełniony (*overfull hbox*) lub niedopełniony (*underfull hbox*).

Powyższy opis odnosi się do sytuacji, gdy `\looseness=0`. Wartości niezerowe tego parametru powodują, że TEX oprócz warunków na brzydotę wierszy rozpatruje również liczbę wierszy tworzonego akapitu. Jeżeli więc na przykład `\looseness=1` i w pierwszej fazie da się połamać akapit, ale nie da się go wydłużyć o jeden wiersz, TEX odrzuca takie rozwiązanie i przechodzi do drugiej fazy. Jedynie w ostatniej TEX pogodzi się z faktem, że nie wypełnił wymagań na `\looseness`.

Niestety przytoczony na wstępie Knuthowski opis działania parametru `\looseness` jest mylący. Wydaje się bowiem, że wartość n jest to ta liczba wierszy, jaką uzyskuje się przy `\looseness=0`. Tymczasem optymalna liczba wierszy jest ustalana z osobna w każdej fazie działania algorytmu. Mamy więc w istocie wartości n_1, n_2 i n_3, które są optymalne dla łamania uzyskanego w pierwszej, drugiej i trzeciej fazie (o ile wszyskie fazy zostały wykonane i w każdej udało się znaleźć jakiś sposób złamania).

Powstaje pytanie, czy wartości te mogą być różne. Jak najbardziej. W przypadku akapitu testowego $n_1 = 19$ zaś $n_2 = 20$ (trzecia faza łamania nie jest wykonywana). Teraz możemy już wyjaśnić, że wersja akapitu testowego na bieżącej stronie została uzyskana przy parametrach

```
\looseness=0
\pretolerance=-1
```

Jest to zatem najlepsze łamanie akapitu spośród znalezionych w drugiej fazie algorytmu.

Łatwo widzieć, dlaczego możliwe jest, żeby pierwsza i druga faza składu akapitu dały różne wyniki. Dzielenie wyrazów przy przenoszeniu jest obarczone grzywną (*penalty*), więc z tego punktu widzenia lepiej jest nie przenosić. Z drugiej strony podzielenie wyrazów daje krótsze segmenty, więc może pozwolić na mniejsze rozciągnięcie lub ściśnięcie odstępów i w konsekwencji mniejszą brzydotę. Te dwa czynniki konkurują ze sobą. Może więc się zdarzyć, że skład bez dzielenia słów daje rozwiązanie akceptowalne, ale podzielenie jakichś słów pozwala go ulepszyć. W takim przypadku, jeżeli zmusi się TEXa do wejścia w drugą fazę łamania, uzyska się inny wynik. Może się również zdarzyć, że te dwa różne składy będą się różnić liczbą wierszy. To właśnie zdarzyło się w akapicie testowym.

Historia akapitu testowego jest więc taka: najpierw TEX bez dzielenia wyrazów uzyskał skład w 19 wierszach. Parametr `\looseness` wynosił 1, więc TEX spróbował uzyskać (nadal bez dzielenia wyrazów) 20 wierszy. To się nie udało, więc TEX przeszedł do drugiej próby. W tym przypadku skład naturalny miał długość 20 wierszy. Oczekiwana liczba wierszy wyniosła więc 21 i okazało się, że taką liczbę da się uzyskać przy zadanej wartości `\tolerance`.

Jak wspomniałem na początku, w zadaniu, z którym się zmagałem, potrzebowałem sposobu niezawodnego komunikowania TEXowi pożądanych długości akapitów. A przedstawiony tu akapit powoduje zasadniczy problem. Trudno bowiem spowodować, aby został on złamany w 20 wierszy. Nie da się tego zrobić przy pomocy samego parametru `\looseness`:

`\looseness`	l. wierszy
< 0	19
0	19
1	21
2	22
3	23
4	24
> 4	24

Uznałem, że najprostszym rozwiązaniem będzie ustawienie parametru `\pretolerance=-1`, czyli zmuszenie TEXa do pominięcia prób składu bez dzielenia wyrazów. Okazało się jednak, że to nie koniec problemu.

Oto kolejny przykład narowistego akapitu. Parametry składu jak poprzednio, z wyjątkiem ustawienia `\pretolerance=-1`, dzięki któremu wiemy, że TEX wykonuje łamanie akapitu w drugiej fazie algorytmu. Z pewnością nie interweniuje więc tutaj zjawisko opisane powyżej. Akapit testowy ma pięć wierszy, przy czym koniec trzeciego jest wymuszony poleceniami `\hfill\break` (w LATEXu to byłoby `\newline`):

aaa aab aac aad aae aaf aag aah aai aaj aak aal
aam aam aao aap aaq aar aas aat aau aav aaw aax
aay aaz aba abb abc abd abe abf abg abh abi abj
aaa aab aac aad aae aaf aag aah aai aaj aaiixxxxxxxx
aak aal aam aan aao

Przy ustawieniu `\looseness=1` TEX łamie ten akapit na sześć wierszy:

aaa aab aac aad aae aaf aag aah aai aaj aak aal
aam aam aao aap aaq aar aas aat aau aav aaw
aax aay aaz aba abb abc abd abe abf abg abh abi
abj
aaa aab aac aad aae aaf aag aah aai aaj aaiixxxxxxxx
aak aal aam aan aao

Nie byłoby w tym nic niezwykłego, gdyby nie to, że `\looseness=-1` daje to samo złamanie w sześć wierszy. Więcej: każda niezerowa wartość `\looseness`, dodatnia lub ujemna, daje tę drugą wersję łamania.

Dla tego akapitu dla dowolnej wartości parametru `\looseness` TEX potrafi znaleźć tylko jeden podział nie dający komunikatu o błędzie łamania. Sposób złamania jest przy tym inny w zależności od tego, czy `\looseness` jest zerowe, czy niezerowe. Spróbujmy rzucić odrobinę światła na to zjawisko.

Mimo, że TEX przy łamaniu tego akapitu nie podaje żadnego komunikatu diagnostycznego, wiersz kończący się słowem 'aaiixxxxxxxx' jest w istocie przepełniony. Wielkość tego przepełnienia wynosi 0.00056pt. TEX nie powiadamia o tym fakcie, ponieważ nie zwykł informować o przepełnieniach mniejszych od wartości parametu `\hfuzz`, który w Plain TEXu wynosi 0.1pt. Zatem akapit jest przetwarzany bez diagnostyki, ale w istocie są w nim podejmowane akcje awaryjne, jak w każdym akapicie z przepełnieniem.

Fragment akapitu przed wymuszonym łamaniem daje się podzielić na 3 lub 4 wiersze (jak widać w przykładach), przy czym lepszy jest podział na 3 wiersze.

TEX w trakcie przebiegu algorytmu wyznacza wszystkie dopuszczalne sposoby złamania początkowego fragmentu akapitu, stopniowo coraz dalej rozszerzając ten fragment. Gdy `\looseness=0` i TEX trafi na wymuszony koniec wiersza, zachowuje tylko najlepszy sposób złamania akapitu aż do tego miejsca, a o innych zapomina. W przypadku naszego akapitu, po natrafieniu na wymuszone łamanie, TEX pamięta tylko o podziale początkowego fragmentu na 3 wiersze.

Gdy `\looseness` jest niezerowe TEX musi pamiętać więcej informacji. Musi mianowicie dla każdego sposobu podziału początkowego fragmentu akapitu pamiętać, ile daje on wierszy w wyniku. Podziały tego samego fragmentu dające różną liczbę wierszy są uważane za różne. Tak więc gdy algorytm dotrze do polecenia `\break` w pamięci programu znajdzie się informacja o dwóch sposobach podzielenia początkowego fragmentu.

Dalej następuje przepełniony wiersz. Jest on tworzony, gdy algorytm odrzuci jako niemożliwe wszystkie sposoby podziału. Wtedy awaryjnie przywraca z niebytu któryś podział, żeby w ogóle wyprodukować jakiś wynik. Jego wybór jest arbitralny — jest to ten podział, który został znaleziony jako ostatni. TEX bowiem uważa wszystkie przepełnione wiersze za nieskończenie brzydkie i nie prowadzi ich rankingu.

W przypadku akapitu testowego dla niezerowego `\looseness` ostatnio znalezionym podziałem okazuje się być ten, w którym fragment przed wymuszonym łamaniem ma 4 wiersze. Dlatego w sumie TEX produkuje akapit o 6 wierszach.

Czytelnik zapewne widzi już, że w omawianym zjawisku odegrały rolę dość skomplikowane zależności między częściami algorytmu łamania akapitów. Pozostaje jeszcze wyjaśnić, dlaczego powodują one problemy.

Otóż sposobem na dowiedzenie się, jakie są wszystkie możliwe sposoby złamania danego akapitu (w szególności jakie liczby wierszy można uzyskać), może być przetworzenie tego akapitu przy dodatniej wartości parametru `\tracingparagraphs` i dużej dodatniej wartości parametru `\looseness`. Wówczas wynik tego przebiegu TEXa będzie zapewne bezużyteczny, ale w pliku `.log` znajdzie się potrzebna informacja. Niestety w przypadku omawianego akapitu w podany sposób nie da się ustalić, ile wierszy ma skład dla `\looseness=0`.

Jeden sposób uporania się z problemem (oprócz zmodyfikowania samego TEXa) to wyzerowanie parametru `\hfuzz`. Wówczas każde przepełnienie wiersza, niezależnie jak małe, będzie powodować stosowny komunikat diagnostyczny. Ten sposób jest jednak mało zadowalający, bo ignorowanie niewielkich przepełnień przy wąskim łamie zauważalnie ułatwia skład. Poniżej przedstawiam inną metodę zwalczenia problemu.

Jak wspominałem, różnica zachowania TEXa wynika z tego, że dla niezerowego `\looseness` TEX pamięta osobno punkty podziału różniące się jedynie numerem wiersza, który kończy się w danym miejscu. Należy więc spowodować, aby mimo że `\looseness=0`, TEX zachowywał się tak samo. W tym celu można zadać niestandardowy kształt akapitów przy pomocy parametru `\parshape` lub `\hangindent`. Wtedy bowiem długość wiersza (potencjalnie) zależy od jego numeru i TEXowi nie wolno utożsamić wierszy o różnych numerach. Pomysł polega więc na tym, aby uzyskać standardowy kształt akapitu narzędziami służącymi do uzyskiwania kształtów niestandardowych. Na przykład można zastosować poniższe wartości parametrów:

```
\hangindent1sp
\hangafter-10000
\leftskip-1sp
```

Sprawiają one, że pierwsze 10 000 wierszy akapitu jest wcięte o 1sp, ale to wcięcie jest skompensowane przez wartość `\leftskip`. Zatem, o ile akapit ma nie więcej niż 10 000 wierszy, jego kształt jest co do joty standardowy. TEX jednak sądzi inaczej i w związku z tym nie może się ujawnić problem opisany wyżej.

Można zadać pytanie, czy omawiane tu zjawiska występują w przyrodzie, a jeżeli tak, to czy wystarczająco często, żeby warto się nimi było przejmować.

Wśród 7 500 akapitów *Anny Kareniny* pierwsze zjawisko przy tolerancji 300 zdarza się jeden raz (w przytoczonym akapicie). Podobnie dzieje się przy tolerancji 200 i `\emergencystretch=4pt`. Przy tolerancji 500 zjawisko powtarza się pięciokrotnie.

W składzie 700-stronicowej książki, który wykonywałem (wąskie łamy, krótkie akapity), kilkakrotnie pojawiły się oba zjawiska. Procentowo to niewiele. Gdy jednak chodzi o realizację wyznaczonego automatycznie sposobu łamania dzieła wystarczy jeden akapit, który „nie posłucha” specyfikacji, aby łamanie na wszystkich dalszych stronach zupełnie się posypało.

Oczywiście to wszystko nie ma znaczenia dla łamania wykonywanego ręcznie. W takim bowiem przypadku łamacz nie ma gwarancji, że jakikolwiek konkretny akapit podda się operacji wydłużenia lub skrócenia i zawsze działa na zasadzie eksperymentu.

Podsumowując mogę stwierdzić, że próba automatycznego sterowania sposobem łamania akapitów z zewnątrz TEXa okazała się trudna, kosztowała wiele czasu i nerwów (wisiała przecież nade mną wizja uciążliwego ręcznego łamania 800 stron tekstu).

Zadanie byłoby znacznie łatwiejsze, gdyby można było bezpośrednio zadawać pożądaną liczbę wierszy dla danego akapitu. Należałoby rozważyć wprowadzenie takiej możliwości do programu eTEX. Z punktu widzenia programistycznego jest to kilka linijek kodu, a sądzę, że przed automatycznym łamaniem jest przyszłość. Ale to już inna historia. . .

Autor bardzo dziękuje za inspirującą dyskusję Bogusławowi «Jacko» Jackowskiemu, Frankowi Mittelbachowi i Piotrowi Strzelczykowi.

◇ Marcin Woliński
wolinski@mimuw.edu.pl