A note on \linepenalty

Udo Wermuth

Abstract

This article analyzes the effect of the line-breaking parameter \linepenalty. First, its rôle in the problem of typesetting a text in one line or in two lines is studied theoretically. Then the effect of different values for \linepenalty are demonstrated for longer paragraphs. Finally, \linepenalty is compared to \looseness.

1 Introduction

The line-breaking algorithm of \TeX selects a shortest path in a network of possible breakpoints ([2] or the reprint in [7], p. 107) using a cost function that calculates demerits. For every line, four values are involved in the computation of its demerits and then the sum of all line demerits stands for the total demerits of a paragraph. Three of the four values are directly related to the characteristics of the created line, while the fourth value is a constant for all lines of a paragraph: the \linepenalty. In \TeX{} this constant was hard-coded into the program but with \TeX{}82 it became an integer parameter that the user can change ([5], or the reprint in [6], pp. 273–274). The plain \TeX{} format sets the default value of \linepenalty.

The next section gives a brief overview of the rules by which \TeX{}’s line-breaking algorithm calculates the demerits. It also introduces some notation in accordance with [9] but as there are only a few symbols in this article the conventions are not repeated here. Section 3 analyzes what happens to a text, for example, a heading, that can be typeset in accordance with [9] but as there are only a few symbols in this article the conventions are not repeated here. Section 3 analyzes what happens to a text, for example, a heading, that can be typeset.

2 Calculation of demerits

Section 2 of [8] contains a detailed description of the rules that \TeX{} uses to compute the demerits. In order to introduce the notation for this article a brief summary of these rules follows.

The formula ([3], p. 98) that computes the demerits of a line, stated as $\Lambda$, is

$$\Lambda = (\lambda + \beta)^2 + \sgn(\pi)\pi^2 + \delta$$  \hspace{1cm} (1)

which names the four parameters with Greek letters.

$\lambda$ is the \linepenalty, a constant value set in the plain \TeX{} format to 10.

The badness assigned to the line is called $\beta$. The badness is itself the result of a computation which looks at the ratio of used and available stretch- or shrinkability in the line ([3], p. 97). It is a nonnegative number $\leq \pretolerance$ in the first pass; otherwise $\leq \tolerance$.

Depending on the type of line break a penalty $\pi$ is charged for the break ([3], p. 96). The value is squared but the sign is kept so that the line demerits are lowered when a negative penalty is given. Penalties lie in the range $-10000 < \pi \leq 10000$; $\pi = -10000$ forces a break but does not add to the line demerits. A break at glue gets $\pi = 0$; otherwise a break at a hyphen uses either \hyphenpenalty or \exhyphenpenalty, a break in math mode either \binoppenalty or \relpenalty, and a break at an explicit \penalty command uses the given value. The plain \TeX{} format sets the value of the four mentioned parameters to 50, 50, 700, and 500, respectively.

The last parameter is named the additional demerits $\delta$. Lines interact with their predecessors: If visually incompatible lines would be output or if two hyphens in a row occur or if the penultimate line of the paragraph ends with a hyphen then additional demerits are added. The term $\delta$ is the sum of the parameters for these three mentioned cases: \adjdemerits, \doublehyphendemerits, and only for the last line \finalhyphendemerits. The default values in plain \TeX{} are 10000, 10000, and 5000, respectively.

The Pascal code in [4, §859] shows that the first summand on the right hand side of equation (1) takes a minimum of two numbers and involves the absolute value of $\lambda + \beta$. Actually the summand is
The line demerits of a paragraph sum certain values that are associated with the path through the network of line breaks that TEX has identified as the shortest path according to its cost function. This calculation can be performed for any path in this network so the notation $\Lambda_p$ for path demerits is introduced.

If the value of \linepenalty is important it is given as an argument to $\Lambda_t$ or $\Lambda_p$, for example, the left hand side of (3) can be written as $\Lambda_t(\lambda)$.

The total demerits also have an upper limit. In [4, §833] this limit is coded and it is best to have

$$\Lambda_t < 2^{30} - 1 = 1,073,741,823$$

otherwise \TeX might output overfull lines although line breaks seem to be possible. \TeX does not stop working but except for the end of the paragraph it does not create useful feasible breakpoints that are required for \TeX’s line-breaking decisions [4, §835].

In order to get familiar with the notation a simple lemma is proved. (The symbol ‘$\triangleright$’ marks the end of a proof or of an example.)

**Lemma 1.** If \linepenalty $\geq 0$ and if for a line that has neither penalties nor additional demerits the line demerits are larger than the line demerits of a second line with a penalty $\geq 0$, additional demerits $\geq 0$, and a summand $\epsilon \geq 0$ then the badness of the first line is larger than the badness of the second.

**Proof:** With (1) the lemma claims for two lines with line demerits $\Lambda$ and $\Lambda'$ that with $\epsilon \geq 0$

$$\Lambda > \Lambda' + \epsilon \implies \beta > \beta'$$

if there are no penalties and additional demerits in $\Lambda$, i.e., $\pi = 0$ and $\delta = 0$, and if $\pi' \geq 0$ and $\delta' \geq 0$.

$$\Lambda > \Lambda' + \epsilon \iff (\beta + \lambda)^2 > (\beta' + \lambda)^2 + \text{sgn}(\pi')\pi'^2 + \delta' + \epsilon$$

by (1). The sum $\text{sgn}(\pi')\pi'^2 + \delta' + \epsilon$ is $\geq 0$ as $\pi' \geq 0$, $\delta' \geq 0$, and $\epsilon \geq 0$. It follows that

$$(\beta + \lambda)^2 - (\beta' + \lambda)^2 > \pi'^2 + \delta' + \epsilon$$

$$\implies (\beta - \beta')(\beta + \beta' + 2\lambda) > 0.$$
value. Without such a penalty a line with badness 0 has the line demerits \(\Lambda_1 = \lambda^2\) and a two-line solution must have at least this value for its second line alone: \(\Lambda'_2 \geq \lambda^2\). The first line adds the positive value \(\Lambda'_1\) to the path demerits as no negative penalties are involved, making a two-line solution worse than the one-line result. There is no other case for the glue of the single line as by Lemma 2 the glue cannot stretch.

But let’s look at the general case. The path demerits of the single line are computed as

\[
\Lambda_p = \Lambda_1 = (\lambda + \beta_1)^2
\]

because no penalty is added in a first pass; i.e., \(\pi_1 = 0\). \(\delta_1 = 0\) except if the line is very loose but that cannot happen by Lemma 2; so this summand can be dropped too.

For the two-line solution the calculation is

\[
\Lambda'_p = \Lambda'_1 + \Lambda'_2
\]

\[
= (\lambda + \beta'_1)^2 + sgn(\pi'_1)\pi'_1^2 + \delta'_1 + \lambda^2 + \delta'_2
\]

because \(\beta'_2 = \pi'_2 = 0\) as the second line must have badness 0 because of the assumptions about the line width and the \texttt{parfillskip}. At a break with a hyphen \(\pi'_1\) is either the value of \texttt{exhyphenpenalty} if the hyphen is part of the text or \texttt{hyphenpenalty} for user entered discretionary hyphens. In both cases the additional demerits of the second line, \(\delta'_2\), must contain the value of \texttt{finalhyphenemitters}, called \(\delta'_f\). And \(\delta'_1\) is either 0, or if this line is very loose \texttt{adjdemerits}, named \(\delta_a\). In this case \(\delta'_2\) contains \(\delta_a\) too. A break in math or at an explicit \texttt{penalty} does not influence the additional demerits.

\TeX \ will break the text into two lines if and only if \(\Lambda_1 = \Lambda'_p \leq \Lambda_p\).

Case 1: No penalty. This means \(\pi'_1 = 0\) and \(\delta'_2\) does not contain \(\delta_f\); thus (6) simplifies to

\[
\Lambda'_p = (\lambda + \beta'_1)^2 + \delta'_1 + \lambda^2 + \delta'_2.
\]

A path that generates two lines is preferred by \TeX \ if the right hand side of (7) is smaller than the right hand side of (5):

\[
(\lambda + \beta_1)^2 > (\lambda + \beta'_1)^2 + \delta'_1 + \lambda^2 + \delta'_2.
\]

To make this inequality true \(\beta'_1\) must be smaller than \(\beta_1\) by Lemma 1; the difference is called the “change” \(\chi\) of badness for the two-line solution, i.e., \(\beta_1 - \chi = \beta'_1\) with \(\chi > 0\). As the badness \(\beta'_1\) is greater than or equal to 0 one more inequality is known

\[
\beta_1 \geq \chi.
\]

All solutions must have a badness of the single line that lies on or above the identity function \(g(\chi) = \chi\).

Now \(\beta'_1 < 100\) so the first line of the pair is not very loose, thus \(\delta'_1 = \delta'_2 = 0\) and inequality (8)

\[
2\chi^2 > (\lambda - \chi)^2
\]

becomes with \(\beta'_1 = \beta_1 - \chi\)

\[
(\lambda + \beta_1)^2 > (\lambda + \beta_1 - \chi)^2 + \lambda^2
\]

\[
\iff
2\beta_1 \chi > (\lambda - \chi)^2.
\]

As \(\chi > 0\), inequality (11) can be written as

\[
\beta_1 > \frac{(\lambda - \chi)^2}{2\chi}.
\]

With (9) the left hand side of (11) is kept equal or made smaller when \(\beta_1\) is replaced by \(\chi\). If this new inequality holds then (12) holds too.

\[
2\chi^2 > (\lambda - \chi)^2
\]

\[
\iff
\sqrt{2}\chi > \lambda - \chi
\]

\[
\iff
\chi > (\sqrt{2} - 1)\lambda
\]

The right-side inequality doesn’t say anything new, as \(\chi > 0\). If \(\lambda = 0\) both inequalities state \(\chi > 0\) so that only (12) counts.

This computation proves the following theorem.

**Theorem 1.** In plain \TeX \ with \texttt{linepenalty} \(\geq 0\) a text that fits into one line is typeset in two lines containing a line break without penalties if the difference between the badness of the single line and the badness of the first line of the pair is larger than

\[
(\sqrt{2} - 1)\texttt{linepenalty}
\]

or this difference, named “change”, is larger than zero and the badness of the single line is larger than

\[
(\texttt{linepenalty} - \text{change})^2/2 \times \text{change}.
\]

When plain \TeX \’s settings are used the value 10 can be plugged in for \texttt{linepenalty}. Thus the identity function is used as lower limit for the badness when the change is larger than 4 as \((\sqrt{2} - 1) \times 10 \approx 4\).
4.14. For $(1, 2, 3, 4)$ the badness of the single line must be larger than or equal to $17$, respectively; see (12). Figure 1 shows the graphs of two functions for the integer values from 1 to 100. First, the identity is shown as $g$. The second function $f$ represents in essence the formula of the right hand side of (12). If the badness of the single line lies on or above the thick points for a given $\chi$ then two lines are typeset. The gray area forms the solution set.

Of course, the \linepenalty can be changed. Figure 2 shows the solution sets for three different values of \linepenalty. For $\lambda = 10$ all three gray areas count (compare the areas to the solution set shown in Fig. 1), for $\lambda = 100$ the light-colored area is excluded, and for $\lambda = 200$ only the dark area builds the solution set.

**Case 2a: Break at hyphen.** This important special case of a break with penalties is treated first. The break must be either at an explicit hyphen in the text or at an inserted discretionary break as the network is built from the first pass of \TeX’s line-breaking algorithm. This means that $\pi_1^j$ equals either \exhyphenpenalty or \hyphenpenalty and as explained above there are additional demerits $\delta_f = \finalhyphenpenalty = \delta_f$.

Now (10) and thus (12) get additional constant terms on their right hand sides; (12) becomes:

$$\beta_1 > \frac{(\lambda - \chi)^2 + \text{sgn}(\pi_1^j)\pi_1^j + \delta_f}{2\chi}.$$

As in plain \TeX $\pi_1^j = 50$ and $\delta_f = 5000$, the sum $\text{sgn}(\pi_1^j)\pi_1^j + \delta_f$ is 7500. A graph for the above inequality similar to Fig. 1 is shown in Fig. 3. The change must be at least 43 to get two lines. With $\beta_1 = 78$ the change must be larger than 75. The comparison of Figs. 1 and 3 shows that in essence the point from which the identity function dominates the other function is moved on this line to a higher value. (The same effect occurs in Fig. 2.)

**Case 2b: Break at positive penalty.** Lemma 1 is applicable. So starting in (10) with $\epsilon \geq 0$, which is the sum of the penalty of the first line of the pair and the additional demerits of both lines added to the right hand side, the equivalent of (12) is

$$\beta_1 > \frac{(\lambda - \chi)^2 + \epsilon}{2\chi}.$$

Similarly, starting with inequality (11) and replacing $\beta_1$ by $\chi$ gives

$$2\chi^2 > (\lambda - \chi)^2 + \epsilon \iff (\lambda + \chi)^2 > 2\lambda^2 + \epsilon \iff \chi > \sqrt{2\lambda^2 + \epsilon - \lambda} \lor \chi < -\sqrt{2\lambda^2 + \epsilon - \lambda}.$$

Obviously the second inequality is not relevant.

Thus a generalization of Theorem 1 is proved:

**Theorem 2.** Given a text that fits into one line or can be typeset in two lines with a line break that has the value $\epsilon \geq 0$ as the sum of penalties and additional demerits. Let $\linepenalty \geq 0$.

The two-line solution is used by plain \TeX if the change $> 0$ is either at least

$$\sqrt{2\linepenalty + \epsilon - \linepenalty}$$

or the badness of the single line is larger than

$$(\text{linepenalty} - \text{change})^2 + \epsilon)/(2 \times \text{change}).$$

A note on \linepenalty
Theorem 2 mentions two limits, and the formula (13) gives for the three penalties 10, 50, and 90, i.e., the three numbers 10², 50², and 90²:

\[ \lambda = 10 \implies \frac{\sqrt{200 + \epsilon}}{10} \approx 7.3 \quad 41.9 \quad 81.1 \]

\[ \lambda = 100 \implies \frac{\sqrt{200000 + \epsilon}}{100} \approx 41.7 \quad 50 \quad 67.6 \]

Thus the values of the limit get larger for \( \epsilon = 10² \) and \( \epsilon = 50² \) when \( \lambda \) is changed to 100 but for \( \epsilon = 90² \), it is smaller! Its solution set is not a subset of the solution set when \( \lambda = 10 \). Figure 5 shows the solution sets for the three values of \( \epsilon \) with \( \lambda = 100 \).

**Case 2c: Negative penalties.** In this case, the inequality (8) is changed to

\[ (\lambda + \beta_1)^2 > (\lambda + \beta'_1)^2 + \delta'_1 + \lambda^2 - \epsilon \]  
(13)

in which \( \epsilon > 0 \) stands for the sum of penalties of the first line and additional demerits of the second line as in case 2b. That means \( \delta'_1 \) is contained in \( \epsilon \) and thus not mentioned in (13). \( \delta'_1 = 0 \) except the first line of the two-line solution, named \( L'_1 \), is very loose. Then \( \delta'_1 \) is the value of \[ \text{adjdemerits} = \delta_a \]. The bracket notation is used to identify this summand: \( \delta_a[L'_1 \text{ very loose}] \). And this means the additional demerits of the second line contains \( \delta_a \) too. But \( \epsilon \) is not changed; instead the term \( \delta_a \) is added twice.

Lemma 1 is not applicable and \( \beta'_1 \) can be larger than \( \beta_1 \). For some “change” \( \chi \), \( -100 \leq \chi \leq 100 \), let \( \beta_1 - \chi = \beta'_1 \). In other words: Instead of (9), now two limits for the badness of the single line are required:

\[ \beta_1 \geq \chi, \quad \text{if } \chi > 0; \] 
\[ \beta_1 \leq 100 + \chi, \quad \text{if } \chi < 0. \]  
(14a, b)

Inequality (13) becomes

\[ (\lambda + \beta_1)^2 > (\lambda + \beta_1 - \chi)^2 + 2\delta_a[L'_1 \text{ very loose}] + \lambda^2 - \epsilon \]

or after the usual transformations

\[ 2\beta_1 \chi > (\chi - \lambda)^2 + 2\delta_a[L'_1 \text{ very loose}] - \epsilon. \]

If \( \chi = 0 \) this inequality states that \( \epsilon \) must be larger than \( \lambda^2 + 2\delta_a[L'_1 \text{ very loose}] \), i.e., the three solution sets for the three values of \( \chi \) are required:

\[ \chi = 100 = \lambda \]

\( \chi = 10 \)

\( \chi = 0 \)

As before \( \chi \) is used to replace \( \beta_1 \) with (14a) to get

\[ 2\chi^2 > \chi^2 - 2\lambda \chi + \lambda^2 - \epsilon \]

\( \iff \)

\[ \chi > \frac{(\chi - \lambda)^2 - \epsilon}{2\chi}. \]  
(15)

Starting from

\[ 2\beta_1 \chi > (\chi - \lambda)^2 + 2\delta_a[L'_1 \text{ very loose}] - \epsilon \]

Figure 4: Solution sets for three different penalties

Figure 4 compares the solution sets for three different \( \epsilon \) similar to Fig. 2.

When the \texttt{linepenalty} is increased to 100 the solution set shrinks as shown in Fig. 2. But Theorem 2 mentions two limits, and the formula \( \sqrt{2\lambda^2 + \epsilon - \lambda} \) gives for the three penalties 10, 50, and 90, i.e., the three \( \epsilon \) amounts 10², 50², and 90²:

Figure 5: Like Fig. 4 for larger \texttt{linepenalty}

Udo Wermuth
the diagram shows the solution set for

\[ 2(100 + \chi)\chi > (\chi - \lambda)^2 + 2\delta_{n}[L'_{1\text{ very loose}}] - \epsilon. \]

With the usual transformations this leads to the relevant solution

\[ \chi > \sqrt{(100 + \lambda)^2 + \lambda^2 + 2\delta_{n}[L'_{1\text{ very loose}}]} - \epsilon - 100 - \lambda. \]

Thus a somewhat complex third theorem is proved:

**Theorem 3.** Given a text that fits into one line or can be typeset in two lines with a line break that has the value \(-\epsilon < 0\) as the sum of penalties and additional demerits except for \texttt{adjdemerits} if the first line is very loose. Let \texttt{linepenalty} \(\geq 0\).

If the change is 0 then there are two cases: If the first line of the pair is very loose \(\epsilon\) must be larger than \(\texttt{linepenalty}^2 + 2\texttt{adjdemerits}; otherwise \(\epsilon > \texttt{linepenalty}^2\) is sufficient.

If the change is \(\epsilon > 0\) then \(\epsilon \geq 2\texttt{linepenalty}^2\) typeset two lines; otherwise if \(\epsilon < 2\texttt{linepenalty}^2\) then either the change must be larger than

\[ \sqrt{2\texttt{linepenalty}^2 - \epsilon - \texttt{linepenalty}} \]

or the badness of the single line is larger than

\[ \frac{(\text{change} - \texttt{linepenalty})^2 - \epsilon}{2 \times \text{change}} \]

(\(\ast\)) to output two lines.

If the change is smaller than 0 but the first line is not very loose and \(\epsilon \geq (100 + \lambda)^2 + \lambda^2\) then two lines are created. Otherwise if \(\epsilon\) is smaller then either the change must be larger than

\[ \sqrt{(100 + \texttt{linepenalty})^2 + \texttt{linepenalty}^2 - \epsilon - 100 - \texttt{linepenalty}} \]

or the badness of the single line must be smaller than \(\ast\) to output two lines.

If the first line of the pair is very loose then two lines are typeset if either the change is smaller than

\[ \sqrt{(100 + \texttt{linepenalty})^2 + \texttt{linepenalty}^2 - \epsilon - 100 - \texttt{linepenalty}} \]

and \(\ast\) + \texttt{adjdemerits}/\text{change} is larger than the badness of the single line or \(\epsilon\) is larger than

\[ (100 + \texttt{linepenalty})^2 + \texttt{linepenalty}^2 + 2\texttt{adjdemerits}. \]

Figure 6 shows in the style of previous figures three instances of negative penalties. All gray areas represent the solution set if \(\epsilon = 159^2\). The dots show the limit when the first line of the pair is very loose. If the dots and the lightest gray area are excluded the diagram shows the solution set for \(\epsilon = 68^2\). It doesn’t capture all negative change but the smallest positive values build the identity function. If \(\epsilon = 5^2\) only the darkest area counts and even for positive change the identity function is not achieved for small values.

Setting \texttt{linepenalty} = 100 makes all areas smaller, the dots disappear, and for \(\epsilon = 159^2\) the left edge of the solution set drops from \((-43, 57)\) to \((-59, 0)\).

### 4 A few consequences of the theorems

The developed theory helps to understand certain cases for plain \TeX\ when a single line can be broken.

**Theorem 1** shows how the two-line solution can be made more likely when no penalties are involved: Reduce the value of \texttt{linepenalty!} The assignment 2 requires a change that must be larger than \(2(\sqrt{2} - 1) < 1\) for all badness values, i.e., a change of 1 or more typesets the two lines.

Theorem 1 also states that with a large value of \texttt{linepenalty!} \TeX\ will typeset a single line, for example, with a value 242 the line break is impossible as the change must be larger than \(242(\sqrt{2} - 1) > 100\).

**Theorem 2** makes, among other things, statements about penalties. It proves that a penalty of 110 forces the single line as \((100)^2 + 110^2)/(2 \times 100) > 100\).

**Theorem 3** implies that a \texttt{penalty} of \(-180\) typesets always two lines even if the first line of this pair is very loose and \texttt{adjdemerits} are involved.

Larger \texttt{linepenalty} might break a single line. One interesting consequence of Theorem 2 is that a
larger \linepenalty in combination with a positive \penalty breaks a line that would be kept as a single line if the default value of \linepenalty is used. See the discussion after Theorem 2 comparing Figs. 4 and 5.

**Example 1:** Description
A single line is broken when \linepenalty is increased.

**TEX input**
\toks0={\noindent It's a surprise, but it's true. See for yourself now. So it\penalty95 is.}
\linepenalty=10 \the\toks0\par
\linepenalty=100 \the\toks0\par

**TeX output**
It’s a surprise, but it’s true. See for yourself now. So it is. It’s a surprise, but it’s true. See for yourself now. So it is.

The single line has badness 86 and without the “is.” the badness drops to 0, that is, the first line of the two-line solution produces a change of 86. Using the formula of Theorem 2, once the values 10 and 95² and 100 and 95² are used for \linepenalty and for $\epsilon$, respectively, the results are that the change must be larger than 86 in the first case and larger than 70 in the second to create two lines. Thus, in the first case the break is avoided and in the second it is made.

Instead of an explicit penalty a hyphen can be the reason for a line break:

**Example 1 continued: TeX input**
\toks0={Bob, tell us, what made you want to look up \texttt{run-in}?
\noindent Do! not! do! it! Never! No! This \texttt{parshape} is bad. Do! not! do! it! Never! No! This \texttt{parshape} is bad.

Of course, the theory could be extended, but it does not seem worth the effort. Such settings of \texttt{parshape} are never applied to normal text. The author wants to generate a certain effect and controls the situation.

**Longer paragraphs.** The theoretical results do not apply without change to paragraphs with more than one line because the penultimate line in a long paragraph might be changed too when the last line is broken, i.e., $(L_{\mu-1}, L_{\mu}) \rightarrow (L'_{\mu-1}, L'_{\mu}, L'_{\mu+1})$ with $L_{\mu-1} \neq L'_{\mu-1}$. And even if it stays unchanged, i.e., $L_{\mu-1} = L'_{\mu-1}$, the line characteristic might influence the next line through additional demerits in different ways. Finally, the paragraph might be broken in the second pass of \TeX’s line-breaking algorithm, thus the badness of $L'_{\mu}$ might be larger than the badness of $L_{\mu}$ even if the change is positive.

OK, enough warning notices: There are nevertheless cases in which the theory is applicable to longer paragraphs.

**Example 3:** Description
Typeset a short text twice with plain \TeX. First with the default settings, next with \texttt{\linepenalty=2}.

**\TeX output**
When you start to count where do you start? With zero or with one? Hmm, I start at 1! A CS nerd uses a 0, or?

Later in Section 5 it is shown that the value 4 for \texttt{\linepenalty} is sufficient. It turns out that the value $-2$ works in this case too; see Section 6.

Next the technique with the penalty of 110 is used. The example also demonstrates that a large \texttt{\linepenalty} does not break the last line if the theory is applicable. Here the minimal required value for \texttt{\linepenalty} is used.

**Example 4:** Description
Typeset a short text thrice with plain \TeX. First with plain \TeX’s default settings, second with a \texttt{\penalty110} inserted between the last two words and a third time without this penalty but with \texttt{\linepenalty=199}.

**\TeX definitions**
\toks0={\noindent This text can be typeset, yes, either in two or in three lines and the theory of this section applies to the}
\toks1={\the\toks0} last line.}
\toks2={\the\toks0} last\penalty110 line.}

Udo Wermuth
This text can be typeset, yes, either in two or in three lines and the theory of this section applies to the last line.

This text can be typeset, yes, either in two or in three lines and the theory of this section applies to the last line.

This text can be typeset, yes, either in two or in three lines and the theory of this section applies to the last line.

As expected the \penalty110 prevents the line break. A tie would do the job too but if the text later grows, a break at the \penalty is still possible.

On the other hand if the text is typeset twice in one paragraph the theory is not applicable; with a line break in the last word the phrase “three” is moved from the penultimate line of the four-line paragraph to the penultimate line of the five-line paragraph.

Example 5: Description
Typeset the paragraph of example 4 two times as one paragraph: once with plain \TeX’s defaults and once with \linepenalty = 385.

In order to bring this paragraph to four lines \linepenalty must be set to 385.

5 Changing the value of \linepenalty
The parameter \linepenalty can be changed by the user. In this section an analysis is made when a different \linepenalty results in different line breaks and what the trade-offs are.

Example 6: Description
Typeset a paragraph several times with different values for \linepenalty. Start with \TeX’s default settings.

\TeX definitions
\linepenalty=10

\TeX output
The line-breaking algorithm of \TeX selects a shortest path in a network of feasible breakpoints using a cost function that calculates demerits. For every line four values are used to compute the demerits for this line and then the sum of all line demerits stands for the total demerits of a paragraph.

The line-breaking decisions of \TeX are listed in the log file if \tracingparagraphs is set to 1. This output helps to explain the effect on the line breaks when \linepenalty is changed; therefore the trace data is shown. See The \TeXbook [3], pp. 98–99, or [8], Section 3, for a description of this data.

Example 6 continued: \tracingparagraphs’ data
1. @firstpass
2. @secondpass
3. []\inerm The line-breaking al-go-rithm of \TeX se-lects a short-
4. @ discretionary via \000 b=5 p=50 d=2725
5. \001: line 1.2- t=2725 \rightarrow \000
6. est path in a net-work of fea-si-ble break-
7. @ via \001 b=82 p=0 d=8464
8. \002: line 2.1 t=11189 \rightarrow \001
9. cost
10. @ via \001 b=20 p=0 d=900
11. \003: line 2.3 t=3625 \rightarrow \001
12. func-tion that cal-cu-lates \ninesl de-mer-its/\inerm . For ev-ery line
13. @ via \002 b=48 p=0 d=3636
14. \004: line 3.1 t=14553 \rightarrow \002
15. four
16. @ via \003 b=45 p=0 d=13025
17. \005: line 3.1 t=16650 \rightarrow \003
18. val-
19. @ discretionary via \003 b=83 p=50 d=11149
20. \006: line 3.3- t=14774 \rightarrow \003
21. ues are used to com-pute the de-mer-its for
22. @ via \004 b=39 p=0 d=2401
23. \007: line 4.1 t=16954 \rightarrow \004
24. line
25. @ via \004 b=36 p=0 d=12116
26. \005: line 4.1 t=16650 \rightarrow \003
27. val-
28. @ discretionary via \003 b=83 p=50 d=11149
29. \006: line 3.3- t=14774 \rightarrow \003
30. and
31. @ via \005 b=20 p=0 d=10900
32. \006: line 4.2 t=14999 \rightarrow \006
33. then the sum of all line de-mer-its stands
34. @ via \007 b=84 p=0 d=8836
35. \0011: line 5.1 t=25790 \rightarrow \007
36. the
37. @ via \007 b=0 p=0 d=100
38. \008: line 5.1 t=25790 \rightarrow \007
39. \009: line 5.2 t=17054 \rightarrow \007
40. to-
41. @ discretionary via \008 b=0 p=50 d=2600
42. \0011: line 5.2 t=25790 \rightarrow \007
43. \009: line 5.2 t=17054 \rightarrow \007
44. @ discretionary via \009 b=0 p=50 d=2600

A note on \linepenalty
Table 1: Badness, penalties, and additional demerits of the line breaks for the six paths of the trace listing

<table>
<thead>
<tr>
<th>Path</th>
<th>Badness</th>
<th>Penalty</th>
<th>Adj. Demerits</th>
<th>Line</th>
<th>Badness</th>
<th>Penalty</th>
<th>Adj. Demerits</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>550</td>
<td>550</td>
<td></td>
<td>1</td>
<td>550</td>
<td>550</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>82</td>
<td>82</td>
<td></td>
<td>3</td>
<td>20</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>48</td>
<td>48</td>
<td></td>
<td>5</td>
<td>45</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>8350</td>
<td>8350</td>
<td></td>
<td>8</td>
<td>63</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>36</td>
<td></td>
<td></td>
<td>10</td>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>84</td>
<td></td>
<td></td>
<td>12</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>050</td>
<td></td>
<td></td>
<td>14</td>
<td>46</td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>15</td>
<td></td>
<td></td>
<td>16</td>
<td>350</td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td>0f</td>
<td>0f</td>
</tr>
</tbody>
</table>

\( \mu = 6 \quad 6 \quad 6 \quad 6 \quad 6 \quad 6 \)

\[ \Lambda_p(10) = 25890 \quad 17154 \quad 29679 \quad 18235 \quad 27394 \quad 22768 \]

The last three rows state the number of lines, \( \mu \), the sum of the badness values of the path, \( B \), and the path demerits \( \Lambda_p \). These values are not found directly in the trace data. They have been computed from the information in the columns.

The theory. An increase of \( \lambda \) by \( \kappa > 0 \) changes the first summand of the formula (1) for the line demerits

\[ (\lambda + \kappa + \beta)^2 = (\lambda + \beta)^2 + 2\beta\kappa + 2\lambda\kappa + \kappa^2. \]

The two summands \( 2\lambda\kappa \) and \( \kappa^2 \) form a “constant” that is added to every line and therefore they do not change the line-breaking decisions by \( \TeX \) — as long as the limits (2) and (4) are obeyed. The third summand \( 2\beta\kappa \) increases the influence of the badness \( \beta \). That means, the penalties and the additional demerits in (1) are less important: If \( \kappa \) is large enough \( \TeX \) selects a path for which more penalties or additional demerits are charged if only the badness values can be made smaller.
In fact, the increment that is necessary to go from one path to another can be calculated with (1) and (3). The path demerits become

\[
\Lambda_p(\lambda + \kappa) = \sum_{i=1}^{\mu} ((\lambda + \kappa + \beta_i)^2 + \text{sgn}(\pi_i)\pi_i^2 + \delta_i) \\
= \mu(\lambda + \kappa)^2 + 2\kappa B + \Lambda_p(\lambda) - \mu \lambda^2.
\]

The task is to determine \( \kappa > 0 \) to change the total line demerits to a path with a lower sum of badness — this path gets all subscripted variables and their sums primed. Therefore starting with \( B' < B \) and \( \Lambda_1(\lambda) = \Lambda_p(\lambda) < \Lambda_p'(\lambda) \) find \( \kappa > 0 \) such that

\[
\Lambda_p(\lambda + \kappa) = \Lambda_p'(\lambda + \kappa) = \Lambda_1(\lambda + \kappa).
\]

In the longer form the inequality is

\[
\mu(\lambda + \kappa)^2 + 2\kappa B + \Lambda_p(\lambda) - \mu \lambda^2 > \\
\mu'(\lambda + \kappa)^2 + 2\kappa B' + \Lambda_p'(\lambda) - \mu' \lambda^2.
\]

A few simple transformations when \( \mu = \mu' \) give

\[
2\kappa(B - B') > \Lambda_p'(\lambda) - \Lambda_p(\lambda)
\]
or as \( B > B' \)

\[
\kappa > \frac{\Lambda_p'(\lambda) - \Lambda_p(\lambda)}{2(B - B')}. \tag{17}
\]

**Its application.** Table 1 shows that there are three paths with a lower sum of badness value than column 12o: 14o with sum 159, 13o with sum 133, and 16o with sum 116. Inequality (17) states the following conditions for \( \kappa \):

Path (Column) 13o 14o 16o
\( \kappa > 152 \) 36 48

(Note, only integer parts of the numbers are shown.)

So \( \kappa = 37 \) (or \( \lambda = 47 \)) typesets the path of column 14o. The path of column 13o cannot be reached: 16o has a lower sum of badness and needs a lower \( \kappa \).

Example 6 continued: \TeX{} definitions

\TeX{} output

The line-breaking algorithm of \TeX{} selects a shortest path in a network of feasible breakpoints using a cost function that calculates demerits. For every line four values are used to compute the demerits for this line and then the sum of all line demerits stands for the total demerits of a paragraph.

All paths with a smaller sum of badness have been used. But these are not all possible paths as \TeX{} ignores a path that cannot become the shortest. Except \linepenalty \neq 10 might change \TeX{}’s viewpoint but unfortunately the path is not shown explicitly in the available trace. In total there are six more paths hidden in the data; Table 1 lists them. The paths are still named by the par information and now the subscript identifies the variant. The notations (10) and (t) mean that the corresponding \&-line does not have its own \&-line in the trace and \&10 follows next. With this data all values of \( \kappa \) for the transitions to a path with a smaller sum of badness can be computed if the paths are sorted by their sum of badness values B:

\begin{align*}
16o & 16o 13o 14o 15o 14o 13o 12o \\
16o & 0 \\
16o & 272 0 \\
13o & 70 -204 0 \\
14o & 52 -201 -185 0 \\
15o & 23 -156 -101 -80 0 \\
14o & 129 52 220 272 657 0 \\
13o & 6 -106 -62 -52 -37 -673 0 \\
12o & 112 48 152 179 299 36 2869 0 
\end{align*}

As noted above 36 is the smallest number in the last row, selecting path 14o. In the row for 14o 52 is the smallest number selecting 16o, and in its row 272 is selecting 16o. So only one more path can be shown for \( \kappa > 272 \), i.e., \( \kappa \) must be 273 and \( \lambda = \kappa + 10 = 283 \).

Example 6 continued: \TeX{} definitions

\TeX{} output

The line-breaking algorithm of \TeX{} selects a shortest path in a network of feasible breakpoints using a cost function that calculates demerits. For every line four values are used to compute the demerits for this line and then the sum of all line demerits stands for the total demerits of a paragraph.

Negative values. A negative amount for the integer \linepenalty does not act directly as a bonus if a line is created, as the sum with the badness is squared in equation (1). But a negative value reverses the meaning of badness! For example, a value
Table 1': Badness, penalties, and additional demerits
of the line breaks for not-shown paths of the trace listing

<table>
<thead>
<tr>
<th>linepenalty</th>
<th>\par via</th>
<th>µ</th>
<th>B</th>
<th>\textbackslash p(10)</th>
</tr>
</thead>
<tbody>
<tr>
<td>\textbackslash linepenalty</td>
<td>\textbackslash linepenalty \textbackslash linepenalty</td>
<td>\textbackslash linepenalty</td>
<td>\textbackslash linepenalty</td>
<td>\textbackslash linepenalty</td>
</tr>
<tr>
<td>*6</td>
<td>\textbackslash linepenalty</td>
<td>\textbackslash linepenalty</td>
<td>\textbackslash linepenalty</td>
<td>\textbackslash linepenalty</td>
</tr>
<tr>
<td>1</td>
<td>v</td>
<td>68</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>d</td>
<td>050</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>t</td>
<td>68</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>d</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>d</td>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>l</td>
<td>32</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>d</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>d</td>
<td>0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The matrix states that from row 11, the value \( \kappa = -672 \) moves on to path 12.2.

Example 6 continued: \TeX definitions
\textbackslash linepenalty=-662

\TeX output

The line-breaking algorithm of \TeX selects a shortest path in a network of feasible breakpoints using a cost function that calculates demerits. For every line four values are used to compute the demerits for this line and then the sum of all line demerits stands for the total demerits of a paragraph.

The limit. The \textbackslash linepenalty has a limit as stated in (2). The setting of 10000 is larger than this limit and the badness values are completely ignored—

Table 2: Badness, penalties, and additional demerits
of the line breaks for the two paths of example 7

<table>
<thead>
<tr>
<th>linepenalty</th>
<th>\par via</th>
<th>µ</th>
<th>B</th>
<th>\textbackslash p(10)</th>
</tr>
</thead>
<tbody>
<tr>
<td>\textbackslash linepenalty</td>
<td>\textbackslash linepenalty \textbackslash linepenalty</td>
<td>\textbackslash linepenalty</td>
<td>\textbackslash linepenalty</td>
<td>\textbackslash linepenalty</td>
</tr>
<tr>
<td>*6</td>
<td>\textbackslash linepenalty</td>
<td>\textbackslash linepenalty</td>
<td>\textbackslash linepenalty</td>
<td>\textbackslash linepenalty</td>
</tr>
<tr>
<td>1</td>
<td>v</td>
<td>68</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>d</td>
<td>050</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>t</td>
<td>68</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>d</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>d</td>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>l</td>
<td>32</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>d</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>d</td>
<td>0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Example 7: Description

Typeset a paragraph three times in a forced second pass, i.e., \textbackslash pretolerance = -1: first with \textbackslash linepenalty = 10, second with \textbackslash linepenalty = -14, and third with \textbackslash linepenalty = 10000.

\TeX output

Hi! \TeX! Tell me: How is the following long word broken ‘pneumonoultramicroscopicsilicovolcanoconiosis’? I am sure that you are an expert in hyphenation, right \TeX? Or shall I ask Siri?

Udo Wermuth
Hi! \TeX! Tell me: How is the following long word broken ‘pneumonoultramicroscopicsilicovolcanoconiosis’? I am sure that you are an expert in hyphenation, right \TeX? Or shall I ask Siri?

Hi! \TeX! Tell me: How is the following long word broken ‘pneumonoultramicroscopicsilicovolcanoconiosis’? I am sure that you are an expert in hyphenation, right \TeX? Or shall I ask Siri?

Table 2 shows: \( \kappa < -23.7 \approx (6509 - 4564)/-82 \) by (18), that is \( \lambda = -14 \), selects the path in column 7. As the path in column 6 contains a hyphen a \linepenalty of 10000 selects the path in column 7, which has no penalties or additional demerits.

**Paths with fewer lines.** In example 5, the parameter \linepenalty must be set to a large value in order to reduce the number of lines that are typeset from five to four. This is not covered by (17) as now \( \mu = \mu' + 1 \).

To determine the \( \kappa \) that selects a path with fewer lines the initial inequality must distinguish between \( \mu \) and \( \mu' = \mu - 1 \):

\[
\begin{align*}
\mu(\lambda + \kappa)^2 + 2\kappa B + \Lambda_p(\lambda) - \mu \lambda^2 & > \\
(\mu - 1)(\lambda + \kappa)^2 + 2\kappa B' + \Lambda'_p(\lambda) - (\mu - 1)\lambda^2.
\end{align*}
\]

The difference between \( \mu \) and \( \mu' \) adds the summand \( (\lambda + \kappa)^2 - \lambda^2 \) to the left hand side and this allows that the sum of badness can be larger for the shorter paragraph.

A simple rearrangement of the terms gives

\[
\kappa^2 + 2\kappa(\lambda + B - B') > \Lambda'_p(\lambda) - \Lambda_p(\lambda).
\]

(19)

Addition of \( (\lambda + B - B')^2 \) to both sides gives a quadratic term on the left

\[
(\kappa + \lambda + B - B')^2 > (\lambda + B - B')^2 + \Lambda'_p(\lambda) - \Lambda_p(\lambda)
\]

and as the right hand side is positive, the square root can be taken. Therefore the following relevant inequality is found

\[
\kappa > B' - B - \lambda \\
+ \sqrt{(\lambda + B - B')^2 + \Lambda'_p(\lambda) - \Lambda_p(\lambda)).
\]

(20)

Table 3 shows the data of the corresponding trace listing for the text of example 5. The notation (10) is explained above: see also “No information in the trace data” in [8], p. 370ff. The path of column 8 is typeset and it has the lowest value for the sum of badness B. Using this data inequality (20) gives \( \kappa > 374.8 \ldots \) for column 6. The required \linepenalty must be set to 385 — as mentioned after example 5.

**Paths with more lines.** In example 3 the parameter \linepenalty was set to 2 to enlarge the number of typeset lines. So this case should be analyzed too.

### Table 3: Badness, penalties, and additional demerits of the line breaks for the four paths of example 5

<table>
<thead>
<tr>
<th>( \par ) via</th>
<th>Class</th>
<th>( \mu )</th>
<th>( \kappa )</th>
<th>( \lambda )</th>
<th>( \Lambda_p(10) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \bullet \bullet )</td>
<td>6</td>
<td>7</td>
<td>*8</td>
<td>9</td>
<td>22760</td>
</tr>
<tr>
<td>1</td>
<td>d</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>196</td>
</tr>
<tr>
<td>2</td>
<td>l</td>
<td>19</td>
<td>19</td>
<td>19</td>
<td>121</td>
</tr>
<tr>
<td>3</td>
<td>t</td>
<td>96</td>
<td>96</td>
<td>96</td>
<td>27</td>
</tr>
<tr>
<td>4</td>
<td>l</td>
<td>87</td>
<td>87</td>
<td>87</td>
<td>119</td>
</tr>
<tr>
<td>5</td>
<td>d</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>d</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>19110</td>
</tr>
<tr>
<td>7</td>
<td>l</td>
<td>13</td>
<td>13</td>
<td>13</td>
<td>1431</td>
</tr>
<tr>
<td>8</td>
<td>d</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>24565</td>
</tr>
<tr>
<td>9</td>
<td>l</td>
<td>19110</td>
<td>19110</td>
<td>19110</td>
<td>24565</td>
</tr>
</tbody>
</table>

This time \( \kappa > 0 \) is subtracted from \( \lambda \):

\[
\Lambda_p(\lambda - \kappa) = \sum_{i=1}^{\mu} (\lambda - \kappa + \beta_i)^2 + \text{sgn}(\pi_i)\pi_i^2 + \delta_i
\]

\[
= \mu(\lambda - \kappa)^2 - 2\kappa B + \Lambda_p(\lambda) - \mu \lambda^2.
\]

Thus the inequality for the path demerits becomes

\[
\mu(\lambda - \kappa)^2 - 2\kappa B + \Lambda_p(\lambda) - \mu \lambda^2 < \\
\mu'(\lambda - \kappa)^2 - 2\kappa B' + \Lambda'_p(\lambda) - \mu' \lambda^2
\]

and with \( \mu' = \mu + 1 \) this is

\[
\Lambda_p(\lambda) - \Lambda'_p(\lambda) < \kappa^2 - 2\kappa(B - B' - \lambda).
\]

This time add the term \( (\lambda + B' - B)^2 \) to both sides and take square roots; then the useful result is

\[
\kappa > \lambda + B' - B \\
- \sqrt{(\lambda + B' - B)^2 + \Lambda_p(\lambda) - \Lambda'_p(\lambda))}. 
\]

(21)

The data of Table 4 gives for the transition from the path of column 2 to 6 with (21): \( \kappa > 5.3 \). Therefore \linepenalty = 10 - 6 = 4 typesets three lines as mentioned above.

### Table 4: Badness, penalties, and additional demerits of the line breaks for the five paths of example 3

<table>
<thead>
<tr>
<th>( \par ) via</th>
<th>Class</th>
<th>( \mu )</th>
<th>( \kappa )</th>
<th>( \lambda )</th>
<th>( \Lambda_p(10) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \bullet \bullet )</td>
<td>*2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>1</td>
<td>l</td>
<td>86</td>
<td>86</td>
<td>86</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>d</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>86</td>
</tr>
<tr>
<td>3</td>
<td>d</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>101</td>
</tr>
<tr>
<td>4</td>
<td>l</td>
<td>57</td>
<td>57</td>
<td>57</td>
<td>101</td>
</tr>
<tr>
<td>5</td>
<td>d</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>101</td>
</tr>
<tr>
<td>6</td>
<td>d</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>101</td>
</tr>
<tr>
<td>7</td>
<td>d</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>101</td>
</tr>
<tr>
<td>8</td>
<td>l</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>101</td>
</tr>
<tr>
<td>9</td>
<td>l</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>(7)</td>
<td>(d)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

\( \mu = 2 \) \quad \( B = 11 \) \quad \( \Lambda_p(10) = 485 \) \quad \( 9512 \) \quad \( 4785 \) \quad \( 9416 \) \quad \( 552 \)
Summary. When the \texttt{linepenalty} value is increased, \TeX's line-breaking algorithm focuses more on the badness values. If a path exists in the network of line breaks that has the same number of lines but a lower sum of badness compared to the path selected with the default settings, that path might be chosen with the larger \texttt{linepenalty}. This means that more breaks in mathematics and/or at positive \texttt{penalty} commands and/or more hyphens and/or more stacks of hyphens and/or more visually incompatible lines are typeset and at least one of these items is increased.

If a path exists that uses fewer lines for the paragraph, this path can be selected with a large \texttt{linepenalty} even if its sum of badness is higher than that of the paragraph with the default settings. Similarly a path can be selected that has more lines if \texttt{linepenalty} stays positive but is made smaller than 10.

Negative values for \texttt{linepenalty} typically create rather ugly paragraphs as \TeX then prefers large badness values for the lines. This effect is not normally desirable for justified text.

6 \texttt{linepenalty} versus \texttt{looseness}

The \TeXbook has an exercise in which the value 100 for the parameter \texttt{linepenalty} is suggested as a replacement for a negative \texttt{looseness} in an application to a single paragraph ([3], exercise 14.25). The reason refers to efficiency to "achieve almost the same result" if the user is not willing to pay the cost that a nonzero \texttt{looseness} generates. (\texttt{looseness} is explained in [3], pp. 103–104 or see Section 5 of [8] for an analysis of this parameter.)

Figure 2 shows that there are a lot of cases in which two lines are typeset instead of only one if \texttt{linepenalty} = 100. And example 1 proves that the increase of \texttt{linepenalty} can make a paragraph longer. Therefore this parameter might not only fail to reduce the number of lines it might be counterproductive. Although passes can have different numbers of lines for the shortest path, with a small enough negative \texttt{looseness} a paragraph can never get more lines than it has in the earliest pass that typesets it if \texttt{pretolerance} $\leq \texttt{tolerance}$, as the paths of the first pass are part of the network of line breaks of the second pass.

Inequality (19) can be used to determine in which cases a \texttt{linepenalty} of 100, i.e., $\kappa = 90$, can be successful in general. Here the plain \TeX values are used:

\begin{align*}
90^2 + 2 \cdot 90(10 + B - B') &> \Lambda_p'(10) - \Lambda_t(10) \\
&\iff 9900 - 180(B' - B) > \Lambda_p'(10) - \Lambda_t(10).
\end{align*}

Therefore the difference between the sum of badness values must be less than $55 = 9900/180$, but of course it must often be much smaller as the difference of the path demerits on the right hand side is positive and usually not very small.

Although Theorem 1 proves that the value 242 for \texttt{linepenalty} acts like $\texttt{looseness} = -1$ for a single line the scenario represents only a special case. For example, as noted in Section 3, the single line is always typeset by \TeX's line-breaking algorithm in the first pass. In general these two parameters behave quite differently.

Second pass. This is the fundamental difference between these two parameters: \texttt{looseness} will try hyphenation, i.e., the second pass, if it is not successful in the first.

Thus, hyphens might be introduced at the end of the lines if \texttt{looseness} is used although no reduction of the number of lines is achieved.

Example 8: Description

Typeset the text of example 7 twice: first with plain \TeX's defaults and second with \texttt{looseness} = -1.

\TeX output

Hi! \TeX! Tell me: How is the following long word broken 'pneumonoultramicroscopicsilicovolcanoconiosis'? I am sure that you are an expert in hyphenation, right \TeX? Or shall I ask Siri?

Hi! \TeX! Tell me: How is the following long word broken 'pneumonoultramicroscopicsilicovolcanoconiosis'? I am sure that you are an expert in hyphenation, right \TeX? Or shall I ask Siri?

The change of \texttt{linepenalty} never forces \TeX's line-breaking algorithm to execute another pass. It uses the pass that is necessary to break the lines when \texttt{linepenalty} = 10.

Example 9: Description

Typeset a paragraph twice: first with \texttt{linepenalty} = 9799 and second with the default \texttt{linepenalty} = 10 and \texttt{looseness} = -1.

\TeX output

A short text that cannot be typeset in two lines although looseness does it in the 2nd pass. A surprise! Or?

A short text that cannot be typeset in two lines although looseness does it in the 2nd pass. A surprise! Or?

\TeX typesets the text of the first paragraph in the first pass. The line-breaking algorithm cannot eliminate the third line in this pass. On the other hand this means that the first pass is a failure in \TeX's view if \texttt{looseness} = -1. But the second pass is a success: Although it also prefers three lines, there is a way to output only two. The demerits for the three line solution are 6926, those for the pair of lines 15773.

Udo Wermuth
Different cost functions. This leads to the next difference: \texttt{looseness} can choose a line-breaking solution that does not represent the shortest path in the network. This never happens for any setting of \texttt{linepenalty}: it must pick the shortest path.

As \texttt{looseness} has a different cost function to be optimized, penalties larger than \(-10000\) and smaller than \(10000\) mark places that are as good as others for a line break.

Example 10: Description
A text with two penalties is typeset twice: first with \texttt{linepenalty} = 9799 and second with \texttt{looseness} = \(-1\) and \texttt{linepenalty} = 10.

\TeX{} input
OK! Even 4-digit penalties, positive or negative, are not important for looseness but linepenalty obeys\texttt{penalty-9999} \ them.

\TeX{} output
OK! Even 4-digit penalties, positive or negative, are not important for looseness but linepenalty obeys them.

OK! Even 4-digit penalties, positive or negative, are not important for looseness but linepenalty obeys them.

Both paragraphs are typeset in the first pass. The \texttt{linepenalty} must pick the shortest path in the network with the cost function of demerits and thus \TeX{} typesets three lines. This cost function is not relevant for \texttt{looseness} if the number of lines of the paragraph can be changed. Only if this is not possible does \TeX{} select the shortest path in the current pass as usual. This means a parameter that does not inhibit some behavior does not count if \texttt{looseness} can be successful.

Success rate. No single value of \texttt{linepenalty} works for all paragraphs but \texttt{looseness} is always successful if the paragraph can be typeset with fewer lines.

Example 4 shows a text that can be typeset in two or three lines; a pair is output if \texttt{linepenalty} is set to 199. Example 5 typesets the text twice and it needs \texttt{linepenalty} = 385 to keep four lines. Each additional repetition of the original text requires a larger \texttt{linepenalty}:

<table>
<thead>
<tr>
<th>number of copies</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>\texttt{linepenalty} = 199</td>
<td>385</td>
<td>557</td>
<td>738</td>
<td>918</td>
<td>1098</td>
<td>1278</td>
<td></td>
</tr>
</tbody>
</table>

Of course, this is a constructed example, but nevertheless with 25 iterations the \texttt{linepenalty} must be 4519. The next step, i.e., a paragraph with only 52 lines in the shortest form, cannot be typeset with a \texttt{linepenalty} of 4541 anymore and it would need 4599 if the progression continues as before. It violates \TeX{}'s limit for the total demerits, see (4), and the text is not typeset correctly. With the default \texttt{linepenalty} together with \texttt{looseness} = \(-1\) no problem occurs.

Note: The large value 9799 for \texttt{linepenalty}, which was used in the last two examples, can be applied for paragraphs with at most ten lines. In order to demonstrate certain effects in examples its usage is needed, but such a large value would not be used for normal copy.

Quality of output. If the number of lines of a paragraph cannot be lowered then \texttt{looseness} still tries to find a line-breaking solution that avoids visually incompatible lines and stacks of hyphens, i.e., it obeys the additional demerits \texttt{adjdemerits} and \texttt{doublehyphendemerits} if possible.

But Section 5 shows that \texttt{linepenalty} trades the smaller sum of badness for penalties and additional demerits. This means that a text that contains no math, no explicit \texttt{penalty} command, and no explicit hyphens must get visually incompatible lines in a first pass if \TeX{} changes the line breaks.

In the second pass \texttt{linepenalty} considers hyphens as \texttt{looseness} does. The latter treats the parameters \texttt{double...} and \texttt{finalhyphendemerits} as usual while the former parameter trades them like \texttt{adjdemerits} for a smaller sum of badness values. That is, \TeX{} not only might typeset more hyphens, but also there might be more visually incompatible lines, more stacks of hyphens, and more hyphenated penultimate lines.

Efficiency. The advantage of increasing the parameter \texttt{linepenalty} instead of using \texttt{looseness} is that \texttt{linepenalty}’s value is always added in the code of the line-breaking algorithm — even if it is zero. A nonzero \texttt{looseness} invokes otherwise unused code and thus slows the algorithm down [4, §§ 873, 875]. As expected, this loss in efficiency is hardly noticed in normal copy with modern equipment.

My outdated computer from 2011 (a 1.8 GHz Dual-Core i7) with my own \TeX{} installation shows the following factors by which the runtime increases. The reference value 1 is used for the time needed by plain \TeX{} to typeset the two lines of example 4 with \texttt{looseness} = 0.

<table>
<thead>
<tr>
<th>copies</th>
<th>\texttt{looseness} = 0</th>
<th>\texttt{looseness} = (-1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>100</td>
<td>3</td>
<td>11</td>
</tr>
<tr>
<td>225</td>
<td>6</td>
<td>100</td>
</tr>
</tbody>
</table>

But even the abnormally long paragraph of the last case with 450 lines needs only one second to get typeset if \texttt{looseness} = \(-1\).

A note on \texttt{linepenalty}
\texttt{looseness} can be positive. A value larger than zero for the parameter \texttt{looseness} tries to make the paragraph longer.

Negative values for \texttt{linepenalty} usually typeset low quality paragraphs. This was proved in Section 5. Thus it is not a good idea to use them except when they would lengthen paragraphs; Section 5 shows that values \(\leq 10\) can be successful.

Example 3 uses the value 2 for \texttt{linepenalty} to typeset three instead of two lines. It happens that the \texttt{linepenalty} can be \(-2\) (but not \(-3\)).

Example 11: Description
Typeset example 3 with \texttt{linepenalty} = \(-2\).

\TeX\ output
When you start to count where do you start? With zero or with one? Hmm, I start at 1! A CS nerd uses a 0, or?

The next example uses a small positive and a high negative value for \texttt{linepenalty}. But even in a forced second pass (used by \texttt{looseness} = 1) these values do not make \TeX\ typeset an additional line.

Example 12: Description
Typeset a paragraph four times: first with plain \TeX's settings, second with \texttt{looseness} = 1, third in a forced second pass with \texttt{linepenalty} = 1, and finally, still in the second pass, with a \texttt{linepenalty} of \(-9999\).

\TeX\ output
This is a short paragraph and two words can have a hyphen in it. The rest of the text is made up of short words only. Well, I think the first sentence is wrong. Wait then one more must be wrong. Two are wrong.

This is a short paragraph and two words can have a hyphen in it. The rest of the text is made up of short words only. Well, I think the first sentence is wrong. Wait then one more must be wrong. Two are wrong.

This is a short paragraph and two words can have a hyphen in it. The rest of the text is made up of short words only. Well, I think the first sentence is wrong. Wait then one more must be wrong. Two are wrong.

Summary. \TeX\ must work harder if \texttt{looseness} is negative but a large value of \texttt{linepenalty} is not a replacement. A large value for the parameter \texttt{linepenalty} might even increase the number of lines output for a paragraph.

- If a paragraph can be typeset with fewer lines than \TeX's default settings produce then \texttt{looseness} = \(n\) for some \(n \leq -1\) is successful; \texttt{linepenalty} > 10 might be successful but only if the pass has not to be changed; otherwise the paragraph is treated like an unsuccessful case.
- If fewer lines for the paragraph are impossible \texttt{looseness} tries the final pass and thus might insert hyphens but outputs the shortest path: \texttt{linepenalty} outputs the (new) shortest path that might now have lines with lower badness, but then more breaks in mathematics or at positive \texttt{penalty} commands or more hyphens or more stacks of hyphens or more visually incompatible lines are used.

Both parameters might have a “negative impact” on paragraphs that cannot be shortened but the outcome with a large \texttt{linepenalty} seems to be worse. Its only advantage is that it does not change the pass and that it obeys other \TeX\ parameters.

References

[7] Donald E. Knuth, Digital Typography, Stanford, California: Center for the Study of Language and Information, CSLI Lecture Notes No. 78, 1999

Errata for previous articles. Here are two corrections for errors that are not merely typographical.

In [8], p. 365, left column, the \texttt{hsize} of the books The Art of Computer Programming by D. E. Knuth should be 348 pt.

In [9], p. 75, left column no. 10, the value of \(t\) is wrong; the stated value represents 24t.

Udo Wermuth