# pTEX のペナルティ

山下 弘展

TEXConf 2019

## 1 はじめに

ペナルティは，TEX において行分割時やページ分割時に「その箇所がブレークポイントとしてどの程度適切であるか」を示す評価値である．pTEX ではペナルティが和文組版に必須の禁則処理を実現するために拡張されている [1]．この仕様や挙動，注意すべき点について議論する．

以下，pTEX p3.8.2 (TEX Live 2019) に基づく．また，特に断らない限り pTEX 系列の標準的なフォーマットでの初期設定のペナルティ値を仮定する（特に下記）．

```
% 禁則ペナルティ
\prebreakpenalty‘.=10000
\prebreakpenalty‘。=10000
\postbreakpenalty‘(=10000
\postbreakpenalty‘（=10000
% 一文字だけの行を抑制
\jcharwidowpenalty=500
```

## 2 禁則処理に使われるペナルティ

欧文では連続する文字列（いわゆる単語）の途中で行分割が起こらない[†1]のに対し，和文ではほとんどの文字間が分割可能であり，その例外として**禁則**（行頭禁則と行末禁則）が存在する．

pTEX は，各文字間を分割可能にするためにグルー (\kanjiskip, \xkanjiskip) の自動挿入を行う．一方，禁則処理はペナルティによって実現されている．より詳細には

- 禁則対象とする文字のコード
- その文字に対するペナルティ値
- ペナルティの挿入位置（行頭禁則ならその文字の前に挿入，行末禁則なら後）

を登録できる**禁則テーブル**と，ここに情報を登録または削除する \prebreakpenalty と \postbreakpenalty というプリミティブを用意している．そして，pTEX は文章を読み込むたびにその文字がテーブルに登録されているかどうかを調べ，登録されていればそのペナルティを文字の前後適切な位置に挿入する．

pTEX の標準的なフォーマットでは約物（句読点やカッコ類）や小書き仮名等が禁則テーブルに登録されているが，ユーザが自由に登録・削除・値を変更することもできる[†2]．ただし，いくつか注意したい点があるので，下記で議論する．

### 2.1 行頭禁則と行末禁則は同時指定不可

同一の文字コード[†3]に対して，\prebreakpenalty と \postbreakpenalty の両方を同時に与えるような指定はできない．もし両方指定された場合，後から指定されたものに置き換えられる．

```
\prebreakpenalty‘〜=10000
\showthe\prebreakpenalty‘〜  % => 10000
\showthe\postbreakpenalty‘〜 % => 0
\postbreakpenalty‘〜=10000
\showthe\prebreakpenalty‘〜  % => 0
\showthe\postbreakpenalty‘〜 % => 10000
```

### 2.2 禁則の指定は文字コードに依る

禁則テーブルには和文文字と欧文文字の区別なく登録できる．これは，和文のみの文章中でも〈、〉や〈。〉等の和文文字以外に〈,〉や〈.〉等の欧文文字も約物として使われる可能性を考慮してのことであろう．

ただし，登録されるのは「文字」そのものではなく「**文字コード**」である．和文文字については内部コー

---

†1 ハイフネーション処理等の特別な場合を除く．

†2 削除できるようになったのはバージョン p3.8.1 以降 [2, 3].

†3 わざわざ「コード」と書いている理由はすぐ次の節参照．

ド（-kanji-internal の値）に依る．pTEX では和文文字の内部コードは 2 バイト，欧文文字は 1 バイトであるから衝突しないが，内部コード uptex の upTEX では和文文字[†4]の文字コードが欧文文字 (8-bit Latin) に被ることがある．

例えば，下記で “CJK” とした文字は約物に該当するが，“non-CJK” とした文字はアクセント付きの通常の文字である（T1 エンコーディングの場合）．約物を禁則扱いすると，同じ文字コードを持つアクセント付き文字も禁則扱いされることに注意が必要である [4].

- 0xA1: ¡ (CJK) vs. ą (non-CJK)
- 0xAB: « (CJK) vs. ń (non-CJK)
- 0xB7: · (CJK) vs. ů (non-CJK)

## 2.3 欧文文字に対する禁則処理

禁則テーブルには和文文字と欧文文字の区別なく登録できるが，実際には細かい注意が必要となる．

以下では pTEX がボックスを構築する様子を明らかにするため，\showlists や \showbox といったプリミティブを活用する．ボックスの中身を完全に表示するため，最初に次のコードを実行しておくとよい．

```
\tracingonline1
\showboxdepth10000
\showboxbreadth10000
```

### 2.3.1 禁則ペナルティの挿入タイミング

和文文字の禁則ペナルティは，その文字ノードをリストに追加する際に一緒に挿入される．このうち行末禁則に相当する \postbreakpenalty は「現在のリストの最後」になりうるため，和文文字の \postbreakpenalty は \lastpenalty で値を取得できるし，\unpenalty で取り除くこともできる．行頭禁則にあたる \prebreakpenalty は原理的に必ず後ろに文字ノードを伴うため，いったん挿入されればその値を取得したり取り除いたりすることはできない．

例として，upTEX で \font\x=upjisr-h \x を実行し，和文フォントに upjisr-h を選択した場合．

- \setbox0=\vbox{です。\showlists}

```
\x で
\x す
\penalty 10000(for kinsoku)
\x 。
\glue(refer from jfm) 5.0
```

- \setbox0=\vbox{まず（\showlists}

```
\x ま
\x ず
\glue(refer from jfm) 5.0 minus 5.0
\x （
\penalty 10000(for kinsoku)
```

一方，欧文文字の禁則ペナルティは，禁則対象の欧文文字が和文文字と隣り合う場合にのみ挿入される．つまり，欧文のみの組版では挿入されず，pTEX はオリジナルの TEX と同様に振る舞うことになる．

- 欧文文字直後が和文文字の場合に限り，その欧文文字に対する \postbreakpenalty を挿入する．
- 欧文文字直前が和文文字の場合に限り，その欧文文字に対する \prebreakpenalty を挿入する．

以下は，上記に続いて \font\A=cmr10 \A を実行し，欧文フォントに cmr10 を選択した場合．

- \setbox0=\vbox{Text end.\showlists}

```
\A T
\kern-0.83334
\A e
\A x
\A t
\glue 3.33333 plus 1.66666 minus 1.11111
\A e
\A n
\A d
\A .
```

- \setbox0=\vbox{終わり.\showlists}

```
\x 終
\x わ
\x り
\penalty 10000(for kinsoku)
\A .
```

- \setbox0=\vbox{(begin\showlists}

```
\A (
\A b
\kern0.27779
```

[†4] 厳密には upTEX では CJK 文字と呼ぶべきかもしれない．

```
\A e
\A g
\A i
\A n
```

- \setbox0=\vbox{(始まり \showlists}

```
\A (
\penalty 10000(for kinsoku)
\x 始
\x ま
\x り
```

隣接する文字が重要であることから，欧文文字に対する \postbreakpenalty の挿入はその文字ノードと同時ではなく，後続の和文文字ノード追加時に起こる．このタイミングのズレにより，欧文文字に対する \postbreakpenalty は \lastpenalty で取得できないし，\unpenalty で取り除くこともできない．

### 2.3.2 グループ境界での禁則ペナルティ

欧文文字の禁則ペナルティは，禁則対象の欧文文字と和文文字が隣接する場合にのみ挿入される…はずであるが，現在の pTEX ではグループ境界で隔てられた場合にペナルティが入ったり入らなかったりする．まずは \ybaselineshift=0pt の場合：

- \setbox0=\vbox{(漢 ({漢}{(}漢 \showlists}

```
\displace 0.0
\A (
\penalty 10000(for kinsoku)
\x 漢
\A (
\penalty 10000(for kinsoku)
\x 漢
\A (
\penalty 10000(for kinsoku)
\x 漢
```

- \setbox0=\vbox{漢.{漢}. 漢{.}\showlists}

```
\displace 0.0
\x 漢
\penalty 10000(for kinsoku)
\A .
\x 漢
\A .
\x 漢
\A .
```

このように，欧文文字の \postbreakpenalty は問題なさそうだが，\prebreakpenalty はグループ境界が挟まると挿入されていない．

次に \ybaselineshift=3pt の場合：

- \setbox0=\vbox{(漢 ({漢}{(}漢 \showlists}

```
\displace 3.0
\A (
\penalty 10000(for kinsoku)
\displace 0.0
\x 漢
\displace 3.0
\A (
\displace 0.0
\x 漢
\displace 3.0
\A (
\displace 0.0
\x 漢
```

- \setbox0=\vbox{漢.{漢}. 漢{.}\showlists}

```
\displace 0.0
\x 漢
\penalty 10000(for kinsoku)
\displace 3.0
\A .
\displace 0.0
\x 漢
\displace 3.0
\A .
\displace 0.0
\x 漢
\displace 3.0
\A .
\displace 0.0
```

ここで \displace は和欧文のベースライン補正を実現するノードを示し，\displace 3.0 は垂直下に 3.0 pt だけシフトし，\displace 0.0 で元の位置に戻している．今度は，欧文文字の \postbreakpenalty も \prebreakpenalty も，グループ境界が挟まる場合に挿入失敗している [5].

### 2.3.3 リガチャに対する禁則ペナルティ

欧文のリガチャといえば

- ⟨f⟩ + ⟨i⟩ で fi ではなく fi
- ⟨f⟩ + ⟨f⟩ + ⟨i⟩ で ffi ではなく ffi

のような例がよく想起されるが，他にも

- ⟨’⟩ + ⟨’⟩ で ”
- ⟨‘⟩ + ⟨‘⟩ で “

のように約物類もいくつか該当する（以上，cmr10 すなわち Computer Modern の例）．しかし，リガチャに禁則ペナルティを正しく設定するのは難しい．

- \postbreakpenalty はリガチャ自体の文字コードから決まる．
- \prebreakpenalty はリガチャ自体ではなく，その構成要素の最初の要素から決まる．

後者の \prebreakpenalty について，upLaTeX で例を見てみよう．T1 エンコーディングで ⟨>⟩ + ⟨>⟩ は合字 ⟨»⟩ を作るが，これはフォントでは ^^T の位置，Unicode では U+00BB にあたる．

```
\usepackage[utf8]{inputenc} % LaTeX 既定
\usepackage[T1]{fontenc}
\prebreakpenalty`\^^T=2560
\prebreakpenalty`\>=128
```

ここで次の入力を考えると，以下の出力が得られる．

- \setbox0=\hbox{あ>> い \char`\^^T う»}
  \showbox0

```
.\displace 0.0
.\x あ
.\penalty 128(for kinsoku)
.\glue(\xkanjiskip) 2.5 plus 2.5 minus 1.25
.\A ^^T (ligature >>)
.\glue 2.5 plus 1.99997 minus 1.00006
.\x い
.\penalty 2560(for kinsoku)
.\A ^^T
.\x う
.\penalty 2560(for kinsoku)
.\A ^^T
```

このように，文字コードから直接指定された ⟨»⟩ の前（いの後）と UTF-8 入力で得た ⟨»⟩ の前（うの後）にはリガチャ自体の文字コードに設定された禁則ペナルティ 2560 が挿入されているが，⟨>⟩ + ⟨>⟩ から生成した ⟨»⟩ の前（あの後）には ⟨>⟩ に設定された禁則ペナルティ 128 が挿入されている．つまり，すべての ⟨»⟩ を行頭禁則にするためには ⟨>⟩ と ⟨^^T⟩ の両方に \prebreakpenalty を設定しなければならないことになる（\postbreakpenalty ではこのような不統一は起きないようである）．

欧文文字のリガチャに関しては，pTeX において他にも「\xkanjiskip の挿入可否 (\xspcode)」でも問題になる．こちらもリガチャの構成要素から決まるようであり，議論を要する [5].

## 2.4 和文文字に対する禁則処理

pTeX には，和文文字の禁則処理についてもいくつかの課題が残っている．

### 2.4.1 和文文字の禁則ペナルティ取得

先に書いた

> 和文文字の \postbreakpenalty は \lastpenalty で値を取得できるし，\unpenalty で取り除くこともできる

についても，少々デリケートな挙動が存在する．

例えば，pLaTeX の標準クラス (jarticle.cls) や upLaTeX の標準クラス (ujarticle.cls), jsarticle.cls などでは以下のようにして ⟨(⟩ の後に挿入された \postbreakpenalty を取得・削除できる．

- \setbox0=\vbox{あ（\showthe\lastpenalty}
  % => 10000.
- \setbox0=\vbox{あ（\showlists}

```
\x あ
\glue(refer from jfm) 5.0 minus 5.0
\x （
\penalty 10000(for kinsoku)
```

- \setbox0=\vbox{あ（\unpenalty\showlists}

```
\x あ
\glue(refer from jfm) 5.0 minus 5.0
\x （
```

ところが，jlreq.cls ではうまくいかない．

- \setbox0=\vbox{あ（\showthe\lastpenalty}
  % => 0.
- \setbox0=\vbox{あ（\showlists}

```
\X あ
\glue(refer from jfm) 5.0 minus 5.0
\X （
\penalty 10000(for kinsoku)
\glue(refer from jfm) 0.0
```

- \setbox0=\vbox{あ（\unpenalty\showlists}

```
\X あ
\glue(refer from jfm) 5.0 minus 5.0
```

```
\X （
\penalty 10000(for kinsoku)
\glue(refer from jfm) 0.0
```

これはリストを見ればわかるように，「現在のリストの最後」が JFM グルーとなってしまったためである [6]．この挙動が問題となる例として，pLaTeX カーネルで脚注マクロ (\footnotetext) を修正するにあたり，\lastpenalty と \unpenalty を使用していることが挙げられる．

### 2.4.2 禁則ペナルティの合算

pTeX の \prebreakpenalty は，既にその位置にペナルティがあった場合は新たに発行されるのではなく，そのペナルティに合算される．これは「既にあったペナルティ」が \penalty プリミティブによる明示的なペナルティか，\postbreakpenalty 由来の自動挿入された禁則ペナルティかの種類によらない．そのため，\break すなわち \penalty -10000 の直後が行頭禁則文字（例えば \prebreakpenalty が 10000）である場合は −10000 + 10000 = 0 の合算が起きてしまい，結果的に \break による強制改行が効かない．

## 3 和文文字の孤立を防ぐペナルティ

本講演では時間の制約上取り扱わないが，講演タイトルが「pTeX のペナルティ」であるので，トピックだけを挙げるに留める．

パラグラフの最終行が

> このように一文字だけ孤立すると嬉しくありません

のように和文文字一字で孤立するのを防ぐために，pTeX には \jcharwidowpenalty というペナルティが用意されている．このペナルティを発行する要件として，現在の pTeX は「段落に 6 文字以上が含まれる」などの要件を課しているが，その挙動はなんとも不統一である [7]．さらに，\jcharwidowpenalty は本来行分割を抑制するためのものであるにもかかわらず，和文文字の前に \jcharwidowpenalty が挿入されることでかえって「その位置での行分割が起こりやすくなる」というケースも指摘されている [8]．このあたりについては引き続き議論を要する．

## 参考文献


[1] 日本語 TeX 開発コミュニティ，「pTeX マニュアル」，TEXMFDIST/doc/ptex/ptex-manual/ptex-manual.pdf

[2] h-kitagawa，「禁則テーブル，\inhibitxspcode 情報テーブルからのエントリ削除」，2017/09/10，https://github.com/texjporg/tex-jp-build/pull/26

[3] Man-Ting-Fang, *[upTeX] Unexpected behaviour in kinsoku processing*, 2018/04/13, https://github.com/texjporg/tex-jp-build/issues/57

[4] aminophen，「中韓フォントの JFM」（2017/07/31 以降のコメント），2017/07/01，https://github.com/texjporg/uptex-fonts/issues/2

[5] aminophen，「グループ境界をはさむ時の欧文文字の禁則ペナルティ」，2019/07/26，https://github.com/texjporg/tex-jp-build/issues/85

[6] aminophen，「\prebreakpenalty の合算と JFM グルー，\lastpenalty」，2018/09/17，https://github.com/texjporg/tex-jp-build/issues/67

[7] h-kitagawa，「[ptex] \jcharwidowpenalty の仕様」，2017/04/24，https://github.com/texjporg/tex-jp-build/issues/13

[8] ZR，「Re: upTeX における kcatcode と禁則処理の連動」，2009/10/25，https://oku.edu.mie-u.ac.jp/~okumura/texfaq/qa/53892.html