

Tabu Package Updates

January 12, 2019

The tabu package patches internal commands of many packages and is liable to stop working if those internal interfaces change.

Unfortunately the original author appears to be out of contact and the package had not been updated for some years.

This manual documents version 2.8 from 2011, but the distributed package contains fixes reported since that time, see <https://github.com/tabu-fixed/tabu>

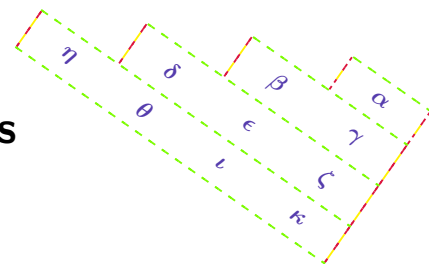
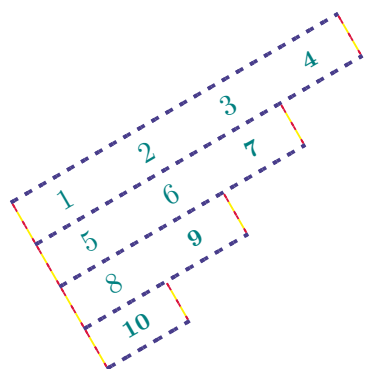


tabu and longtabu

Flexible L^AT_EX tabulars

FC

2011/02/26 – version 2.8 release



Abstract

This package defines a single environment `tabu` to make all kinds of tabulars in text or in math mode provided that they do not split across pages.

An environment `longtabu` – based on D. Carlisle `longtable` package – is also provided to make tabulars that can stretch out on several pages, while keeping some features (not all of them) of the `tabu` environment.

`tabu` is more flexible than `tabular`, `tabular*`, `tabularx` and `array` and extends the possibilities. All tabulars in this document were made with the `tabu` environment, *of course...* The implementation is optimised to minimise the measurements required to put all together.

`\tau_N b c` likes colors too, with special lines that are able to keep the alignment of the surrounded text... and also like numbers with the possibility to embed `siunitx` `S` (or `s`) columns. `\tau_N b c` does not modify any of the macro defined by `array.sty` or in the L^AT_EX kernel¹.

`\tau_N b c` requires ϵ -T_EX and the standard package `array.sty`. Natural widths of columns are computed (but not printed) by the code of `varwidth` by D. Arseneau. Finally `longtabu` is based on `longtable`.

Contents

Summary of the features provided by <code>\tau_N b c</code>	3
1 Examples and counterexamples	5
1.1 “Locally global” settings and their scopes	5
1.2 X column widths computation	6
1.3 Inserting Verbatim material (<code>fancyvrb</code>)	7
1.4 Maths inside <code>tabu</code> X columns	7
1.5 Embedding <code>sunitx</code> <code>S</code> columns inside X columns	8
2 The <code>tabu</code> environment	8
2.1 <code>tabu</code> , <code>tabu to</code> and <code>tabu spread</code>	8
2.2 <code>longtabu</code> , <code>longtabu to</code> and <code>longtabu spread</code>	9
2.3 <code>tabu</code> X columns – Mastering horizontal space.....	10
2.3.1 X columns with “ <code>tabu spread</code> ”	
2.3.2 <i>Negativ width coefficients for X columns</i>	
2.3.3 <i>Multicolumn in <code>tabu</code></i>	
2.4 <code>\tabulinesep</code> and <code>\extrarowsep</code> – Mastering vertical space.....	12
2.5 <code>tabu</code> in math mode	14
3 Lines leaders and colors inside <code>tabu</code>	14
3.1 First important remark	14
3.2 Vertical lines: <code> </code> has an optional parameter	14
3.3 Multiple <code>\firstline</code> and <code>\lastline</code>	15
3.4 More style for lines	16
3.5 Automatic horizontal lines and row colors	17
4 Modifying the font and the alignment in one row: <code>\rowfont</code>	18
5 Saving and restoring a <code>tabu</code>: <code>\savetabu</code>, <code>\usetabu</code> and <code>\preamble</code>	19

* This documentation is produced with the DocStrip utility, and required `\tau_N b c` with its `linegoal` option.

→ To get the package, run: `etex tabu.dtx`
→ To get the documentation run (thrice): `pdflatex tabu.dtx`
To get the index, run: `makeindex -s gind.ist tabu.idx`

The `.dtx` file is embedded into this pdf file thank to `embedfile` by H. Oberdiek.

1. Inside the `tabu` environment a few macros are modified... this was compulsory !

6	Some other features	20
6.1	Printing numbers inside <code>tabu</code> with <code>numprint</code> and <code>siunitx</code>	20
	6.1.1 <code>\tabudecimal</code>	
	6.1.2 <i>You should know how it works...</i>	
6.2	Paragraph indentation	21
6.3	<code>delarray</code> shortcuts	22
7	Differences between <code>tabu</code>, <code>tabular</code>, <code>tabularx</code> and <code>longtable</code>	22
7.1	Paragraph indentation	22
7.2	Custom environments	22
7.3	Inversion of tokens.....	22
7.4	Improved process for rewriting columns (<i>for keen readers</i>).....	23
8	The package options	24
8.1	The <code>debugshow</code> package option.....	24
8.2	The <code>delarray</code> package option.....	24
8.3	The <code>linegoal</code> package option	24
9	Corrections of some bugs (<i>available only inside <code>tabu</code></i>)	24
9.1	Correction for <code>colortbl</code> and <code>arydshln</code> : compatibility with <code>delarray</code>	24
9.2	Correction for <code>arydshln</code> : @ columns.....	25
10	To do for even better <code>tabus</code>	25
11	TECHNICAL NOTICE AND IMPLEMENTATION	26
11.1	Drawing a tabular - The \mathcal{T}_{Nbc} approach.....	26
11.2	Algorithms	26
11.3	The <code>tabu</code> strategies.....	30
11.4	Identification and Requirements	31
11.5	Flow chart of expansion	31
11.6	Some constants	33
11.7	Rules, colors and vertical adjustment	38
11.8	The entry inside <code>tabu</code>	47
11.9	The rewriting process: inside the “ <code>\@mkpream group</code> ”.....	51
11.10	Implementing the strategy at the exit of the <code>\@mkpream group</code>	59
11.11	One trial after the other (<code>\tabu@strategy</code>)	61
11.12	The algorithms: Measuring the <code>tabu</code> box.....	64
11.13	Measuring the natural width of columns (<code>varwidth</code> code from D. Arseneau).....	69
11.14	Measuring the height and depths of rows	70
11.15	<code>\tabuphantomline</code>	71
11.16	Horizontal lines inside <code>tabu</code> : <code>\tabucline</code> , <code>\firstline</code> and <code>\lastline</code>	72
11.17	Numbers in <code>tabu</code>	75
11.18	Verbatim inside <code>tabu</code> with X columns.....	77
11.19	<code>\savetabu</code>	78
11.20	<code>\rowfont</code>	79
11.21	Taking care of footnotes and <code>\arraybackslash</code>	84
11.22	Corrections	85
11.23	Package options and Initialisation	87
12	References	91
13	History	91
	[2011/02/26 v2.8]	91
	[2011/02/25 v2.7]	91
	[2011/02/24 v2.6]	91
	[2011/02/19 v2.5]	91
	[2011/02/17 v2.4]	91
	[2011/02/13 v2.3]	91
	[2011/02/12 v2.2 – New implementation - Absolutely no modification of <code>array.sty</code>]	91
	[2011/01/19 v2.1]	92
	[2011/01/18 v2.0]	92
	[2011/01/15 v1.9]	92
	[2010/12/28 v1.8]	92
	[2010/12/18 v1.7]	93
	[2010/12/07 v1.5]	93
	[2010/11/22 v1.4]	93
	[2010/11/18 v1.3]	93
	[2010/11/15 v1.2]	94
	[2010/10/28 v1.1]	94
14	Index	94

Summary of the features provided by τ_{Nbc}

<code>tabu</code>	is like <code>tabular</code> in text mode and like <code>array</code> in math mode when there is no X column in its preamble
<code>longtabu</code>	is like <code>longtable</code> with the possibility to use <code>tabu</code> X columns and vertical lines with the extended syntax.
<code>{tabu} to <dimen></code>	specifies the target width of the whole tabular. This is like <code>tabular*</code> with an automatic stretchability that can be overwritten with <code>@{\extracolsep {dimen}}</code> in front of the preamble.
<code>{tabu} spread <dimen></code>	has no equivalent in L ^A T _E X: the final width is <code><dimen></code> wider than the natural width that can be obtained with <code>spread Opt</code> .
<code> [width,color]</code>	vertical lines have an optional parameter.
<code>X[coef,align,type]</code> <code>X[coef,align,type,\$]</code>	X columns widths are adjusted in order for the whole tabular to fit the target width. The target width is a dimension either: <ul style="list-style-type: none"> → directly specified with <code>{tabu} to<dimen></code> → computed from the natural width: <code>{tabu} spread<dimen></code> → by default <code>\linewidth</code> (or <code>\linegoal</code> with the <code>linegoal</code> package option). coef scales the widths of the X columns, if there are more than one X column. align is either r, c, l or j (or R C L J) and type can be p (default), m or b. X[\$] makes a math X column (<i>ie.</i> <code>>{X}<{X}</code>) X[\$\$] display math X column: <code>>{\displaystyle }X<{X}</code>
<code>X[-coef,align,type]</code>	X columns widths are first computed with the absolute value: <code> coef </code> . Then the width is made narrower down to the natural width of the column if possible. In any case, the final width does not exceed the one obtained with <code>X[coef]</code> .
<code>X[X options]{S[S options]}</code>	Embed a <code>siunitx</code> S column into a <code>tabu</code> -X column.
<code>\everyrow {code}</code>	Allows to add horizontal lines automatically for every row. The settings can be changed inside the <code>tabu</code>
<code>\rowfont [align]{font spec}</code>	Modify the font and optionally the alignment of each cell in one row.
<code>\tabulinesep =<dimen></code>	More control on vertical spacing of lines in a way very close to <code>cellspace</code> 's method (dynamic vertical spacing adjustment).
<code>\extrarowsep =<dimen></code>	Control vertical spacing (<code>\extrarowheight</code> and <code>\extrarowdepth</code>): fixed vertical spacing adjustment. <code>\tabulinesep</code> generally gives better results.
<code>\tabudecimal {\usermacro }</code>	a help to align numbers easily inside a column.
<code>\savetabu {user-name}</code>	Saves the <code>tabu</code> preamble and its parameters. The command must appear at the end of a line.
<code>\usetabu {user-name}</code>	Makes a <code>tabu</code> of exactly the same shape as the one saved with <code>\savetabu</code> . All parameters (target, preamble, stretch <i>etc.</i>) are restored. This command is put alone in the preamble in place of the columns specifications.
<code>\preamble {user-name}</code>	Makes a <code>tabu</code> with the same preamble as the one saved with <code>\savetabu</code> . The only <code>preamble</code> is restored, not the <code>target</code> nor any other parameter. This command is put alone in the preamble in place of the columns specifications.

Summary of the features provided by τ_{Nbc}

<code>\tabulinestyle {line spec}</code>	Sets the current line style to be used for <code> </code> and <code>\tabucline</code>
<code>\newtabulinestyle {name=spec,...}</code>	Defines a line style for use with <code>\tabucline [name]</code> or with <code> [name]</code>
<code>\tabucline [spec]{start-stop}</code>	Draws a line comparable to <code>\hrule</code> . The line <code><spec></code> can contain information for making a dash or dotted line (f.ex. <code>[on 3pt off 6pt]</code>) and a color name. The line spec can also be defined with <code>\newtabulinestyle</code>
<code>\taburulecolor dbl rule sep {rule color}</code>	sets the color for rules (<code>\hrule</code> , <code>\firstline...</code>)
<code>\taburowcolors [skip]<number>{first .. last}</code>	Sets the color series to make alternate background colors for rows
<code>\tabuphantomline</code>	inserts a phantom (<i>ie.invisible</i>) line inside the <code>tabu</code> May be usefull with <code>\multicolumn</code> in some cases.
<code>\tracingtabu = 0, 1, 2, 3, 4</code>	Reports informations in the <code>.log</code> file about the steps of the algorithm for <code>tabu X</code> columns, and the informations saved by <code>\savetabu</code> .

1 Examples and counterexamples

Let's begin in colors !

$\tau_{\text{N}bc}$ provides facilities to put horizontal and vertical leaders in a tabular. The package `xcolor` must be loaded of course. Background colors for cells are left to package `colortbl` which is fully compatible with $\tau_{\text{N}bc}$.

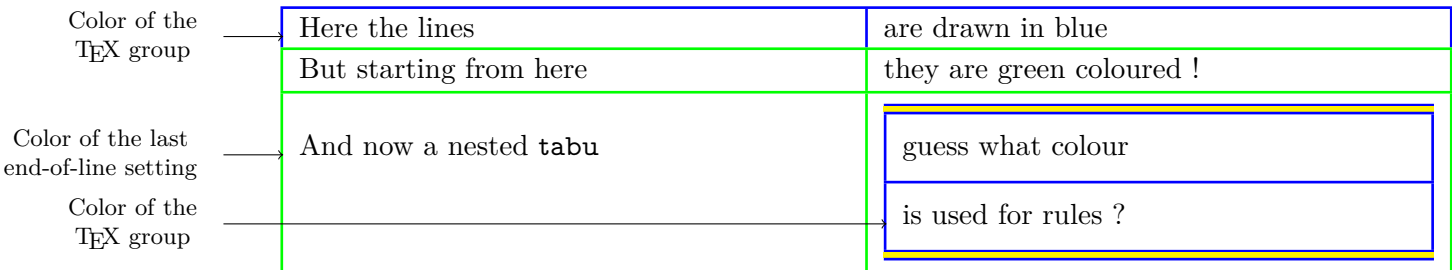
1.1 “Locally global” settings and their scopes

`\tabulinestyle`
`\taburulecolor`
`\taburowcolors`
`\everyrow`

$\tau_{\text{N}bc}$ observes T_EX grouping levels for the settings of rule colors (`\taburulecolor`) and styles (`\tabulinestyle`), and `\everyrow`. There is however a subtlety for nested `tabu` environments as described in this example:

Listing 1: Locally global settings and their scopes

```
\taburulecolor |gray!50|{red} \arrayrulewidth=1pt
{
  \taburulecolor |yellow|{blue}
  \begin{tabu}{|X|X|} \hline
  Here the lines & are drawn in blue \\ \taburulecolor{green} \hline
  But starting from here & they are green coloured ! \\ \hline
  And now a nested tabu & \begin{tabu}{X} \firsthline\hline
                           guess what colour \\ \hline
                           is used for rules ? \\ \lasthline\hline
                           \end{tabu} \\ \hline
  \end{tabu}
  % Inside the group, rule colors are blue
}
% After the group, rule colors are red again !
\begin{tabu}{X}\hline\hline\indent\end{tabu}
```



Inside the group, rule colors are blue After the group, rule colors are red again !

The “rules” are the following:

- If outside of a `tabu` environment, the settings are local to the T_EX group. Every tabular drawn inside this group will inherit from the settings of that group.
- If `\taburulecolor` (or `\everyrow` or `\tabulinestyle`) is used inside a cell of the tabular, this is the same: the settings a local to that cell, and any nested tabular will inherit from the setting of that cell.
- When used after the end of a row, the settings are globally changed from that point until the end of the tabular, or until a new setting is set at the end of a further row (T_EXnically, this is done inside a `\noalign` group). But a nested `tabu` does not inherit from this “global” setting, and inherits from the settings of the T_EX group instead.

If `\arrayrulecolor` or `\doublerulesepcolor` (from package `colortbl`) are used instead of `\taburulecolor` then colors are globally overwritten.

A counterexample from the `xcolor` package: `\rowcolors` does not like `\cline`, `\cmidrule` etc.²

```
\rowcolors{2}{green!25}{yellow!50}
\begin{tabular}{cc} \toprule
\repeatcell{2}{
  rows=5,
  text/col1=test ,
  text/col2=row \number\rownum} \
  test & other row \number\rownum \ \ \cmidrule
    {1-2}
  test & other row \number\rownum \ \
  test & other row \number\rownum \ \ \
    \bottomrule
\end{tabular}
```

test	row 1
test	row 2
test	row 3
test	row 4
test	row 5
test	other row 6
test	other row 8
test	other row 9

The `\rownum` counter is not reliable in the case of `\cline` or `\cmidrule`.
 In addition, the first coloured row is yellow, while one could have expected it green...

For $\tau_{\text{nb}}c$ color changes are called at `\everyrow`:

```
\taburowcolors [2] 2{green!25 .. yellow!50}
\begin{tabu}{*2{X[c]}} \toprule
\repeatcell{2}{
  rows=5,
  text/col1=test ,
  text/col2=row \thetaburow} \
  test & other row \thetaburow \ \ \cmidrule
    {1-2}
  test & other row \thetaburow \ \
  test & other row \thetaburow \ \ \bottomrule
\end{tabu}
```

test	row 1
test	row 2
test	row 3
test	row 4
test	row 5
test	other row 6
test	other row 7
test	other row 8

$\tau_{\text{nb}}c$ does not use “real” alternate colors but colorseries provided by package `xcolor`. This allow some gradations:

```
\taburowcolors 5{green!25 .. yellow!50}
\begin{tabu}{X[-1]X}
\repeatcell 2{
  rows=10,
  text/col1=test ,
  text/col2={Row number
    \row$=\thetaburow},
}
\end{tabu}
```

test	Row number 1=1
test	Row number 2=2
test	Row number 3=3
test	Row number 4=4
test	Row number 5=5
test	Row number 6=6
test	Row number 7=7
test	Row number 8=8
test	Row number 9=9
test	Row number 10=10

1.2 X column widths computation

The new algorithm implemented in version 2.8 requires only one measure of the width of the table in any case. This speeds up the convergence of the algorithm.

```
\begin{tabu} to 140mm {|X[1,1] | X[2,c] | X[3,c] | X[1,r]|}
|\dotfill | & Text & & Text & & Text \
Text & & Text & & Text
\end{tabu}
```

.	Text	Text	Text
Text	Text	Text	Text
1X = 17.5mm	2X = 35mm	3X = 52.5mm	1X = 17.5mm

$$X = (140mm - 8 \times \text{\tabcolsep} - 5 \times \text{\arrayrulewidth}) / 7 = 17.4896mm$$

2. Because color changes are done at `\everycr`, which is not exactly the same as $\tau_{\text{nb}}c$ `\everyrow`!

Example provided by
M. Manuel Kuehner

1.3 Inserting Verbatim material (fancyvrb)

Though the content of the `tabu` environment is collected for measuring purpose, it is possible to insert verbatim material with the `tabu*` variant of the environment. The content is then carefully collected and re-scanned (with `\scantokens`). During the process, the `@` letter is read with the category code it has been given at the entry inside the environment (it is possible to say `\makeatletter` before `\begin {tabu* }`).

Example:

It is possible to insert Verbatim material with some `\csname` control sequences `\endcsname` inside a `tabu` and inside X columns. Negative coefficients work well too, adjusting the width of the X column to the natural width if it is finally less than the width computed with the absolute value of the coefficient. A complete `Verbatim` environment is also admissible. But you must use the star form of the environment: `tabu*` which uses `\scantokens`.

Verbatim environments must be put alone on their lines (in the input file) for nothing is allowed after `\begin{Verbatim}` or `\end{Verbatim}`.

Another point to know is that `\begin` and `\end` control sequences should match otherwise, you must enclose the `Verbatim` environment inside braces.

This is related to the fact that `tabu` collects its body, and looks for matching pairs of `\begin ... \end !`

`tabu*` is useless when nested inside another tabular. The star form of the environment should be used only for the outermost table ! Comments are removed, unless the `%` character is given a category code of 12 (or 11) before the entry inside the environment.

```
\tabulinestyle{on2pt Crimson!60 off3pt yellow!50} \tabulinesep=2mm
\makeatletter \@makeother\%
\begin{tabu*}spread 0pt {|X[-1]X|} \tabucline -
This is a small \Verb+\Verbatim+\par
insertion
&
\begin{Verbatim}[listparameters={\topsep=-\ht\strutbox}]
And this is a complete % with some comments
Verbatim environment % every now and then
\end{Verbatim}
\\ \tabucline -
\end{tabu*}
```

Here a small <code>\Verbatim</code> insertion	And this is a complete <code>% with some comments</code> Verbatim environment <code>% every now and then</code>
---	--

It's not possible to insert a `lstlisting` environment presently, but you can save such an environment in a `\vbox` and insert it inside the `tabu` of course.

1.4 Maths inside tabu X columns

<pre>\$\begin{tabu}spread .5in {*3{X[\$c]}} \alpha & \beta & \gamma \\ \sum_i \frac{a_i}{x_i} & 0 & \cdot \\ \end{tabu}\$</pre>		$\begin{matrix} \alpha & \beta & \gamma \\ \sum_i \frac{a_i}{x_i} & 0 & \cdot \end{matrix}$	
---	--	---	--

X[\$] columns

<pre>\$\begin{tabu}spread .5in {*3{X[\$\$c]}} \alpha & \beta & \gamma \\ \sum_i \frac{a_i}{x_i} & 0 & \cdot \\ \end{tabu}\$</pre>		$\begin{matrix} \alpha & \beta & \gamma \\ \sum_i \frac{a_i}{x_i} & 0 & \cdot \end{matrix}$	
---	--	---	--

X[\$\$] columns

ragged2e settings) and the **column type** (p, m, or b).
 tabu has a **default target width** when used with X columns, making nesting even easier.

- You are used to the tabular environment in text mode, and array environment in math mode, but tabu works in both modes and its name does not change... X columns are also possible in math mode; delarray shortcuts for delimiters are available in both math and text modes.
- A tabu environment can contain another tabular of any kind: tabular, tabular* , tabularx or tabu itself can be placed in any cell of a tabu. Conversely, tabu can be placed in a tabular, tabularx etc..
- tabu provides facilities for vertical and horizontal lines, and for the insertion of verbatim text inside X columns.
- tabu is fully compatible with colortbl, delarray, hline, makecell, booktabs, siunitx, dcolumn, warpcol, etc.. When you are inside a tabu environment, you can use \raggedleft, \raggedright and \centering without special care about \arraybackslash and conversely \\ has its “normal” meaning inside a list of items that may appear in a X column...

\begin {tabu} to<dimen> is like tabular* but the inter-columns space is given a stretchability of 1fil, in other words @{\extracolsep {0pt plus 1fil}} is inserted by default at the beginning of the tabular preamble, unless another value for \extracolsep is specified. Therefore “tabu to” fills in width the specified <dimen>.

\begin {tabu} spread<dimen> does a tabular whose width is <dimen> wider than its natural width. @{\extracolsep {0pt plus 1fil}} is inserted by default if <dimen> > 0.

2.2 longtabu, longtabu to and longtabu spread

```
\begin {longtabu} [l | c | r] {tabular preamble}
\begin {longtabu} to <dimen> [l | c | r] {tabular preamble}
\begin {longtabu} spread <dimen> [l | c | r] {tabular preamble}
```

longtabu is just like tabu but page breaks are allowed between rows of the table. longtabu is based on the longtable package which must be loaded, and all features of the longtable environment works inside longtabu: \endhead, \endfirsthead, \endfoot, \endlastfoot and \caption.

longtabu enhances the longtable environment with the possibility to use X columns and line specifications for horizontal and vertical rules. longtabu is thus much easier than ltxtable.

The following commands provided for tabu do not work with longtabu:

tabu command	Not available	Not (yet) implemented	Comment
\tabucline		✿	\tabucline does not care of page breaks presently: use \hline instead.
\usetabu	✗		but \savetabu and \preamble work.
mathematical mode	✗		longtable is not designed to work in math mode.
delarray shortcuts	✗		a delimiter cannot be spanned over pages...
\tabuphantomline	✗		useless inside longtabu

However, tabu X columns, \rowfont, \extrarowsep, \tabulinesep, \tabudecimal, \tabucline (with restrictions on page breaks), \taburulecolor, \tabulinestyle, \taburowcolors, \preamble, {longtabu} to , {longtabu} spread work inside longtabu.

2.3 tabu X columns – Mastering horizontal space

tabu X columns can be viewed as an enhancement of tabularx X columns, but do not interact with them, for they are defined only for a short time during the parsing of the preamble:

- **width coefficients** can optionally be given to X columns
 ex. `X[2.5]X[1]` is the same as `X[2.5]X` and the same as `X[5]X[2]`
 This means that the first X column will be two and a half wider than the second one or that the first X column width will be $\frac{5}{7}$ of the whole tabular width.

X[2.5]	X
--------	---

- **negativ width coefficients** can be given to X columns:
 ex. `X[-2.5]X[1]` or `X[-2.5]X` or `X[-5]X[2]`
 In this case, the first X column will be *at most* two and a half wider than the second one, and if the *natural width* of the first X column is finally less than $2.5 \times$ (the width of the second column) then it will be narrowed down to this natural width.

The following tabus have the same preamble:

`\begin{tabu} to\linewidth { |X[-2.5c|X[c]| }:`

X[-2.5]	X
Negativ coefficients make X columns close to standard l, c and r columns.	X

- horizontal alignment specification is made easier with `X[5,r]X[2,c]` for example. Vertical alignment can be specified as well with `X[5,r,m]X[2,p,c]` (commas are not required, but `X[2cm]` or `X[4pc]` could be misunderstood – not by T_EX: by you...).

Modifier	Meaning	Default
l, c, r, j, L, C, R, J	left, centered, right, justified	j
p, m, b	X column is converted into p, m or b column	p
\$	X[\$] is a shortcut for: <code>>{ \$ }X<{ \$ }</code>	
\$\$	X[\$\$] is a shortcut for: <code>>{ \$ \displaystyle }X<{ \$ }</code>	

- tabu X columns can be spanned with `\multicolumn`.
- tabu X columns can be used with “tabu spread” for small tabulars.
- tabu X columns can contain any type of tabular, tabular*, tabularx or tabu without special care about the syntax. tabu can also be put inside tabular, tabular* and tabularx. As long as tabu with X columns has a *default target*, nesting tabu with X columns is easy. Furthermore, the default global alignment of a nested tabu is t (for top) while the default global alignment of a tabu in a paragraph is c (for centered).
- The “algorithm” (or the arithmetic) to get the target width for tabu X columns is the same as the one used by tabularx. `\hfuzz` is the “tolerance” for the whole tabular width. We use `\epsilon-TEX \dimexpr` instead of T_EX primitives (with round/truncate bias correction).
- Convergence to the target width is optimised: the `\halign` preamble is not re-built at each trial, but only expanded again, until the target is reached. Though optimized, the process is the same as the one implemented for tabularx and in particular the content of the tabu environment is collected as soon as a tabu X column is found in the preamble. This implies restrictions on catcode modifications and verbatim text inside a tabu with X columns.
- If the width of the whole tabular is not specified with “tabu to” it is considered to be `\linewidth`. The [linegoal package option](#) makes the default width equal to `\linegoal`. Compilation must then be done with pdfT_EX either in pdf or dvi mode, and package `linegoal` is loaded. `\linegoal` requires pdfT_EX for its `\pdfsavepos` primitive and the `zref-savepos`: if the tabu is not alone in its paragraph *ie.* if the target is not `\linewidth`, then two compilations (or more) are required to get the correct target. Default target for nested tabu environments is always `\linewidth`, which equals to the column width inside p, m, b and X columns.
- As long as the `\halign` content is expanded more than once, protections against counters

incrementation, whatsits (*write*) index entries, footnotes *etc.*. are set up: the mechanism of `tabularx` is reimplemented and enhanced for `tabu X` columns. `\tabuDisableCommands` can be used to neutralize the expansion of additional macros during the trials.

X columns with “tabu spread”

`tabu X` columns can be used with “`tabu spread`” to adjust the column widths of tabulars that contain only small pieces of text. The question is: how to make a tabular the width of the line, with 6 columns; the columns 1, 2, 5 and 6 are of equal widths and the widths of columns 3 and 4 are only one half. As possible solution:

```
\begin{tabu} to\linewidth{ |X[2]|X[2]|X|X|X[2]|X[2]|} \hline
1 & 2 & 3 & 4 & 5 & 6 \\ \hline
\end{tabu}
```

1	2	3	4	5	6
---	---	---	---	---	---

But the text in each cell is very short: one single character, and you prefer the table to be tight, but don't know the exact width of the whole:

```
\begin{tabu} spread 0pt{ |X[2]|X[2]|X|X|X[2]|X[2]|} \hline
1 & 2 & 3 & 4 & 5 & 6 \\ \hline
\end{tabu}
```

1	2	3	4	5	6
---	---	---	---	---	---

But now it's definitely too narrow, then give it some more space:

```
\begin{tabu} spread 2in{ |X[2]|X[2]|X|X|X[2]|X[2]|} \hline
1 & 2 & 3 & 4 & 5 & 6 \\ \hline
\end{tabu}
```

1	2	3	4	5	6
---	---	---	---	---	---

`tabu spread` is useless with long columns: the following tabular was made with this preamble:

```
\begin{tabu} spread 3cm{ @{}X[9]X[4]|X|}
```

"Like the air we breathe, Sherlock Holmes is everywhere. His pipe-smoking, deer stalkered image peers at us from ads in Yellow Pages, to signs for neighbourhood crime-watch; from billboards to the classroom; from film and television to the public library, and now over the Internet. He long ago transcended the boundaries of 19th Century London³ to become an international best-seller and has been accepted as part of British folklore. Holmes is alive to millions."

There the text was too long, and `tabu spread` behaves as if you didn't give it a target.

The result of this example is the same as if one had written `\begin{tabu}to\linewidth.`

Sherlock Holmes

The “official” web site: http://www.sherlockholmes.com/
--

In the preamble, `@{}` means that the margin is removed.

Negativ width coefficients for X columns

```
\tabulinestyle{3pt ForestGreen}
\begin{tabu}{|X[-1m]|X[c m]|}
\tabucline - \savetabu{FirstNegativTest}
$\begin{tabu}{(X[-1$]X[-1$c])}
\alpha & \beta \\
\gamma & \delta + \epsilon + \zeta + \eta + \theta
\end{tabu}$
&
This is a tabu with negativ width coefficients for \texttt X columns
\\ \tabucline -
\end{tabu}
```

$\begin{pmatrix} \alpha & \beta \\ \gamma & \delta + \epsilon + \zeta + \eta + \theta \end{pmatrix}$	This is a tabu with negativ width coefficients for X columns
--	--

3. Capital of the U.K. (too see a linked footnote)

$\begin{pmatrix} \alpha & \beta \\ \gamma & \delta + \epsilon + \zeta + \eta + \theta \end{pmatrix}$	And this is the same with <code>\tabulinesep</code> set to <i>2pt</i> .
--	---

Multicolumn in tabu

`\tabuphantomline`

The process of `\multicolumn` implies the T_EX primitive `\omit` which discards the tabular preamble for the spanned columns. Discarding the preamble means discarding the information about the widths of the columns. This explains why the following example does not work properly:

```
\begin{tabu}{|X|X|X[2]|} \tabucline-
\multicolumn2{|c|}{Hello} & World \\ \tabucline-
\end{tabu}
```

The correct result can be obtained by the mean of a phantom line, that will remain invisible unless your preamble contains special `@` or `!` columns that prints some text:

```
\begin{tabu}{|X|X|X[2]|} \tabucline-
\multicolumn2{|c|}{Hello} & World \\ \tabucline-
\tabuphantomline
\end{tabu}
```

Hello	World
-------	-------

Remember you may need `\tabuphantomline` in conjunction with `\savetabu` and `\usetabu` with `\multicolumn`. Even if it is possible to add a `\tabuphantomline` in any line of the `tabu`, it is a good practice to append it *at the end* of the `tabu`, for it may introduce undesirable side effects on vertical alignment otherwise, when `tabu` is nested inside another tabular.

In particular, `\tabuphantomline` should not be followed by `\cr` or `\\` or `\tabularnewline...`

The need for this command could disappear in a future release, but this requires a complete new implementation of `\multicolumn...`

2.4 `\tabulinesep` and `\extrarowsep` – Mastering vertical space

```
\tabulinesep = <dimen>
\tabulinesep = ^<dimen>
\tabulinesep = _<dimen>
\tabulinesep = ^<dimen>_<dimen>
\tabulinesep = _<dimen>^<dimen>
```

`\tabulinesep` sets the *minimal* vertical space allowed between the cell content and the cell border. The macro may be prefixed by `\global` (even inside a `\noalign` group)⁴.

It is possible to set the “top limit” (a T_EX dimension called `\abovetabulinesep`) and the “bottom limit” independently with the syntaxes:

```
\tabulinesep = ^<dimen>           sets \abovetabulinesep
\tabulinesep = _<dimen>           sets \belowtabulinesep
\tabulinesep = _<dimen>^<dimen>  sets \belowtabulinesep and \abovetabulinesep.
```

These parameters can be used in text and math modes to give more vertical space between lines, especially when using math formulae.

Examples (with `\tracingtabu = 3` and `interfaces-\papergraduate` to see the struts):

4. However `\tabulinesep` is not a dimension ! You can't test, for example, `\ifdim \tabulinesep > 0pt` ! Test `\abovetabulinesep` and `\belowtabulinesep` instead, if needed.

$\frac{\partial\Phi}{\partial\theta}$	$\frac{d\theta}{dt}$	$\frac{\partial\Phi}{\partial\theta}$	$\frac{d\theta}{dt}$	$\frac{\partial\Phi}{\partial\theta}$	$\frac{d\theta}{dt}$
<code>\tabulinesep = 0mm</code>		<code>\tabulinesep = 1mm</code>		<code>\tabulinesep = 3mm</code>	

`\tabulinesep` is a soft parameter, and leads to rows which do not share the same height.

<code>\extrarowsep = <dimen></code>
<code>\extrarowsep = ^<dimen></code>
<code>\extrarowsep = _<dimen></code>
<code>\extrarowsep = ^<dimen>_<dimen></code>
<code>\extrarowsep = _<dimen>^<dimen></code>

`\extrarowsep` is an extra vertical space which is added to each row, unconditionally. `array.sty` provides the T_EX dimension `\extrarowheight` and $\tau_{\text{nb}}c$ provides `\extrarowdepth` in addition. As a result, the rows can share the same height/depth but the spacing is not dynamic. `\tabulinesep` can be used even with positive values for `\extrarowsep`, for `tabu` inserts only one strut per row and vertical spacing computations are possible in all cases.

The macro can be prefixed by `\global` as well, even inside a `\noalign` group⁵.

Set `\extrarowheight` and `\extrarowdepth` to different values, with the syntaxes:

<code>\extrarowsep = ^<dimen></code>	sets <code>\extrarowheight</code> <code>\extrarowdepth</code> is unchanged
<code>\extrarowsep = _<dimen></code>	sets <code>\extrarowdepth</code> <code>\extrarowheight</code> is unchanged
<code>\extrarowsep = _<dimen>^<dimen></code>	sets <code>\extrarowdepth</code> and <code>\extrarowheight</code> .

Both `\extrarowheight` and `\extrarowdepth` are scaled by `\arraystretch` (a scaling *macro*⁶ of `array.sty`) if `\arraystretch > 1...`

These parameters can be used in text and math modes.

Examples (with `\tracingtabu = 3` and `interfaces-\papergraduate` to see the struts):

Standard c column	X[-1,c] columns	Math mode	Mixed: X[-1c]X[-1c\$]
One Two	One Two	α β	First $\frac{\partial\Phi}{\partial\theta}$
Three Four	Three Four	$\frac{\Phi}{\theta}$ Γ_x^t	Second $\frac{d\theta}{dt}$
<code>\extrarowsep = 3mm</code>			

Standard c column	X[-1,c] columns	Math mode	Mixed: X[-1c]X[-1c\$]
One Two	One Two	α β	First $\frac{\partial\Phi}{\partial\theta}$
Three Four	Three Four	$\frac{\Phi}{\theta}$ Γ_x^t	Second $\frac{d\theta}{dt}$
<code>\extrarowsep = 0mm</code>			

5. However `\extrarowsep` is not a dimension ! You can't test, for example, `\ifdim \extrarowsep > 0pt` ! Test `\extrarowheight` and `\extrarowdepth` instead, if needed.

6. `\arraystretch` is not a dimension but a macro that stores a scaling factor.

2.5 tabu in math mode

On the left, you can see the famous Maxwell-Lorentz equations for electromagnetic field in vacuum, published in 1873.

$$\begin{cases} \operatorname{div} \vec{E} = \frac{\rho}{\epsilon_0} \\ \operatorname{div} \vec{B} = 0 \\ \operatorname{rot} \vec{E} = -\frac{\partial \vec{B}}{\partial t} \\ \operatorname{rot} \vec{B} = \mu_0 \vec{j} + \mu_0 \epsilon_0 \frac{\partial \vec{E}}{\partial t} \end{cases}$$

In this example, the big `tabu` is: `\begin{tabu} to\linewidth {XX[-1$]}`.

The nested `tabu` (in math mode) uses `delarray` shortcut: its preamble is: `\begin{tabu}{\rl}`.

`\tabulinesep` has been set to `2pt`. Horizontal rules are `booktabs \toprule` and `\bottomrule`.

array	tabu	tabu spread 1em
$\left \begin{array}{cc} \alpha & \beta \\ \gamma & \delta \end{array} \right $	$\left \begin{array}{cc} \alpha & \beta \\ \gamma & \delta \end{array} \right $	$\left \begin{array}{cc} \alpha & \beta \\ \gamma & \delta \end{array} \right $

Here, vertical lines are made with `delarray` shortcuts: `\begin{tabu} spread 1em |{cc}|`

Vertical lines inside the tabular preamble gives: $\left| \begin{array}{cc} \alpha & \beta \\ \gamma & \delta \end{array} \right|$

This was an example of `\savetabu... \usetabu` to keep the alignment.

3 Lines leaders and colors inside tabu

3.1 First important remark

The features provided in this section are quite experimental: they are not generally taken for good typography. You can use `TNbC` with package `booktabs` for example, which provides properly designed commands for horizontal rules in tabulars. `arydshln` is pretty good too, but it modifies a huge amount of macros of `array.sty`, something that `TNbC` does not.

Lines in `tabu` printed in this document are mostly made with `booktabs`.

3.2 Vertical lines: | has an optional parameter

Inside `tabu` environment, the vertical line marker `|` has an *optional* argument which is the width of the vertical rule. The default width remains `\arrayrulewidth` of course. The optional argument for `|` can also contain the name of a color. *color names* are only possible, not a color specification by the mean of a color model. The width of the line if specified, must come before the color name and... as for `X columns` parameters, commas are optional.

Example:

	<p>The <code>tabu</code> you see on the left was made with the code on the right.</p>	<pre>\begin{tabu}{ [5pt] c c [5pt] } Hello & World \end{tabu}</pre>
	<p>The <code>tabu</code> you see on the left was made with the code on the right.</p>	<pre>\begin{tabu}{ [5pt red] c c [5pt Indigo] } Hello & World \end{tabu}</pre>

This example was printed inside a `tabu*` whose preamble is: `X[-1m]X[m]X[-2m]`

It is not a necessary to protect the optional argument with braces: `[{...}]`. because `TNbC` takes care the `|` token to be rewritten before any other column type (the same for `tabu X columns`, and `sunitx S columns`). The rewriting process is divided into three stages under control inside a `tabu` environment.

T_{NbC} [rev.2.8 release] ©2010 – 2011
 Use `tabu*` to insert Verbatim environments !

3.3 Multiple `\firstline` and `\lastline`

```
\firstline [extratabsurround] make multiple lines !
\firstline [extratabsurround]\hline
\lastline [extratabsurround]
\lastline [extratabsurround]\hline
```

`\firstline` and `\lastline` are defined in `array.sty` and can be used to preserve the alignment of text, when using horizontal lines. Besides, the optional argument can be used to change (locally) the `\extratabsurround` dimension.

The example of `array` documentation is:

Tables with no line commands used	versus	Tables with no line commands used
tables with some line commands used.		tables with some line commands used.
with <code>\firstline</code> and <code>\lastline</code>		with <code>\hline</code> (text alignment is not preserved)

Now with `tabu` you can make double, triple (or more) `\firstline` or `\lastline` as in:

Top alignment	<pre>Tables \begin {tabu}{t}{c} with no\line \ commands \ used \end {tabu} versus tables \begin {tabu}{t}{ c } \firstline \hline \hline \hline with some \line \ commands \ \lastline \hline \hline \hline \end {tabu} used.</pre>	
Bottom alignment	<pre>Tables \begin {tabu}{t}{c} with no\line \ commands \ used \end {tabu} versus tables \begin {tabu}{b}{ c } \firstline \hline \hline \hline with some \line \ commands \ \lastline \hline \hline \hline \end {tabu} used.</pre>	

`\firstline \firstline \firstline` is equivalent to: `\firstline \hline \hline`
and also to: `\firstline \hline \hline \hline`

But the optional argument must come in *first position*: `\firstline [extratabsurround] ...`

The same for `\lastline`.

In yellow you can see the `\extratabsurround` strut, because `\tracingtabu = 3` for this `tabu`

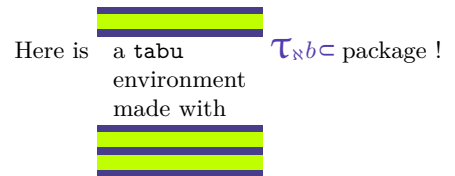
3.4 More style for lines

```
\taburulecolor {<rule color>}
\taburulecolor |<double rule sep color>|{<rule color>}
```

`\taburulecolor` sets (in a “locally-global” way) the color to be used for `\hline`, `\firsthline`, `\lasthline` and also vertical lines if the standard line style is used (the standard line style is active after `\tabulinestyle {}` or after `\tabureset`).

The optional parameter enclosed by vertical bars: `|<double rule sep color>|` is the color to set between two adjacent rules. If not specified, double (or triple...) rules are separated by a vertical space (`\vskip`).

```
\taburulecolor |lime|{DarkSlateBlue}
\arrayrulewidth=1mm \doublerulesep=2mm
Here is
\begin{tabu}spread 0pt {X[-1]}
      \firsthline \hline
a tabu \\\
environment \\\
made with \\\ \lasthline [5mm]\hline \hline
\end{tabu} \TabU package !\par
And the next paragraph follows...
```



And the next paragraph follows...

```
\tabulinestyle {<line style specification>}
```

`\tabulinestyle` sets the line style for vertical (|) and horizontal lines (*ie.* `\tabucline`: `\hline`, `\firsthline` etc. are not modified by `\tabulinestyle`)

The line specification is of the form:

```
3pt rule color on 4pt dash color off 5pt gap color
rule color on 4pt dash color off 5pt gap color
on 4pt dash color off 6pt gap color
3pt rule color
on 4pt dash color
off 5pt
3pt
Named style defined by \newtabulinestyle
```

Well... any parameter is optional. Obviously the rule color is the same as the dash color and the former overwrites the latter if both are given.

Your color names can contain spaces but:

- If the first character in the line specification is not a letter, then it is taken as a dimension: the thickness of the line. Otherwise, the default thickness is used *ie.* `\arrayrulewidth`.
- Your color names must not contain any series of characters that match one the patterns:

```
on?          off?
```

where `?` is a character of category 12, different from `!` and possibly preceded by spaces. I don't think this is a real limitation...

```
\newtabulinestyle {<style=line spec., style=line spec., ...>}babel
```

This command defines a line style to be used in the first optional argument of `\tabucline` (horizontal lines) or the optional argument of `|` (vertical lines) or with `\tabulinestyle` (locally-global style).

Style names and color names are `babel`-protected.

```
\tabucline [style or spec.]{start-end}
```

`\tabucline` is an attempt to give a versatile command to make horizontal lines:

- `\tabucline` is pretty good with vertical lines even if the thickness of the line grows up,

- `\tabucline` takes care of `\extrarowheight`,
- `\tabucline` can make horizontal dashed lines, with a pgf/TikZ syntax:
`\tabucline [width] ondash offgap]{first column-last column}`
- alternatively, you can give `\tabucline` a `\hbox` to make a leader with it: The `<spec.>` must then begin with `\hbox`, `\box` or `\copy`,
- finally you can give `\tabucline` a color *name*, after the line specification.

Any parameter can be omitted.

[1pt on 1.5pt off 2pt]	<code>\tabucline [1pt on 1.5pt off 2pt]{1-4}</code>	draws a horizontal dashed line of width 1pt. Dashes are 1.5pt long and gap width is 2pt. The line is drawn between columns 1 and 4. Here there are only 2 columns and the line stops at column 2.
[1.5pt]	<code>\tabucline [1.5pt]{-}</code>	draws a horizontal solid line of width 1.5pt between the first and the last column.
default	<code>\tabucline {2-}</code>	draws a horizontal solid line of width <code>\arrayrulewidth</code> between the second column and the last one.
[on 2pt red]	<code>\tabucline [on 2pt red]{-5}</code>	draws a horizontal dashed line between columns 1 and 5 of width <code>\arrayrulewidth</code> . Dashed are 2pt long and gap width is 4pt (the default).

Define the line style	Use the line style
<code>\newtabulinestyle {myline=0.4pt on 2.5pt off 1pt red}</code>	<code>\tabucline [myline]{-}</code>

Or use a leader or a box to make a leader with it directly in the argument of `\tabucline`

`\tabucline [\hbox {$\scriptstyle \star $}]{1-3}`

Dashed or dotted And below is the default	Dash Gap This one was thick	Dash Gap... ...or leader Interesting ?
--	--------------------------------------	---

3.5 Automatic horizontal lines and row colors

`\everyrow {code}`

`\everyrow` can be used to insert horizontal lines automatically:

```

\begin{tabu}to .5\linewidth{cX[2mc]X} \tabucline[1pt]-
\everyrow{\tabucline[on 2pt]-}
This is      & a small example & of a \texttt{tabu}           \\
which       & automatically & inserts                     \\
a horizontal & line after    & each of its row \everyrow{} \\
pt]-
\end{tabu}
```

This is	a small example	of a tabu
which	automatically	inserts
a horizontal	line after	each of its
		row

`\everyrow` can be used in `longtabu` as well. The syntax is like `\everycr`: a token-like syntax, and braces are mandatory: `\everyrow {argument}`.

`\taburowcolors [first line]<number>\{first .. last}`

`\taburowcolors` sets the alternate colors to be used on every row of the tabular. The command can be used before a `tabu` environment or inside it, at the end of a row.

The optional parameter `[first line]` tells the first row from which background colors are starting – this optional parameter has no effect when `\taburowcolors` is used at the end of a row: background are starting immediately in this case.

`<number>` is the number of colors in the color series. If not specified, it defaults to 2 (for alternate rows color).

Finally `<first>` and `<last>` are the first and the last colors in the colorseries.

Example:

```
\taburowcolors [2] 3{Crimson!30 .. ForestGreen!40}
\taburulecolor |GreenYellow|{OrangeRed}
\arrayrulewidth=1pt \doublerulesep=1.5pt
\everyrow{\hline\hline}
\begin{tabu} {X[-1]X}
This is      &just a test      & \\
and i think &&it will          & \\
look        &&rather bad      & \\
for         &&i've not        & \\
chosen      &&the colors      & \\
with care.  &&i can't         & \\
say         &&less...         & \\
\taburowcolors 2{Crimson .. ForestGreen}
1           &&This is Crimson & \\
2           &&This is ForestGreen & \\
3           &&This is Crimson & \\
4           &&This is ForestGreen & \\
\end{tabu}
```

This is	just a test
and i think	it will
look	rather bad
for	i've not
chosen	the colors
with care.	i can't
say	less...
1	This is Crimson
2	This is ForestGreen
3	This is Crimson
4	This is ForestGreen

`\tabureset`

To go back to “standard” parameters, T_{NbC} provides the command `\tabureset` which basically does:

```
\tabulinesep = 0pt           \extrarowsep = 0pt           \extratabsurround = 0pt
\tabulinestyle {}           \everyrow {}           \taburulecolor ||{}
\taburowcolors {}
```

4 Modifying the font and the alignment in one row: `\rowfont`

`\rowfont [alignment]{font specification}`

Inside a `tabu` environment, you can modify the font for each cell in a row. `\rowfont` has priority over column font specification, exactly like `\rowcolor` (package `colortbl`) has priority over `\columncolor`.

The alignment of each cell in one row can also be changed to:

- l = left or for `ragged2e` settings: L
- c = center C
- r = right R
- j = justify J

Any other value for the optional `<alignment>` parameter is silently ignored. If `ragged2e` is not loaded, L R C and J are synonymous with the lowercase equivalent.

```
\begin{tabu}{|X|X[-1]|} \tabucline-
\rowfont[c]\bfseries
This & &Is & \\
\xpackage{tabu} & &\xpackage & \\
\\ \tabucline[on 2pt,blue]-
\\ \tabucline[off 2pt blue]-
```

```
\rowfont[r]\itshape
for          &\textt{tabu} and \textt{longtabu} \\ \tabucline-
\end{tabu}
```

This	Is
tabu	package
<i>for</i>	<i>tabu and longtabu</i>

5 Saving and restoring a tabu

```
\savetabu {<user-name>}
```

The command `\savetabu` can be used at the end of any line of a `tabu` environment to save the parameters of a `tabu` environment. The saving is always global. This allows to easily make tabulars which share exactly the same shape throughout your document. This can also be used as a kind of `tabbing` environment which is able to remember the tabs positions...

If the `<user-name>` has been used before, an info is displayed in the `.log` file and the previous settings are overwritten.

With the `\tracingtabu > 0`, informations about the saved parameters are reported in the `.log` file.

Recalling saved parameters are done with `\usetabu` (complete recovery) or `\preamble` (partial recovery of the preamble only).

```
\usetabu {<user-name>}
```

`\usetabu` is the complement of `\savetabu`: it can be put alone in the `tabu` preamble instead of the usual columns specifications to restore any previous settings saved with `\savetabu`.

The `<user-name>` must exist otherwise, you get an error.

`\usetabu` is a help to **make several tabulars of exactly the same shape, same target, same preamble**. The only parameter that can be changed is the optional vertical position parameter for the whole tabular.

`\usetabu` does not work with `longtabu`.

`\usetabu` locally restores:

- the preamble⁷.
- the vertical position [c], [b] or [t], unless another position is specified.
- the target width of the `tabu` in points: the saved target width does not contain any control sequence: it is fixed and stored in points.
- the width of `tabu X` columns: those widths are not calculated any more – even in the case of negativ coefficients – and `X` columns are directly transformed into `p`, `m` or `b` columns of the same widths as the ones that where calculated at the time of `\savetabu`
- `\tabcolsep` (or `\arraycolsep` in math mode) `\extrarowheight`, `\extrarowdepth`, `\arraystretch` and `\extratabsurround`
- `\arrayrulewidth`, `\doublerulesep` and the parameters for `\everyrow` `\taburulecolor`, `\tabulinestyle`, and `\taburowcolors`
- `\minrowclearance`, (package `colortbl`)

`\abovetabulinesep` and `\belowtabulinesep` are not restored, because they are related to the content of the tabular rather than to its shape.

Example:

```
\tabcolsep=12pt \extrarowsep=1mm
\tabulinestyle{on 1pt ForestGreen}
```

7. The complete `\halign`-preamble is restored.

```
\begin{tabu}to .7\linewidth{|XXX|X[c]|} \savetabu{mytabu} \tabucline -
This & is & tabu & package \\ \tabucline -
\end{tabu}
```

This	is	tabu	package
------	----	------	---------

```
\tabureset
\begin{tabu}{\usetabu{mytabu}} \tabucline -
\multicolumn3{c}{This is tabu} & package \\ \tabucline -
\tabuphantomline
\end{tabu}
```

This is tabu			package
--------------	--	--	---------

If one day you use `tabu`, you will have the idea to restore a `tabu` while modifying its target, or adding new columns... `\savetabu` and `\usetabu` have not been thought for this purpose, and you may have unexpected results.

`\preamble {<user-name>}`

`\preamble` can also be used after `\savetabu`. This is a variant of `\usetabu` that locally restores:

- the `tabu` (or `longtabu`) preamble.
- the vertical position [c], [b] or [t] (or [c], [l] or [r] for `longtabu`), unless another position is specified.
- the `tabu` / `longtabu` target width, unless another target is specified.

Any other tabular parameter is not restored.

Put `\preamble {<user-name>}` alone inside the `tabu` (or `longtabu`) preamble in place of the usual columns specifications.

`\preamble` works exactly as if you defined a `custom environment` for `tabu`.

`\preamble` works with `longtabu` .

Example (continued...):

```
\tabulinestyle{1pt off1pt}
\begin{tabu} to\linewidth{\preamble{mytabu}} \tabucline -
This & is & tabu & package \\ \tabucline -
\end{tabu}
```

This	is	tabu	package
------	----	------	---------

`\tabcolsep`, `rule colors` etc. are not restored from `\savetabu`: the only `tabu` preamble is restored.

6 Some other features

6.1 Printing numbers inside `tabu` with `numprint` and `siunitx`

`\tabudecimal`

T_{NbC} provides a *facility* to print numbers inside columns. This facility is not implemented to replace `siunitx` `S` and `s` columns or `numprint` `n` and `N` columns or other packages that provide alignment such as `warpcol`, `dcolumn` or `rccol`. It just make easy to apply a macro you get already on each number in a column of a `tabu`.

`\tabudecimal` has been developed mainly because it makes possible to align numbers inside `tabu` `X` columns.

`\tabudecimal {⟨user-macro⟩}`

`\tabudecimal` can be used in the preamble of a `tabu` before a column specification. The `⟨user-macro⟩` is a macro with one parameter that has to be defined before.

Example with `\numprint`:

```
\def\usermacro#1{\numprint[\officialeuro]{\zap@space #1 \@empty}}
\nprouddigits{2} \npprintnull \npthousandsep{\,} \npunitseparator{~}
```

<code>\rowfont [c]{\bf }</code> January & February \\	January	February	...
12.324 & 745.32 \\	12,32 €	745,32 €	...
21.13 & 0 \\	21,13 €	0,00 €	...
213.3245 & 12.342 \\	213,32 €	12,34 €	...
2143.12 & 324.325 \\	2 143,12 €	324,33 €	...

Example with `\SI`:

```
\def\usermacro#1{\SI[group-four-digits=true, % thousand separator
round-mode=places, % round numbers
round-precision=2, % with 2 decimal digits
round-integer-to-decimal=true, % add trailing 0 if necessary
per-mode=symbol]{#1}{\officialeuro\per\kilo\gram}}
```

```
\begin{tabu}spread 0pt{ |[GreenYellow]*2{>{\tabudecimal \usermacro}X[r] |[GreenYellow] } ...
```

January	February	...
12.32 €/kg	745.32 €/kg	...
21.13 €/kg	0.00 €/kg	...
213.32 €/kg	12.34 €/kg	...
2 143.12 €/kg	324.33 €/kg	...

As you can see, the columns widths are exactly the same, whatever their content.

Here `\tabulinesep` has been set to `3pt`.

You should know how it works...

Yes you should know how it works to avoid problems. `tabu` has a small scanner based on `\futurelet` to grab all numbers, blank spaces, commas and dots + and - sign and also the letter e and E for exponents. The scanner stops as soon as something else than a number, blank space, comma, dot, +, -, e, E is found, and even if it is a macro that contains a number.

This explains why there is `\zap@space` in the definition of `\usermacro`: because the scanner scans blank spaces and because `\numprint` does not allow blank spaces in its mandatory argument, quite strangely...

6.2 Paragraph indentation

`tabu` takes care of paragraph indentation when it is used with X columns and its default target, no matter if it has been loaded or not with the `linegoal` option. Example with L^AT_EX default: `\parindent = 20pt`.

This is `tabu` with its default target in an indented paragraph.

This is `tabu` with its default target, preceded by `\noindent`

This is `tabularx` with target: `\linewidth` in an indented paragraph.

This is `tabularx` with target: `\linewidth`, preceded by `\noindent`

6.3 delarray shortcuts

When you enclose your tabular with math delimiters using `delarray` shortcuts, $\tau_{\text{N}bc}$ tries to reach its target for the whole: the tabular and the delimiter(s). You can see the difference:

with overflow hboxes
(17.5pt and 17.8pt two wide)
for tabularx

$\left(\begin{array}{c} \text{This is } \text{tabu} \text{ with delar-} \\ \text{ray shortcuts for paren-} \\ \text{thesis around.} \end{array} \right)$	$\left\{ \begin{array}{c} \text{This is } \text{tabu} \text{ with de-} \\ \text{larray shortcuts for curly} \\ \text{brackets around.} \end{array} \right\}$
$\left(\begin{array}{c} \text{This is } \text{tabularx} \text{ with de-} \\ \text{larray shortcuts for paren-} \\ \text{thesis around.} \end{array} \right)$	$\left\{ \begin{array}{c} \text{This is } \text{tabularx} \text{ with de-} \\ \text{larray shortcuts for curly} \\ \text{brackets around.} \end{array} \right\}$

Here `\tabulinesep = 3mm`

7 Differences between `tabu`, `tabular`, `tabularx` and `longtable`

7.1 Paragraph indentation

See [Paragraph indentation](#)

7.2 Custom environments

Unlike `tabularx`, it is possible to define your own environment using `tabu`:

```
\newenvironment{foo}
  {\begin{tabu}{X[1.2]|[1pt gray]X}}
  {\end{tabu}}
```

`tabu` environment, even when `X` columns are used, may appear in the definition of your custom `tabular` environment.

You can also use the commands `\savetabu` `\preamble` (or `\usetabu`) for this purpose.

7.3 Inversion of tokens

When you typeset the following `tabular`:

```
\begin{tabular}{|>{\bfseries}>{ before }|<{ one }<{ two }|}
  cell content
\end{tabular}
```

You get the following result: `| before cell content two one |`

→ The word *before* is not bold, and *two* comes before *one*.

The reason is explained in the documentation of `array.sty`, and is related to the `array` environment in math mode when using `\newcolumntype`.

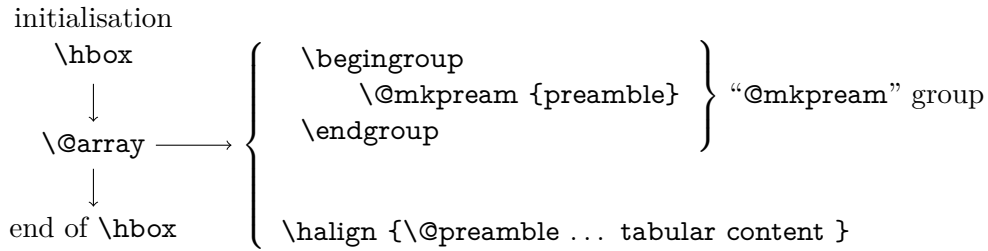
This rather strange inversion of tokens may be justified in math mode (otherwise, errors may occur) but not in text mode in our opinion. Inside a `tabu` environment, when not in math mode, the tokens are not reversed and you get the intuitively expected result:

`| before cell content one two |`

In math mode however, tokens are in the reverse order in the `tabu` environment like they are in the `array` environment.

7.4 Improved process for rewriting columns *(for keen readers)*

Any tabular that does not split across pages is made with the following process:



For more details, see the [Flow chart of expansion](#).

`\@mkpream` works in two times inside a (semi-simple) group:

First the rewriting process:

Each special column in the tabular preamble is transformed into one the columns defined by `array.sty`.

Second the building of the `\halign` preamble:

The “rewritten preamble” is parsed and transformed in a preamble for the T_EX primitive `\halign`. The result is stored into the `\@preamble` macro.

Any special columns of `tabu` are defined only inside the “@mkpream” group.

In the following example, you get an error with `tabular` and no error with `tabu`. With `tabular`, and `siunitx` `S` column, the *rewriting process* is as follow:

<pre> \documentclass {minimal} \usepackage {numprint,siunitx,xcolor} \usepackage {tabu} \begin {document} \begin {tabular}{*2{S[color=green]}} 123,45 \end {tabular} \begin {tabu}{*2{S[color=green]}} 123,45 \end {tabu} \end {document} </pre>	<p>Inside <code>tabular</code>:</p> <ol style="list-style-type: none"> 1) Rewrite <code>S</code>: not found because inside <code>{...}</code> 2) Rewrite <code>*</code> 3) Rewrite <code>n</code> column defined by package <code>numprint</code> Then the ‘n’ in <code>green</code> is rewritten → problem <p>Inside <code>tabu</code>:</p> <ol style="list-style-type: none"> 1) Rewrite <code>*</code> 2) Rewrite <code> </code> (there is none here) <p>go back</p> <ol style="list-style-type: none"> 3) Rewrite <code>*</code> 4) Rewrite <code> </code> 5) Rewrite <code>S</code> 6) Rewrite <code>n</code> → not found because <code>S</code> was rewritten before, according to <code>siunitx</code> definition.
---	--

The process of rewriting columns is usually longer inside `tabu` than inside `tabular`, but conversely `tabu` with `X` columns is optimised compared to `tabularx`, because the preamble is built only once, and not rebuilt before each trial as `tabularx` does. Thus `tabu` is much quicker than `tabularx`.

The process of rewriting is very sensitiv to the order in which columns are actually rewritten. This becomes critical when columns are defined with an optional argument like `tabu X` and `|` columns or `siunitx` `S` column.

8 The package options

8.1 The debugshow package option

`\tracingtabu`
`\tracingtabu = 1, 2, 3 or 4`

The control sequence `\tracingtabu` has the same effect as the `debugshow` option:

- $\tau_{\text{nb}}c$ will report the widths it computes at each attempt to read the target, when X columns are used.
- Saved informations on the `tabu` are reported in the `.log` file when `\savetabu` is used.

`\tracingtabu = 2` gives more information on the measures of the natural widths.

`\tracingtabu = 3` shows the struts inserted inside the `tabu` environment and gives more information about the measures of the height and depth of every row.

`\tracingtabu = 4` displays information on the insertions made by `\tabucline`.

Typical information in the `.log` file:

```
(tabu) Try      tabu X      tabu Width      Target      Coefs      Update
(tabu) 1)  386.67296pt  797.34592pt  386.67296pt  2.0pt  -205.33649pt
(tabu) 2)  181.33647pt  386.67294pt  386.67296pt  2.0pt   0.00002pt
(tabu) 2) Target reached (hfuzz=0.1pt) *****
```

What does it mean?

- 1) The first attempt was performed with $X=386.67296\text{pt}$
 The `tabu` width (797.34592pt) exceeded the target by 410.67296pt .
 Thus X has been updated: $410.67296\text{pt} / 2 = 205.33649\text{pt}$ and then:
 $X = 386.67296\text{pt} - 205.33649\text{pt} = 181.33647\text{pt}$
- 2) The second attempt lead to a `tabu` width of 386.67294pt : the target is reached.
 The final width of each X column is the product of `tabu X` by its width coefficient.

8.2 The delarray package option

`delarray` option has the single effect to load `delarray.sty` for delimiters shortcuts around `tabu`. Delimiters shortcuts work both in math and text mode.

8.3 The linegoal package option

With the `linegoal` option, the default target for `tabu` with X columns is `\linegoal` instead of `\linewidth`. The `linegoal` package must be loaded and compilation must be done with `pdfLATEX`, otherwise, a warning is displayed and the `linegoal` option has no effect: the default target remains `\linewidth`. `\linegoal` works with `pdfLATEX` in `pdf` mode **and in dvi** mode.

If for some reason, you wish to turn down the `linegoal` option in your document, you can say (in a group for example): `\let\tabudefaulttarget=\linewidth`

In any case, specifying the target overwrites the default: `\begin {tabu} to\linewidth`

9 Corrections of some bugs (*available only inside tabu*)

9.1 Correction for `colortbl` and `arydshln`: compatibility with `delarray`

Both `colortbl` and `arydshln` forget the control sequence `\@arrayright` in their implementation, quite strangely because both of them take care of `\@arrayleft`. As a result, `delarray` shortcuts for delimiters around a tabular does not work if `colortbl` and/or `arydshln` are loaded.

Those control sequences are used by the `delarray` package to put variable size delimiters around

the array:

<code>\begin {tabu} \{{X}.</code>		<code>\left \{\begin {tabu}{X}</code>
<code>...</code>		<code>...</code>
<code>\end {tabu}</code>	is like:	<code>\end {tabu} \right .</code>

9.2 Correction for arydashln: @ columns

A bug in `\adl@xarraydashrule: !-arg` columns (class 1) and `@-arg` columns (class 5) should be treated the same as far as rules are concerned.

With this correction, the “known problem number 1” in `arydashln` documentation is solved.

10 To do for even better tabus

In decreasing order of priority:

- ➔ Make double `\tabucline` compatible with `colortbl \doublerulesepcolor`
- ➔ Multiple `\tabucline` between different columns: extended specs:
`\tabucline [line spec]{start-stop, start-stop}[line spec]{start-stop} ...`
- ➔ Reimplement `\multicolumn` in order to allow the `X` token in `\multicolumn` preamble. Provide `\multicell` to allow spanning columns and rows at the same time.
- ➔ Presently, `longtabu` with `X` columns works only if `\LTchunksize` is greater than the number of rows. I compiled a `longtabu` of 56 pages on my PC with `\LTchunksize = 2000` without problem. Presently `\LTchunksize` is set to 10 000 during trials when `longtabu` contains `X` columns.
- ➔ Make `\tabucline` work with page breaks (one line on the top of the page, one line on the bottom of the previous).

11 Technical notice and Implementation

11.1 Drawing a tabular - The $\mathcal{T}_{\mathbb{N}b\subset}$ approach

$\mathcal{T}_{\mathbb{N}b\subset}$ has a different approach than almost any other package providing facilities for tabulars. `colortbl` and `arydshln` both put the cells contents into a box for measuring purpose, and then use the dimensions of each box to make their setups:

`colortbl` needs the dimensions of the box to put a rule in the background of the cell,

`arydshln` needs the dimensions to set the length of its leaders (dash lines).

This is achieved by modifying the macros defined in `array.sty` to insert columns inside the `\halign` preamble.

Instead, $\mathcal{T}_{\mathbb{N}b\subset}$ proceeds as follow:

1. It first measures (if there are some negative width coefficients, or if `tabu spread` is used) the natural widths of the cells / the columns,
2. Then it always measures the height and depth of each cell / row,
3. Thereafter, the tabular is printed exactly as if `array.sty` was entitle to print it: no “extra” boxing of the cells material. The measurements have been stored and can be used to set the struts (only one per row) and the lengths of vertical leaders.
4. No macros of `array.sty` is modified at stage 3.

$\mathcal{T}_{\mathbb{N}b\subset}$ material inserted in the tabular for vertical leaders, `\rowfont` etc. is put inside the special “free” tokens provided by `array.sty`:

- A vertical leader is put inside a ! column: `!{vertical leader}`
- Changing font and alignment in one row requires some setup in > tokens: `>{rowfont material}`.

This way, the commands of `array.sty` that build each column definition (or preamble, in the sense of `\halign`) are never modified.

11.2 Algorithms

`tabu to target`

The algorithm of `\tabu@arith` computes the desired widths to reach the target. In any case, only one measure of the tabular is required to get the widths for all columns. Here we describe the method with an example and some equations too to show that this handle all cases in generality.

Notations and initialisation of X In the case of `tabu to the target` $T = 300$ is given : it is the target specified by the user or the default `tabu target` which is `\linewidth - \langle parindent correction \rangle` or `\linegoal`. Each X column has a width coefficient which is given too (or default to 1). The coefficients are: c_1, c_2, \dots, c_n .

X is the main dimension that drives the widths of all columns with a non negative coefficient, and limit the widths of columns with a negative coefficient.

Then we have first:

Coef c_i	c_1	c_2	c_3	c_4	c_5	c_6	Σ	Δ
	-1	-2	-5	-2	2	3	15	
Target T	300							

Some coefficients are negative and we have to measure the natural widths of the corresponding columns, for columns always have a width:

$$\lambda_i = \begin{cases} c_i \cdot X & \text{if } c_i > 0 \\ \text{Min} (|c_i| \cdot X, \nu_i) & \text{if } c_i < 0 \end{cases} \quad \text{with} \quad \nu_i \leq T \quad \forall i$$

ν_i is the “natural width of the column” in the sense that it is the maximum of the natural widths of each cell in the *i*th X column, limited to the `tabu target`: $\nu_i \leq T \quad \forall i$.

The whole width of the tabular is always:

$$\text{wd}(\text{table}) = \sum_i \lambda_i + \text{incompressible material} \left\{ \begin{array}{l} \bullet \backslash \text{tabcolsep} \\ \bullet \text{ vertical lines/leaders thickness} \\ \bullet \text{ non X columns} \end{array} \right.$$

and should finally be equal to T , by the correct computation of the λ_i .

So what is X at first ? Columns that have a non negative coefficients always have a width equal to $\lambda_i = c_i \cdot X$ therefore, if we only have non negative coefficients, we can safely set:

$$X = \frac{T}{\sum_i c_i}$$

then: $\sum_i \lambda_i = \sum_i c_i \cdot X \geq T$ at the first trial. But this is not the same if some coefficients are negative, because in this case the column width λ_i can shrink until its natural width ν_i and may be until to $0pt$! And then if every column has a negative coefficient, one of them can have a width close to the target T . We have to ensure that the first measure of the natural widths does not limit them artificially:

$$\begin{aligned} \forall i \quad c_i < 0 &\implies |c_i| \cdot X \geq T \\ \exists c_i > 0 &\implies \sum_{c_i > 0} c_i \cdot X \geq T \end{aligned}$$

And finally, for the measure: $X = \text{Max} \left[\text{Max}_{c_i < 0} \frac{T}{|c_i|}; \frac{T}{\sum_{c_i > 0} c_i} \right]$

Coef c_i	c_1	c_2	c_3	c_4	c_5	c_6	Σ	Δ
	-1	-2	-5	-2	2	3	15	
Target T	300							
X	300							
ν_i	10	300	80	80				
λ_i	10	300	80	80	600	900	1970	1800

First step of the algorithm: reducing the width After having measured the table we get: $\text{wd}(\text{table}) = 2100$. The *incompressible material* is $2100 - 1970 = 130$ wide and the gap to the target is $\Delta = 2100 - 300 = 1800$.

We now choose a new value for X :

$$\sum_i \lambda_i = \sum_i \text{Min}(\nu_i; c_i \cdot X) + \sum_{c_i > 0} c_i \cdot X \leq \sum_i |c_i| \cdot X$$

Let's try $X' = \frac{\sum_i \lambda_i - \Delta}{\sum_i |c_i|}$ so that $\sum_i \lambda'_i \leq \sum_i |c_i| \cdot X' \leq \sum_i \lambda_i - \Delta$:

Coef c_i	c_1	c_2	c_3	c_4	c_5	c_6	Σ	Δ
	-1	-2	-5	-2	2	3	15	
Target T	300							
X	300							
ν_i	10	300	80	80				
λ_i	10	300	80	80	600	900	1970	1800
X'	11.33							

$X' = \frac{1970 - 1800}{15} = \frac{170}{15} = 11.33$ Note that the computation of X' does not involve any measurement.

Coef c_i	c_1	c_2	c_3	c_4	c_5	c_6	\sum	Δ
	-1	-2	-5	-2	2	3	15	
Target T	300							
X	300							
ν_i	10	300	80	80				
λ_i	10	300	80	80	600	900	1970	1800
X'	11.33							
λ'_i	10,00	22,67	56,67	22,67	22,67	34,00	168,67	-1.33

Here we are in the case where the table width:

$$\begin{aligned} \text{wd}(\text{table}) &= \sum_i \lambda_i + \text{incompressible material} = T + \Delta \\ \implies \sum_i \lambda'_i + \text{incompressible material} &\leq \sum_i \lambda_i - \Delta + I = T \end{aligned}$$

Without any measure, we can say that the final table width will be less than the target, if we choose X' . The free space to share among the X columns (computed with X') is now $\Delta' = T - (\sum_i \lambda'_i + I) = 300 - (168.67 + 130) = -1.33$, where I is the incompressible material.

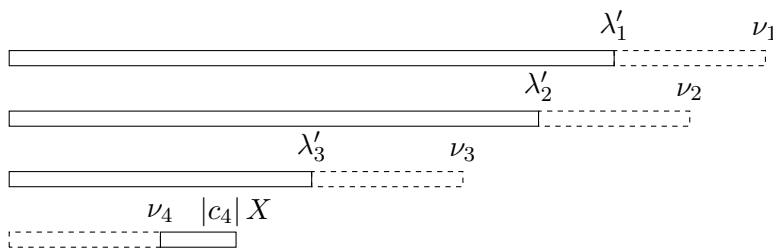
Giving space We say that a column is *saturated* (ie.full) if its natural width is greater than $|c_i| \cdot X$, or all the same that $\lambda_i < \nu_i$. We also will consider that the columns with $c_i > 0$ have a “natural width” which is always equal to $c_i \cdot X$: in other words, a column with a non negative coefficient is always *saturated*.

Giving space (or “refunding” space) to the columns must be done in priority to the *saturated* columns. If all columns are finally underfull, then we will distribute the extra space to each, according a distribution rule. But this case can only occur if $\forall i \ c_i < 0$ because we first choosed X so that:

$$X \geq \frac{T}{\sum_{c_i > 0} c_i}$$

and hence, the sum of the widths of the “non negative” columns exceeds the target.

Let’s rank the columns widths:



we first give space to the saturated columns 1, 2 and 3

Because of the saturation, the total amount of space to give: $|\Delta|$ shall be shared among the columns according to their widths coefficients. We shall not give too much space: the columns shall remain saturated. Let $0 < \epsilon \leq |\Delta|'$ the amount of space to give, then after the operation:

$$\begin{aligned} \lambda_1'' + \lambda_2'' + \lambda_3'' &= \lambda'_1 + \lambda'_2 + \lambda'_3 + \epsilon \\ &= |c_1| X' + |c_2| X' + |c_3| X' + \epsilon \end{aligned}$$

Let’s say $X'' = X' + \frac{|\Delta|'}{\sum_{c_i \text{ saturated}} |c_i|}$ then it’s possible, without any measure, to compute:

$$\sum_{c_i \text{ saturated}} \lambda''_i + \nu_4 \leq \sum_{c_i \text{ saturated}} |c_i| \cdot X'' + \nu_4 \leq \sum_i |c_i| X' + \Delta' = \sum_i \lambda'_i + \Delta' \leq T - I$$

Or for clarity: $\sum_i \lambda_i'' + I = \text{wd}(\text{table}) \leq T$ and the new free space to share is now :

$$\Delta'' = \left| T - \left(\sum_i \lambda_i'' + I \right) \right|$$

At each step of the computation, and without any measure but the first, X grows, Δ decreases, and finally the target is reached for X such that $\Delta \leq \text{hfuzz}$.

Coef c_i	c_1	c_2	c_3	c_4	c_5	c_6	\sum	Δ
	-1	-2	-5	-2	2	3	15	
Target T	300							
X	300							
ν_i	10	300	80	80				
λ_i	10	300	80	80	600	900	1970	1800
X'	11.33							
λ'_i	10,00	22,67	56,67	22,67	22,67	34,00	168,67	-1.33
X''	11.43							
λ''_i	10,00	22,86	57,14	22,86	22,86	34,29	170,00	0

Now if the width of the table is less that the target, because 1) every column has a negative coefficient and 2) their natural widths are so small than the tabular don't fill the wanted horizontal space, the algorithm artificially raise the natural widths, according to a linear distribution:

$$\lambda'_i = \lambda_i + \Delta \cdot \frac{\lambda_i}{\sum_i \lambda_i} = \nu_i + \Delta \cdot \frac{\nu_i}{\sum_i \nu_i} = \nu_i \cdot \left(1 + \frac{\Delta}{\sum_i \nu_i} \right)$$

tabu spread dimen

The case of `tabu spread` is interesting and quite complex...

Here, the aim of the game is to give a target to the table, depending on its natural width. `tabu` has a default target (`\linewidth` in general, but it is possible to `\let \tabudefaultrightarget` to another value... for example `\linegoal`) which is a maximum for the final target of `tabu spread`. The case where the spread is `Opt` is not simpler nor more difficult.

If every column has a negative coefficient, it's rather easy because either the table exceeds the target, and then the new target will be the default target (the *maximum*), or the table width is less than the default target and we fix the new target to be that width + the spread, in the limit of the default target.

The condition that must hold on coefficient is not restritive if every column has a negative coefficient because if you say, for example: $X = \text{Max}_i \frac{\nu_i}{|c_i|}$ then:

$$\sum_i \lambda_i = \sum_i \text{Min}(\nu_i; |c_i| \cdot X)$$

is true. It's always possible to find a X such that the behaviour announced in the documentation is observed !

Then let's get some non negative coefficients. The natural widths of such columns must be measured, but the natural width of the tabular is not the same, for the proportions between column widths – expressed by their positive coefficient c_i – must be respected.

The real natural width of the tabular, which observe the proportions between columns with a non negative coefficient is:

$$\text{wd}(\text{table}) + \text{Max}_{c_i > 0} \left(\frac{\nu_i}{c_i} \right) \times \sum_{c_i > 0} c_i - \sum_{c_i > 0} \nu_i > \text{wd}(\text{table})$$

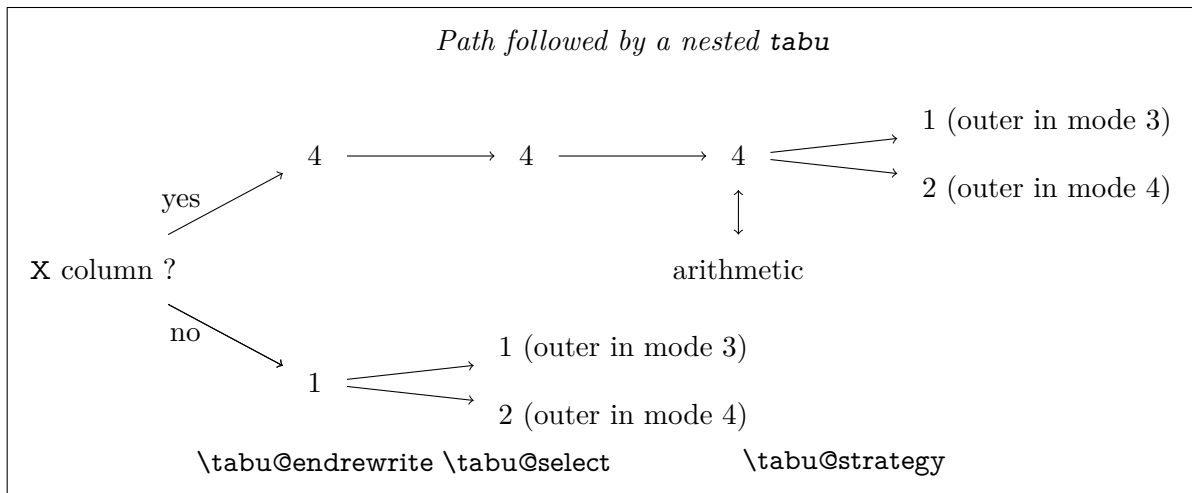
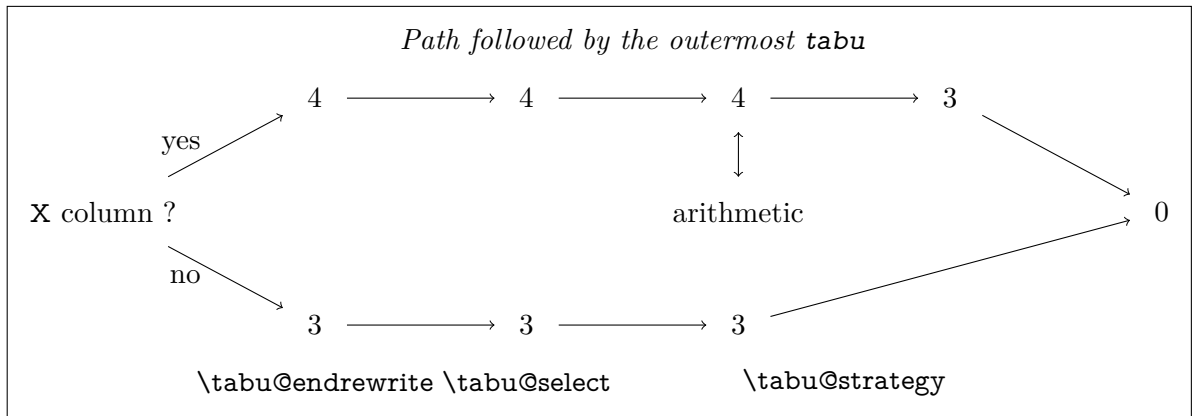
This quantity is computed, τ_{nb} adds the spread and fix the new target to the sum, in the limit of the default target.

Then X is initialized such that:
$$X = \text{Max} \left[\begin{array}{c} \frac{T}{\sum_{c_i < 0} |c_i|}; \frac{T}{\sum_{c_i > 0} c_i} \end{array} \right]$$

and the algorithm described in the former section works, without any new measurement of the tabular. Unless this was not possible or deemed inconvenient for clarity, the code is presented in the same order it executes.

11.3 The tabu strategies

	Not nested (outer)	Nested
	<code>\count@</code> condition	<code>\count@</code> condition
<code>\tabu@endrewrite</code>	3 no X column 4 X columns	0 outer is in mode 0 1 no X column 3 X column
<code>\tabu@select</code>	3 or 4 needs trials	0 outer in mode 0 ⇒ print 1 outer in mode 3 2 from 1 in <code>\tabu@endrewrite</code> if outer in mode 4 3 or 4 needs trials
<code>\tabu@strategy</code>	0 print out 3 Vertical measure 4 Horizontal measure	1 Exit in vertical measure (outer in mode 3) 2 Exit with a rule (outer in mode 4) 4 Horizontal measure (nested in <code>coef < 0</code> or spread)



11.4 Identification and Requirements

\mathcal{T}_{NBC} requires `array.sty` and `varwidth.sty`. The package namespace is `tabu@`.

```

1 (*package)
2 \NeedsTeXFormat{LaTeX2e}[2005/12/01]
3 \ProvidesPackage{tabu}[2011/02/26 v2.8 - flexible LaTeX tabulars (FC)]
4 \RequirePackage{array}[2008/09/09]
5 \RequirePackage{varwidth}[2009/03/30]

```

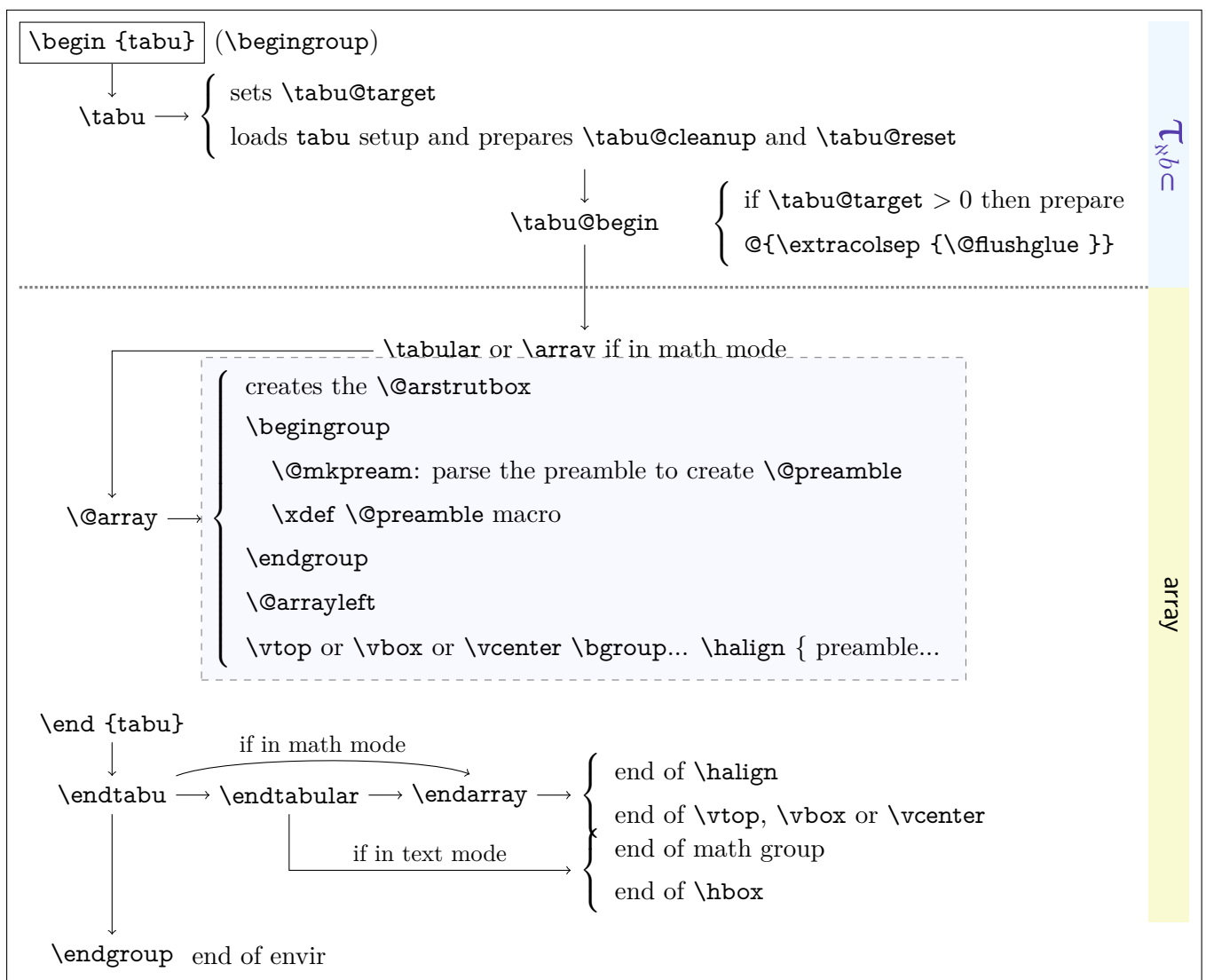
Minimal catcode ascertaining for loading \mathcal{T}_{NBC} in good conditions:

```

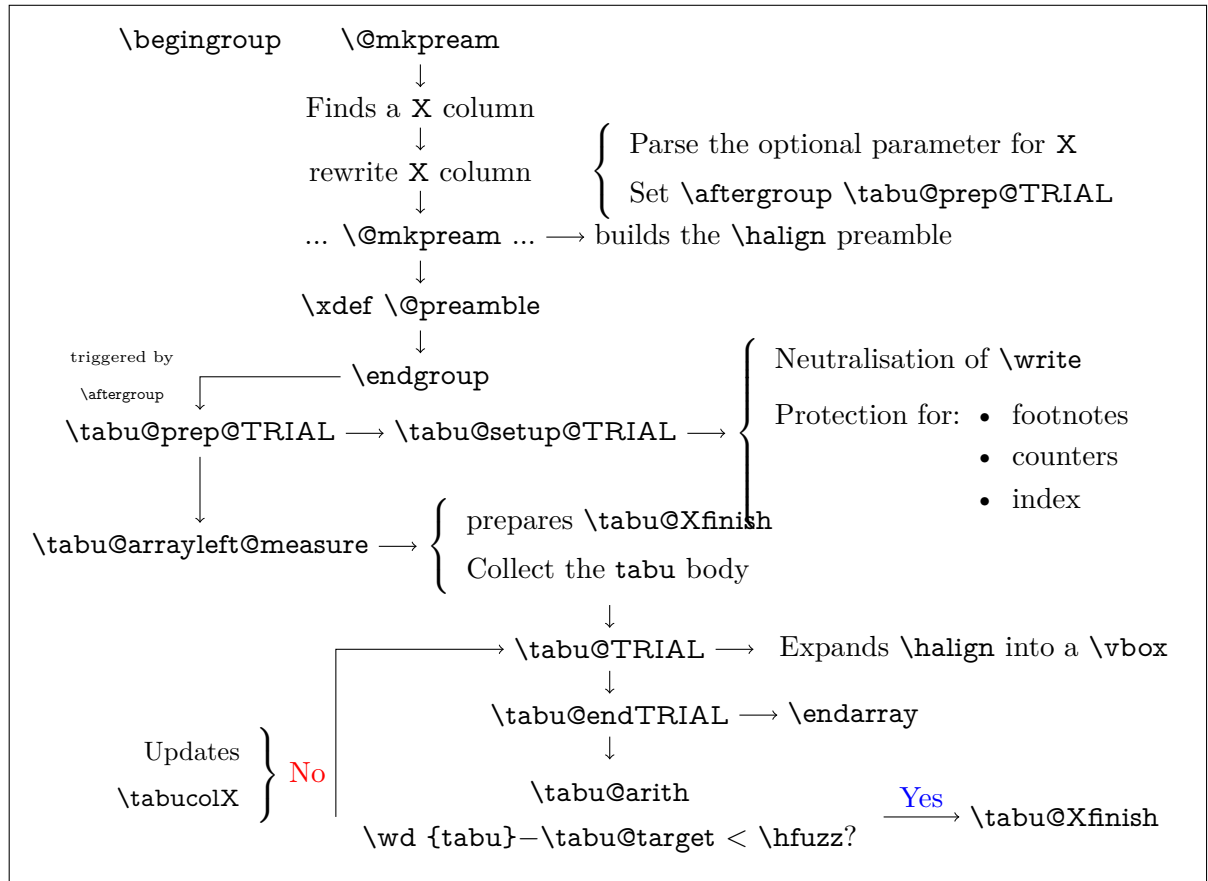
6 \AtEndOfPackage{\tabu@AtEnd \let\tabu@AtEnd \@undefined}
7 \let\tabu@AtEnd\@empty
8 \def\TMP@EnsureCode#1={%
9   \edef\tabu@AtEnd{\tabu@AtEnd
10      \catcode#1 \the\catcode#1}%
11   \catcode#1=%
12 }% \TMP@EnsureCode
13 \TMP@EnsureCode 33 = 12 % !
14 \TMP@EnsureCode 58 = 12 % : (for siunitx)
15 \TMP@EnsureCode 124 = 12 % |
16 \TMP@EnsureCode 36 = 3 % $ = math shift
17 \TMP@EnsureCode 38 = 4 % & = tab alignment character
18 \TMP@EnsureCode 32 = 10 % space
19 \TMP@EnsureCode 94 = 7 % ^
20 \TMP@EnsureCode 95 = 8 % _

```

11.5 Flow chart of expansion



tabu to with X column The important part of the job is made inside the dashed box above: `\@mkpream` expands the columns definitions, which can be user defined. Hopefully, it does its job inside a group, therefore a user-column can set a macro to be expanded `\aftergroup`. This implementation allows much modifications in the tabular preparation, without any change in the macros of `array.sty`.



tabu spread with X column

In the case of “`tabu spread`” with `X` columns, the process is the same as the one described for “`tabu to`” with `X` columns. However, the first trial is different because we have first to measure the *natural width* of the tabular. The process is the following:

- `\tabu@target` is first set to `\linewidth` (or `\linegoal` with the `linegoal` package option).
- The `X` column corresponds to a `\vbox` with `\hsize` fixed to `\tabu@target`.
- Inside this `\vbox` the cell content is written into a `\hbox` whose width is limited to `\tabu@target`. This `\hbox` is captured into the box register `\tabu@box`.
- At the end of the cell, the `\badness` of the `\hbox` is checked:
 - if the `\badness` is > 1000 then the text is too long and “`tabu spread`” is useless: `tabu to \tabu@target` give the same result.
 - Otherwise, we get the natural width of the cell content by:

$$\setbox \tabu@box \hbox {\unhbox \tabu @box}$$
- At the end of the first trial, `\tabu@spreadarith` checks if:

$$\text{width}(\text{tabular}) + \text{spread} < \text{\linewidth} \text{ (or \linegoal)}$$

- if not, then `tabu to \tabu@target` give the same result
- Otherwise, the target for `tabu to` will be:

$$\text{width}(\text{tabular}) + \text{spread} - \underbrace{\sum_i \text{natural widths } X_i + \text{Max}_i \left(\frac{\text{natural width } X_i}{\text{coef}_i} \right) \times \sum_i \text{coef}_i}_{\text{minimal natural width that can be obtained with the given coefs}}$$

And the next trial will be done as if the user called “`tabu to`” with this target.

11.6 Some constants

Here we define the constants used by \mathcal{T}_{Nbc} : T_EX registers and a few *helper* macros.

When working inside a tabular (*ie.* `\halign`) each cell is a T_EX group. Probably the most important property of each register defined here is whether it is global or not. A *local* register does not suffer, never, any global assignment.

T_EX registers

taburow L^AT_EX counter that globally stores the value of the current row. It is updated at `\everyrow`, rather than at `\everycr`⁸. `\thetaburow` expands to the (arabic) number.

This counter can be read by the user, but she **must not change its value** because it is used internally to store the height/depth of every row, for vertical spacing adjustment (and vertical leaders).

\tabu@nbcols T_EX counter that – locally – saves the total number of columns of the `tabu`. Special `@` and `!` columns are not counted (they are not *real* columns for `\halign`, but only insertions into the preamble).

The value is used by `\tabucline` to ensure that the leader does not jut out over the last column...

\tabu@cnt T_EX counter that – locally – stores the number of trials. Incidentally, it is also temporarily used to parse the width coefficient for `X` columns, during the rewriting process.

\tabu@Xcol T_EX counter that – locally – stores the number of `tabu X` columns. Defined while rewriting the `X` token, it is used in the specification of the width of the column (`\tabu@hsize {Rank of the X column}{coef}`).

It is also used to store the natural width of `X` columns (in the cases of a negativ coefficient or if `tabu spread` is used).

\tabu@alloc A global counter whose initial value (`-1`) is incremented for each nested tabular. The end of the outermost tabular globally resets the value to `-1`. `\tabu@nested` stores locally the value of `\tabu@alloc` and is therefore the “index” of the current tabular (the one that is actually in construction).

This influences the initialisation process (cf. `\tabu@setup` and `\tabu@init`).

\tabu@start They are used locally by `\tabucline` and `\everyrow` while parsing the parameters: this is, for clarity, the local name for `\@tempcnta` and `\@tempcntb`.

```

21 \newcount \c@taburow          \def\thetaburow {\number\c@taburow}
22 \newcount \tabu@nbcols
23 \newcount \tabu@cnt
24 \newcount \tabu@Xcol
25 \let\tabu@start \@tempcnta
26 \let\tabu@stop  \@tempcntb
27 \newcount \tabu@alloc  \tabu@alloc=\m@ne
28 \newcount \tabu@nested
29 \def\tabu@alloc@{\global\advance\tabu@alloc \@ne \tabu@nested\tabu@alloc}

```

\tabu@target T_EX dimen that – locally – stores the `tabu` target (either “to” or “spread”).

\tabu@spreadtarget T_EX dimen that – locally – stores the `tabu` spread given by the user.

\tabu@naturalX T_EX dimen that – globally – stores the total natural widths of the `X` columns, in the cases of negativ coefficients and/or `tabu spread`. The value is reset to `0pt` at `\everyrow`, and *maxima/minima* are stored into the macro `\tabu@naturalXmin` and `\tabu@naturalXmax`: those are required for the algorithm of `tabu spread` (`\tabu@spreadarith`).

\tabu@colX T_EX dimen that – locally – stores the width corresponding to the preamble token `X[1]`: the standard width of `X` columns.

\tabu@DELTA This is for clarity, the local name of `\@tempdimc` in `\tabu@arith`.

\tabu@thick They are used locally by `\tabu@getline`, while parsing the parameters for a line specification. This is for clarity, the local name for `\@tempdima`, `\@tempdimb` and `\@tempdimc`.

\tabu@on

\tabu@off

⁸. Package `xcolor` defines the `\rownum` T_EX counter, which is globally updated at `\everycr`. Hence this `\rownum` counter is not reliable in case the user invokes `\cline` or `\cmidrule` for example...

```

30 \newdimen \tabu@target
31 \newdimen \tabu@spreadtarget
32 \newdimen \tabu@naturalX
33 \newdimen \tabu@colX
34 \let\tabu@DELTA \@tempdimc
35 \let\tabu@thick \@tempdima
36 \let\tabu@on \@tempdimb
37 \let\tabu@off \@tempdimc

```

\tabu@Xsum T_EX dimen that – locally – stores the sum of all width coefficients for X columns. This is required to fix the initial value for `\tabu@colX` and then in the algorithms (`\tabu@arith` and `\tabu@arithnegcoef`).

```
38 \newdimen \tabu@Xsum
```

\extrarowdepth `array.sty` defines `\extrarowheight` as a T_EX dimen register: the extra height to be finally added to each row of a table. \mathcal{T}_{Nbc} defines `\extrarowdepth` in addition: the *extra depth*. Though `\extrarowheight` and `\extrarowdepth` can be set by the user, the official interface is `\extrarowsep`.

\abovetabulinesep T_EX dimensions `\abovetabulinesep` and `\belowtabulinesep` store the minimum allowed vertical space between the contents of the cells and their borders. Their values are ignored if non positive. Though they can be set by the user, the official interface is `\tabulinesep`.

The philosophy and the technics are similar to the one provided by the `cellspace` package. However, limitations of `cellspace` are lifted (nested `tabu` environments, use of colors... see the [cellspace limitations](#) in the revision history). \mathcal{T}_{Nbc} inserts only one strut per line, whose name is `\@arstrut`.

\tabustrutrule The T_EX dimen `\tabustrutrule` is here only for debugging purpose: its value must be *Opt*. It behaves mostly like T_EX primitive `\overfullrule`, and allow to see the struts introduced in the tabular, and to control vertical spacing. Setting `\tabustrutrule` to a positive value has no effect unless `\tracingtabu` is ≥ 3 . The official interface is `\tracingtabu = 3`.

```

39 \newdimen \extrarowdepth
40 \newdimen \abovetabulinesep
41 \newdimen \belowtabulinesep
42 \newdimen \tabustrutrule \tabustrutrule \z@

```

\tabu@thebody This token stores – locally – the collected content of the `tabu` environment during the measuring process.

\tabu@footnotes Token that globally stores the footnotes inside the `tabu` environment, for `\insert` does not work inside such a level of groupings...

```

43 \newtoks \tabu@thebody
44 \newtoks \tabu@footnotes

```

\tabu@box Stores – locally – the whole `tabu` when an attempt to adjust X columns is performed.

\tabu@arstrutbox While the `\@arstrutbox` may redefined globally at the end of each line (for vertical spacing adjustment), we define a new box and `\let \@arstrutbox` to be that box inside the `tabu` environment.

Hence, the `\@arstrutbox` used by other tabular environment does not suffer any modification.

\tabu@hleads Those boxes are used to built horizontal and vertical leaders. In order not to rebuilt the boxes every time a leader is inserted, the box is globally defined if a line style is specified (*via* `|[line style]` or `\tabucline [line style]{...}` or `\tabulinestyle {line style}`).

```

45 \newsavebox \tabu@box
46 \newsavebox \tabu@arstrutbox
47 \newsavebox \tabu@hleads
48 \newsavebox \tabu@vleads

```

Switches

- `\iftabu@colortbl` The global switch `\iftabu@colortbl` is used by `\rowfont` when modifying the alignments, because `colortbl` changes the glues put inside the `\halign` preamble to make standard alignments. This switch is set `At Begin Document`.
- `\iftabu@siunitx` Global switch set `\AtBeginDocument`. `true` if `siunitx` package is detected.
- `\iftabu@measuring` This switch is somewhat *magic* in the sense that it has several meanings... It is temporarily set to `true` by `\tabu@arith` in the trial group, to say that the `tabu` did not reach its target yet. It is also set to `true` in the `\@mkpream` group when the first `X` column is encountered in the preamble. Finally, it is true in the `trialS` group when the outermost tabular is in strategy number 2 or number 3.
- `\iftabu@spread` A switch whether “`tabu spread`” is used or not. A nested `tabu` inside a `X` column whose coefficient is negative has a default target set to `spread 0pt`.
- `\iftabu@negcoef` A switch set to true in case of negativ coef (natural width if less than `X[coef]`).
- `\iftabu@everyrow` A very important global switch: `true` when outside any `tabu` environment, `true` as well when inside a cell of a `tabu`, but globally set to `false` at `\everycr` and therefore inside any `\noalign` command. This allows to insert leaders (by `\omit \span \omit \cr \noalign {...}`) or first/last line corrections only once, even if `\everycr` is executed more than once.
- `\iftabu@long` Finally the swith `\iftabu@long` is set to `true` inside `longtabu` and to `false` inside `tabu`. This is convenient because some setup are slightly different between `tabu` and `longtabu`.

```

49 \newif \iftabu@colortbl
50 \newif \iftabu@siunitx
51 \newif \iftabu@measuring
52 \newif \iftabu@spread
53 \newif \iftabu@negcoef
54 \newif \iftabu@everyrow
55 \def\tabu@everyrowtrue {\global\let\iftabu@everyrow \iftrue}
56 \def\tabu@everyrowfalse{\global\let\iftabu@everyrow \iffalse}
57 \newif \iftabu@long

```

- `\iftabuscantokens` `\iftabuscantokens` is the switch for whether or not `tabu` will use `\scantokens`. Though the user can set `\tabu@rescan` `\iftabuscantokens` to `\iftrue` or `\iffalse`, the official interface is `tabu*` .

It does not make sense to use `\scantokens` in a nested `tabu`: only the outermost `tabu` can use `\scantokens`, for the environment body must be collected with care !

`\tabu@rescan` is the helper macro for scanning tokens.

```

58 \newif \iftabuscantokens
59 \def\tabu@rescan {\tabu@verbatim \scantokens }

```

Some helper macros

- `\tabu@gobblespace` Two macros which are needed when scanning tokens with `\futurelet`.
- `\tabu@gobbletoken`
`\tabu@gobblex` This gobbles the character number 10 in ASCII (`^^J` in T_EX).
- `\tabu@ifenvir` Checks if the current environment is `tabu` or `longtabu` (for `\multicolumn` inside `tabu`).
- `\tabu@modulo` Computes the modulo (for `\taburowcolors`). The method is taken from H.O. `intcalc` package.

```

60 \def\tabu@gobblespace #1 {#1}
61 \def\tabu@gobbletoken #1#2{#1}
62 \def\tabu@gobbleX{\futurelet\@let@token \tabu@gobblex}
63 \def\tabu@gobblex{\if ^^J\noexpand\@let@token \expandafter\@gobble
64 \else\ifx \@sptoken\@let@token
65 \expandafter\tabu@gobblespace\expandafter\tabu@gobblex
66 \fi\fi}
67 }% \tabu@gobblex
68 \def\tabu@X{^^J}

```

```

69 {\obeyspaces
70 \global\let\tabu@spxiiii= % saves an active space (for \ifx)
71 \gdef\tabu@@spxiiii{ }}
72 \def\tabu@ifenvir {% only for \multicolumn
73   \expandafter\tabu@if@nvir\csname\@currenvir\endcsname
74 }% \tabu@ifenvir
75 \def\tabu@if@nvir #1{\csname @\ifx\tabu#1first\else
76   \ifx\longtabu#1first\else
77   second\fi\fi oftwo\endcsname
78 }% \tabu@ifenvir
79 \def\tabu@modulo #1#2{\numexpr\ifnum\numexpr#1=\z@ 0\else #1-(#1-(#2-1)/2)/(#2)*(#2)\fi}

```

\tabu@strtrim Trimming spaces at low cost...

```

80 {\catcode`\&=3
81 \gdef\tabu@strtrim #1{% #1 = control sequence to trim
82   \ifodd 1\ifx #1\@empty \else \ifx #1\space \else 0\fi \fi
83   \let\tabu@c@l@r \@empty \let#1\@empty
84   \else \expandafter \tabu@trimspaces #1&#1\@nnil
85   \fi
86 }% \tabu@strtrim
87 \gdef\tabu@trimspaces #1&#2\@nnil{\let\tabu@c@l@r=#2\tabu@firstspace .#1& &#2}%
88 \gdef\tabu@firstspace #1#2#3 &{\tabu@lastspace #2#3&}
89 \gdef\tabu@lastspace #1&#2&#3{\def #3{#1}%
90   \ifx #3\tabu@c@l@r \def\tabu@c@l@r{\protect\color{#1}}\expandafter\remove@to@nnil \fi
91   \tabu@trimspaces #1&#3\@nnil}
92 }% \catcode

```

\tabu@sanitizearg Sanitize an argument (babel compliant).

```

93 \def\tabu@sanitizearg #1#2{%
94   \csname \ifcsname if@safe@actives\endcsname % <babel>
95     @safe@activetrue\else
96     relax\fi \endcsname
97   \edef#2{#1}\tabu@strtrim#2\@onelevel@sanitize#2%
98   \expandafter\expandafter\def\expandafter#2\expandafter{#2}%
99 }% \tabu@sanitizearg

```

\tabu@textbar The character | may have a special category code inside the document, depending on the language setting or for example, | can be the delimiter shortcut for verbatim. We use \scantokens to allow an \ifx test even if the category code of | changes along the compilation.

```

100 \def\tabu@textbar #1{\begingroup \endlinechar\m@ne \scantokens{\def\:{|}}%
101   \expandafter\endgroup \expandafter#1:% !!! semi simple group !!!
102 }% \tabu@textbar

```

\tabu@everyrow@bgroup Commands like \everyrow, \taburulecolor, \tabulinestyle, \taburowcolors can be expanded either in a cell or outside a tabu environment or at the end of a row, inside a \noalign group.

To avoid the insertion of an empty math atom (equivalent to \hbox to0pt{ }) we open a semi-simple group rather than a math group if not in \noalign. \toks@ is used to define the local-to-the- \TeX -group setting (post-fixed by @L).

```

103 \def\tabu@everyrow@bgroup{\iftabu@everyrow \begingroup \else \noalign{\ifnum0='}\fi \fi}
104 \def\tabu@everyrow@egroup{%
105   \iftabu@everyrow \expandafter \endgroup \the\toks@
106   \else \ifnum0='{\fi}%
107   \fi
108 }% \tabu@everyrow@egroup

```

Rebuild the \@arstrutbox

\tabu@arstrut The macros rebuilds the \@arstrutbox (a \hbox). With the *debug* variants when `\tracingtabu = 3` and `\tabu@rearstrut` `\tabustrutrule > 0`.

```

109 \def\tabu@arstrut {\global\setbox\@arstrutbox \hbox{\vrule
110   height \arraystretch \dimexpr\ht\strutbox+\extrarowheight
111   depth \arraystretch \dimexpr\dp\strutbox+\extrarowdepth
112   width \z@}%
113 }% \tabu@arstrut
114 \def\tabu@rearstrut {%
115   \@tempdima \arraystretch\dimexpr\ht\strutbox+\extrarowheight \relax
116   \@tempdimb \arraystretch\dimexpr\dp\strutbox+\extrarowdepth \relax
117   \ifodd 1\ifdim \ht\@arstrutbox=\@tempdima
118     \ifdim \dp\@arstrutbox=\@tempdimb 0 \fi\fi
119   \tabu@mkarstrut
120   \fi
121 }% \tabu@rearstrut

```

\tabu@DBG@arstrut This is the “debug” version of `\tabu@arstrut`: used when `\tracingtabu = 3` or more to show the struts inserted in the tabular.

```

122 \def\tabu@DBG #1{\ifdim\tabustrutrule>\z@ \color{#1}\fi}
123 \def\tabu@DBG@arstrut {\global\setbox\@arstrutbox
124   \hbox to\z@{\hbox to\z@{\hss
125     {\tabu@DBG{cyan}\vrule
126     height \arraystretch \dimexpr\ht\strutbox+\extrarowheight
127     depth \z@
128     width \tabustrutrule}\kern-\tabustrutrule
129     {\tabu@DBG{pink}\vrule
130     height \z@
131     depth \arraystretch \dimexpr\dp\strutbox+\extrarowdepth
132     width \tabustrutrule}}}%
133 }% \tabu@DBG@arstrut

```

\tabu@save@decl No inversion on tokens in the `tabu` preamble, when not in math mode.

```

134 \def\tabu@save@decl{\toks\count@ \expandafter{\the\toks\expandafter\count@
135   \nextchar}}%
136 \def\tabu@savdecl{\ifcat$\d@llarend\else
137   \let\save@decl \tabu@save@decl \fi % no inversion of tokens in text mode
138 }% \tabu@savdecl

```

\tabu@finalstrut

```

139 \def\tabu@finalstrut #1{\unskip\ifhmode\nobreak\fi\vrule height\z@ depth\z@ width\z@}

```

Disable some commands during trials

\tabuDisableCommands Following the model of `hyperref` `\pdfstringdefDisableCommands`, `\tabuDisableCommands` allow the user to change the definition of some commands during the trial loops, by the mean of a hook to be expanded by `\tabu@setstrategy`.

```

140 \newcommand*\tabuDisableCommands {\g@addto@macro\tabu@trialh@@k }
141 \let\tabu@trialh@@k \@empty

```

\tabu@nowrite A trick (from the T_EX-book) to forbid `\write` when a trial is done on the `\halign`.

\tabu@noxfootnotes Disable footnotes during trials.

```

142 \def\tabu@nowrite #1#{{\afterassignment}\toks@}
143 \let\tabu@write\write
144 \let\tabu@immediate\immediate
145 \def\tabu@WRITE{\begingroup
146   \def\immediate\write{\aftergroup\endgroup}

```

```

147             \tabu@immediate\tabu@write}%
148 }% \tabu@WRITE
149 \expandafter\def\expandafter\tabu@GenericError\expandafter{%
150             \expandafter\tabu@WRITE\GenericError}
151 \def\tabu@warn{\tabu@WRITE\PackageWarning\tabu}}
152 \def\tabu@noxfootnote [#1]{\@gobble}

```

\tabu@nocolor For optimisation purpose, color changes are deactivated during trials, for they do not affect the measures.

```

\tabu@norowcolor 153 \def\tabu@nocolor #1#{\@gobble}
154 \newcommand*\tabu@norowcolor[2][{}

```

siunitx S and s columns management

\tabu@maybesiunitx A macro that encloses the definition of `\tabu@celllalign`, in order to check if the column is a siunitx S (or s) column, and neutralise the setup of `\rowfont` in this case, for siunitx provides its own key=value options to set fonts inside S (or s) columns.

```

155 \def\tabu@maybesiunitx #1{\def\tabu@temp{#1}%
156             \futurelet\@let@token \tabu@m@ybesiunitx}
157 \def\tabu@m@ybesiunitx #1{\def\tabu@m@ybesiunitx {%
158     \ifx #1\@let@token \let\tabu@celleft \@empty \let\tabu@cellright \@empty \fi
159     \tabu@temp}% \tabu@m@ybesiunitx
160 }\expandafter\tabu@m@ybesiunitx \cename siunitx_table_collect_begin:Nn\endcsname
161 \def\tabu@celllalign@def #1{\def\tabu@celllalign{\tabu@maybesiunitx{#1}}}%

```

11.7 Rules, colors and vertical adjustment

`\extrarowsep` and `\tabulinesep`

\extrarowsep `\extrarowsep` makes the assignment for both `\extrarowheight` and `\extrarowdepth`.

The macro may be prefixed by `\global`.

```

162 \newcommand*\extrarowsep{\edef\tabu@C@extra{\the\numexpr\tabu@C@extra+1}%
163     \iftabu@everyrow \aftergroup\tabu@Gextra
164     \else \aftergroup\tabu@n@Gextra
165     \fi
166     \@ifnextchar={\tabu@gobbletoken\tabu@extra} \tabu@extra
167 }% \extrarowsep
168 \def\tabu@extra {\@ifnextchar_%
169     {\tabu@gobbletoken{\tabu@setextra\extrarowheight \extrarowdepth}}
170     {\ifx ^\@let@token \def\tabu@temp{%
171         \tabu@gobbletoken{\tabu@setextra\extrarowdepth \extrarowheight}}%
172     \else \let\tabu@temp \@empty
173         \afterassignment \tabu@setextrasep \extrarowdepth
174     \fi \tabu@temp}%
175 }% \tabu@extra
176 \def\tabu@setextra #1#2{\def\tabu@temp{\tabu@extr@#1#2}\afterassignment\tabu@temp#2}
177 \def\tabu@extr@ #1#2{\@ifnextchar^%
178     {\tabu@gobbletoken{\tabu@setextra\extrarowdepth \extrarowheight}}
179     {\ifx _\@let@token \def\tabu@temp{%
180         \tabu@gobbletoken{\tabu@setextra\extrarowheight \extrarowdepth}}%
181     \else \let\tabu@temp \@empty
182         \tabu@Gsave \tabu@G@extra \tabu@C@extra \extrarowheight \extrarowdepth
183     \fi \tabu@temp}%
184 }% \tabu@extr@
185 \def\tabu@setextrasep {\extrarowheight=\extrarowdepth
186     \tabu@Gsave \tabu@G@extra \tabu@C@extra \extrarowheight \extrarowdepth
187 }% \tabu@setextrasep
188 \def\tabu@Gextra{\ifx \tabu@G@extra\@empty \else {\tabu@Rextra}\fi}
189 \def\tabu@n@Gextra{\ifx \tabu@G@extra\@empty \else \noalign{\tabu@Rextra}\fi}

```

```
190 \def\tabu@Rextra{\tabu@Grestore \tabu@G@extra \tabu@C@extra}
191 \let\tabu@C@extra \z@
192 \let\tabu@G@extra \@empty
```

\tabulinesep `\tabulinesep` makes the assignment for both `\abovetabulinesep` and `\belowtabulinesep`.

The macro may be prefixed by `\global`.

```
193 \newcommand*\tabulinesep{\edef\tabu@C@linesep{\the\numexpr\tabu@C@linesep+1}%
194   \iftabu@everyrow   \aftergroup\tabu@Glinesep
195   \else               \aftergroup\tabu@n@Glinesep
196   \fi
197   \@ifnextchar={\tabu@gobbletoken\tabu@linesep} \tabu@linesep
198 }% \tabulinesep
199 \def\tabu@linesep {\@ifnextchar_%
200   {\tabu@gobbletoken{\tabu@setsep\abovetabulinesep \belowtabulinesep}}
201   {\ifx ^\@let@token \def\tabu@temp{%
202     \tabu@gobbletoken{\tabu@setsep\belowtabulinesep \abovetabulinesep}}}%
203   \else \let\tabu@temp \@empty
204     \afterassignment \tabu@setlinesep \abovetabulinesep
205   \fi \tabu@temp}%
206 }% \tabu@linesep
207 \def\tabu@setsep #1#2{\def\tabu@temp{\tabu@sets@p#1#2}\afterassignment\tabu@temp#2}
208 \def\tabu@sets@p #1#2{\@ifnextchar^%
209   {\tabu@gobbletoken{\tabu@setsep\belowtabulinesep \abovetabulinesep}}
210   {\ifx _\@let@token \def\tabu@temp{%
211     \tabu@gobbletoken{\tabu@setsep\abovetabulinesep \belowtabulinesep}}}%
212   \else \let\tabu@temp \@empty
213     \tabu@Gsave \tabu@G@linesep \tabu@C@linesep \abovetabulinesep \belowtabulinesep
214   \fi \tabu@temp}%
215 }% \tabu@sets@p
216 \def\tabu@setlinesep {\belowtabulinesep=\abovetabulinesep
217   \tabu@Gsave \tabu@G@linesep \tabu@C@linesep \abovetabulinesep \belowtabulinesep
218 }% \tabu@setlinesep
219 \def\tabu@Glinesep{\ifx \tabu@G@linesep\@empty \else {\tabu@Rlinesep}\fi}
220 \def\tabu@n@Glinesep{\ifx \tabu@G@linesep\@empty \else \noalign{\tabu@Rlinesep}\fi}
221 \def\tabu@Rlinesep{\tabu@Grestore \tabu@G@linesep \tabu@C@linesep}
222 \let\tabu@C@linesep \z@
223 \let\tabu@G@linesep \@empty
```

\tabu@Gsave Utility macros to implement the possibility to prefix a macro by `\global`.

\tabu@Grestore

```
224 \def\tabu@Gsave #1#2#3#4{\xdef#1{#1%
225   \toks#2{\toks\the\currentgrouplevel{\global#3\the#3\global#4\the#4}}}%
226 }% \tabu@Gsave
227 \def\tabu@Grestore#1#2{%
228   \toks#2{#1}\toks\currentgrouplevel\expandafter{\expandafter}\the\toks#2\relax
229   \ifcat$\the\toks\currentgrouplevel$\else
230     \global\let#1\@empty \global\let#2\z@
231     \the\toks\currentgrouplevel
232   \fi
233 }% \tabu@Grestore
```


Setting code for every row

\everyrow As long as `tabu` needs to execute some code at `\everycr`, it's not difficult to provide a command to give the user the opportunity to execute its own arbitrary code. However, `\everyrow` will be used almost only with `\hline` (or `\tabucline` or `\midrule`).

`\everyrow` can be changed anywhere inside the `tabu`: at the end of a row, or even inside a cell.

The rows L^AT_EX counter `taburow` must not be changed by the user!.

The settings are saved in a “locally-global” way...

```

234 \newcommand*\everyrow{\tabu@everyrow@bgroup
235           \tabu@start \z@ \tabu@stop \z@ \tabu@evrstartstop
236 }% \everyrow
237 \def\tabu@evrstartstop {\@ifnextchar^%
238   {\afterassignment \tabu@evrstartstop \tabu@stop=}%
239   {\ifx ^\let@token
240     \afterassignment\tabu@evrstartstop \tabu@start=%
241     \else \afterassignment\tabu@everyr@w \toks@
242     \fi}%
243 }% \tabu@evrstartstop
244 \def\tabu@everyr@w {%
245   \xdef\tabu@everyrow{%
246     \noexpand\tabu@everyrowfalse
247     \let\noalign \relax
248     \noexpand\tabu@rowfontreset
249     \iftabu@colortbl \noexpand\tabu@rc@ \fi % \taburowcolors
250     \let\noexpand\tabu@docline \noexpand\tabu@docline@evr
251     \the\toks@
252     \noexpand\tabu@evrh@@k
253     \noexpand\tabu@rearstrut
254     \global\advance\c@taburow \@ne}%
255   \iftabu@everyrow \toks@\expandafter
256     {\expandafter\def\expandafter\tabu@evr@L\expandafter{\the\toks@}\ignorespaces}%
257   \else \xdef\tabu@evr@G{\the\toks@}%
258   \fi
259   \tabu@everyrow@egroup
260 }% \tabu@everyr@w
261 \def\tabu@evr {\def\tabu@evrh@@k           % for internal use only
262 \tabu@evr{}
```

Setting line styles and colors

\newtabulinestyle `\newtabulinestyle {style=spec.,style=spec,style=spec}`

All the job is done by `\tabu@getline`. New line style specification are always defined globally, and can be overwritten without warning...

```

263 \newcommand*\newtabulinestyle [1]{%
264   {\@for \@tempa :=#1\do{\expandafter\tabu@newlinestyle \@tempa==\@nil}}%
265 }% \newtabulinestyle
266 \def\tabu@newlinestyle #1=#2=#3\@nil{\tabu@getline {#2}%
267   \tabu@sanitizearg {#1}\@tempa
268   \ifodd 1\ifx \@tempa\@empty \ifdefined\tabu@linestyle@ 0 \fi\fi
269   \global\expandafter\let
270     \csname tabu@linestyle@\@tempa \endcsname =\tabu@thestyle \fi
271 }% \tabu@newlinestyle
```

\tabulinestyle `\tabulinestyle {style name}` or `\tabulinestyle`line specs / leader

The job is done by `\tabu@getline`. The settings as usual, are stored in a “locally-global” way...

```

272 \newcommand*\tabulinestyle [1]{\tabu@everyrow@bgroup \tabu@getline{#1}%
```

```

273 \iftabu@everyrow
274     \toks@\expandafter{\expandafter \def \expandafter
275         \tabu@ls@L\expandafter{\tabu@thestyle}\ignorespaces}%
276     \gdef\tabu@ls@{\tabu@ls@L}%
277 \else
278     \global\let\tabu@ls@G \tabu@thestyle
279     \gdef\tabu@ls@{\tabu@ls@G}%
280 \fi
281 \tabu@everyrow@egroup
282 }% \tabulinestyle

```

`\taburulecolor` [colortbl](#) provides `\arrayrulecolor`, but the definition is global and must be restored manually after the table. `\taburulecolor` works with the same scheme as `\everyrow`: even if the definition of the rules colors must be global (because we it can be changed inside the tabular) the value is not restored globally at the end of the environment.

Instead, `\tabu@arc@L` stores locally the color definition (*ie.* its definition is relative to the group level before the entry inside the `tabu` environment).

This is the same for `\doublerulesepcolor` (which may be given as an optional argument to `\taburulecolor`): [colortbl](#) makes the definition global, while $\mathcal{T}_N b c$ keeps grouping level into mind (“locally-global” settings).

```

283 \newcommand*\taburulecolor{\tabu@everyrow@bgroup \tabu@textbar \tabu@rulecolor}
284 \def\tabu@rulecolor #1{\toks@{}}%
285     \def\tabu@temp #1##1#1{\tabu@ruledrsc{##1}}\@ifnextchar #1%
286         \tabu@temp
287         \tabu@rulearc
288 }% \tabu@rulecolor
289 \def\tabu@ruledrsc #1{\edef\tabu@temp{#1}\tabu@strtrim\tabu@temp
290     \ifx \tabu@temp\empty \def\tabu@temp{\tabu@rule@drsc@ {}{}}%
291     \else \edef\tabu@temp{\noexpand\tabu@rule@drsc@ {}{\tabu@temp}}%
292     \fi
293     \tabu@temp
294 }% \tabu@ruledrsc@
295 \def\tabu@ruledrsc@ #1#{\tabu@rule@drsc@ {#1}}
296 \def\tabu@rule@drsc@ #1#2{%
297     \iftabu@everyrow
298         \ifx \\\#1#2\\\toks@\let\CT@drsc@ \relax}%
299         \else \toks@\def\CT@drsc@{\color #1{#2}}}%
300     \fi
301     \else
302         \ifx \\\#1#2\\\global\let\CT@drsc@ \relax
303         \else \gdef\CT@drsc@{\color #1{#2}}}%
304     \fi
305     \fi
306     \tabu@rulearc
307 }% \tabu@rule@drsc@
308 \def\tabu@rulearc #1#{\tabu@rule@arc@ {#1}}
309 \def\tabu@rule@arc@ #1#2{%
310     \iftabu@everyrow
311         \ifx \\\#1#2\\\toks@\expandafter{\the\toks@ \def\CT@arc@{}}%
312         \else \toks@\expandafter{\the\toks@ \def\CT@arc@{\color #1{#2}}}%
313         \fi
314         \toks@\expandafter{\the\toks@
315             \let\tabu@arc@L \CT@arc@
316             \let\tabu@drsc@L \CT@drsc@
317             \ignorespaces}%
318     \else
319         \ifx \\\#1#2\\\gdef\CT@arc@{}}%
320     \else \gdef\CT@arc@{\color #1{#2}}}%

```

```

321     \fi
322     \global\let\tabu@arc@G \CT@arc@
323     \global\let\tabu@drsc@G \CT@drsc@
324     \fi
325     \tabu@everyrow@egroup
326 }% \tabu@rule@arc@

```

\taburowcolors `\taburowcolors {number}<number>{first color .. last color}`

The aim of the game is to define the process that will be executed at `\everyrow`.

After that, the usual process for “locally-global” settings is plugged into `\tabu@cleanup` and `\tabu@reset...`

```

327 \def\taburowcolors {\tabu@everyrow@bgroup \@testopt \tabu@rowcolors 1}
328 \def\tabu@rowcolors [#1]#2#{\tabu@rowc@lors{#1}{#2}}
329 \def\tabu@rowc@lors #1#2#3{%
330     \toks@{\@defaultunits \count@          =\number0#2\relax \@nnil
331         \@defaultunits \tabu@start    =\number0#1\relax \@nnil
332     \ifnum \count@<\tw@ \count@=\tw@ \fi
333     \advance\tabu@start \m@ne
334     \ifnum \tabu@start<\z@ \tabu@start \z@ \fi
335     \tabu@rowcolorseries #3\in@..\in@ \@nnil
336 }% \tabu@rowcolors
337 \def\tabu@rowcolorseries #1..#2\in@ #3\@nnil {%
338     \ifx \in@#1\relax
339         \iftabu@everyrow \toks@{\def\tabu@rc@{\let\tabu@rc@L \tabu@rc@}}%
340         \else \gdef\tabu@rc@{\global\let\tabu@rc@G \tabu@rc@}
341         \fi
342     \else
343         \ifx \\#2\\\tabu@rowcolorserieserror \fi
344         \tabu@sanitizearg{#1}\tabu@temp
345         \tabu@sanitizearg{#2}\@tempa
346         \advance\count@ \m@ne
347     \iftabu@everyrow
348         \def\tabu@rc@ ##1##2##3##4{\def\tabu@rc@{%
349             \ifnum ##2=\c@taburow
350                 \definecolorseries\tabu@rcseries@{\the\tabu@nested}{rgb}{last}{##3}{##4}\fi
351             \ifnum \c@taburow<##2 \else
352                 \ifnum \tabu@modulo {\c@taburow-##2}{##1+1}=\z@
353                     \resetcolorseries[##1]{\tabu@rcseries@{\the\tabu@nested}}\fi
354                 \xglobal\colorlet\tabu@rc@{\the\tabu@nested}{\tabu@rcseries@{\the\tabu@nested!!}}%
355                 \rowcolor\tabu@rc@{\the\tabu@nested}\fi}%
356             }\edef\x{\noexpand\tabu@rc@          {\the\count@}
357                 {\the\tabu@start}
358                 {\tabu@temp}
359                 {\@tempa}}%
360             }\x
361         \toks@\expandafter{\expandafter\def\expandafter\tabu@rc@\expandafter{\tabu@rc@}}%
362         \toks@\expandafter{\the\toks@ \let\tabu@rc@L \tabu@rc@ \ignorespaces}%
363     \else % inside \noalign
364         \definecolorseries\tabu@rcseries@{\the\tabu@nested}{rgb}{last}{\tabu@temp}{\@tempa}%
365         \expandafter\resetcolorseries\expandafter[\the\count@]{\tabu@rcseries@{\the\tabu@nested}}%
366         \xglobal\colorlet\tabu@rc@{\the\tabu@nested}{\tabu@rcseries@{\the\tabu@nested!!}}%
367         \let\noalign \relax \rowcolor\tabu@rc@{\the\tabu@nested}%
368     \def\tabu@rc@ ##1##2{\gdef\tabu@rc@{%
369         \ifnum \tabu@modulo {\c@taburow-##2}{##1+1}=\@ne
370             \resetcolorseries[##1]{\tabu@rcseries@{\the\tabu@nested}}\fi
371         \xglobal\colorlet\tabu@rc@{\the\tabu@nested}{\tabu@rcseries@{\the\tabu@nested!!}}%
372         \rowcolor\tabu@rc@{\the\tabu@nested}}%
373     }\edef\x{\noexpand\tabu@rc@{\the\count@}{\the\c@taburow}}\x

```

```

374     \global\let\tabu@rc@G \tabu@rc@
375     \fi
376     \fi
377     \tabu@everyrow@egroup
378 }% \tabu@rowcolorseries
379 \tabuDisableCommands {\let\tabu@rc@ \@empty }
380 \def\tabu@rowcolorserieserror {\PackageError{tabu}
381     {Invalid syntax for \string\taburowcolors
382     \MessageBreak Please look at the documentation!}\@ehd
383 }% \tabu@rowcolorserieserror

```

\tabureset Simply – and locally – reset the default values for `\tabulinesep` (0pt), `\extrarowsep` (0pt), `\extratabsurround` (0pt), `\tabulinestyle {}`, `\everyrow {}` and `\taburulecolor []{}.`

```

384 \newcommand*\tabureset {%
385     \tabulinesep=\z@ \extrarowsep=\z@ \extratabsurround=\z@
386     \tabulinestyle{}\everyrow{}\taburulecolor|{}\taburowcolors}%
387 }% \tabureset

```

Parsing line styles

\tabu@getline This macro parses a line specification argument of the form:

3pt BlunchedAlmond on 4pt Crimsom off 2pt ForestGreen

Note that Crimsom will overwrite BlunchedAlmond in this case: the color for the line dash may be specified after the line width or after the line dash length.

The process uses `\scantokens` on the argument given by the user, which is first expanded in a context where the `babel` switch `\if@save@actives` is set to `true`. Then `\scantokens` is used on the argument in a group where the letter “o” is active, and defined to be a macro which rewrites the line specification. Incidentally, the comma is active too, and expands to a space. This way the initial argument is “genetically modified”, so that it becomes very easy to assign dimensions (thickness, dash length and gap length) and colors separately.

For example: **3pt BlunchedAlmond on 4pt Crimsom** will be expanded in a context where “o” is active (and equal to `\tabu@oxiii`, the `xiii` suffix means “active” *ie.* `\catcode = 13`).

Then the “o” in **BlunchedAlmond** is rewritten as follow:

1. “o” sees “n” after itself, then it expands `\tabu@onxiii`.
2. `\tabu@onxiii` sees a character whose catcode is not other, then the rewriting process is aborted, and “**ond**” is rewritten as “**ond**” where the “o” is not active but the usual letter “o”.

The next “o” is rewritten as follow:

1. “o” sees “n” after itself, then it expands `\tabu@onxiii`.
2. `\tabu@onxiii` sees a space (which is active): it calls back itself again,
3. `\tabu@onxiii` sees a character whose catcode is other: then the sequence “**on_l3**” is rewritten as:
“`\tabu@ \tabu@on =4pt Crimsom`”

Finally the whole argument is rewritten as:

`\tabu@ \tabu@thick =3pt BlunchedAlmond \tabu@ \tabu@on =4pt Crimsom \tabu@ \tabu@`

Define `\tabu@` as an appropriate macro which uses `\afterassignment` to:

1. Assign the corresponding dimension (thickness, dash length or gap length).
2. Collect the rest until the next `\tabu@`, trim spaces and check if the color exists.

Limitation: A color name must not contain a sequence that matches on of the patterns:

`...on<a character of category 12>...` or `...off<a character of category 12>...`

But this “limitation” is not too heavy, I suppose...

The result is `\tabu@thestyle`: a `\tabu` line style to be used to rewrite a `|` column, for `\tabucline`.

We use locally the L^AT_EX defined dimen registers `\@tempdima`, `\@tempdimb` and `\@tempdimc`. For clarity, their names are `\tabu@thick`, `\tabu@on` and `\tabu@off` here...

```

388 \def\tabu@getline #1{\begingroup
389   \csname \ifcsname if@safe@actives\endcsname          % <babel>
390     @safe@activestruel\else
391       relax\fi          \endcsname
392   \edef\tabu@temp{#1}\tabu@sanitizearg{#1}\@tempa
393   \let\tabu@thestyle \relax
394   \ifcsname tabu@linestyle@\@tempa \endcsname
395     \edef\tabu@thestyle{\endgroup
396       \def\tabu@thestyle{\expandafter\noexpand
397         \csname tabu@linestyle@\@tempa\endcsname}%
398     }\tabu@thestyle
399   \else \expandafter\tabu@definestyle \tabu@temp \@nil
400   \fi
401 }% \tabu@getline

```

`\tabu@definestyle` Here is the `\scantokens` stuff.

```

402 \def\tabu@definestyle #1#2\@nil {\endlinechar \m@ne \makeatletter
403   \tabu@thick \maxdimen \tabu@on \maxdimen \tabu@off \maxdimen
404   \let\tabu@c@lon \@undefined \let\tabu@c@loff \@undefined
405   \ifodd 1\ifcat .#1\else\ifcat \relax #1\else 0\fi\fi % catcode 12 or non expandable cs
406   \def\tabu@temp{\tabu@getparam{thick}}%
407   \else \def\tabu@temp{\tabu@getparam{thick}\maxdimen}%
408   \fi
409   {%
410     \let\tabu@ \relax
411     \def\:{\obeyspaces \tabu@oXIII \tabu@commaXIII \edef\:% (space active \: happy ;-)}
412     \scantokens{\:\{\tabu@temp #1#2 \tabu@\tabu@}}%
413     \expandafter}\expandafter
414     \def\expandafter\:\expandafter{\:% line spec rewritten now ;-)}
415     \def\;\{\def\:%}
416     \scantokens\expandafter{\expandafter\;\expandafter{\:%} space is now inactive (catcode 10)
417     \let\tabu@ \tabu@getcolor \:% all arguments are ready now ;-)}
418     \ifdefined\tabu@c@lon \else \let\tabu@c@lon\@empty \fi
419     \ifx \tabu@c@lon\@empty \def\tabu@c@lon{\CT@arc@}\fi
420     \ifdefined\tabu@c@loff \else \let\tabu@c@loff \@empty \fi
421     \ifdim \tabu@on=\maxdimen \ifdim \tabu@off<\maxdimen
422       \tabu@on \tabulineon \fi\fi
423     \ifdim \tabu@off=\maxdimen \ifdim \tabu@on<\maxdimen
424       \tabu@off \tabulineoff \fi\fi
425     \ifodd 1\ifdim \tabu@off=\maxdimen \ifdim \tabu@on=\maxdimen 0 \fi\fi
426     \in@true % <leaders>
427     \else \in@false % <rule>
428     \fi
429     \ifdim\tabu@thick=\maxdimen \def\tabu@thick{\arrayrulewidth}%
430     \else \edef\tabu@thick{\the\tabu@thick}%
431     \fi
432     \edef \tabu@thestyle ##1##2{\endgroup
433       \def\tabu@thestyle{%
434         \ifin@ \noexpand\tabu@leadersstyle {\tabu@thick}
435           {\the\tabu@on}{##1}
436           {\the\tabu@off}{##2}%
437         \else \noexpand\tabu@rulesstyle
438           {##1\vrule width \tabu@thick}%
439           {##1\leaders \hrule height \tabu@thick \hfil}%

```

```

440         \fi}%
441     }\expandafter \expandafter
442         \expandafter \tabu@thestyle \expandafter
443         \expandafter \expandafter
444         {\expandafter\tabu@c@lon\expandafter}\expandafter{\tabu@c@loff}%
445 }% \tabu@definestyle

```

\tabu@onxiii We have to define the active “o” character, which looks for the next tokens, trying to find a pattern like **on**(*category 12*) or **off**(*category 12*) (possibly with – active – spaces between **on** or **off** and the next character of catcode 12).

\tabu@ofxiii
\tabu@offiii

```

446 {\catcode`\o=\active \lccode`\O=`\o \catcode`\,=\active
447     \lowercase{\gdef\tabu@oXIII {\catcode`\o=\active \let O=\tabu@oxiii}}
448     \gdef\tabu@commaXIII {\catcode`\,=\active \let ,=\space}
449 }% \catcode
450 \def\tabu@oxiii #1{%
451     \ifcase \ifx n#1\z@ \else
452         \ifx f#1\@ne\else
453             \tw@ \fi\fi
454             \expandafter\tabu@onxiii
455     \or \expandafter\tabu@ofxiii
456     \else o%
457     \fi#1}%
458 \def\tabu@onxiii #1#2{%
459     \ifcase \ifx !#2\tw@ \else
460         \ifcat.\noexpand#2\z@ \else
461         \ifx \tabu@spxiii#2\@ne\else
462         \tw@ \fi\fi\fi
463         \tabu@getparam{on}#2\expandafter\@gobble
464     \or \expandafter\tabu@onxiii % (space is active)
465     \else o\expandafter\@firstofone
466     \fi{#1#2}}%
467 \def\tabu@ofxiii #1#2{%
468     \ifx #2f\expandafter\tabu@offxiii
469     \else o\expandafter\@firstofone
470     \fi{#1#2}}
471 \def\tabu@offxiii #1#2{%
472     \ifcase \ifx !#2\tw@ \else
473         \ifcat.\noexpand#2\z@ \else
474         \ifx \tabu@spxiii#2\@ne \else
475         \tw@ \fi\fi\fi
476         \tabu@getparam{off}#2\expandafter\@gobble
477     \or \expandafter\tabu@offxiii % (space is active)
478     \else o\expandafter\@firstofone
479     \fi{#1#2}}

```

\tabu@getparam The rewritten stuff.

```

480 \def\tabu@getparam #1{\tabu@ \csname tabu@#1\endcsname=}

```

\tabu@getcolor **\tabu@ \tabu@on** = *<3pt>* **Crimson****\tabu@**

\tabu@getcolor first makes the assignment to **\tabu@on** and then looks for the color name which might have been placed before the next **\tabu@**.

```

481 \def\tabu@getcolor #1{% \tabu@ <- \tabu@getcolor after \edef
482     \ifx \tabu@#1\else % no more spec
483         \let\tabu@theparam=#1\afterassignment \tabu@getc@l@r #1\fi
484 }% \tabu@getcolor
485 \def\tabu@getc@l@r #1\tabu@ {%
486     \def\tabu@temp{#1}\tabu@strtrim \tabu@temp

```

```

487 \ifx \tabu@temp\@empty
488 \else%\ifcsname \string\color@\tabu@temp \endcsname % if the color exists
489 \ifx \tabu@theparam \tabu@off \let\tabu@c@loff \tabu@c@l@r
490 \else \let\tabu@c@lon \tabu@c@l@r
491 \fi
492 %\else \tabu@warncolour{\tabu@temp}%
493 \fi%\fi
494 \tabu@ % next spec
495 }% \tabu@getc@l@r
496 \def\tabu@warncolour #1{\PackageWarning\tabu
497 {Color #1 is not defined. Default color used}%
498 }% \tabu@warncolour

```

\tabu@leadersstyle When a style is executed, it expands either `\tabu@leadersstyle` or `\tabu@rulesstyle` depending on whether `\tabu@rulesstyle` or not it contains leaders (dashed lines) or simple rules (solid lines): T_EX internals allow to insert solid lines easily inside a tabular, while inserting leaders is more complex.

`\tabu@leadersstyle` eventually rebuilds the (horizontal and vertical) leaders boxes, and then define two macros: `\tabu@thevleaders` and `\tabu@thehleads`, suitable to draw vertical and horizontal lines respectively. Incidentally, `\tabu@leaders` is defined to be the parameters for the leaders.

`\tabu@rulesstyle` only defines the two macros `\tabu@therule` and `\tabu@thehrule`. The control sequence `\tabu@leaders` is undefined so that we know if the style contains a leader or a rule.

```

499 \def\tabu@leadersstyle #1#2#3#4#5{\def\tabu@leaders{#1}{#2}{#3}{#4}{#5}}%
500 \ifx \tabu@leaders\tabu@leaders@G \else
501 \tabu@LEADERS{#1}{#2}{#3}{#4}{#5}\fi
502 }% \tabu@leadersstyle
503 \def\tabu@rulesstyle #1#2{\let\tabu@leaders \@undefined
504 \gdef\tabu@therule{#1}\gdef\tabu@thehrule{#2}%
505 }% \tabu@rulesstyle

```

\tabu@LEADERS Here the two leaders boxes `\tabu@hleads` and `\tabu@vleads` are built, as well as the leaders macros `\tabu@thehleaders` and `\tabu@thevleaders`.

```

506 \def\tabu@LEADERS #1#2#3#4#5{%% width, dash, dash color, gap, gap color
507 {\let\color \tabu@color % => during trials -> \color = \tabu@nocolor
508 {% % but the leaders boxes should have colors !
509 \def\@therule{\vrule}\def\@thick{height}\def\@length{width}%
510 \def\@box{\hbox}\def\@unbox{\unhbox}\def\@elt{\wd}%
511 \def\@skip{\hskip}\def\@ss{\hss}\def\tabu@leads{\tabu@hleads}%
512 \tabu@l@@d@rs {#1}{#2}{#3}{#4}{#5}%
513 \global\let\tabu@thehleaders \tabu@theleaders
514 }%
515 {%
516 \def\@therule{\hrule}\def\@thick{width}\def\@length{height}%
517 \def\@box{\vbox}\def\@unbox{\unvbox}\def\@elt{\ht}%
518 \def\@skip{\vskip}\def\@ss{\vss}\def\tabu@leads{\tabu@vleads}%
519 \tabu@l@@d@rs {#1}{#2}{#3}{#4}{#5}%
520 \global\let\tabu@thevleaders \tabu@theleaders
521 }%
522 \gdef\tabu@leaders@G{#1}{#2}{#3}{#4}{#5}}%
523 }%
524 }% \tabu@LEADERS
525 \def\tabu@therule #1#2{\@therule \@thick#1\@length\dimexpr#2/2 \@depth\z@}
526 \def\tabu@l@@d@rs #1#2#3#4#5{%% width, dash, dash color, gap, gap color
527 \global\setbox \tabu@leads=\@box{%
528 {#3\tabu@therule{#1}{#2}}%
529 \ifx\@#5\@skip#4\else{#5\tabu@therule{#1}{#4*2}}\fi
530 {#3\tabu@therule{#1}{#2}}}%
531 \global\setbox\tabu@leads=\@box to\@elt\tabu@leads{\@ss

```

```

532     {#3\tabu@therule{#1}{#2}}\@unbox\tabu@leads}%
533 \edef\tabu@theleaders ##1{\def\noexpand\tabu@theleaders {%
534     {##1\tabu@therule{#1}{#2}}%
535     \xleaders \copy\tabu@leads \@ss
536     \tabu@therule{0pt}{-#2}{##1\tabu@therule{#1}{#2}}}%
537 } \tabu@theleaders{#3}%
538 }% \tabu@l@@@ders

```

11.8 The entry inside tabu

`\tabu`, `\endtabu`, `\longtabu` and `\endlontabu`

`\tabu` `\tabu` and `\longtabu` are the commands of the environments.

`\endtabu` `\endtabu` is `\endtabular` or `\endarray` in math mode.

```

539 \newcommand*\tabu {\tabu@longfalse
540     \ifmmode \def\tabu@ {\array}\def\endtabu {\endarray}%
541     \else \def\tabu@ {\tabu@tabular}\def\endtabu {\endtabular}\fi
542     \expandafter\let\csname tabu*\endcsname \tabu
543     \expandafter\def\csname endtabu*\endcsname{\endtabu}%
544     \tabu@spreadfalse \tabu@negcoeffalse \tabu@settarget
545 }% {tabu}
546 \let\tabu@tabular \tabular % <For LyX: some users redefine \tabular...>
547 \expandafter\def\csname tabu*\endcsname{\tabuscantokenstrue \tabu}
548 \newcommand*\longtabu {\tabu@longtrue
549     \ifmmode\PackageError{tabu}{longtabu not allowed in math mode}\fi
550     \def\tabu@{\longtable}\def\endlongtabu{\endlongtable}%
551     \LTchunksize=\@M
552     \expandafter\let\csname tabu*\endcsname \tabu
553     \expandafter\def\csname endlongtabu*\endcsname{\endlongtabu}%
554     \let\LT@startpbox \tabu@LT@startpbox % \everypar{ array struts }
555     \tabu@spreadfalse \tabu@negcoeffalse \tabu@settarget
556 }% {longtabu}
557 \expandafter\def\csname longtabu*\endcsname{\tabuscantokenstrue \longtabu}
558 \def\tabu@nolongtabu{\PackageError{tabu}
559     {longtabu requires the longtable package}\@ehd}

```

Setting the tabu target

`\tabu@settarget` The macro sets `\tabu@target` (a dimen) to the value specified for “tabu to” or “tabu spread”.

```

\begin{tabu}
\begin{tabu}
560 \def\tabu@settarget {\futurelet\@let@token \tabu@settarget }
561 \def\tabu@settarget {\tabu@target \z@
562     \ifcase \ifx \bgroup\@let@token \z@ \else
563         \ifx \@sptoken\@let@token \@ne \else
564         \if t\@let@token \tw@ \else
565         \if s\@let@token \thr@@\else
566         \z@\fi\fi\fi\fi
567     \expandafter\begin{tabu}
568     \or \expandafter\begin{tabu}gobblespace\expandafter\begin{tabu}settarget
569     \or \expandafter\begin{tabu}to
570     \or \expandafter\begin{tabu}spread
571     \fi
572 }% \begin{tabu}settarget
573 \def\begin{tabu}to to{\def\begin{tabu}halignto{to}\begin{tabu}gettarget}
574 \def\begin{tabu}spread spread{\begin{tabu}spreadtrue\def\begin{tabu}halignto{spread}\begin{tabu}gettarget}
575 \def\begin{tabu}gettarget {\afterassignment\begin{tabu}linegoaltarget \begin{tabu}target }
576 \def\begin{tabu}linegoaltarget {\futurelet\begin{tabu}temp \begin{tabu}linegoaltarget }
577 \def\begin{tabu}linegoaltarget {%

```



```

578 \ifx \tabu@temp\LNGL@setlinegoal
579 \LNGL@setlinegoal \expandafter \@firstoftwo \fi % @gobbles \LNGL@setlinegoal
580 \tabu@begin
581 }% \tabu@linegoal\trget
582 \def\tabu@begin #1#{%
583 \iftabu@measuring \expandafter\tabu@nestedmeasure \fi
584 \ifdim \tabu@target=\z@ \let\tabu@halignto \@empty
585 \else \edef\tabu@halignto{\tabu@halignto\the\tabu@target}%
586 \fi
587 \@testopt \tabu@tabu@ \tabu@aligndefault #1\@nil
588 }% \tabu@begin
589 \long\def\tabu@tabu@ [#1]#2\@nil #3{\tabu@setup
590 \def\tabu@align {#1}\def\tabu@savdpream{\NC@find #3}%
591 \tabu@ [\tabu@align ]#2{#3\tabu@rewritefirst }%
592 }% \tabu@tabu@
593 \def\tabu@nestedmeasure {%
594 \ifodd 1\iftabu@spread \else \ifdim\tabu@target=\z@ \else 0 \fi\fi\relax
595 \tabu@spreadtrue
596 \else \beginingroup \iffalse{\fi \ifnum0='}\fi
597 \toks@{}\def\tabu@stack{b}%
598 \expandafter\tabu@collectbody\expandafter\tabu@quickrule
599 \expandafter\endgroup
600 \fi
601 }% \tabu@nestedmeasure
602 \def\tabu@quickrule {\indent\vrule height\z@ depth\z@ width\tabu@target}

```

\tabu@setup \tabu@init is expanded only when tabu is not nested. In this case, and if \parindent > 0, and if
\tabu@init \tabu@defaulttarget =\linewidth, the correction of the default target for paragraph indentation is executed
\tabu@indent (see [paragraph indentation](#)).

```

603 \def\tabu@setup{\tabu@alloc@
604 \ifcase \tabu@nested
605 \ifmmode \else \iftabu@spread\else \ifdim\tabu@target=\z@
606 \let\tabu@afterendpar \par
607 \fi\fi\fi
608 \def\tabu@aligndefault{c}\tabu@init \tabu@indent
609 \else % <nested tabu>
610 \def\tabu@aligndefault{t}\let\tabu@defaulttarget \linewidth
611 \fi
612 \let\tabu@thetarget \tabu@defaulttarget \let\tabu@restored \@undefined
613 \edef\tabu@NC@list{\the\NC@list}\NC@list{\NC@do \tabu@rewritefirst}%
614 \everycr{}\let\@startpbox \tabu@startpbox % for nested tabu inside longtabu...
615 \let\@endpbox \tabu@endpbox % idem " " " " " "
616 \let\@tabarray \tabu@tabarray % idem " " " " " "
617 \tabu@setcleanup \tabu@setreset
618 }% \tabu@setup
619 \def\tabu@init{\tabu@starttimer \tabu@measuringfalse
620 \edef\tabu@hfuzz {\the\dimexpr\hfuzz+1sp}\global\tabu@footnotes{}%
621 \let\firsthline \tabu@firsthline \let\lasthline \tabu@lasthline
622 \let\firstline \tabu@firstline \let\lastline \tabu@lastline
623 \let\hline \tabu@hline \let\@xhline \tabu@xhline
624 \let\color \tabu@color \let\@arstrutbox \tabu@arstrutbox
625 \iftabu@colortbl\else\let\LT@@hline \tabu@LT@@hline \fi
626 \tabu@trivlist %<restore \\=\@normalcr inside lists>
627 \let\@footnotetext \tabu@footnotetext \let\@xfootnotetext \tabu@xfootnotetext
628 \let\@xfootnote \tabu@xfootnote \let\centering \tabu@centering
629 \let\raggedright \tabu@raggedright \let\raggedleft \tabu@raggedleft
630 \let\tabu@decimal \tabu@tabu@decimal \let\Centering \tabu@Centering
631 \let\RaggedRight \tabu@RaggedRight \let\RaggedLeft \tabu@RaggedLeft

```

```

632 \let\justifying \tabu@justifying \let\rowfont \tabu@rowfont
633 \let\fbbox \tabu@fbbox \let\color@b@x \tabu@color@b@x
634 \let\tabu@@everycr \everycr \let\tabu@@everypar \everypar
635 \let\tabu@prepnext@tokORI \prepnext@tok\let\prepnext@tok \tabu@prepnext@tok
636 \let\tabu@multicolumnORI\multicolumn \let\multicolumn \tabu@multicolumn
637 \let\tabu@startpbox \@startpbox % for nested tabu inside longtabu pfff !!!
638 \let\tabu@endpbox \@endpbox % idem " " " " " " " "
639 \let\tabu@tabarray \@tabarray % idem " " " " " " " "
640 \tabu@adl@fix \let\endarray \tabu@endarray % <fix> colortbl & arydshln (delarray)
641 \iftabu@colortbl\CT@everycr\expandafter\expandafter\iftabu@everyrow \the\CT@everycr \fi\fi
642 }% \tabu@init
643 \def\tabu@indent{% correction for indentation
644 \ifdim \parindent>\z@\ifx \linewidth\tabu@defaulttarget
645 %
646 \everypar\expandafter{%
647 \the\everypar\everypar\expandafter{\the\everypar}%
648 \setbox\z@=\lastbox
649 \ifdim\wd\z@>\z@ \edef\tabu@thetarget
650 {\the\dimexpr -\wd\z@+\tabu@defaulttarget}\fi
651 \box\z@}%
652 \fi\fi
653 }% \tabu@indent

```

\tabu@setcleanup We have to save locally (in the group of the environment) the current value of the last global assignments to `\CT@arc@`, `\CT@drsc@`, `\tabu@ls@` etc.

\tabu@cleanup Restoration will be done globally after the box that contains the tabular by `\tabu@cleanup`.

```

654 \def\tabu@setcleanup {% saves last global assignments
655 \ifodd 1\ifmmode \else \iftabu@long \else 0\fi\fi\relax
656 \def\tabu@aftergroupcleanup{%
657 \def\tabu@aftergroupcleanup{\aftergroup\tabu@cleanup}}%
658 \else
659 \def\tabu@aftergroupcleanup{%
660 \aftergroup\aftergroup\aftergroup\tabu@cleanup
661 \let\tabu@aftergroupcleanup \relax}%
662 \fi
663 \let\tabu@arc@Gsave \tabu@arc@G
664 \let\tabu@arc@G \tabu@arc@L % <init>
665 \let\tabu@drsc@Gsave \tabu@drsc@G
666 \let\tabu@drsc@G \tabu@drsc@L % <init>
667 \let\tabu@ls@Gsave \tabu@ls@G
668 \let\tabu@ls@G \tabu@ls@L % <init>
669 \let\tabu@rc@Gsave \tabu@rc@G
670 \let\tabu@rc@G \tabu@rc@L % <init>
671 \let\tabu@evr@Gsave \tabu@evr@G
672 \let\tabu@evr@G \tabu@evr@L % <init>
673 \let\tabu@celllalign@save \tabu@celllalign
674 \let\tabu@cellralign@save \tabu@cellralign
675 \let\tabu@cellleft@save \tabu@cellleft
676 \let\tabu@cellright@save \tabu@cellright
677 \let\tabu@@celllalign@save \tabu@@celllalign
678 \let\tabu@@cellralign@save \tabu@@cellralign
679 \let\tabu@@cellleft@save \tabu@@cellleft
680 \let\tabu@@cellright@save \tabu@@cellright
681 \let\tabu@rowfontreset@save \tabu@rowfontreset
682 \let\tabu@@rowfontreset@save\tabu@@rowfontreset
683 \let\tabu@rowfontreset \@empty
684 \edef\tabu@alloc@save {\the\tabu@alloc}% restore at \tabu@reset

```

```

685 \edef\c@taburow@save      {\the\c@taburow}%
686 \edef\tabu@naturalX@save  {\the\tabu@naturalX}%
687 \let\tabu@naturalXmin@save \tabu@naturalXmin
688 \let\tabu@naturalXmax@save \tabu@naturalXmax
689 \let\tabu@mkarstrut@save  \tabu@mkarstrut
690 \edef\tabu@clarstrut{%
691     \extrarowheight \the\dimexpr \ht\@arstrutbox-\ht\strutbox \relax
692     \extrarowdepth \the\dimexpr \dp\@arstrutbox-\dp\strutbox \relax
693     \let\noexpand\@arraystretch \@ne \noexpand\tabu@rearstrut}%
694 }% \tabu@setcleanup
695 \def\tabu@cleanup {\begingroup
696     \globaldefs\@ne      \tabu@everyrowtrue
697     \let\tabu@arc@G      \tabu@arc@Gsave
698     \let\CT@arc@         \tabu@arc@G
699     \let\tabu@drsc@G     \tabu@drsc@Gsave
700     \let\CT@drsc@        \tabu@drsc@G
701     \let\tabu@ls@G       \tabu@ls@Gsave
702     \let\tabu@ls@        \tabu@ls@G
703     \let\tabu@rc@G       \tabu@rc@Gsave
704     \let\tabu@rc@        \tabu@rc@G
705     \let\CT@do@color     \relax
706     \let\tabu@evr@G      \tabu@evr@Gsave
707     \let\tabu@celllalign \tabu@celllalign@save
708     \let\tabu@cellralign \tabu@cellralign@save
709     \let\tabu@cellleft   \tabu@cellleft@save
710     \let\tabu@cellright  \tabu@cellright@save
711     \let\tabu@@celllalign \tabu@@celllalign@save
712     \let\tabu@@cellralign \tabu@@cellralign@save
713     \let\tabu@@cellleft  \tabu@@cellleft@save
714     \let\tabu@@cellright \tabu@@cellright@save
715     \let\tabu@rowfontreset \tabu@rowfontreset@save
716     \let\tabu@@rowfontreset \tabu@@rowfontreset@save
717     \tabu@naturalX       =\tabu@naturalX@save
718     \let\tabu@naturalXmax \tabu@naturalXmax@save
719     \let\tabu@naturalXmin \tabu@naturalXmin@save
720     \let\tabu@mkarstrut   \tabu@mkarstrut@save
721     \c@taburow            =\c@taburow@save
722     \ifcase \tabu@nested  \tabu@alloc \m@ne\fi
723     \endgroup            % <end of \globaldefs>
724     \ifcase \tabu@nested
725         \the\tabu@footnotes \global\tabu@footnotes{}%
726         \tabu@afterendpar  \tabu@elapsedtime
727     \fi
728     \tabu@clarstrut
729     \everyrow\expandafter {\tabu@evr@G}%
730 }% \tabu@cleanup
731 \let\tabu@afterendpar \relax

```

\tabu@setreset At the beginning of each trial, we have to restore the current value that were active at the entry in the tabu environment (for they could have been globally overwritten inside the tabular).

The same must occur when using `\usetabu` as a preamble. Values are restored locally inside the `tabu` box.

\tabu@reset `\tabu@setreset` defines `\tabu@reset` to be expanded at the beginning of each trial and when `\usetabu` is used.

```

732 \def\tabu@setreset {%
733     \edef\tabu@savparams {%          \relax for \tabu@message@save
734         \ifmmode \col@sep \the\arraycolsep
735         \else     \col@sep \the\tabc@colsep \fi \relax

```

```

736 \arrayrulewidth \the\arrayrulewidth \relax
737 \doublerulesep \the\doublerulesep \relax
738 \extratabsurround \the\extratabsurround \relax
739 \extrarowheight \the\extrarowheight \relax
740 \extrarowdepth \the\extrarowdepth \relax
741 \abovetabulinesep \the\abovetabulinesep \relax
742 \belowtabulinesep \the\belowtabulinesep \relax
743 \def\noexpand\arraystretch{\arraystretch}%
744 \ifdefined\minrowclearance \minrowclearance\the\minrowclearance\relax\fi}%
745 \begingroup
746 \@temptokena\expandafter{\tabu@savparams}% => only for \savetabu / \usetabu
747 \ifx \tabu@arc@L\relax \else \tabu@savesave \tabu@arc@L \fi
748 \ifx \tabu@drsc@L\relax \else \tabu@savesave \tabu@drsc@L \fi
749 \tabu@savesave \tabu@ls@L \tabu@savesave \tabu@evr@L
750 \expandafter \endgroup \expandafter
751 \def\expandafter\tabu@sav@ \expandafter{\the\@temptokena
752 \let\tabu@arc@G \tabu@arc@L
753 \let\tabu@drsc@G \tabu@drsc@L
754 \let\tabu@ls@G \tabu@ls@L
755 \let\tabu@rc@G \tabu@rc@L
756 \let\tabu@evr@G \tabu@evr@L}%
757 \def\tabu@reset{\tabu@savparams
758 \tabu@everyrowtrue \c@taburow \z@
759 \let\CT@arc@ \tabu@arc@L
760 \let\CT@drsc@ \tabu@drsc@L
761 \let\tabu@ls@ \tabu@ls@L
762 \let\tabu@rc@ \tabu@rc@L
763 \global\tabu@alloc \tabu@alloc@save
764 \everyrow\expandafter{\tabu@evr@L}}%
765 }% \tabu@reset
766 \def\tabu@savesave #1{\expandafter\tabu@savesave #1\@nil{#1}}
767 \long\def\tabu@savesave #1\@nil #2{\@temptokena\expandafter{\the\@temptokena \def#2{#1}}}
```

11.9 The rewriting process: inside the “\@mkpream group”

New column types and private (new) column types

\tabu@newcolumntype A helper macro to create new column types for tabu.

The column types **are not appended** to \NC@list in order to keep them local to tabu.

```

768 \def\tabu@newcolumntype #1{%
769 \expandafter\tabu@new@columntype
770 \csname NC@find@\string#1\expandafter\endcsname
771 \csname NC@rewrite@\string#1\endcsname
772 {#1}%
773 }% \tabu@newcolumntype
774 \def\tabu@new@columntype #1#2#3{%
775 \def#1##1#3{\NC@{##1}}%
776 \let#2\relax \newcommand*#2%
777 }% \tabu@new@columntype
```

\tabu@privatecolumntype Columns types defined with \tabu@privatecolumntype are "mounted" only inside the \@mkpream group of tabu.

```

778 \def\tabu@privatecolumntype #1{%
779 \expandafter\tabu@private@columntype
780 \csname NC@find@\string#1\expandafter\endcsname
781 \csname NC@rewrite@\string#1\expandafter\endcsname
782 \csname tabu@NC@find@\string#1\expandafter\endcsname
783 \csname tabu@NC@rewrite@\string#1\endcsname
```

```

784      {#1}%
785 }% \tabu@privatecolumntype
786 \def\tabu@private@columntype#1#2#3#4{%
787     \g@addto@macro\tabu@private@columns{\let#1#3\let#2#4}%
788     \tabu@new@columntype#3#4%
789 }% \tabu@private@columntype
790 \let\tabu@private@columns \@empty

```

High priority columns

\tabucolumn \tabucolumn puts a user-defined column in high priority in the tabu rewriting process.

```

791 \newcommand*\tabucolumn [1]{\expandafter \def \expandafter
792     \tabu@highprioritycolumns\expandafter{\tabu@highprioritycolumns
793                                     \NC@do #1}}%
794 \let\tabu@highprioritycolumns \@empty

```

Rewriting vertical lines and leaders

| (private column type) This is the rewrite macro for the | column type inside tabu and longtabu.

Vertical lines are *simply rewritten* as special ! columns.

```

795 \tabu@privatecolumntype |{\tabu@rewritevline}
796 \newcommand*\tabu@rewritevline[1][\tabu@vlinearg{#1}%
797     \expandafter \NC@find \tabu@rewritten}

```

\tabu@lines The | token for vertical lines may have a special catcode. **array.sty** makes the test with \if and therefore, it is catcode insensitiv. Here, we use \scantokens and check if | is not an *other* character.

```

798 \def\tabu@lines #1{%
799     \ifx|#1\else \tabu@privatecolumntype #1{\tabu@rewritevline}\fi
800     \NC@list\expandafter{\the\NC@list \NC@do #1}%
801 }% \tabu@lines@

```

\tabu@vlinearg The macro that parses the optional argument of | vertical lines...

```

802 \def\tabu@vlinearg #1{%
803     \ifx\#1\def\tabu@thestyle {\tabu@ls@}%
804     \else\tabu@getline {#1}%
805     \fi
806     \def\tabu@rewritten ##1{\def\tabu@rewritten{!##1\tabu@thevline}}%
807     }\expandafter\tabu@rewritten\expandafter{\tabu@thestyle}%
808     \expandafter \tabu@keepls \tabu@thestyle \@nil
809 }% \tabu@vlinearg
810 \def\tabu@keepls #1\@nil{%
811     \ifcat $\@cdr #1\@nil $\%
812     \ifx \relax#1\else
813     \ifx \tabu@ls@#1\else
814         \let#1\relax
815         \xdef\tabu@mkpreamblebuffer{\tabu@mkpreamblebuffer
816             \tabu@savels\noexpand#1}\fi\fi\fi
817 }% \tabu@keepls
818 \def\tabu@thevline {\begingroup
819     \ifdefined\tabu@leaders
820         \setbox\@tempboxa=\vtop to\dimexpr
821             \ht\@arstrutbox+\dp\@arstrutbox{\tabu@thevleaders}}%
822         \ht\@tempboxa=\ht\@arstrutbox \dp\@tempboxa=\dp\@arstrutbox
823         \box\@tempboxa
824     \else
825         \tabu@thevrule
826     \fi
827 }% \tabu@thevline

```

```

828 \def\tabu@savels #1{%
829     \expandafter\let\csname\string#1\endcsname #1%
830     \expandafter\def\expandafter\tabu@reset\expandafter{\tabu@reset
831                                     \tabu@resetls#1}}%
832 \def\tabu@resetls #1{\expandafter\let\expandafter#1\csname\string#1\endcsname}%

```

Vertical lines and leaders in the `\multicolumn` preamble

`\tabu@rewritemulticolumn` A special rewrite to allow `[...]` in `\multicolumn` preamble inside `tabu` environment.

As long as `\multicolumn` begins with `\omit` (via `\multispan`) special care has to be taken: everything shall be purely expandable until `\omit`.

`\multicolumn` is not an environment: no group is opened apart the `\@mkpream` group. We open a semi simple group for `\multicolumn` when inside `tabu`, in order for the setup to be local (in case a user would try to embed a `tabular` inside the argument of `\multicolumn...`)

```

833 \tabu@newcolumnntype \tabu@rewritemulticolumn{%
834     \aftergroup \tabu@endrewritemulticolumn % after \@mkpream group
835     \NC@list{\NC@do *}\tabu@textbar \tabu@lines
836     \tabu@savdecl
837     \tabu@privatecolumns
838     \NC@list\expandafter{\the\expandafter\NC@list \tabu\NC@list}%
839     \let\tabu@savels \relax
840     \NC@find
841 }% \tabu@rewritemulticolumn
842 \def\tabu@endrewritemulticolumn{\gdef\tabu@mkpreambuffer{\endgroup}
843 \def\tabu@multicolumn{\tabu@ifenvir \tabu@multic@lumn \tabu@multicolumnORI}
844 \long\def\tabu@multic@lumn #1#2#3{\multispan{#1}\begingroup
845     \tabu@everyrowtrue
846     \NC@list{\NC@do \tabu@rewritemulticolumn}%
847     \expandafter\@gobbletwo % gobbles \multispan{#1}
848         \tabu@multicolumnORI{#1}{\tabu@rewritemulticolumn #2}%
849         {\iftabuscantokens \tabu@rescan \else \expandafter\@firstofone \fi
850          {#3}}%
851 }% \tabu@multic@lumn

```

Rewriting `tabu X` columns

X (private column type) This is the rewrite macro for `tabu X` columns. Such a column has an optional argument: the width coefficient for the `tabu X` column whose default value is 1, and may be some alignments parameters. The coefficient is used in the expression: `p{\dimexpr <coef>\tabucolX }`

```

852 \tabu@privatecolumnntype X[1][]{\begingroup \tabu@siunitx{\endgroup \tabu@rewriteX {#1}}}
853 \def\tabu@nosiunitx #1{#1}{\expandafter \NC@find \tabu@rewritten }
854 \def\tabu@siunitx #1{\@ifnextchar \bgroup
855     {\tabu@rewriteX@Ss{#1}}
856     {\tabu@nosiunitx{#1}}}
857 \def\tabu@rewriteX@Ss #1#2{\@temptokena{}}%
858     \@defaultunits \let\tabu@temp =#2\relax\@nnil
859     \ifodd 1\ifx S\tabu@temp \else \ifx s\tabu@temp \else 0 \fi\fi
860     \def\NC@find{\def\NC@find >####1###2<####3\relax{#1 {####1}{####3}}%
861         }\expandafter\NC@find \the\@temptokena \relax
862         }\expandafter\NC@rewrite@S \@gobble #2\relax
863     \else \tabu@siunitxerror
864     \fi
865     \expandafter \NC@find \tabu@rewritten
866 }% \tabu@rewriteX@Ss
867 \def\tabu@siunitxerror {\PackageError{tabu}{Not a S nor s column !
868     \MessageBreak X column can only embed siunitx S or s columns}\@ehd
869 }% \tabu@siunitxerror

```

\tabu@rewriteX This macro is expanded by during the rewriting process in case a X column is found.

\tabu@Xsum (a dimen) stores the sum of the (absolute) width coefficients.

For the first X column found in the preamble, a special setup occurs:

- if the default target is used (no target specified or **\tabu spread** with X columns), the target: **\tabu@target** is set to the default, with a message in the .log file.
- **\@halignto** is **\let** to **\relax** to avoid its expansion in **\xdef \@preamble** just after **\@mkpream**. Indeed as long as we have to measure the natural width of the tabular, **\@halign** must be empty for trial steps.
- The rest of the setup is made **\aftergroup** (*ie.* after **\xdef \@preamble** which occurs inside a group) by **\tabu@prep@TRIAL**.

```
870 \def\tabu@rewriteX #1#2#3{\tabu@Xarg {#1}{#2}{#3}%
871   \iftabu@measuring
872   \else \tabu@measuringtrue % first X column found in the preamble
873     \let\@halignto \relax   \let\tabu@halignto \relax
874     \iftabu@spread \tabu@spreadtarget \tabu@target \tabu@target \z@
875     \else          \tabu@spreadtarget \z@ \fi
876     \ifdim \tabu@target=\z@
877       \setlength\tabu@target \tabu@thetarget
878       \tabu@message{\tabu@message@defaulttarget}%
879     \else \tabu@message{\tabu@message@target}\fi
880   \fi
881 }% \tabu@rewriteX
```

\tabu@rewriteXrestore This macro replaces **\tabu@rewriteX** in the case of **\usetabu**.

```
882 \def\tabu@rewriteXrestore #1#2#3{\let\@halignto \relax
883   \def\tabu@rewritten{1}}
```

\tabu@Xarg A tedious (and fastidious) macro to parse the optional argument of X columns. The aim is to build **\tabu@Xparse** **\tabu@rewritten** which expands to the column specification:

>{alignment} p or m or b {dimexpr coef \tabucolX \relax }

After that **array.sty** make it easy: **\expandafter \NC@find \tabu@rewritten**

```
884 \def\tabu@Xarg #1#2#3{%
885   \advance\tabu@Xcol \@ne   \let\tabu@Xlcr \@empty
886   \let\tabu@Xdisp \@empty  \let\tabu@Xmath \@empty
887   \ifx\#1\% <shortcut when no option>
888     \def\tabu@rewritten{p}\tabucolX \p@           % <default coef = 1>
889   \else
890     \let\tabu@rewritten \@empty \let\tabu@temp \@empty \tabucolX \z@
891     \tabu@Xparse {}#1\relax
892   \fi
893   \tabu@Xrewritten{#2}{#3}%
894 }% \tabu@Xarg
895 \def\tabu@Xparse #1{\futurelet\@let@token \tabu@Xtest}
896 \expandafter\def\expandafter\tabu@Xparsespace\space{\tabu@Xparse{}}
897 \def\tabu@Xtest{%
898   \ifcase \ifx \relax\@let@token \z@ \else
899     \if ,\@let@token \m@ne\else
900     \if p\@let@token 1\else
901     \if m\@let@token 2\else
902     \if b\@let@token 3\else
903     \if l\@let@token 4\else
904     \if c\@let@token 5\else
905     \if r\@let@token 6\else
906     \if j\@let@token 7\else
907     \if L\@let@token 8\else
```

```

908         \if C{@let@token 9}else
909         \if R{@let@token 10}else
910         \if J{@let@token 11}else
911         \ifx \@sptoken\@let@token 12}else
912         \if .{@let@token 13}else
913         \if -{@let@token 13}else
914         \ifcat ${@let@token 14}else
915         15\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\relax
916 \or \tabu@Xtype {p}%
917 \or \tabu@Xtype {m}%
918 \or \tabu@Xtype {b}%
919 \or \tabu@Xalign \raggedright\relax
920 \or \tabu@Xalign \centering\relax
921 \or \tabu@Xalign \raggedleft\relax
922 \or \tabu@Xalign \tabu@justify\relax
923 \or \tabu@Xalign \RaggedRight\raggedright
924 \or \tabu@Xalign \Centering\centering
925 \or \tabu@Xalign \RaggedLeft\raggedleft
926 \or \tabu@Xalign \justifying\tabu@justify
927 \or \expandafter \tabu@Xparespace
928 \or \expandafter \tabu@Xcoef
929 \or \expandafter \tabu@Xm@th
930 \or \tabu@Xcoef{}%
931 \else\expandafter \tabu@Xparse
932 \fi
933 }% \tabu@Xtest
934 \def\tabu@Xalign #1#2{%
935     \ifx \tabu@Xlcr\@empty \else \PackageWarning{tabu}
936         {Duplicate horizontal alignment specification}\fi
937     \ifdefined#1\def\tabu@Xlcr{#1}\let#1\relax
938     \else \def\tabu@Xlcr{#2}\let#2\relax\fi
939     \expandafter\tabu@Xparse
940 }% \tabu@Xalign
941 \def\tabu@Xtype #1{%
942     \ifx \tabu@rewritten\@empty \else \PackageWarning{tabu}
943         {Duplicate vertical alignment specification}\fi
944     \def\tabu@rewritten{#1}\expandafter\tabu@Xparse
945 }% \tabu@Xtype
946 \def\tabu@Xcoef#1{\edef\tabu@temp{\tabu@temp#1}%
947     \afterassignment\tabu@Xc@ef \tabu@cnt\number\if-#10\fi
948 }% \tabu@Xcoef
949 \def\tabu@Xc@ef{\advance\tabu@colX \tabu@temp\the\tabu@cnt\p@
950     \tabu@Xparse{}%
951 }% \tabu@Xc@ef
952 \def\tabu@Xm@th #1{\futurelet \@let@token \tabu@Xd@sp}
953 \def\tabu@Xd@sp{\let\tabu@Xmath=$%
954     \ifx ${@let@token \def\tabu@Xdisp{\displaystyle}%
955         \expandafter\tabu@Xparse
956     \else \expandafter\tabu@Xparse\expandafter{\expandafter}%
957     \fi
958 }% \tabu@Xd@sp

```

\tabu@Xrewritten Final step: the whole optional argument has been read, then builds the rewritten column specification.

```

959 \def\tabu@Xrewritten {%
960     \ifx \tabu@rewritten\@empty \def\tabu@rewritten{p}\fi
961     \ifdim \tabu@colX<\z@ \tabu@negcoeftrue
962     \else\ifdim \tabu@colX=\z@ \tabu@colX \p@
963     \fi\fi

```



```

964 \edef\tabu@temp{\the\tabu@Xcol}\tabu@strippt\tabucolX}%
965 \edef\tabu@Xcoefs{\tabu@Xcoefs \tabu@ \tabu@temp}%
966 \edef\tabu@rewritten##1##2{\def\noexpand\tabu@rewritten{%
967 >\tabu@Xlcr \ifx$\tabu@Xmath$\tabu@Xdisp\fi ##1}%
968 \tabu@rewritten {\tabu@hsize \tabu@temp}%
969 <{##2\ifx$\tabu@Xmath$\fi}}%
970 }\tabu@rewritten
971}% \tabu@Xrewritten

```

\tabu@hsize `\tabu@hsize {X column number}{X column width coefficient}`

Depending on the sign of the coefficient, and of the stored value for the natural width of the column the X cell belongs to, `\tabu@hsize` returns the wanted width for the *par-box* that contains the cell content.

```

972 \def\tabu@hsize #1#2{%
973 \ifdim #2\p@<\z@
974 \ifdim \tabucolX=\maxdimen \tabu@wd{#1}\else
975 \ifdim \tabu@wd{#1}<-\#2\tabucolX \tabu@wd{#1}\else -\#2\tabucolX\fi
976 \fi
977 \else #2\tabucolX
978 \fi
979}% \tabu@hsize

```

Rewriting `\usetabu` and `\preamble`

The rewriting process is very simple, when all the job has been done cleverly at the time of `\savetabu!!`

The `\savetabu` macro is a bit more complex...

\usetabu (private column type) `\usetabu` is defined as a `tabu` new column type: loaded only inside the `\@mkpream` group inside the `tabu` environment.

```

980 \tabu@privatecolumntype \usetabu [1]{%
981 \ifx\#1\#1\tabu@saveerr{}\else
982 \ifundefined{tabu@saved@\string#1}
983 {\tabu@saveerr{#1}}
984 {\let\tabu@rewriteX \tabu@rewriteXrestore
985 \csname tabu@saved@\string#1\expandafter\endcsname\expandafter\@ne}%
986 \fi
987}% \NC@rewrite@\usetabu

```

\preamble (private column type) `\preamble` is defined as a `tabu` new column type: loaded only inside the `\@mkpream` group inside the `tabu` environment.

```

988 \tabu@privatecolumntype \preamble [1]{%
989 \ifx\#1\#1\tabu@saveerr{}\else
990 \ifundefined{tabu@saved@\string#1}
991 {\tabu@saveerr{#1}}
992 {\csname tabu@saved@\string#1\expandafter\endcsname\expandafter\z@}%
993 \fi
994}% \NC@rewrite@\preamble

```

Controlling the rewriting process

\tabu@rewritefirst This new column type is not really a column type! It is always added to a `tabu` preamble in order to do some setup before any other column is rewritten by `\@mkpream`.

Thus, `\NC@list` is simply set to `{\NC@do \tabu@rewritefirst }`. The rewriting of `\tabu@rewritefirst` will restore the original list `\NC@list`.

This “column type” sets:

- `\tabu@select` to be expanded `\aftergroup` (after the closing of the `\@mkpream` group. All the thick is there: all information collected during the rewriting of X columns (and vertical lines or leaders) can be *reinject*ed into the group below the `\@mkpream` group, by the mean of the

`\tabu@mkpreamblebuffer` (globally defined).

- The private columns types are loaded by `\tabu@rewritefirst`: they will be rewritten afterwards, during the rewriting loop. This way, X column definition for `tabu` are only available during the rewriting process of the `tabu` preamble, making it possible (and easy) to embed a `tabularx` inside a cell of a `tabu`.
- `\save@decl` is modified inside the `\@mkpream` group, if `tabu` is in text mode.

```

995 \tabu@newcolumnntype \tabu@rewritefirst{%
996   \iftabu@long \aftergroup \tabu@longpream % <the whole implementation is here !>
997   \else \aftergroup \tabu@pream
998   \fi
999   \let\tabu@ \relax \let\tabu@hsize \relax
1000  \let\tabu@Xcoefs \@empty \let\tabu@savels \relax
1001  \tabu@Xcol \z@ \tabu@cnt \tw@
1002  \gdef\tabu@mkpreamblebuffer{\tabu@{}}\tabu@measuringfalse
1003  \global\setbox\@arstrutbox \box\@arstrutbox
1004  \NC@list{\NC@do *}\tabu@textbar \tabu@lines
1005  \NC@list\expandafter{\the\NC@list \NC@do X}%
1006  \iftabu@siunitx % <siunitx S and s columns>
1007    \NC@list\expandafter{\the\NC@list \NC@do S\NC@do s}\fi
1008  \NC@list\expandafter{\the\expandafter\NC@list \tabu@highprioritycolumns}%
1009  \expandafter\def\expandafter\tabu@NC@list\expandafter{%
1010    \the\expandafter\NC@list \tabu@NC@list}% * | X S <original>
1011  \NC@list\expandafter{\expandafter \NC@do \expandafter\usetabu
1012    \expandafter \NC@do \expandafter\preamble
1013    \the\NC@list \NC@do \tabu@rewritemiddle
1014    \NC@do \tabu@rewritelast}%
1015  \tabu@savdecl
1016  \tabu@privatecolumns
1017  \edef\tabu@prev{\the\@temptokena}\NC@find \tabu@rewritemiddle
1018 }% \NC@rewrite@\tabu@rewritefirst

```

`\tabu@rewritemiddle` This new column type is rewritten after X columns, because it is declared by when the column `\tabu@rewritelast` `\tabu@rewritefirst` is actually rewritten. In the case where `\tabu@target` is > 0 (either because of “`tabu to`” or “`tabu spread`” has been called) and if there is no X column, then `\extracolsep \flushglue` is added at the beginning of the preamble.

To avoid duplicate margin in the `tabu` we have to test the next token in the preamble. If the next token is `|` or `!` then no margin must be added and `\extracolsep \flushglue` can be inserted at the beginning of the preamble.

Otherwise, we must insert `!\extracolsep \flushglue` in order to keep the margin.

`\tabu@rewritelast` column type is loaded by `\tabu@rewritefirst` column type, only inside the `\@mkpream` group inside the `tabu` environment.

```

1019 \tabu@newcolumnntype \tabu@rewritemiddle{%
1020   \edef\tabu@temp{\the\@temptokena}\NC@find \tabu@rewritelast
1021 }% \NC@rewrite@\tabu@rewritemiddle
1022 \tabu@newcolumnntype \tabu@rewritelast{%
1023   \ifx \tabu@temp\tabu@prev \advance\tabu@cnt \m@ne
1024     \NC@list\expandafter{\tabu@NC@list \NC@do \tabu@rewritemiddle
1025     \NC@do \tabu@rewritelast}%
1026   \else \let\tabu@prev\tabu@temp
1027   \fi
1028   \ifcase \tabu@cnt \expandafter\tabu@endrewrite
1029   \else \expandafter\NC@find \expandafter\tabu@rewritemiddle
1030   \fi
1031 }% \NC@rewrite@\tabu@rewritelast

```

The end of the rewriting process: determining the tabu strategy

\tabu@endrewrite Determines the strategy to be executed **\aftergroup** (at the closing of the **\@mkpream** group):

- 0) There is no real strategy: **tabu** behaves like **tabular**, there no **X** column, and no need to measure the vertical dimensions of the cells (no dynamic spacing, no vertical leader). In case a target has been given to **tabu**, it behaves like **tabular*** and a infinite stretchability is given to the column inter-space. This is done (if required) by **\tabu@extracolsep**.
- 1) Measuring natural width of some (or all) columns is compulsory for **tabu spread** of **X** columns with negativ coefficients. Thereafter, the strategy nr 2 will bring into play.
- 2) Measuring the natural width is not necessary, or has been done before. But **tabu** contains **X** columns and trials have to be performed to reach the desired target, adjusting the **\tabucolX** dimension accordingly. Then, the strategy nr 3 may bring into play, if vertical measure is required.
- 3) Vertical measure of the cells is required, for vertical spacing adjustment or vertical leaders. This step can be done only if the width are known.

> 3 The **tabu** is finished and ready to be printed !!

```

1032 \def\tabu@endrewrite {%
1033     \let\tabu@temp \NC@find
1034     \ifx \@arrayright\relax \let\@arrayright \@empty \fi
1035     \count@=%
1036     \ifx \@finalstrut\tabu@finalstrut \z@ % outer in mode 0 print
1037         \iftabu@measuring
1038             \xdef\tabu@mkpreambuffer{\tabu@mkpreambuffer
1039                 \tabu@target         \csname tabu@\the\tabu@nested.T\endcsname
1040                 \tabucolX             \csname tabu@\the\tabu@nested.X\endcsname
1041                 \edef\@halignto {\ifx\@arrayright\@empty to\tabu@target\fi}}%
1042             \fi
1043         \else\iftabu@measuring         4           % X columns
1044             \xdef\tabu@mkpreambuffer{\tabu@{\tabu@mkpreambuffer
1045                 \tabu@target         \the\tabu@target
1046                 \tabu@spreadtarget   \the\tabu@spreadtarget}%
1047                 \def\noexpand\tabu@Xcoefs{\tabu@Xcoefs}%
1048                 \edef\tabu@halignto{\ifx \@arrayright\@empty to\tabu@target\fi}}%
1049             \let\tabu@Xcoefs \relax
1050         \else\ifcase\tabu@nested \thr@@ % outer, no X
1051             \global\let\tabu@afterendpar \relax
1052         \else                       \@ne    % inner, no X, outer in mode 1 or 2
1053             \fi
1054         \ifdefined\tabu@usetabu
1055         \else \ifdim\tabu@target=\z@
1056         \else \let\tabu@temp \tabu@extracolsep
1057         \fi\fi
1058     \fi
1059     \fi
1060     \xdef\tabu@mkpreambuffer{\count@ \the\count@ \tabu@mkpreambuffer}%
1061     \tabu@temp
1062 }% \tabu@endrewrite

```

\tabu@extracolsep Inserts **\extracolsep \@flushglue** in front of the preamble, unless another value for **\extracolsep** has been specified.

\@flushglue is **Opt plus 1fl**.

```

1063 \def\tabu@extracolsep{\@defaultunits \expandafter\let
1064     \expandafter\tabu@temp \expandafter=\the\@temptokena \relax\@nnil
1065     \ifx \tabu@temp\@sptoken
1066         \expandafter\tabu@gobblespace \expandafter\tabu@extracolsep

```

```

1067 \else
1068     \edef\tabu@temp{\noexpand\NC@find
1069         \if |\noexpand\tabu@temp      @%
1070         \else\if !\noexpand\tabu@temp  @%
1071         \else                            !%
1072         \fi\fi
1073         {\noexpand\extracolsep\noexpand\@flushglue}}%
1074 \fi
1075 \tabu@temp
1076 }% \tabu@extrac@lsep

```

11.10 Implementing the strategy at the exit of the \@mkpream group

\tabu@select

\tabu@pream Triggered `\aftergroup` by the rewriting of `\tabu@rewritefirst`.

The `\tabu@mkpreambuffer` macro is expanded twice: first it injects `\count@` (the strategy number) and `\tabu@nbc`, and redefines itself.

Second – and only if measurements are necessary – it expands into the *trials group* to inject `\tabu@Xcoefs` (the coefficients of *X* columns), `\tabu@Xsum` (the sum of the absolute coefficients), `\tabu@target`, `\tabu@spreadtarget`, and `\tabu@vertical`, which is the number by which one have to increment the strategy number after step 2 (either 1: then a last measure is done for the vertical dimensions, or 255 then the strategy number is > 3 and `\tabu@strategy` orders to finish.)

\tabu@longpream This is the long version for `longtabu`: the material to collect until `\@preamble` is different !

```

1077 \long\def\tabu@pream #1\@preamble {%
1078     \let\tabu@ \tabu@@ \tabu@mkpreambuffer \tabu@aftergroupcleanup
1079     \NC@list\expandafter {\tabu@NC@list}% in case of nesting...
1080     \ifdefined\tabu@usetabu \tabu@usetabu \tabu@target \z@ \fi
1081     \let\tabu@savdpreamble \@preamble
1082     \global\let\tabu@elapsedtime \relax
1083     \tabu@thebody ={\#1\tabu@aftergroupcleanup}%
1084     \tabu@thebody =\expandafter{\the\expandafter\tabu@thebody
1085                                     \@preamble}%
1086     \edef\tabuthepreamble {\the\tabu@thebody}% ( no @ allowed for \scantokens )
1087     \tabu@select
1088 }% \tabu@pream
1089 \long\def\tabu@longpream #1\LT@bchunk #2\LT@bchunk{%
1090     \let\tabu@ \tabu@@ \tabu@mkpreambuffer \tabu@aftergroupcleanup
1091     \NC@list\expandafter {\tabu@NC@list}% in case of nesting...
1092     \let\tabu@savdpreamble \@preamble
1093     \global\let\tabu@elapsedtime \relax
1094     \tabu@thebody ={\#1\LT@bchunk #2\tabu@aftergroupcleanup \LT@bchunk}%
1095     \edef\tabuthepreamble {\the\tabu@thebody}% ( no @ allowed for \scantokens )
1096     \tabu@select
1097 }% \tabu@longpream

```

\tabu@select Here we check if trials are required or not: depending on the value of `\count@` (set at `\tabu@endrewrite`, and *injected* here by `\tabu@mkpreambuffer`), on `\iftabu@measuring` (nested trials).

When trials are required, `\tabu@select` give control to `\tabu@setstrategy` (to prepare the neutralisation of commands, save counters etc).

When trials are not required, we just have to expand `\tabuthepreamble`, after having set up the `\everyrow` stuff properly (for vertical adjustment or vertical measure, if needed).

```

1098 \def\tabu@select {%
1099     \ifnum\tabu@nested>\z@ \tabuscantokensfalse \fi
1100     \ifnum \count@=\@ne \iftabu@measuring \count@=\tw@ \fi\fi
1101     \ifcase \count@

```

```

1102     \global\let\tabu@elapsedtime \relax
1103     \tabu@seteverycr
1104     \expandafter \tabu@hepreamble           % vertical adjustment (inherited from outer)
1105 \or     % exit in vertical measure + struts per cell because no X and outer in mode 3
1106     \tabu@evr{\tabu@verticalinit}\tabu@celllalign@def{\tabu@verticalmeasure}%
1107     \def\tabu@cellralign{\tabu@verticalspacing}%
1108     \tabu@seteverycr
1109     \expandafter \tabu@hepreamble
1110 \or     % exit without measure because no X and outer in mode 4
1111     \tabu@evr{\tabu@celllalign@def{}\let\tabu@cellralign \@empty
1112     \tabu@seteverycr
1113     \expandafter \tabu@hepreamble
1114 \else     % needs trials
1115     \tabu@evr{\tabu@celllalign@def{}\let\tabu@cellralign \@empty
1116     \tabu@savecounters
1117     \expandafter \tabu@setstrategy
1118 \fi
1119 }% \tabu@select
1120 \def\tabu@@ {\gdef\tabu@mkpreambuffer}

```

General setup for trials: neutralisation of \write etc.

\tabu@setstrategy This is the general setup for trials: the tabu will be expanded more than once, thus some protections are set: the value of global counters are saved, footnotes have a special setup, \hbadness and \hfuzz are neutralised etc.

The initial value for \tabucolX is computed with the coefficients stored into \tabu@Wvoefs:
 $\tabu@{coef1} \tabu@{coef2} \tabu@{coef3}$ etc.

is very suitable for loops on the column width coefficients (without the need of \@for or whatsoever).

```

1121 \def\tabu@setstrategy {\begingroup % <trials group>
1122     \tabu@trialh@k \tabu@cnt \z@ % number of trials
1123     \hbadness \@M \let\hbadness \@tempcnta
1124     \hfuzz \maxdimen \let\hfuzz \@tempdima
1125     \let\write \tabu@nowrite\let\GenericError \tabu@GenericError
1126     \let\savetabu \@gobble \let\tabubefaulttarget \linewidth
1127     \let\@footnotetext \@gobble \let\@xfootnote \tabu@xfootnote
1128     \let\color \tabu@nocolor\let\rowcolor \tabu@norowcolor
1129     \let\tabu@aftergroupcleanup \relax % only after the last trial
1130     \tabu@mkpreambuffer
1131     \ifnum \count@>\thr@@ \let\@halignto \@empty \tabucolX@init
1132     \def\tabu@lasttry{\m@ne\p@}\fi
1133     \begingroup \iffalse{\fi \ifnum 0='}\fi
1134     \toks@{\}\def\tabu@stack{b}\iftabuscantokens \endlinechar=10 \obeyspaces \fi %
1135     \tabu@collectbody \tabu@strategy %
1136 }% \tabu@setstrategy
1137 \def\tabu@savecounters{%
1138     \def\@elt ##1{\csname c@##1\endcsname\the\csname c@##1\endcsname}%
1139     \edef\tabu@clckpt {\begingroup \globaldefs=\@ne \cl@ckpt \endgroup}\let\@elt \relax
1140 }% \tabu@savecounters
1141 \def\tabucolX@init {% \tabucolX <= \tabu@target / (sum coefs > 0)
1142     \dimen@ \z@ \tabu@Xsum \z@ \tabucolX \z@ \let\tabu@ \tabu@Xinit \tabu@Xcoefs
1143     \ifdim \dimen@>\z@
1144         \@tempdima \dimexpr \tabu@target * \p@/\dimen@ + \tabu@hfuzz\relax
1145         \ifdim \tabucolX<@\tempdima \tabucolX \@tempdima \fi
1146     \fi
1147 }% \tabucolX@init
1148 \def\tabu@Xinit #1#2{\tabu@Xcol #1 \advance \tabu@Xsum
1149     \ifdim #2\p@>\z@ #2\p@ \advance\dimen@ #2\p@

```

```

1150 \else -#2\p@ \tabu@negcoeftrue
1151 \@tempdima \dimexpr \tabu@target*\p@/\dimexpr-#2\p@\relax \relax
1152 \ifdim \tabucolX<\@tempdima \tabucolX \@tempdima \fi
1153 \tabu@wdddef{#1}{0pt}%
1154 \fi
1155 }% \tabu@Xinit

```

Collecting the tabu body

The macro collect the stuff inside `\@array`: depending on the global vertical alignment parameter for the whole tabular, the tabular is built inside a `\vbox`, `\vtop` or `\vcenter` (the default – unless `tabu` is nested).

At this time, we define `\tabu@trial` (which inherits from the `\vbox`, `\vtop` or `\vcenter`) and `\tabu@Xfinish` as well.

`\tabu@collectbody` The mechanism is the same as \mathcal{AMS} -`\collect@body` (also defined in `environ.sty`). The content of the tabular is captured inside `\toks@`, expanded by `\tabu@trial`.

`\tabu@endofcollect`

```

1156 \long\def\tabu@collectbody #1#2\end #3{%
1157 \edef\tabu@stack{\tabu@pushbegins #2\begin\end\expandafter\@gobble\tabu@stack}%
1158 \ifx \tabu@stack\@empty
1159 \toks@\expandafter{\expandafter\tabu@thebody\expandafter{\the\toks@ #2}%
1160 \def\tabu@end@envir{\end{#3}}%
1161 \iftabuscantokens
1162 \iftabu@long \def\tabu@endenvir {\end{#3}\tabu@gobbleX}%
1163 \else \def\tabu@endenvir {\let\endarray \@empty
1164 \end{#3}\tabu@gobbleX}%
1165 \fi
1166 \else \def\tabu@endenvir {\end{#3}}\fi}%
1167 \let\tabu@collectbody \tabu@endofcollect
1168 \else\def\tabu@temp{#3}%
1169 \ifx \tabu@temp\@empty \toks@\expandafter{\the\toks@ #2\end }%
1170 \else \ifx\tabu@temp\tabu@@spxiix \toks@\expandafter{\the\toks@ #2\end #3}%
1171 \else \ifx\tabu@temp\tabu@X \toks@\expandafter{\the\toks@ #2\end #3}%
1172 \else \toks@\expandafter{\the\toks@ #2\end{#3}}%
1173 \fi\fi\fi
1174 \fi
1175 \tabu@collectbody{#1}%
1176 }% \tabu@collectbody
1177 \long\def\tabu@pushbegins#1\begin#2{\ifx\end#2\else b\expandafter\tabu@pushbegins\fi}%
1178 \def\tabu@endofcollect #1{\ifnum0=}`}\fi
1179 \expandafter\endgroup \the\toks@ #1%
1180 }% \tabu@endofcollect

```

11.11 One trial after the other (`\tabu@strategy`)

Switching between the strategies

`\tabu@strategy` This macro does some specific setup depending on the strategy (1, 2 or 3), and orders to finish when all measurements are done.

This consists in a switch (`\ifcase`) which is done before the trials by `\tabu@strategy`, and after the trials by `\tabu@endtrial`.

```

1181 \def\tabu@strategy {\relax % stops \count@ assignment !
1182 \ifcase\count@ % case 0 = print with vertical adjustment (outer is finished)
1183 \expandafter \tabu@endoftrials
1184 \or % case 1 = exit in vertical measure (outer in mode 3)
1185 \expandafter\xdef\csname tabu@\the\tabu@nested.T\endcsname{\the\tabu@target}%
1186 \expandafter\xdef\csname tabu@\the\tabu@nested.X\endcsname{\the\tabucolX}%
1187 \expandafter \tabu@endoftrials
1188 \or % case 2 = exit with a rule replacing the table (outer in mode 4)

```

```

1189     \expandafter \tabu@quickend
1190     \or                                     % case 3 = outer is in mode 3 because of no X
1191     \begingroup
1192         \tabu@evr{\tabu@verticalinit}\tabu@celllalign@def{\tabu@verticalmeasure}%
1193         \def\tabu@cellralign{\tabu@verticalspacing}%
1194         \expandafter \tabu@measuring
1195     \else                                     % case 4 = horizontal measure
1196     \begingroup
1197         \global\let\tabu@elapsedtime \tabu@message@etime
1198         \long\def\multicolumn##1##2##3{\multispan{##1}}%
1199         \let\tabu@startpboxORI \@startpbox
1200         \iftabu@spread
1201             \def\tabu@naturalXmax {\z@}%
1202             \let\tabu@naturalXmin \tabu@naturalXmax
1203             \tabu@evr{\global\tabu@naturalX \z@}%
1204             \let\@startpbox \tabu@startpboxmeasure
1205         \else\iftabu@negcoef
1206             \let\@startpbox \tabu@startpboxmeasure
1207         \else \let\@startpbox \tabu@startpboxquick
1208         \fi\fi
1209         \expandafter \tabu@measuring
1210     \fi
1211 }% \tabu@strategy

```

\tabu@measuring Expands `\tabu@trial` with the whole content of the environment stored in `\toks@` by `\tabu@collectbody`. At the end of the trial, `\count@` will be reassigned to the value it had before the trial. Then `\tabu@endtrial` will choose the algorithm depending on the strategy number, and set the new strategy number (into `\count@` again) for the next step.

\tabu@trial This is the starting point of trials: `\halign` is expanded here.

\tabu@longtrial This is the long version of `\tabu@trial` for `longtabu`. Almost the same apart for the math group and the end (a `longtable` environment does not finish with `\endarray`).

```

1212 \def\tabu@measuring{\expandafter \tabu@trial \expandafter
1213                                     \count@ \the\count@ \tabu@endtrial
1214 }% \tabu@measuring
1215 \def\tabu@trial{\iftabu@long \tabu@longtrial \else \tabu@shorttrial \fi}
1216 \def\tabu@shorttrial {\setbox\tabu@box \hbox\bgroup \tabu@seteverycr
1217     \ifx \tabu@savecounters\relax \else
1218         \let\tabu@savecounters \relax \tabu@clckpt \fi
1219     $\iftabuscantokens \tabu@rescan \else \expandafter\@secondoftwo \fi
1220     \expandafter{\expandafter \tabu@the preamble
1221                 \the\tabu@thebody
1222                 \csname tabu@adl@endtrial\endcsname
1223                 \endarray}$\egroup % got \tabu@box
1224 }% \tabu@shorttrial
1225 \def\tabu@longtrial {\setbox\tabu@box \hbox\bgroup \tabu@seteverycr
1226     \ifx \tabu@savecounters\relax \else
1227         \let\tabu@savecounters \relax \tabu@clckpt \fi
1228     \iftabuscantokens \tabu@rescan \else \expandafter\@secondoftwo \fi
1229     \expandafter{\expandafter \tabu@the preamble
1230                 \the\tabu@thebody
1231                 \tabu@endlongtrial}\egroup % got \tabu@box
1232 }% \tabu@longtrial
1233 \def\tabu@endlongtrial{% no @ allowed for \scantokens
1234     \LT@echunk \global\setbox\@ne \hbox{\unhbox\@ne}\kern\wd\@ne
1235     \LT@get@widths
1236 }% \tabu@endlongtrial

```

```
1237 \def\tabu@adl@endtrial{% <arydshln in nested trials - problem for global column counters!>
1238     \crrc \noalign{\global\adl@ncol \tabu@nbccols}}% anything global is crap, junky and fails !
```

\tabu@seteverycr \ialign resets \everycr to an empty token. This macro sets \everycr for the tabu environment : a *bridge* around \ialign is built: \everycr redefines itself \afterassignment!

```
1239 \def\tabu@seteverycr {\tabu@reset
1240     \everycr \expandafter{\the\everycr \tabu@everycr}%
1241     \let\everycr \tabu@noeverycr % <for ialign>
1242 }% \tabu@seteverycr
1243 \def\tabu@noeverycr{{\aftergroup\tabu@restoreeverycr \afterassignment}\toks@}
1244 \def\tabu@restoreeverycr {\let\everycr \tabu@@everycr}
1245 \def\tabu@everycr {\iftabu@everyrow \noalign{\tabu@everyrow}\fi}
```

\tabu@endoftrials When the algorithm said the tabular was ready to be printed, \tabu@endoftrials closes the trials group and prints the tabular...

The required values (column widths, struts etc.) are *injected* into the group by the mean of the buffer \tabu@bufferX (locally defined).

\tabu@closetrialsgroup This closes the group in which all the trials are done.

```
1246 \def\tabu@endoftrials {%
1247     \iftabuscantokens \expandafter\@firstoftwo
1248     \else \expandafter\@secondoftwo
1249     \fi
1250     {\expandafter \tabu@closetrialsgroup \expandafter
1251     \tabu@rescan \expandafter{%
1252         \expandafter\tabuthepreamble
1253         \the\expandafter\tabu@thebody
1254         \iftabu@long \else \endarray \fi}}
1255     {\expandafter\tabu@closetrialsgroup \expandafter
1256     \tabuthepreamble
1257     \the\tabu@thebody}%
1258     \tabu@endenvir % Finish !
1259 }% \tabu@endoftrials
1260 \def\tabu@closetrialsgroup {%
1261     \toks@\expandafter{\tabu@endenvir}%
1262     \edef\tabu@bufferX{\endgroup
1263         \tabuacolX \the\tabuacolX
1264         \tabu@target \the\tabu@target
1265         \tabu@cnt \the\tabu@cnt
1266         \def\noexpand\tabu@endenvir{\the\toks@}%
1267         %Quid de \@halignto = \tabu@halignto ??
1268     }% \tabu@bufferX
1269     \tabu@bufferX
1270     \ifcase\tabu@nested % print out (outer in mode 0)
1271         \global\tabu@cnt \tabu@cnt
1272         \tabu@evr{\tabu@verticaldynamicadjustment}%
1273         \tabu@celllalign@def{\everypar{}}\let\tabu@cellralign \@empty
1274         \let\@finalstrut \tabu@finalstrut
1275     \else % vertical measure of nested tabu
1276         \tabu@evr{\tabu@verticalinit}%
1277         \tabu@celllalign@def{\tabu@verticalmeasure}%
1278         \def\tabu@cellralign{\tabu@verticalspacing}%
1279     \fi
1280     \tabu@clckpt \let\@halignto \tabu@halignto
1281     \let\@halignto \@empty
1282     \tabu@seteverycr
1283     \ifdim \tabustrutrule>\z@ \ifnum\tabu@nested=\z@
1284         \setbox\@arstrutbox \box\voidb@x % force \@arstrutbox to be rebuilt (visible struts)
```



```
1285 \fi\fi
1286 }% \tabu@closetrialsgroup
```

\tabu@quickend Quick exit after having measuring the natural width of a nested **tabu**.

```
1287 \def\tabu@quickend {\expandafter \endgroup \expandafter
1288 \tabu@target \the\tabu@target \tabu@quickrule
1289 \let\endarray \relax \tabu@endenvir
1290 }% \tabu@quickend
```

\tabu@endtrial Depending on the strategy that was just applied, **\tabu@endtrial** chooses the algorithm and determines the number of the strategy for the next step.

```
1291 \def\tabu@endtrial {\relax % stops \count@ assignment !
1292 \ifcase \count@ \tabu@err % case 0 = impossible here
1293 \or \tabu@err % case 1 = impossible here
1294 \or \tabu@err % case 2 = impossible here
1295 \or % case 3 = outer goes into mode 0
1296 \def\tabu@bufferX{\endgroup}\count@ \z@
1297 \else % case 4 = outer goes into mode 3
1298 \iftabu@spread \tabu@spreadarith % inner into mode 1 (outer in mode 3)
1299 \else \tabu@arith % or 2 (outer in mode 4)
1300 \fi
1301 \count@=%
1302 \ifcase\tabu@nested \thr@@ % outer goes into mode 3
1303 \else\iftabu@measuring \tw@ % outer is in mode 4
1304 \else \@ne % outer is in mode 3
1305 \fi\fi
1306 \edef\tabu@bufferX{\endgroup
1307 \tabu@colX \the\tabu@colX
1308 \tabu@target \the\tabu@target}%
1309 \fi
1310 \expandafter \tabu@bufferX \expandafter
1311 \count@ \the\count@ \tabu@strategy
1312 }% \tabu@endtrial
1313 \def\tabu@err{\errmessage{(tabu) Internal impossible error! (\count@=\the\count@)}}
```

11.12 The algorithms: Measuring the tabu box

At the end of each trial, we call **\tabu@arith** (or **\tabu@spreadarith**) to computes the widths and update the values.

At the exit, **\iftabu@measuring** is set to **\iftrue**: a further trial is necessary, or **\iffalse**: the target width is reached.

The arithmetic of X columns: the tabu to case

\tabu@arithnegcoef This is a loop against the width coefficients. There is no **\@for** or **\@whiles** because **\tabu@Xcoefs** stores the series in the form: **\tabu@ {coef1} \tabu@ {coef2} \tabu@ {coef3}**.

Thus, just **\let \tabu@** to be **\tabu@arith@negcoef** and expand **\tabu@Xcoefs**!

The aim of the game is to *neutralize* some X columns: when their natural width are less than **coef**×**\tabu@colX**.

```
1314 \def\tabu@arithnegcoef {%
1315 \@tempdima \z@ \dimen@ \z@ \let\tabu@ \tabu@arith@negcoef \tabu@Xcoefs
1316 }% \tabu@arithnegcoef
1317 \def\tabu@arith@negcoef #1#2{%
1318 \ifdim #2\p@>\z@ \advance\dimen@ #2\p@ % saturated by definition
1319 \advance\@tempdima #2\tabu@colX
1320 \else
1321 \ifdim -#2\tabu@colX <\tabu@wd{#1}% c_i X < natural width <= \tabu@target-> saturated
1322 \advance\dimen@ -#2\p@
```

```

1323          \advance\@tempdima -#2\tabucolX
1324      \else
1325          \advance\@tempdima \tabu@wd{#1}% natural width <= c_i X => neutralised
1326          \ifdim \tabu@wd{#1}<\tabu@target \else % neutralised
1327          \advance\dimen@ -#2\p@ % saturated (natural width = tabu@target)
1328          \fi
1329      \fi
1330  \fi
1331 }% \tabu@arith@negcoef

```

\tabu@arith General algorithms for tabu to with X columns.

```

1332 \def\tabu@givespace #1#2{% here \tabu@DELTA < \z@
1333     \ifdim \@tempdima=\z@
1334         \tabu@wddef{#1}{\the\dimexpr -\tabu@DELTA*\p@/\tabu@Xsum}%
1335     \else
1336         \tabu@wddef{#1}{\the\dimexpr \tabu@hsize{#1}{#2}
1337             * (\p@ -\tabu@DELTA*\p@/\@tempdima) /\p@\relax}%
1338     \fi
1339 }% \tabu@givespace
1340 \def\tabu@arith {\advance\tabu@cnt \@ne
1341     \ifnum \tabu@cnt=\@ne \tabu@message{\tabu@titles}\fi
1342     \tabu@arithnegcoef
1343     \@tempdimb \dimexpr \wd\tabu@box -\@tempdima \relax % <incompressible material>
1344     \tabu@DELTA = \dimexpr \wd\tabu@box - \tabu@target \relax
1345     \tabu@message{\tabu@message@arith}%
1346     \ifdim \tabu@DELTA <\tabu@hfuzz
1347         \ifdim \tabu@DELTA<\z@ % wd (tabu)<\tabu@target ?
1348             \let\tabu@ \tabu@givespace \tabu@Xcoefs
1349             \advance\@tempdima \@tempdimb \advance\@tempdima -\tabu@DELTA % for message
1350             \else % already converged: nothing to do but nearly impossible...
1351             \fi
1352             \tabucolX \maxdimen
1353             \tabu@measuringfalse
1354         \else % need for narrower X columns
1355             \tabucolX =\dimexpr (\@tempdima -\tabu@DELTA) *\p@/\tabu@Xsum \relax
1356             \tabu@measuringtrue
1357             \@whilesw \iftabu@measuring\fi {%
1358                 \advance\tabu@cnt \@ne
1359                 \tabu@arithnegcoef
1360                 \tabu@DELTA =\dimexpr \@tempdima+\@tempdimb -\tabu@target \relax % always < 0 here
1361                 \tabu@message{\tabu@header
1362                     \tabu@msgalign \tabucolX { } { } { } { } { } \@@
1363                     \tabu@msgalign \@tempdima+\@tempdimb { } { } { } { } { } \@@
1364                     \tabu@msgalign \tabu@target { } { } { } { } { } \@@
1365                     \tabu@msgalign@PT \dimen@ { } { } { } { } { } { } \@@
1366                     \ifdim -\tabu@DELTA<\tabu@hfuzz \tabu@spaces target ok\else
1367                     \tabu@msgalign \dimexpr -\tabu@DELTA *\p@/\dimen@ { } { } { } { } { } \@@
1368                     \fi}%
1369                 \ifdim -\tabu@DELTA<\tabu@hfuzz
1370                     \advance\@tempdima \@tempdimb % for message
1371                     \tabu@measuringfalse
1372                 \else
1373                     \advance\tabucolX \dimexpr -\tabu@DELTA *\p@/\dimen@ \relax
1374                 \fi
1375             }%
1376     \fi
1377     \tabu@message{\tabu@message@reached}%
1378     \edef\tabu@bufferX{\endgroup \tabu@cnt \the\tabu@cnt

```

```
1379 \tabucolX \the\tabucolX
1380 \tabu@target \the\tabu@target}%
1381 }% \tabu@arith
```

The arithmetic of X columns for tabu spread

\tabu@spreadarith Algorithm for tabu spread with X columns: the aim of the game is to compute the target (relative to the natural width of the tabular) and go to \tabu@arith afterwards.

```
1382 \def\tabu@spreadarith {%
1383   \dimen@ \z@ \@tempdima \tabu@naturalXmax \let\tabu@ \tabu@spread@arith \tabu@Xcoefs
1384   \edef\tabu@naturalXmin {\the\dimexpr\tabu@naturalXmin*\dimen@/\p@}%
1385   \@tempdimc =\dimexpr \wd\tabu@box -\tabu@naturalXmax+\tabu@naturalXmin \relax
1386   \iftabu@measuring
1387     \tabu@target =\dimexpr \@tempdimc+\tabu@spreadtarget \relax
1388     \edef\tabu@bufferX{\endgroup \tabucolX \the\tabucolX \tabu@target\the\tabu@target}%
1389   \else
1390     \tabu@message{\tabu@message@spreadarith}%
1391     \ifdim \dimexpr \@tempdimc+\tabu@spreadtarget >\tabu@target
1392       \tabu@message{(tabu) spread
1393         \ifdim \@tempdimc>\tabu@target useless here: default target used%
1394         \else too large: reduced to fit default target\fi.}%
1395     \else
1396       \tabu@target =\dimexpr \@tempdimc+\tabu@spreadtarget \relax
1397       \tabu@message{(tabu) spread: New target set to \the\tabu@target^^J}%
1398     \fi
1399     \begingroup \let\tabu@wdddef \@gobbletwo
1400     \@tempdimb \@tempdima
1401     \tabucolX@init
1402     \tabu@arithnegcoef
1403     \wd\tabu@box =\dimexpr \wd\tabu@box +\@tempdima-\@tempdimb \relax
1404     \expandafter\endgroup \expandafter\tabucolX \the\tabucolX
1405     \tabu@arith
1406   \fi
1407 }% \tabu@spreadarith
1408 \def\tabu@spread@arith #1#2{%
1409   \ifdim #2\p@>\z@ \advance\dimen@ #2\p@
1410   \else \advance\@tempdima \tabu@wd{#1}\relax
1411   \fi
1412 }% \tabu@spread@arith
```

Reporting in the .log file (debugshow option)

\tabu@message@defaulttarget

```
1413 \def\tabu@message@defaulttarget{%
1414   \ifnum\tabu@nested=\z@^^J(tabu) Default target:
1415   \ifx\tabu@defaulttarget\linewidth \string\linewidth
1416   \ifdim \tabu@thetarget=\linewidth \else
1417     -\the\dimexpr\linewidth-\tabu@thetarget\fi =
1418   \else\ifx\tabu@defaulttarget\linegoal\string\linegoal=
1419   \fi\fi
1420   \else (tabu) Default target (nested): \fi
1421   \the\tabu@target \on@line
1422   \ifnum\tabu@nested=\z@ , page \the\c@page\fi}
1423 \def\tabu@message@target {^^J(tabu) Target specified:
1424   \the\tabu@target \on@line, page \the\c@page}
```

\tabu@message@arith

```
1425 \def\tabu@message@arith {\tabu@header
```


1530 } \do{ }

11.13 Measuring the natural width of columns (varwidth code from D. Arseneau)

\tabu@startpboxmeasure The important job is done at the end: by **\tabu@endpboxmeasure**.

When “**tabu spread**” is used with **X** columns, the first trial must measure the natural width of the columns. When **X** columns have negativ coefficient, the natural is computed after the target has been reached, with the absolute coefficients.

Nested trials may occur (**tabu spread** inside a **X** column with negativ coefficient for example).

For the furthur trials, the standard scheme for **X** column is used: the natural width is measured only once.

pdfT_EX font expansion is disabled inside the **varwidth** environment (we set **\pdfadjustspacing** to 0).

```

1531 \def\tabu@startpboxmeasure #1{\bgroup % entering \vtop
1532   \edef\tabu@temp{\expandafter\@secondoftwo \ifx\tabu@hsize #1\else\relax\fi}%
1533   \ifodd 1\ifx \tabu@temp\@empty 0 \else % starts with \tabu@hsize ?
1534     \iftabu@spread \else % if spread -> measure
1535     \ifdim \tabu@temp\p@>\z@ 0 \fi\fi\fi% if coef>0 -> do not measure
1536     \let\@startpbox \tabu@startpboxORI % restore immediately (nesting)
1537     \tabu@measuringtrue % for the quick option...
1538     \tabu@Xcol =\expandafter\@firstoftwo\ifx\tabu@hsize #1\fi
1539     \ifdim \tabu@temp\p@>\z@ \ifdim \tabu@temp\tabucolX<\tabu@target
1540       \tabu@target=\tabu@temp\tabucolX \fi\fi
1541     \setbox\tabu@box \hbox \bgroup
1542       \begin{varwidth}\tabu@target
1543         \let\FV@ListProcessLine \tabu@FV@ListProcessLine % \hbox to natural width...
1544         \narrowragged \arraybackslash \parfillskip \@flushglue
1545         \ifdefined\pdfadjustspacing \pdfadjustspacing\z@ \fi
1546         \bgroup \aftergroup\tabu@endpboxmeasure
1547         \ifdefined \cellspacetoplimit \tabu@cellspacepatch \fi
1548       \else \expandafter\@gobble
1549         \tabu@startpboxquick{#1}% \@gobble \bgroup
1550     \fi
1551 }% \tabu@startpboxmeasure
1552 \def\tabu@cellspacepatch{\def\bcolumn##1\@nil{}\let\ecolumn\@empty
1553   \bgroup\color@begingroup}

```

\tabu@endpboxmeasure The cell has been built inside a box: we have to get its dimensions, and update **\tabu@naturalX**, **\tabu@naturalXmin** and **\tabu@naturalXmax** accordingly (for **tabu spread**), and even store (globally) each column width: the column width is the maximum width of the cells it contains.

```

1554 \def\tabu@endpboxmeasure {%
1555   \@finalstrut \@arstrutbox
1556   \end{varwidth}\egroup % <got my \tabu@box>
1557   \ifdim \tabu@temp\p@ <\z@ % neg coef
1558     \ifdim \tabu@wd\tabu@Xcol <\wd\tabu@box
1559       \tabu@wddef\tabu@Xcol {\the\wd\tabu@box}%
1560       \tabu@debug{\tabu@message@endpboxmeasure}%
1561     \fi
1562   \else % spread coef>0
1563     \global\advance \tabu@naturalX \wd\tabu@box
1564     \@tempdima =\dimexpr \wd\tabu@box * \p@/\dimexpr \tabu@temp\p@\relax \relax
1565     \ifdim \tabu@naturalXmax <\tabu@naturalX
1566       \xdef\tabu@naturalXmax {\the\tabu@naturalX}\fi
1567     \ifdim \tabu@naturalXmin <\@tempdima
1568       \xdef\tabu@naturalXmin {\the\@tempdima}\fi
1569   \fi
1570   \box\tabu@box \egroup % end of \vtop (measure) restore \tabu@target

```

```

1571 }% \tabu@endpboxmeasure
1572 \def\tabu@wdef #1{\expandafter\xdef
1573             \csname tabu@\the\tabu@nested.W\number#1\endcsname}
1574 \def\tabu@wd #1{\csname tabu@\the\tabu@nested.W\number#1\endcsname}
1575 \def\tabu@message@endpboxmeasure{\tabu@spaces\tabu@spaces<-> % <-> save natural wd
1576     \the\tabu@Xcol. X[\tabu@temp]:
1577     target = \the\tabu@colX \space
1578     \expandafter\expandafter\expandafter\string\tabu@wd\tabu@Xcol
1579     =\tabu@wd\tabu@Xcol
1580 }% \tabu@message@endpboxmeasure

```

\backslash tabu@startpboxquick Contents of paragraph columns are not built during trials in strategy number 4.

```

1581 \def\tabu@startpboxquick {\bgroup
1582     \let\@startpbox \tabu@startpboxORI % restore immediately
1583     \let\tabu \tabu@quick % \begin is expanded before...
1584     \expandafter\@gobble \@startpbox % gobbles \bgroup
1585 }% \tabu@startpboxquick
1586 \def\tabu@quick {\begingroup \iffalse{\fi \ifnum0='}\fi
1587     \toks@{\def\tabu@stack{b}\tabu@collectbody \tabu@endquick
1588 }% \tabu@quick
1589 \def\tabu@endquick {%
1590     \ifodd 1\ifx\tabu@end@envir\tabu@endtabu \else
1591         \ifx\tabu@end@envir\tabu@endtabus \else 0\fi\fi\relax
1592     \endgroup
1593     \else \let\endtabu \relax
1594     \tabu@end@envir
1595     \fi
1596 }% \tabu@quick
1597 \def\tabu@endtabu {\end{tabu}}
1598 \def\tabu@endtabus {\end{tabu*}}

```

11.14 Measuring the height and depths of rows

\backslash tabu@verticalmeasure Starting point for vertical measure of every cell. Only the maxima/minima are stored, for \mathcal{T}_{Nbc} must know the height/depth of every row.

```

1599 \def\tabu@verticalmeasure{\everypar{}}%
1600     \ifnum \currentgrouptype>12 % 14=semi-simple, 15=math shift group
1601         \setbox\tabu@box =\hbox\bgroup
1602         \let\tabu@verticalspacing \tabu@verticalsp@lcr
1603         \d@llarbegin % after \hbox ...
1604     \else
1605         \edef\tabu@temp{\ifnum\currentgrouptype=5\vtop
1606             \else\ifnum\currentgrouptype=12\vcenter
1607             \else\vbox\fi\fi}%
1608         \setbox\tabu@box \hbox\bgroup$\tabu@temp \bgroup
1609         \let\tabu@verticalspacing \tabu@verticalsp@pmb
1610     \fi
1611 }% \tabu@verticalmeasure

```

\backslash tabu@verticalsp@lcr Vertical spacing adjustment for standard l, c, r columns.

```

1612 \def\tabu@verticalsp@lcr{%
1613     \d@llarend \egroup % <got my \tabu@box>
1614     \@tempdima \dimexpr \ht\tabu@box+\abovetabulinesep
1615     \@tempdimb \dimexpr \dp\tabu@box+\belowtabulinesep \relax
1616     \ifdim\tabu@strutrule>\z@ \tabu@debug{\tabu@message@verticalsp}\fi
1617     \ifdim \tabu@ht<\@tempdima \tabu@htdef{\the\@tempdima}\fi
1618     \ifdim \tabu@dp<\@tempdimb \tabu@dpdef{\the\@tempdimb}\fi
1619     \noindent\vrule height\@tempdima depth\@tempdimb

```

```
1620 }% \tabu@verticalsp@lcr
```

\tabu@verticalsp@pmb Vertical spacing adjustment with struts for p, m, or b columns.

```
1621 \def\tabu@verticalsp@pmb{% inserts struts as needed
1622     \par \expandafter\egroup
1623         \expandafter$\expandafter
1624             \egroup \expandafter
1625                 \@tempdimc \the\prevdepth
1626     \@tempdima \dimexpr \ht\tabu@box+\abovetabulinesep
1627     \@tempdimb \dimexpr \dp\tabu@box+\belowtabulinesep \relax
1628     \ifdim\tabustrutrule>\z@ \tabu@debug{\tabu@message@verticalsp}\fi
1629     \ifdim \tabu@ht<\@tempdima \tabu@htdef{\the\@tempdima}\fi
1630     \ifdim \tabu@dp<\@tempdimb \tabu@dpdef{\the\@tempdimb}\fi
1631     \let\@finalstrut \@gobble
1632     \hrule height\@tempdima depth\@tempdimb width\hsize
1633 %% \box\tabu@box
1634 }% \tabu@verticalsp@pmb
```

\tabu@verticalinit Initialisation of \tabu@ht and \tabu@dp. Done at \everyrow.

```
1635 \def\tabu@verticalinit{%
1636     \ifnum \c@taburow=\z@ \tabu@rearstrut \fi % after \tabu@reset !
1637     \advance\c@taburow \@ne
1638     \tabu@htdef{\the\ht\@arstrutbox}\tabu@dpdef{\the\dp\@arstrutbox}%
1639     \advance\c@taburow \m@ne
1640 }% \tabu@verticalinit
1641 \def\tabu@htdef {\expandafter\xdef \csname tabu@the\tabu@nested.H\the\c@taburow\endcsname}
1642 \def\tabu@ht \csname tabu@the\tabu@nested.H\the\c@taburow\endcsname
1643 \def\tabu@dpdef {\expandafter\xdef \csname tabu@the\tabu@nested.D\the\c@taburow\endcsname}
1644 \def\tabu@dp \csname tabu@the\tabu@nested.D\the\c@taburow\endcsname
```

\tabu@verticaldynamicadjustment This updates the \@arstrutbox at \everyrow (ie.\everycr) in order to adjust the vertical spacing of cells.

```
1645 \def\tabu@verticaldynamicadjustment {%
1646     \advance\c@taburow \@ne
1647     \extrarowheight \dimexpr\tabu@ht - \ht\strutbox
1648     \extrarowdepth \dimexpr\tabu@dp - \dp\strutbox
1649     \let\arraystretch \@empty
1650     \advance\c@taburow \m@ne
1651 }% \tabu@verticaldynamicadjustment
```

11.15 \tabuphantomline

\tabuphantomline This macro inserts a phantom line in front of a tabu. This is necessary when you use \usetabu with tabu X column, with a single line containing \multicolumn...

```
1652 \def\tabuphantomline{\crrc \noalign{%
1653     {\globaldefs \@ne
1654         \setbox\@arstrutbox \box\voidb@x
1655         \let\tabu@celllalign \tabu@celllalign
1656         \let\tabu@cellralign \tabu@cellralign
1657         \let\tabu@cellleft \tabu@cellleft
1658         \let\tabu@cellright \tabu@cellright
1659         \let\tabu@thevline \tabu@thevline
1660         \let\tabu@celllalign \@empty
1661         \let\tabu@cellralign \@empty
1662         \let\tabu@cellright \@empty
1663         \let\tabu@cellleft \@empty
1664         \let\tabu@thevline \relax}%
1665     \edef\tabu@temp{\tabu@multispan \tabu@nbc\{ \noindent & }%}
```



```

1666 \toks@{\expandafter{\tabu@temp \noindent\tabu@everyrowfalse \cr
1667 \noalign{\tabu@rearstrut
1668 {\globaldefs\@ne
1669 \let\tabu@celllalign \tabu@@celllalign
1670 \let\tabu@cellralign \tabu@@cellralign
1671 \let\tabu@cellleft \tabu@@cellleft
1672 \let\tabu@cellright \tabu@@cellright
1673 \let\tabu@thevline \tabu@@thevline}}}%
1674 \expandafter}\the\toks@
1675 }% \tabuphantomline

```

11.16 Horizontal lines inside tabu: `\tabucline`, `\firsthline` and `\lasthline`

Horizontal lines: multiple `\firsthline` / `\lasthline`

`\tabu@firsthline` `\firsthline` and `\lasthline` are `\let` to `\tabu@firsthline` and `\tabu@lasthline` inside the `tabu` environment.

`\tabu@lasthline` This allows to duplicate horizontal lines, while keeping the alignment:

`\tabu@firsthline` `\firsthline \firsthline \firsthline` is allowed inside `tabu` and is the same as:

`\tabu@lasthline`

`\firsthline \hline \hline`.

```

1676 \def\tabu@firstline {\tabu@hlineAZ \tabu@firstlinecorrection {}}
1677 \def\tabu@firsthline{\tabu@hlineAZ \tabu@firstlinecorrection \hline}
1678 \def\tabu@lastline {\tabu@hlineAZ \tabu@lastlinecorrection {}}
1679 \def\tabu@lasthline {\tabu@hlineAZ \tabu@lastlinecorrection \hline}
1680 \def\tabu@hline {% replaces \hline if no colortbl (see \AtBeginDocument)
1681 \noalign{\ifnum0='}\fi
1682 {\CT@arc@\hrule height\arrayrulewidth}%
1683 \futurelet \tabu@temp \tabu@xhline
1684 }% \tabu@hline
1685 \def\tabu@xhline{%
1686 \ifx \tabu@temp \hline
1687 {\ifx \CT@drsc@\relax \vskip
1688 \else\ifx \CT@drsc@\empty \vskip
1689 \else \CT@drsc@\hrule height
1690 \fi\fi
1691 \doublerulesep}%
1692 \fi
1693 \ifnum0='{ \fi}%
1694 }% \tabu@xhline

```

`\tabu@hlineAZ` Here we go, inside a `\noalign` group, we collect the next tokens:

`\tabu@nexthlineAZ`

1. first the option,

2. and then the next tokens if they are `\hline` or `\firsthline`.

`\tabu@xhlineAZ`

The code to be executed at the end of the `\noalign` group is built into `\toks@`.

```

1695 \def\tabu@hlineAZ #1#2{\noalign{\ifnum0='}\fi \dimen@ \z@ \count@ \z@
1696 \toks@{\}\def\tabu@hlinecorrection{#1}\def\tabu@temp{#2}%
1697 \tabu@hlineAZsurround
1698 }% \tabu@hlineAZ
1699 \newcommand*\tabu@hlineAZsurround[1][\extratabsurround]{%
1700 \extratabsurround #1\let\tabucline \tabucline@scan
1701 \let\hline \tabu@hlinescan \let\firsthline \hline
1702 \let\cline \tabu@clinescan \let\lasthline \hline
1703 \expandafter \futurelet \expandafter \tabu@temp
1704 \expandafter \tabu@nexthlineAZ \tabu@temp
1705 }% \tabu@hlineAZsurround
1706 \def\tabu@hlinescan {\tabu@thick \arrayrulewidth \tabu@xhlineAZ \hline}

```

```

1707 \def\tabu@clinescan #1{\tabu@thick \arrayrulewidth \tabu@xhlineAZ {\cline{#1}}}
1708 \def\tabu@cline@scan{\@testopt \tabu@cline@sc@n {}}
1709 \def\tabu@cline@sc@n #1[#2]{\tabu@xhlineAZ {\tabu@cline[{#1}]{#2}}}
1710 \def\tabu@nexthlineAZ{%
1711     \ifx \tabu@temp\hline \else
1712     \ifx \tabu@temp\cline \else
1713     \ifx \tabu@temp\tabu@cline \else
1714         \tabu@hlinecorrection
1715     \fi\fi\fi
1716 }% \tabu@nexthlineAZ
1717 \def\tabu@xhlineAZ #1{%
1718     \toks@\expandafter{\the\toks@ #1}%
1719     \@tempdimc \tabu@thick % The last line width
1720     \ifcase\count@ \@tempdimb \tabu@thick % The first line width
1721     \else \advance\dimen@ \dimexpr \tabu@thick+\doublerulesep \relax
1722     \fi
1723     \advance\count@ \@ne \futurelet \tabu@temp \tabu@nexthlineAZ
1724 }% \tabu@xhlineAZ

```

\tabu@firstlinecorrection This is the “correction macro” for `\firstline`, *ie.*a strut and a skip are inserted **before** the first `\hline`.

```

1725 \def\tabu@firstlinecorrection{% \count@ = number of \hline -1
1726     \@tempdima \dimexpr \ht\@arstrutbox+\dimen@
1727     \edef\firstline{% <local in \noalign>
1728         \omit \hbox to\z@{\hss{\noexpand\tabu@DBG{yellow}\vrule
1729             height \the\dimexpr\@tempdima+\extratabsurround
1730             depth \dp\@arstrutbox
1731             width \tabustrutrule}\hss}\cr
1732         \noalign{\vskip -\the\dimexpr \@tempdima+\@tempdimb
1733             +\dp\@arstrutbox \relax}%
1734         \the\toks@
1735     }\ifnum0=\fi
1736         \expandafter\firstline % we are then !
1737 }% \tabu@firstlinecorrection

```

\tabu@lastlinecorrection This is the “correction macro” for `\lastline`, *ie.*a strut and a skip are inserted **after** the last `\hline`.

```

1738 \def\tabu@lastlinecorrection{%
1739     \@tempdima \dimexpr \dp\@arstrutbox+\dimen@+\@tempdimb+\@tempdimc
1740     \edef\lastline{% <local in \noalign>
1741         \the\toks@
1742         \noalign{\vskip -\the\dimexpr\dimen@+\@tempdimb+\dp\@arstrutbox}%
1743         \omit \hbox to\z@{\hss{\noexpand\tabu@DBG{yellow}\vrule
1744             depth \the\dimexpr \dp\@arstrutbox+\@tempdimb+\dimen@
1745             +\extratabsurround-\@tempdimc
1746             height \z@
1747             width \tabustrutrule}\hss}\cr
1748     }\ifnum0=\fi
1749         \expandafter\lastline % we are then !
1750 }% \tabu@lastlinecorrection

```

\tabu@LT@@hline Allowing colored rules even if `colortbl` is not loaded.

```

1751 \def\tabu@LT@@hline{%
1752     \ifx\LT@next\hline
1753         \global\let\LT@next \@gobble
1754         \ifx \CT@drsc@\relax
1755             \gdef\CT@LT@sep{%
1756                 \noalign{\penalty-\@medpenalty\vskip\doublerulesep}}%

```

```

1757         \else
1758             \gdef\CT@LT@sep{%
1759                 \multispan\LT@cols{%
1760                     \CT@drsc@\leaders\hrule\@height\doublerulesep\hfill}\cr}%
1761         \fi
1762     \else
1763         \global\let\LT@next\empty
1764         \gdef\CT@LT@sep{%
1765             \noalign{\penalty-\@lowpenalty\vskip-\arrayrulewidth}}%
1766     \fi
1767     \ifnum0='\fi}%
1768     \multispan\LT@cols
1769         {\CT@arc@\leaders\hrule\@height\arrayrulewidth\hfill}\cr
1770     \CT@LT@sep
1771     \multispan\LT@cols
1772         {\CT@arc@\leaders\hrule\@height\arrayrulewidth\hfill}\cr
1773     \noalign{\penalty\@M}%
1774     \LT@next
1775 }% \tabu@LT@@hline

```

Horizontal lines: `\tabucline`

`\tabucline` `\tabucline` [style or spec.]{start-end}

`\tabucline` appears only at the end of a line: this is the place where we can insert a `\noalign` group. The line to be inserted inside the `tabu` is build inside this `\noalign` group.

`\tabu@start` and `\tabu@stop` store the limits for the line: they are, for clarity, the local name of `\@tempcnta` and `\@tempcntb`.

```

1776 \let\tabu@start \@tempcnta
1777 \let\tabu@stop \@tempcntb
1778 \newcommand*\tabucline{\noalign{\ifnum0='\fi \tabu@ccline}}
1779 \newcommand*\tabu@ccline[2][\tabu@startstop{#2}]%
1780     \ifnum \tabu@stop<\z@ \toks@{}%
1781     \else \tabu@cclinearg{#1}\tabu@thestyle
1782         \edef\tabucline{\toks@{%
1783             \ifnum \tabu@start>\z@ \omit
1784                 \tabu@multispan\tabu@start {\span\omit}&\fi
1785             \omit \tabu@multispan\tabu@stop {\span\omit}%
1786                 \tabu@thehline\cr
1787         }}\tabucline
1788     \tabu@tracinglines{(tabu:tabucline) Style: #1^^J\the\toks@^^J^^J}%
1789     \fi
1790     \futurelet \tabu@temp \tabu@xccline
1791 }% \tabu@ccline
1792 \def\tabu@cclinearg #1{%
1793     \ifx\#1\\\let\tabu@thestyle \tabu@ls@
1794     \else \@defaultunits \expandafter\let\expandafter\@tempa
1795         \romannumeral-\@0#1\relax \@nnil
1796     \ifx \hbox\@tempa \tabu@cclinebox{#1}%
1797     \else\ifx \box\@tempa \tabu@cclinebox{#1}%
1798     \else\ifx \vbox\@tempa \tabu@cclinebox{#1}%
1799     \else\ifx \vtop\@tempa \tabu@cclinebox{#1}%
1800     \else\ifx \copy\@tempa \tabu@cclinebox{#1}%
1801     \else\ifx \leaders\@tempa \tabu@cclineleads{#1}%
1802     \else\ifx \cleaders\@tempa \tabu@cclineleads{#1}%
1803     \else\ifx \xleaders\@tempa \tabu@cclineleads{#1}%
1804     \else\tabu@getline {#1}%
1805     \fi\fi\fi\fi\fi\fi\fi\fi

```

```

1806     \fi
1807 }% \tabu@clinearg
1808 \def\tabu@clinebox #1{\tabu@clineleads{\xleaders#1\hss}}
1809 \def\tabu@clineleads #1{%
1810     \let\tabu@thestyle \relax \let\tabu@leaders \@undefined
1811     \gdef\tabu@thehrule{#1}}
1812 \def\tabu@thehline{\begingroup
1813     \ifdefined\tabu@leaders
1814         \noexpand\tabu@thehleaders
1815     \else \noexpand\tabu@thehrule
1816     \fi \endgroup
1817 }% \tabu@thehline
1818 \def\tabu@xcline{%
1819     \ifx \tabu@temp\tabucline
1820         \toks@\expandafter{\the\toks@ \noalign
1821         {\ifx\CT@drsc@\relax \vskip
1822         \else \CT@drsc@\hrule height
1823         \fi
1824         \doublerulesep}}%
1825     \fi
1826     \tabu@docline
1827 }% \tabu@xcline
1828 \def\tabu@docline {\ifnum0='\fi \expandafter}\the\toks@}
1829 \def\tabu@docline@evr {\xdef\tabu@doclineafter{\the\toks@}%
1830     \ifnum0='\fi}\aftergroup\tabu@doclineafter}
1831 \def\tabu@multispan #1#2{%
1832     \ifnum\numexpr#1>\@ne #2\expandafter\tabu@multispan
1833     \else \expandafter\@gobbletwo
1834     \fi {#1-1}{#2}%
1835 }% \tabu@multispan

```

\tabu@startstop This macro parses the mandatory argument of **\tabucline**: start-column and end-column of the **\cline**.

```

1836 \def\tabu@startstop #1{\tabu@start@stop #1\relax 1-\tabu@nbcols \@nnil}
1837 \def\tabu@start@stop #1-#2\@nnil{%
1838     \@defaultunits \tabu@start\number 0#1\relax \@nnil
1839     \@defaultunits \tabu@stop \number 0#2\relax \@nnil
1840     \tabu@stop \ifnum \tabu@start>\tabu@nbcols \m@ne
1841     \else\ifnum \tabu@stop=\z@ \tabu@nbcols
1842     \else\ifnum \tabu@stop>\tabu@nbcols \tabu@nbcols
1843     \else \tabu@stop
1844     \fi\fi\fi
1845     \advance\tabu@start \m@ne
1846     \ifnum \tabu@start>\z@ \advance\tabu@stop -\tabu@start \fi
1847 }% \tabu@start@stop

```

11.17 Numbers in tabu

\tabudecimal

\tabudecimal **\tabu@tabudecimal** is **\tabudecimal** inside the tabu environment.

```

1848 \def\tabu@tabudecimal #1{%
1849     \def\tabu@decimal{#1}\@temptokena{}%
1850     \let\tabu@getdecimal@ \tabu@getdecimal@ignorespaces
1851     \tabu@scandecimal
1852 }% \tabu@tabudecimal
1853 \def\tabu@scandecimal{\futurelet \tabu@temp \tabu@getdecimal@}
1854 \def\tabu@skipdecimal#1{#1\tabu@scandecimal}
1855 \def\tabu@getdecimal@ignorespaces{%

```

```

1856 \ifcase 0\ifx\tabu@temp\ignorespaces\else
1857 \ifx\tabu@temp\@sptoken1\else
1858 2\fi\fi\relax
1859 \let\tabu@getdecimal@ \tabu@getdecimal
1860 \expandafter\tabu@skipdecimal
1861 \or \expandafter\tabu@gobblespace\expandafter\tabu@scandecimal
1862 \else \expandafter\tabu@skipdecimal
1863 \fi
1864 }% \tabu@getdecimal@ignorespaces
1865 \def\tabu@get@decimal#1{\@temptokena\expandafter{\the\@temptokena #1}%
1866 \tabu@scandecimal}
1867 \def\do#1{%
1868 \def\tabu@get@decimalspace#1{%
1869 \@temptokena\expandafter{\the\@temptokena #1}\tabu@scandecimal}%
1870 }\do{ }
1871 \let\tabu@@tabudecimal \tabu@tabudecimal

```

\tabu@getdecimal

```

1872 \def\tabu@getdecimal{%
1873 \ifcase 0\ifx 0\tabu@temp\else
1874 \ifx 1\tabu@temp\else
1875 \ifx 2\tabu@temp\else
1876 \ifx 3\tabu@temp\else
1877 \ifx 4\tabu@temp\else
1878 \ifx 5\tabu@temp\else
1879 \ifx 6\tabu@temp\else
1880 \ifx 7\tabu@temp\else
1881 \ifx 8\tabu@temp\else
1882 \ifx 9\tabu@temp\else
1883 \ifx .\tabu@temp\else
1884 \ifx ,\tabu@temp\else
1885 \ifx -\tabu@temp\else
1886 \ifx +\tabu@temp\else
1887 \ifx e\tabu@temp\else
1888 \ifx E\tabu@temp\else
1889 \ifx\tabu@celleft\tabu@templ\else
1890 \ifx\ignorespaces\tabu@templ\else
1891 \ifx\@sptoken\tabu@temp2\else
1892 3\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\relax
1893 \expandafter\tabu@get@decimal
1894 \or \expandafter\tabu@skipdecimal
1895 \or \expandafter\tabu@get@decimalspace
1896 \else\expandafter\tabu@printdecimal
1897 \fi
1898 }% \tabu@getdecimal
1899 \def\tabu@printdecimal{%
1900 \edef\tabu@temp{\the\@temptokena}%
1901 \ifx\tabu@temp\@empty\else
1902 \ifx\tabu@temp\space\else
1903 \expandafter\tabu@decimal\expandafter{\the\@temptokena}%
1904 \fi\fi
1905 }% \tabu@printdecimal

```

11.18 Verbatim inside tabu with X columns

\tabu@verbatim Setup to be done before `\scantokens` to allow verbatim inside the `tabu` environment.

```
1906 \def\tabu@verbatim{%
1907     \let\verb \tabu@verb
1908     \let\FV@DefineCheckEnd \tabu@FV@DefineCheckEnd
1909 }% \tabu@verbatim
```

Compatibility with L^AT_EX's kernel `\verb` command

\tabu@verb The `\verb` macro from the latex kernel expands `\@ifstar` in a context where the space token: `\` has a category code of 12.

This is not compatible with `\scantokens` since `\scantokens` adds a space after each control sequence, including `\verb`:

`\verb +some verbatim text+` becomes:

`\verb \ +some verbatim text+`

and thus, the space token `\` is set as the `\verb` delimiter.

We therefore use (a silly) `\@ifstar` in order to gobble the possible space token.

```
1910 \let\tabu@ltx@verb \verb
1911 \def\tabu@verb{\@ifstar {\tabu@ltx@verb*} \tabu@ltx@verb}
```

Compatibility with the `fancyvrb` package

\tabu@FV@DefineCheckEnd This is quite the same issue as for L^AT_EX `\verb` command: a space is inserted after each control sequence scanned by `\scantoken`.

This leads to a break in the macro that checks the end of a `Verbatim` environment, since this macro basically checks for a line that conforms to the pattern:

#1\end {#2}#3

while with `\scantokens`, such a line becomes:

#1\end \ {#2}#3

in a context where the space token is not of category 10 (space).

Thus we replace the end-check for the `Verbatim` environment by a check on the detokenized-line (with ε -T_EX `\detokenize`):

```
1912 \def\tabu@fancyvrb {%
1913     \def\tabu@FV@DefineCheckEnd ##1{%
1914         \def\tabu@FV@DefineCheckEnd{%
1915             ##1% <original definition (if fancyvrb is loaded)>
1916             \let\FV@CheckEnd \tabu@FV@CheckEnd
1917             \let\FV@@@CheckEnd \tabu@FV@@@CheckEnd
1918             \let\FV@@@CheckEnd \tabu@FV@@@CheckEnd
1919             \edef\FV@EndScanning{%
1920                 \def\noexpand\next{\noexpand\end{\FV@EnvironName}}%
1921                 \global\let\noexpand\FV@EnvironName\relax
1922                 \noexpand\next}%
1923                 \xdef\FV@EnvironName{\detokenize\expandafter{\FV@EnvironName}}}%
1924             }\expandafter\tabu@FV@DefineCheckEnd\expandafter{\FV@DefineCheckEnd}
1925 }% \tabu@fancyvrb
1926 \def\tabu@FV@CheckEnd #1{\expandafter\FV@@@CheckEnd \detokenize{#1\end{}}\@nil}
1927 \edef\tabu@FV@@@CheckEnd {\detokenize{\end{}}}
1928 \begingroup
1929 \catcode`\[1 \catcode`\]2
1930 \@makeother\{ \makeother\}
1931 \edef\x[\endgroup
```

```
1932 \def\noexpand\tabu@FV@@CheckEnd ##1\detokenize[\end{##2}\detokenize[]]##3%
1933 ]\x \@nil{\def\@tempa{#2}\def\@tempb{#3}}
```

\tabu@FV@ListProcessLine This macro replaces **\FV@ListProcessLine** when measuring the natural width of a Verbatim environment (see **\tabu@startpboxmeasure**)

```
1934 \def\tabu@FV@ListProcessLine #1{%
1935 \hbox {%to \hsize{%
1936 \kern\leftmargin
1937 \hbox {%to \linewidth{%
1938 \FV@LeftListNumber
1939 \FV@LeftListFrame
1940 \FancyVerbFormatLine{#1}\hss
1941 %% DG/SR modification begin - Jan. 28, 1998 (for numbers=right add-on)
1942 %% \FV@RightListFrame}%
1943 \FV@RightListFrame
1944 \FV@RightListNumber}%
1945 %% DG/SR modification end
1946 \hss}}
```

11.19 \savetabu

\savetabu When this command is called by the user, the **tabu** preamble and target are globally stored into a macro **\tabu@saved@*(user-name)***.

```
1947 \newcommand*\savetabu[1]{\noalign{%
1948 \tabu@sanitizearg{#1}\tabu@temp
1949 \ifx \tabu@temp\@empty \tabu@savewarn(){The tabu will not be saved}\else
1950 \ifundefined\tabu@saved@\tabu@temp{}{\tabu@savewarn{#1}{Overwriting}}%
1951 \ifdefined\tabu@restored \expandafter\let
1952 \csname tabu@saved@\tabu@temp \endcsname \tabu@restored
1953 \else {\tabu@save}%
1954 \fi
1955 \fi}%
1956 }% \savetabu
1957 \def\tabu@save {%
1958 \toks0\expandafter{\tabu@saved@}%
1959 \iftabu@negcoef
1960 \let\tabu@wdddef \relax \let\tabu@ \tabu@savewd \edef\tabu@savewd{\tabu@Xcoefs}%
1961 \toks0\expandafter{\the\toks\expandafter0\tabu@savewd}\fi
1962 \toks1\expandafter{\tabu@savedpream}%
1963 \toks2\expandafter{\tabu@savedpreamble}%
1964 \let\@preamble \relax
1965 \let\tabu@savedpream \relax \let\tabu@savedparams \relax
1966 \edef\tabu@preamble{%
1967 \def\noexpand\tabu@aligndefault{\tabu@align}%
1968 \def\tabu@savedparams {\noexpand\the\toks0}%
1969 \def\tabu@savedpream {\noexpand\the\toks1}}%
1970 \edef\tabu@usetabu{%
1971 \def\@preamble {\noexpand\the\toks2}%
1972 \tabu@target \the\tabu@target \relax
1973 \tabu@colX \the\tabu@colX \relax
1974 \tabu@nbcols \the\tabu@nbcols \relax
1975 \def\noexpand\tabu@aligndefault{\tabu@align}%
1976 \def\tabu@savedparams {\noexpand\the\toks0}%
1977 \def\tabu@savedpream {\noexpand\the\toks1}}%
1978 \let\tabu@aligndefault \relax \let\@sharp \relax
1979 \edef\@tempa{\noexpand\tabu@s@ved
1980 \tabu@usetabu}
1981 \tabu@preamble}
```

```

1982             {\the\toks1}}\@tempa
1983   \tabu@message@save
1984 }% \tabu@save
1985 \long\def\tabu@s@ved #1#2#3{%
1986   \def\tabu@usetabu{#1}% <for \tabu@message@save>
1987   \expandafter\gdef\csname tabu@s@ved@\tabu@temp\endcsname ##1{%
1988     \ifodd ##1%       \usetabu
1989       \tabu@measuringfalse \tabu@spreadfalse % Just in case...
1990     \gdef\tabu@usetabu {%
1991       \ifdim \tabu@target>\z@ \tabu@warn@usetabu \fi
1992       \global\let\tabu@usetabu \@undefined
1993       \def\@halignto {to\tabu@target}%
1994       #1%
1995       \ifx \tabu@align\tabu@aligndefault@text
1996         \ifnum \tabu@nested=\z@
1997           \let\tabu@align \tabu@aligndefault \fi\fi}%
1998     \else % \preamble
1999       \gdef\tabu@preamble {%
2000         \global\let\tabu@preamble \@undefined
2001         #2%
2002         \ifx \tabu@align\tabu@aligndefault@text
2003           \ifnum \tabu@nested=\z@
2004             \let\tabu@align \tabu@aligndefault \fi\fi}%
2005     \fi
2006     #3}%
2007 }% \tabu@s@ved
2008 \def\tabu@aligndefault@text {\tabu@aligndefault}%
2009 \def\tabu@warn@usetabu {\PackageWarning{tabu}
2010   {Specifying a target with \string\usetabu\space is useless
2011   \MessageBreak The target cannot be changed!}}
2012 \def\tabu@s@vewd #1#2{\ifdim #2\p@<\z@ \tabu@wddef{#1}{\tabu@wd{#1}}\fi}

```

\tabu@s@vewarn Info for overwriting when **\savetabu** is used.

\tabu@saveerr Error if **\usetabu** is called with an unknown argument.

```

2013 \def\tabu@s@vewarn#1#2{\PackageInfo{tabu}
2014   {User-name '#1' already used for \string\savetabu
2015   \MessageBreak #2}}%
2016 \def\tabu@saveerr#1{\PackageError{tabu}
2017   {User-name '#1' is unknown for \string\usetabu
2018   \MessageBreak I cannot restore an unknown preamble!}\@ehd}

```

11.20 **\rowfont**

Setting font and alignment specification

\rowfont **\rowfont** uses the control sequences **\tabu@celllalign**, **\tabu@celleft**, **\tabu@cellright** and **\tabu@cellralign** which have been placed on purpose into the user-defined tokens inserted in any preamble by the array package.

\tabu@celllalign and **\tabu@cellralign** are used to modify the alignment. If the optional [alignment] parameter of **\rowfont** is not specified, then those control sequences expand to **\@empty**.

\tabu@celleft contains the font-modification information.

Placement of those control sequences into the user-tokens that are inserted in the preamble by the array package is explained below under the macro **\tabu@prepnxt@tok**.

```

2019 \newskip \tabu@cellskip
2020 \def\tabu@rowfont{\ifdim \baselineskip=\z@\noalign\fi
2021   {\ifnum0='}\fi \tabu@row@font}

```



```

2022 \newcommand*\tabu@row@font[2][\fi]{%
2023     \ifnum7=\currentgrouptype
2024         \global\let\tabu@@cellleft \tabu@cellleft
2025         \global\let\tabu@@cellright \tabu@cellright
2026         \global\let\tabu@@celllalign \tabu@celllalign
2027         \global\let\tabu@@cellralign \tabu@cellralign
2028         \global\let\tabu@@rowfontreset\tabu@rowfontreset
2029     \fi
2030     \global\let\tabu@rowfontreset \tabu@rowfont@reset
2031     \expandafter\gdef\expandafter\tabu@cellleft\expandafter{\tabu@cellleft #2}%
2032     \ifcsname tabu@cell@#1\endcsname % row alignment
2033         \csname tabu@cell@#1\endcsname \fi
2034     \ifnum0='\fi}% end of group / noalign group
2035 }% \rowfont
2036 \def\tabu@ifcolorleavevmode #1{\let\color \tabu@leavevmodecolor #1\let\color\tabu@color}%

```

\tabu@rowfont@reset This macro restores `\tabu@celllalign`, `\tabu@cellleft`, `\tabu@cellright`, and `\tabu@cellralign` to the value they had before the expansion of `\rowfont`.

It expands when a new row is inserted into the tabular or array.

```

2037 \def\tabu@rowfont@reset{%
2038     \global\let\tabu@rowfontreset \tabu@@rowfontreset
2039     \global\let\tabu@cellleft \tabu@@cellleft
2040     \global\let\tabu@cellright \tabu@@cellright
2041     \global\let\tabu@cellfont \@empty
2042     \global\let\tabu@celllalign \tabu@@celllalign
2043     \global\let\tabu@cellralign \tabu@@cellralign
2044 }% \tabu@@rowfontreset
2045 \let\tabu@rowfontreset \@empty % overwritten \AtBeginDocument if colortbl

```

Preparing stuff to be able to use `\rowfont`

\tabu@prepnex@tok `\tabu@prepnex@tok` will replace `\prepnex@tok` defined in `array.sty` (only inside a `tabu` environment). its purpose is to count the number of columns, and to insert the control sequences `\tabu@celllalign`, `\tabu@cellleft`, `\tabu@cellright` and `\tabu@cellralign` at the edge of each cell of the tabular. This is done by putting them inside the user-tokens placed around each column by the `array` package.

`\prepnex@tok` in `array.sty` initialises each user-token to an empty one, each time there is a need for a new one ! The macro has a very simple definition, but its expansion is the occasion to look carefully at the counters `\count@` and `\@tempcnta` which gives us all information required to decide if the token in preparation will be finally placed on the left or on the right of a column.

$$\underbrace{\langle \backslash \text{bfseries} \text{color} \{ \text{red} \} \rangle}_{\backslash \text{toks} \langle i \rangle} \text{ r } \underbrace{\langle \backslash \text{color} \{ \text{black} \} \backslash, \backslash \$ \rangle}_{\backslash \text{toks} \langle i + 1 \rangle}$$

When a column is inserted in the tabular preamble (`\@preamble`), the T_EX counter `\count@` is equal to $i + 1$ (*ie.* the right token) and the counter `\@tempcnta` is equal to i (*ie.* the left token). If the column is special (*ie.* `@` or `!`) `\@tempcnta` is not updated.

Thus, when a new token is “prepared” by `\prepnex@tok`:

either: `i = \count @ = \@ tempcnta` : the token to prepare (*ie.* `\toks < i + 1 >`) is the right one of a “normal” column. The switch `\iftabu@cellright` is set to `true`.

The *previous* token (`\toks < i > = \toks \count@`) is necessarily the left one of this “normal” column: we prepend `\tabu@celllalign` and append `\tabu@cellleft` to this token (`\toks < i >`). This token is finished and will not change afterwards.

or: `i = \count @ = \@ tempcnta + 1` : the token to prepare (`\toks < i + 1 >`) is either the left one of a normal column, or the single one of a special `@` or `!` column.

If the switch `\iftabu@cellright` is true, then the *previous* token `\toks < i >` is the right one of the last inserted column (which was a “normal” column, thus); `\tabu@cellright \tabu@cellralign` is appended to it, and the switch `\ittabu@cellright` is reset to `false`. May be `\prepnex@tok` will be

expanded again (by `\save@decl`): if it happens, then again `\count@ = \@tempcnta + 1` (same case) but `\iftabu@cellright` is false and nothing is changed.

else: The token to prepare (which is `\toks < i + 1 > = \toks \count@ + 1`), cannot be the right one of a “normal” column: `\iftabu@cellright` is set to false.

The fact that `|\count@ - \@tempcnta| > 1` tells us that the previous token `\toks < i >` is necessarily the single one of a “special” `@` or `!` column. We don’t modify this token, as long as *special columns are always inserted as is*: `\rowcolor` has no effect on special columns, nor `\rowfont`.

Thereafter, the original initialisation sequence occurs: `\advance \count@ by \@one` and initialize the token to prepare (`\toks \count@ = \toks < i + 1 >`) to an empty one.

```

2046 \newif \iftabu@cellright
2047 \def\tabu@prepnext@tok{%
2048     \ifnum \count@<\z@ % <first initialisation>
2049         \@tempcnta \@M % <not initialized by array.sty>
2050         \tabu@nbc@ \z@
2051         \let\@fornoopORI \@fornoop
2052         \tabu@cellrightfalse
2053     \else
2054         \ifcase \numexpr \count@-\@tempcnta \relax % (case 0): prev. token is left
2055             \advance \tabu@nbc@ \@ne
2056             \iftabu@cellright % before-previous token is right and is finished
2057                 \tabu@cellrightfalse % <only once>
2058                 \tabu@righttok
2059             \fi
2060             \tabu@lefttok
2061         \or % (case 1) previous token is right
2062             \tabu@cellrighttrue \let\@fornoop \tabu@lastnoop
2063         \else % special column: do not change the token
2064             \iftabu@cellright % before-previous token is right
2065                 \tabu@cellrightfalse
2066                 \tabu@righttok
2067             \fi
2068         \fi % \ifcase
2069     \fi
2070     \tabu@prepnext@tokORI
2071 }% \tabu@prepnext@tok
2072 \long\def\@tabu@lastnoop#1\@#2#3{\@tabu@lastn@p #2\@nextchar \in@\in@}
2073 \def\@tabu@lastn@p #1\@nextchar #2#3\in@{#1%
2074     \ifx \in@#2\else
2075         \let\@fornoop \tabu@fornoopORI
2076         \xdef\@tabu@mkpreamblebuffer{\@tabu@nbc@\the\@tabu@nbc@ \tabu@mkpreamblebuffer}%
2077         \toks0\expandafter{\expandafter\@tabu@everyrowtrue \the\toks0}%
2078         \expandafter\prepnext@tok
2079     \fi
2080 }% \@tabu@lastnoop
2081 \def\@tabu@righttok{%
2082     \advance \count@ \@ne
2083     \toks\count@\expandafter {\the\toks\count@ \tabu@cellright \tabu@cellralign}%
2084     \advance \count@ \@ne
2085 }% \@tabu@righttok
2086 \def\@tabu@lefttok{\toks\count@\expandafter{\expandafter\@tabu@celllalign
2087     \the\toks\count@ \tabu@cellleft}% after because of $
2088 }% \@tabu@lefttok

```

Neutralisation of glues and alignment modification

\tabu@celleft First initialisation to \@empty.

```

2089 \let\tabu@celleft \@empty
\tabu@celllalign 2090 \let\tabu@cellright \@empty
\tabu@cellright 2091 \tabu@celllalign@def{\tabu@celleft}%
\tabu@cellralign 2092 \let\tabu@cellralign \@empty

```

\tabu@cell@align

```

2093 \def\tabu@cell@align #1#2#3{%
2094     \let\tabu@maybesiunitx \toks@ \tabu@celllalign
2095     \global \expandafter \tabu@celllalign@def \expandafter {\the\toks@ #1}%
2096     \toks@\expandafter{\tabu@cellralign #2}%
2097     \xdef\tabu@cellralign{\the\toks@}%
2098     \toks@\expandafter{\tabu@celleft #3}%
2099     \xdef\tabu@celleft{\the\toks@}%
2100 }% \tabu@cell@align

```

\tabu@cell@l Setup macros to modify the alignment. The skips inserted to make the standard alignment specified in the tabular preamble are not the same with standard array tabulars and colortbl tabulars, hence the switch **\iftabu@colortbl**.

```

\tabu@cell@c 2101 \def\tabu@cell@l{% force alignment to left
\tabu@cell@r 2102     \tabu@cell@align
\tabu@cell@j 2103         {\tabu@removehfil \raggedright \tabu@celleft}% left
2104         {\tabu@flush1\tabu@ignorehfil}% right
2105         \raggedright
2106 }% \tabu@cell@l
2107 \def\tabu@cell@c{% force alignment to center
2108     \tabu@cell@align
2109         {\tabu@removehfil \centering \tabu@flush{.5}\tabu@celleft}
2110         {\tabu@flush{.5}\tabu@ignorehfil}
2111         \centering
2112 }% \tabu@cell@c
2113 \def\tabu@cell@r{% force alignment to right
2114     \tabu@cell@align
2115         {\tabu@removehfil \raggedleft \tabu@flush1\tabu@celleft}
2116         \tabu@ignorehfil
2117         \raggedleft
2118 }% \tabu@cell@r
2119 \def\tabu@cell@j{% force justification (for p, m, b columns)
2120     \tabu@cell@align
2121         {\tabu@justify\tabu@celleft}
2122         {}
2123         \tabu@justify
2124 }% \tabu@cell@j
2125 \def\tabu@justify{%
2126     \leftskip\vz@skip \@rightskip\leftskip \rightskip\@rightskip
2127     \parfillskip\@flushglue
2128 }% \tabu@justify
2129 %% ragged2e settings
2130 \def\tabu@cell@L{% force alignment to left (ragged2e)
2131     \tabu@cell@align
2132         {\tabu@removehfil \RaggedRight \tabu@celleft}
2133         {\tabu@flush 1\tabu@ignorehfil}
2134         \RaggedRight
2135 }% \tabu@cell@L
2136 \def\tabu@cell@C{% force alignment to center (ragged2e)

```

```

2137 \tabu@cell@align
2138     {\tabu@removehfil \Centering \tabu@flush{.5}\tabu@celleft}
2139     {\tabu@flush{.5}\tabu@ignorehfil}
2140     \Centering
2141 }% \tabu@cell@C
2142 \def\tabu@cell@R{% force alignment to right (ragged2e)
2143 \tabu@cell@align
2144     {\tabu@removehfil \RaggedLeft \tabu@flush 1\tabu@celleft}
2145     \tabu@ignorehfil
2146     \RaggedLeft
2147 }% \tabu@cell@R
2148 \def\tabu@cell@J{% force justification (ragged2e)
2149 \tabu@cell@align
2150     {\justifying \tabu@celleft}
2151     {}
2152     \justifying
2153 }% \tabu@cell@J
2154 \def\tabu@flush#1{%
2155 \iftabu@colortbl % colortbl uses \hfill rather than \hfil
2156     \hskip \ifnum13<\currentgrouptype \stretch{#1}%
2157     \else \ifdim#1pt<\p@ \tabu@cellskip
2158     \else \stretch{#1}
2159     \fi\fi \relax
2160 \else % array.sty
2161     \ifnum 13<\currentgrouptype
2162         \hfil \hskip1sp \relax \fi
2163 \fi
2164 }% \tabu@flush

```

\tabu@removehfil **\tabu@removehfil** removes (eventually) the infinite stretchable glue inserted *before* the cell (in the preamble of **\halign**) to make the column alignment.

```

2165 \let\tabu@hfil \hfil
2166 \let\tabu@hfill \hfill
2167 \let\tabu@hskip \hskip
2168 \def\tabu@removehfil{%
2169 \iftabu@colortbl
2170     \unkern \tabu@cellskip =\lastskip
2171     \ifnum\gluestretchorder\tabu@cellskip =\tw@ \hskip-\tabu@cellskip
2172     \else \tabu@cellskip \z@skip
2173     \fi
2174 \else
2175     \ifdim\lastskip=1sp\unskip\fi
2176     \ifnum\gluestretchorder\lastskip =\@ne
2177         \hfilneg % \hfilneg for array.sty but not for colortbl...
2178     \fi
2179 \fi
2180 }% \tabu@removehfil

```

\tabu@ignorehfil **\tabu@ignorehfil** removes (eventually) the infinite stretchable glue inserted *after* the cell (in the preamble of **\halign**) to make the column alignment.

```

2181 \def\tabu@ignorehfil{\aftergroup \tabu@nohfil}
2182 \def\tabu@nohfil{% \hfil -> do nothing + restore original \hfil
2183 \def\hfil{\let\hfil \tabu@hfil}% local to (alignment template) group
2184 }% \tabu@nohfil
2185 \def\tabu@colortblalignments {% if colortbl
2186 \def\tabu@nohfil{%
2187     \def\hfil {\let\hfil \tabu@hfil}% local to (alignment template) group

```

```

2188     \def\hfill {\let\hfill \tabu@hfill}% (colortbl uses \hfill) pfff...
2189     \def\hskip ###1\relax{\let\hskip \tabu@hskip}}% local
2190 }% \tabu@colortblalignments

```

11.21 Taking care of footnotes and `\arraybackslash`

Footnotes and hyperfootnotes

`\tabu@footnotetext` The macros in case `hyperref` is not used, or used with the option `hyperfootnotes=false`:

```

2191 \long\def\tabu@footnotetext #1{%
2192     \edef\@tempa{\the\tabu@footnotes
2193         \noexpand\footnotetext [\the\csname c@\@mpfn\endcsname]}%
2194     \global\tabu@footnotes\expandafter{\@tempa {#1}}}%
2195 \long\def\tabu@xfootnotetext [#1]#2{%
2196     \global\tabu@footnotes\expandafter{\the\tabu@footnotes
2197         \footnotetext [{#1}]{#2}}
2198 \let\tabu@xfootnote \@xfootnote

```

`\tabu@Hy@ftntext` The macros in case `hyperref` is loaded with the option `hyperfootnotes=true`:

```

\tabu@Hy@xfootnote 2199 \long\def\tabu@Hy@ftntext{\tabu@Hy@ftntxt {\the \c@footnote }}
2200 \long\def\tabu@Hy@xfootnote [#1] {%
2201     \begingroup
2202         \value\@mpfn #1\relax
2203         \protected@xdef \@thefnmark {\thempfn}%
2204     \endgroup
2205     \@footnotemark \tabu@Hy@ftntxt {#1}%
2206 }% \tabu@Hy@xfootnote
2207 \long\def\tabu@Hy@ftntxt #1#2{%
2208     \edef\@tempa{%
2209         \the\tabu@footnotes
2210         \begingroup
2211             \value\@mpfn #1\relax
2212             \noexpand\protected@xdef\noexpand\@thefnmark {\noexpand\thempfn}%
2213             \expandafter \noexpand \expandafter
2214                 \tabu@Hy@footnotetext \expandafter{\Hy@footnote@currentHref}%
2215         }%
2216     \global\tabu@footnotes\expandafter{\@tempa {#2}%
2217         \endgroup}%
2218 }% \tabu@Hy@ftntxt
2219 \long\def\tabu@Hy@footnotetext #1#2{%
2220     \H@@footnotetext{%
2221         \ifHy@nesting
2222             \hyper@@anchor {#1}{#2}%
2223         \else
2224             \Hy@raisedlink{%
2225                 \hyper@@anchor {#1}{\relax}%
2226             }%
2227             \def\@currentHref {#1}%
2228             \let\@currentlabelname \@empty
2229             #2%
2230         \fi
2231     }%
2232 }% \tabu@Hy@footnotetext

```

`\centering`, `\raggedright`, `\raggedleft` and `\@normalcr`

Inside `tabu` environment, no need to add `\arraybackslash` after such commands.

```

2233 \def\tabu@latextwoe {%
2234 \def\tabu@temp##1##2##3{{\toks@\expandafter{##2##3}\xdef##1{\the\toks@}}}
2235 \tabu@temp \tabu@centering \centering \arraybackslash
2236 \tabu@temp \tabu@raggedleft \raggedleft \arraybackslash
2237 \tabu@temp \tabu@raggedright \raggedright \arraybackslash
2238 }% \tabu@latextwoe
2239 \def\tabu@raggedtwoe {%
2240 \def\tabu@temp ##1##2##3{{\toks@\expandafter{##2##3}\xdef##1{\the\toks@}}}
2241 \tabu@temp \tabu@Centering \Centering \arraybackslash
2242 \tabu@temp \tabu@RaggedLeft \RaggedLeft \arraybackslash
2243 \tabu@temp \tabu@RaggedRight \RaggedRight \arraybackslash
2244 \tabu@temp \tabu@justifying \justifying \arraybackslash
2245 }% \tabu@raggedtwoe
2246 \def\tabu@normalcrbackslash{\let\\\@normalcr}
2247 \def\tabu@trivlist{\expandafter\def\expandafter\@trivlist\expandafter{%
2248 \expandafter\tabu@normalcrbackslash \@trivlist}}
```

Utilities: `tabu` `\fbox`

`\tabu@fbox` works exactly like L^AT_EX `\fbox` but allows the syntax: `\fbox \bgroup...\egroup` suitable for use inside tabular columns. `\fbox` is `\let` to `\tabu@fbox` at the entry inside a `tabu` environment.

```

2249 \def\tabu@fbox \leavevmode\afterassignment\tabu@beginfbox \setbox\@tempboxa\hbox}
2250 \def\tabu@beginfbox {\bgroup \kern\fboxsep
2251 \bgroup\aftergroup\tabu@endfbox}
2252 \def\tabu@endfbox {\kern\fboxsep\egroup\egroup
2253 \@frameb@x\relax}
```

`\tabu@fcolorbox` works exactly like `xcolor` `\fcolorbox` but allows the syntax:

`\fcolorbox {frame color}{background color}\bgroup...\egroup`

suitable for use insed tabular columns. `\fcolorbox` is `\let` to `\tabu@fcolorbox` at the entry inside a `tabu` environment.

```

2254 \def\tabu@color@b@x #1#2{\leavevmode \bgroup
2255 \def\tabu@docolor@b@x{#1{#2\color@block{\wd\z@}{\ht\z@}{\dp\z@}\box\z@}}%
2256 \afterassignment\tabu@begincolor@b@x \setbox\z@ \hbox
2257 }% \tabu@color@b@x
2258 \def\tabu@begincolor@b@x {\kern\fboxsep \bgroup
2259 \aftergroup\tabu@endcolor@b@x \set@color}
2260 \def\tabu@endcolor@b@x {\kern\fboxsep \egroup
2261 \dimen@ht\z@ \advance\dimen@ \fboxsep \ht\z@ \dimen@
2262 \dimen@dp\z@ \advance\dimen@ \fboxsep \dp\z@ \dimen@
2263 \tabu@docolor@b@x \egroup
2264 }% \tabu@endcolor@b@x
```

11.22 Corrections

delarray comptability fix for `colortbl` and `arydshln`

Both `colortbl` and `arydshln` forgot the control sequence `\@arrayright` which must be expanded by `\endarray`. Originally defined for `delarray`, this control sequence is used by `tabu` environments when `tabu X` columns are present in the preamble.

Here is the fix. We test if `\endarray` contains `\@arrayright` before modifying the control sequence, in case `colortbl` and/or `arydshln` modify their definition.

```

2265 \def\tabu@fix@arrayright {%% \@arrayright is missing from \endarray
```

```

2266 \iftabu@colortbl
2267 \ifdefined\adl@array % <colortbl + arydshln>
2268 \def\tabu@endarray{%
2269 \adl@endarray \egroup \adl@arrayrestore \CT@end \egroup %<original>
2270 \@arrayright % <FC>
2271 \gdef\@preamble{}}% <FC>
2272 \else % <colortbl / no arydshln>
2273 \def\tabu@endarray{%
2274 \crcr \egroup \egroup %<original>
2275 \@arrayright % <FC>
2276 \gdef\@preamble{\CT@end}%
2277 \fi
2278 \else
2279 \ifdefined\adl@array % <arydshln / no colortbl>
2280 \def\tabu@endarray{%
2281 \adl@endarray \egroup \adl@arrayrestore \egroup %<original>
2282 \@arrayright % <FC>
2283 \gdef\@preamble{}}% <FC>
2284 \else % <no arydshln / no colortbl + \@arrayright missing>
2285 \PackageWarning{tabu}
2286 {\string\@arrayright\space is missing from the
2287 \MessageBreak definition of \string\endarray.
2288 \MessageBreak Comptability with delarray.sty is broken.}%
2289 \fi\fi
2290 }% \tabu@fix@arrayright

```

arydshln @ columns

```

2291 \def\tabu@adl@xarraydashrule #1#2#3{%
2292 \ifnum\@lastchclass=\adl@class@start\else
2293 \ifnum\@lastchclass=\@ne\else
2294 \ifnum\@lastchclass=5 \else % <FC> @-arg (class 5) and !-arg (class 1)
2295 \adl@leftrulefalse \fi\fi % must be treated the same
2296 \fi
2297 \ifadl@zwvrule\else \ifadl@inactive\else
2298 \@addtopreamble{\vrule\@width\arrayrulewidth
2299 \@height\z@ \@depth\z@}\fi \fi
2300 \ifadl@leftrule
2301 \@addtopreamble{\adl@vlineL{\CT@arc@}{\adl@dashgapcolor}%
2302 {\number#1}#3}%
2303 \else \@addtopreamble{\adl@vlineR{\CT@arc@}{\adl@dashgapcolor}%
2304 {\number#2}#3}
2305 \fi
2306 }% \tabu@adl@xarraydashrule

```

arydshln, colors without colortbl and empty p columns

arydshln redefines \@endpbox for p columns. The definition is stored in \adl@act@endpbox. Here it is:

```

\unskip \ifhmode \nobreak
\vrule\@width\z@\@height\z@\@depth\dp\@arstrutbox
\fi
\egroup \adl@colhtdp \box\adl@box \hfil

```

The \vrule inserted is exactly what package array calls: \@finalstrut \@arstrutbox.

However, just like in array.sty, this array-strut should be inserted incondionnally, and \ifhmode applies only to \nobreak (misplaced \fi in arydshln definition).

Finally, arydshln is not compatible with colors in columns, such that: >{\color {red}}p3in, Unless colortbl is also loaded, the color group is missing.

Fixed inside `tabu` environment.

```

2307 \def\tabu@adl@act@endpbox {%
2308     \unskip \ifhmode \nobreak \fi     \@finalstrut \@arstrutbox
2309     \egroup \egroup
2310     \adl@colhtdp \box\adl@box \hfil
2311 }% \tabu@adl@act@endpbox
2312 \def\tabu@adl@fix {%
2313     \let\adl@xarraydashrule \tabu@adl@xarraydashrule % <fix> arydshln
2314     \let\adl@act@endpbox \tabu@adl@act@endpbox % <fix> arydshln
2315     \let\adl@act@@endpbox \tabu@adl@act@endpbox % <fix> arydshln
2316     \let\@preamerror \@preamerr % <fix> arydshln
2317 }% \tabu@adl@fix

```

longtable \@startpbox: \everypar needed

`\tabu@LT@startpbox` The leading strut should be inserted at `\everypar` in order for `\tabulinesep` to work (otherwise, T_EX is in horizontal mode and `\nointerlineskip` breaks).

```

2318 \def\tabu@LT@startpbox #1{%
2319     \bgroup
2320     \let\@footnotetext\LT@p@ftntext
2321     \setlength\hsize{#1}%
2322     \@arrayparboxrestore
2323     \everypar{%
2324         \vrule \@height \ht\@arstrutbox \@width \z@
2325         \everypar{}}%
2326 }% \tabu@LT@startpbox

```

11.23 Package options and Initialisation

`\tracingtabu` and the package options

`delarray` (package option) The `delarray` package option is only there for convenience: it simply loads the `delarray` package.

```

2327 \DeclareOption{delarray}{\AtEndOfPackage{\RequirePackage{delarray}}}

```

`linegoal` (package option) The `linegoal` package option only sets `\tabudefaultright` to be equal to `\linegoal`. The required package `linegoal` is loaded.

```

2328 \DeclareOption{linegoal}{%
2329     \AtEndOfPackage{%
2330         \RequirePackage{linegoal}[2010/12/07]%
2331         \let\tabudefaultright \linegoal% \linegoal is \linewidth if not pdfTeX
2332     }}

```

`\scantokens` (package option) The `scantokens` package option makes `tabu` equal to `tabu*`.

```

2333 \DeclareOption{scantokens}{\tabuscantokenstrue}

```

`\tracingtabu` `\tracingtabu` is the same as the package option `debugshow`.

`debugshow` (package option)

```

2334 \DeclareOption{debugshow}{\AtEndOfPackage{\tracingtabu=\tw@}}
2335 \def\tracingtabu {\begingroup\@ifnextchar=%
2336     {\afterassignment\tabu@tracing\count@}
2337     {\afterassignment\tabu@tracing\count@1\relax}}
2338 \def\tabu@tracing{\expandafter\endgroup
2339     \expandafter\tabu@tr@cing \the\count@ \relax
2340 }% \tabu@tracing
2341 \def\tabu@tr@cing #1\relax {%
2342     \ifnum#1>\thr@@ \let\tabu@tracinglines\message
2343     \else \let\tabu@tracinglines\@gobble
2344     \fi
2345     \ifnum#1>\tw@ \let\tabu@DBG \tabu@@DBG

```



```

2346         \def\tabu@mkarstrut {\tabu@DBG@arstrut}%
2347         \tabustrurule      1.5\p@
2348     \else         \let\tabu@DBG          \@gobble
2349         \def\tabu@mkarstrut {\tabu@arstrut}%
2350         \tabustrurule      \z@
2351     \fi
2352     \ifnum#1>\@ne   \let\tabu@debug      \message
2353     \else         \let\tabu@debug      \@gobble
2354     \fi
2355     \ifnum#1>\z@
2356         \let\tabu@message          \message
2357         \let\tabu@tracing@save    \tabu@message@save
2358         \let\tabu@starttimer     \tabu@pdftimer
2359     \else
2360         \let\tabu@message          \@gobble
2361         \let\tabu@tracing@save    \@gobble
2362         \let\tabu@starttimer     \relax
2363     \fi
2364 }% \tabu@tr@cing

```

Initialisation and setup \AtBeginDocument

At the end of the tabu package:

- `\tracingtabu` is set to 0: this initialises the message commands. Eventually,, t he value will be overwritten by the `debugshow` package option later.
- `\everyrow` is set to empty: this initialises the process at `\everycr` to the default process,
- a new *empty* line style is defined, to be equivalent to `\hline`: this creates the *default leaders*, which will be used if a line style specification cannot be parsed successfully.
Then this default line style is set to be the current one.

At Begin Document,a fix for `arydshln` and `colortbl` comptability with `delarray` shortcuts available inside `tabu`: requirement for this fix is checked by `\tabu@fix@arrayright`.

Then the switch `\iftabu@colortbl` is set.

Finally, the `longtabu` environment is defined only if the `longtable` package is detected.

```

2365 \AtBeginDocument{\tabu@AtBeginDocument}
2366 \def\tabu@AtBeginDocument{\let\tabu@AtBeginDocument \@undefined
2367     \ifdefined\arrayrulecolor   \tabu@colortbltrue      % <colortbl>
2368                                 \tabu@colortblalignments % different glues are used
2369     \else                         \tabu@colortblfalse \fi
2370     \ifdefined\CT@arc@ \else \let\CT@arc@ \relax \fi
2371     \ifdefined\CT@drsc@\else \let\CT@drsc@ \relax \fi
2372     \let\tabu@arc@L \CT@arc@ \let\tabu@drsc@L \CT@drsc@
2373     \ifodd 1\ifcsname siunitx_table_collect_begin:Nn\endcsname % <siunitx: ok>
2374         \expandafter\ifx
2375             \csname siunitx_table_collect_begin:Nn\endcsname\relax 0\fi\fi\relax
2376         \tabu@siunitxtrue
2377     \else \let\tabu@maybesiunitx \@firstofone % <not siunitx: setup>
2378         \let\tabu@siunitx \tabu@nosiunitx
2379         \tabu@siunitxfalse
2380     \fi
2381     \ifdefined\adl@array % <arydshln>
2382     \else \let\tabu@adl@fix \relax
2383         \let\tabu@adl@endtrial \@empty \fi
2384     \ifdefined\longtable % <longtable>
2385     \else \let\longtabu \tabu@nolongtabu \fi
2386     \ifdefined\cellspacetoplimit \tabu@warn@cellspace\fi

```

```

2387 \csname\ifcsname ifHy@hyperfootnotes\endcsname % <hyperfootnotes>
2388     ifHy@hyperfootnotes\else iffalse\fi\endcsname
2389     \let\tabu@footnotetext \tabu@Hy@ftntext
2390     \let\tabu@xfootnote \tabu@Hy@xfootnote \fi
2391 \ifdefined\FV@DefineCheckEnd% <fancyvrb>
2392     \tabu@fancyvrb \fi
2393 \ifdefined\color % <color / xcolor>
2394     \let\tabu@color \color
2395     \def\tabu@leavevmodecolor ##1{%
2396         \def\tabu@leavevmodecolor {\leavevmode ##1}%
2397     }\expandafter\tabu@leavevmodecolor\expandafter{\color}%
2398 \else
2399     \let\tabu@color \tabu@nocolor
2400     \let\tabu@leavevmodecolor \@firstofone \fi
2401 \tabu@latexttwoe
2402 \ifdefined\@raggedtwoe@everyselectfont % <ragged2e>
2403     \tabu@raggedtwoe
2404 \else
2405     \let\tabu@cell@L \tabu@cell@l
2406     \let\tabu@cell@R \tabu@cell@r
2407     \let\tabu@cell@C \tabu@cell@c
2408     \let\tabu@cell@J \tabu@cell@j \fi
2409 \expandafter\in@ \expandafter\@arrayright \expandafter{\endarray}%
2410 \ifin@ \let\tabu@endarray \endarray
2411 \else \tabu@fix@arrayright \fi% <fix for colortbl & arydshln (delarray)>
2412 \everyrow{}%
2413}% \tabu@AtBeginDocument
2414\def\tabu@warn@cellspace{%
2415    \PackageWarning{tabu}{%
2416        Package cellspace has some limitations
2417    \MessageBreak And redefines some macros of array.sty.
2418    \MessageBreak Please use \string\tabulinesep\space to control
2419    \MessageBreak vertical spacing of lines inside tabu environnement}%
2420}% \tabu@warn@cellspace

```

\ProcessOption * is much quicker than without the star...

```

2421 \tabuscantokensfalse
2422 \let\tabu@arc@G \relax
2423 \let\tabu@drsc@G \relax
2424 \let\tabu@evr@G \@empty
2425 \let\tabu@rc@G \@empty
2426 \def\tabu@ls@G {\tabu@linestyle@}%
2427 \let\tabu@@rowfontreset \@empty % <init>
2428 \let\tabu@@celllalign \@empty
2429 \let\tabu@@cellralign \@empty
2430 \let\tabu@@cellleft \@empty
2431 \let\tabu@@cellright \@empty
2432 \def\tabu@naturalXmin {\z@}
2433 \def\tabu@naturalXmax {\z@}
2434 \let\tabu@rowfontreset \@empty
2435 \def\tabulineon {4pt}\let\tabulineoff \tabulineon
2436 \tabu@everyrowtrue
2437 \ifdefined\pdfelapsedtime % <pdfTeX>
2438     \def\tabu@pdftimer {\xdef\tabu@starttime{\the\pdfelapsedtime}}%
2439 \else \let\tabu@pdftimer \relax \let\tabu@message@etime \relax
2440 \fi
2441 \tracingtabu=\z@
2442 \newtabulinesyle {=\maxdimen}% creates the 'factory' settings \tabu@linestyle@

```

```
2443 \tabulinestyle{}
2444 \taburowcolors{}
2445 \let\tabudefaulttarget \linewidth
2446 \ProcessOptions*           % \ProcessOptions* is quicker !

2447 </package>
```

12 References

- [1] *A new implementation of L^AT_EX's tabular and array environments* by Frank Mittelbach
2008/09/09 v2.4c – Tabular extension package (FMi)
CTAN:help/Catalogue/entries/array.html
- [2] *The varwidth package* by Donald Arseneau
2009/03/30 ver 0.92 – Variable-width minipages
CTAN:help/Catalogue/entries/varwidth.html
- [3] *The enumitem-zref package* by **FC**
2011/02/18 ver 1.8 – Extended references for enumitem pkg
CTAN:help/Catalogue/entries/enumitem-zref.html

13 History

[2011/02/26 v2.8]

- Bug in the starred version (with `\scantokens`) of the `longtabu*` environment.

[2011/02/25 v2.7]

- Automatic `\par` after the end of the `tabu` environment used with its default target is removed in case of `tabu spread`: this was a bug.
- Some `\ignorespaces` were missing (in `\everyrow`, `\taburulecolor`, `\taburowcolors` and `\tabulinestyle`).

[2011/02/24 v2.6]

- `\savetabu` now also saves `\tabulinesep` (*ie.* `\abovetabulinesep` and `\belowtabulinesep`)
- Bug fixed for `custom-environments` when nested.
- `\taburulecolor` works even if `colortbl` is not loaded for the `tabu` environment. This is now the same for the `longtabu` environment.

[2011/02/19 v2.5]

- Bug fixed for `\pdfelapsedtime` when compilation without pdfL^AT_EX.
- Modification of `\@finalstrut` (“null-rule” added) to avoid problems with `\columncolor`.

[2011/02/17 v2.4]

- Documentation revisited

[2011/02/13 v2.3]

- Fixed two bugs for nested `tabu` environment: when using `\rowfont` and when `tabu` is nested inside `longtabu`

[2011/02/12 v2.2 – New implementation - Absolutely no modification of array.sty]

- $\tau_{\text{nb}}\subset$ has been totally reimplemented, including the algorithms. In particular, outside of the `tabu` environment, absolutely none of the macros of `array.sty`, (and obviously none of L^AT_EX) is modified.

The process has been completely reinvented: `tabu` follows a path along different modes (or strategies) measuring natural width of cells, fixing X column widths, measuring vertical length of rows and then printing the final tabular. The process is optimized, especially in the case of nested `tabu` environments: a tabular is not built twice for measuring purpose... As a result, many new features are now possible... vertical leaders (dashed lines), dynamic vertical spacing adjustment, and hopefully still more in a next release.

`tabu` now systematically collects the environment body. But with `\scantokens`, it is possible to insert verbatim material inside the columns: use `tabu*` instead of `tabu`, for the outer most tabular.

- New: `\firstline` and `\lastline` can draw multiple lines, and there is an option to set `\extratabsurround` instantly, and locally.
- New: `\taburulecolor` with a good behaviour with groupings (like `\everyrow`)
- Modification: `\tabulinestyle` sets the line style for the tabu, `\newtabulinestyle` defines a new line style.

This	is	the	new	$\mathcal{T}_{\mathcal{N}bc}$	package
------	----	-----	-----	-------------------------------	---------

[2011/01/19 v2.1]

- `Vertical spacing` had a bug with `longtabu` and paragraph columns. Fixed.
- New: `\everyrow`.
- Fix a bug of `\rowfont` when using `siunitx` S columns.
- Some code optimisation.
- To do (if possible): a syntax `X[6mc]S[...]` to “embed” `siunitx` S column inside `tabu` and `longtabu` X columns...

[2011/01/18 v2.0]

- `Vertical spacing` of lines implemented ! See `\tabulinesep` and `\extrarowsep`.
- `\tabulinestyle` : user defined line style can now be used inside the optional argument of the `|[...]` preamble token.
- `|[...]` is now allowed in `\multicolumn` preamble inside `tabu` environment.
- Bug fixed inside `\tabu@prepnexxtok` (again !!! - a difficult case !)
- Incompatibility of package `cellspace` with `tabu spread` and `tabu` with `negativ coefficients` for X columns with has been lifted.
However, as said in the documentation of package `cellspace`, S column modifier does not work in the case of nested tabulars.
The S column modifier becomes C when the package `siunitx` is loaded (see `siunitx` documentation).
Moreover, `cellspace` does not work with `color` or `xcolor` and paragraph column types !!
Finally, `cellspace` redefines **globally** `\@startpbox` and `\@endpbox` and is therefore not fully compatible with `array.sty` and therefore with $\mathcal{T}_{\mathcal{N}bc}$.
For all those reasons, $\mathcal{T}_{\mathcal{N}bc}$ displays a warning to discourage the use of `cellspace` with the `tabu` environment.

[2011/01/15 v1.9]

- Bug in `\savetabu` when used inside `longtabu`...
- Bug when `tabu` with X column is nested inside `lontabu`.
- Documentation (`\rowfont` was missing in the `summary`).

[2010/12/28 v1.8]

- `\tracingtabu` / `debugshow` package option: reporting of the time elapsed during trials (if `\pdfelapsedtime` and thus `pdfTEX` is available) Slight modifications for better reporting on the `.log` file.
- Fix a bug when `\savetabu` is used after `\multicolumn` (`\multicolumn` globally redefines `\@preamble`).
- Fix a bug with `\tabucline` and `\CT@arc@` (`colortbl`).
- Better privacy of columns types specifically defined for `tabu`.
- Improvement in the rewriting process (but only very few people should notice...)
- Documentation.

[2010/12/18 v1.7]

- Code optimisation
- Modification in the columns rewriting process (bug with some new column types defined by the user).

[2010/12/07 v1.5]

- Implementation of negativ width coefficients for X columns (cf. [tabu X columns – Mastering horizontal space point 2](#)).
- Columns natural widths computation (for `tabu spread` with X columns and negativ coefficients) is based on the code of the `varwidth` package by Donald Arseneau.
- `longtabu` is now provided, based on the `longtable` package by David Carlisle. `longtabu` can be used just like `tabu`.
- Vertical lines can be used whatever the catcode of `|` is.
- `\savetabu` reports saved informations in the `.log` (`debugshow` option).
- `\savetabu... \usetabu` now restores the `\halign` preamble rather than the `tabu` preamble! `\preamble` can be use in the `tabu` preamble to restore a `tabu` preamble.
- `\tabucline` is more robust with “special” preambles containing `>` or `<` tokens. `\tabucline` now takes care of `\arrayrulecolor` (package `colortbl`).
- `enumitem-zref` package has been added to the documentation (see the link [point 1](#))
- Optimisation of some parts of the code.

[2010/11/22 v1.4]

- Compatibility improvement with `linegoal` for the syntax: `\begin {tabu} to\linegoal {...}`
- Hyper footnotes now work correctly.
- Fix a bug when using colored vertical lines in `tabu` in math mode.
- Fix a bug with vertical lines and `colortbl \arrayrulecolor` specification.
- Fix a compatibility bug with `arydshln`: when nesting a tabular that use vertical dashed lines (`arydshln`) inside `tabu spread` with X columns.

[2010/11/18 v1.3]

- Fix a bug that may appear in `\tabucline` depending on the preamble due to arbitrary `\countdef`.
- Improvement in the use of `\everycr`: no `\global` stuff. Thus bug fixed when nesting `tabu` inside `\mathcal{align}` environment for example. Same issue with `\rowfont` which now works without global modification of `\everycr`.
- No phantom line is added to `tabu` but a command `\tabuphantomline` is provided for this purpose (required with `\multicolumn` in some cases).
- Improvement on vertical alignment.
- To do: an example file to test a wide range of possibilities...
- Documentation.

[2010/11/15 v1.2]

- Improvement in parameters parsing for optional parameters (`|` and `\tabucline`).
- Modification / optimization in `\tabu@prepnex@tok`.
- Modification of `\tabucline` to get better results with `m` columns (`X[m]`) and also when `\minrowclearance > 0` (package `colortbl`).

[2010/10/28 v1.1]

- First version.

14 Index

Numbers written in *italic* refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; plain numbers refer to the code lines where the entry is used.

Symbols	
<code>\&</code>	80
<code>\@arrayparboxrestore</code>	2322
<code>\@arrayright</code>	1034, 1041, 1048, 2265, 2270, 2275, 2282, 2284, 2286, 2409
<code>\@arraystretch</code>	693
<code>\@arstrutbox</code> 109, 117, 118, 123, 624, 691, 692, 821, 822, 1003, 1284, 1555, 1638, 1654, 1726, 1730, 1733, 1739, 1742, 1744, 2308, 2324	
<code>\@defaultunits</code> 330, 331, 858, 1063, 1794, 1838, 1839	
<code>\@endpbox</code>	615, 638
<code>\@finalstrut</code>	1036, 1274, 1555, 1631, 2308
<code>\@flushglue</code>	1073, 1544, 2127
<code>\@footnotemark</code>	2205
<code>\@footnotetext</code>	627, 1127, 2320
<code>\@gobbletwo</code>	847, 1399, 1525, 1833
<code>\@halignto</code>	873, 882, 1041, 1131, 1267, 1280, 1281, 1993
<code>\@ifnextchar</code>	166, 168, 177, 197, 199, 208, 237, 285, 854, 2335
<code>\@mkpream</code>	834
<code>\@normalcr</code>	626, 2246
<code>\@onelevel@sanitize</code>	97
<code>\@preamble</code>	1077, 1081, 1085, 1092, 1502, 1964, 1971, 2271, 2276, 2283
<code>\@raggedtwoe@everyselectfont</code>	2402
<code>\@rightskip</code>	2126
<code>\@sharp</code>	1978
<code>\@sptoken</code>	64, 563, 911, 1065, 1857, 1891
<code>\@tabarray</code>	616, 639
<code>\@tempb</code>	1933
<code>\@tempcnta</code>	25, 1123, 1776, 2049, 2054
<code>\@tempcntb</code>	26, 1777
<code>\@tempdimb</code>	36, 116, 118, 1343, 1349, 1360, 1363, 1370, 1400, 1403, 1463, 1467, 1472, 1475, 1615, 1618, 1619, 1627, 1630, 1632, 1720, 1732, 1739, 1742, 1744
<code>\@tempdimc</code> 34, 37, 1385, 1387, 1391, 1393, 1396, 1439, 1440, 1625, 1719, 1739, 1745	
<code>\@temptokena</code> 746, 751, 767, 857, 861, 1017, 1020, 1064, 1849, 1865, 1869, 1900, 1903	
<code>\@thefnmark</code>	2203, 2212
<code>\@trivlist</code>	2247, 2248
<code>\@whilesw</code>	1357
<code>\@xfootnote</code>	628, 1127, 2198
<code>\@xfootnotetext</code>	627
<code>\@xhline</code>	623
<code>\{</code>	1930
<code>\}</code>	1930
A	
<code>\abovetabulinesep</code> .	<u>39</u> , 200, 202, 204, 209, 211, 213, 216, 217, 741, 1515, 1614, 1626
<code>\adl@endarray</code>	2269, 2281
<code>\afterassignment</code>	142, 173, 176, 204, 207, 238, 240, 241, 483, 575, 947, 1243, 2249, 2256, 2336, 2337
<code>\aftergroup</code>	146, 163, 164, 194, 195, 657, 660, 834, 996, 997, 1243, 1546, 1830, 2181, 2251, 2259
<code>\array</code>	540
<code>\arraybackslash</code>	1544, 2235, 2236, 2237, 2241, 2242, 2243, 2244
<code>\arraycolsep</code>	734
<code>\arrayrulecolor</code>	2367
<code>\arrayrulewidth</code>	429, 736, 1510, 1682, 1706, 1707, 1765, 1769, 1772, 2298
<code>\arraystretch</code>	110, 111, 115, 116, 126, 131, 743, 1517, 1649
B	
<code>\baselineskip</code>	2020
<code>\begin</code>	8, 9
<code>\belowtabulinesep</code>	<u>39</u> , 200, 202, 209, 211, 213, 216, 217, 742, 1516, 1615, 1627
C	
<code>\c@footnote</code>	2199
<code>\c@page</code>	1422, 1424
<code>\c@taburow</code>	21, 254, 349, 351, 352, 369, 373, 685, 721, 758, 1636, 1637, 1639, 1641, 1642, 1643, 1644, 1646, 1650
<code>\c@taburow@save</code>	685, 721
<code>\catcode</code> 10, 11, 80, 92, 446, 447, 448, 449, 1929	
<code>\Centering</code>	630, 924, 2138, 2140, 2241
<code>\centering</code> ...	628, 920, 924, 2109, 2111, 2235
<code>\cl@ckcpt</code>	1139
<code>\cleaders</code>	1802
<code>\cline</code>	1702, 1707, 1712
<code>\color</code>	90, 122, 299, 303, 312, 320, 507, 624, 1128, 1501, 2036, 2393, 2394, 2397
<code>\color@</code>	488
<code>\color@b@x</code>	633

<code>\color@begingroup</code>	1553	<code>\FV@LeftListNumber</code>	1938
<code>\color@block</code>	2255	<code>\FV@ListProcessLine</code>	1543
<code>\colorlet</code>	354, 366, 371	G	
<code>\CT@do@color</code>	705	<code>\GenericError</code>	150, 1125
<code>\CT@everycr</code>	641	<code>\globaldefs</code>	696, 723, 1139, 1653, 1668
<code>\CT@LT@sep</code>	1755, 1758, 1764, 1770	<code>\gluestretchorder</code>	2171, 2176
<code>\currentgrouplevel</code>	225, 228, 229, 231	H	
<code>\currentgroupstype</code>	1600, 1605, 1606, 2023, 2156, 2161	<code>\H@footnotetext</code>	2220
D			
<code>\d@llarbegin</code>	1603	<code>\hbadness</code>	1123
<code>\d@llarend</code>	136, 1613	<code>\hfilneg</code>	2177
<code>\debugshow_␣(package_␣option)</code>	2334	<code>\hfuzz</code>	620, 1124
<code>\definecolorseries</code>	350, 364	<code>\hline</code>	623, 1677, 1679, 1680, 1686, 1701, 1702, 1706, 1711, 1725, 1752
<code>\delarray_␣(package_␣option)</code>	2327	<code>\hrule</code>	439, 516, 1632, 1682, 1689, 1760, 1769, 1772, 1822
<code>\detokenize</code>	1501, 1521, 1522, 1523, 1923, 1926, 1927, 1932	<code>\ht</code> 110, 115, 117, 126, 517, 691, 821, 822, 1614, 1626, 1638, 1647, 1726, 2255, 2261, 2324	
<code>\dimen@</code>	1142, 1143, 1144, 1149, 1315, 1318, 1322, 1327, 1365, 1367, 1373, 1383, 1384, 1409, 1429, 1695, 1721, 1726, 1739, 1742, 1744, 2261, 2262	I	
<code>\displaystyle</code>	954	<code>\ifin@</code>	434, 2410
<code>\doublerulesep</code>	737, 1511, 1691, 1721, 1756, 1760, 1824	<code>\iftabu@cellright</code>	2046, 2056, 2064
<code>\dp</code>	111, 116, 118, 131, 692, 821, 822, 1615, 1627, 1638, 1648, 1730, 1733, 1739, 1742, 1744, 2255, 2262	<code>\iftabu@colortbl</code> 49, 249, 625, 641, 2155, 2169, 2266	
E			
<code>\ecolumn</code>	1552	<code>\iftabu@everyrow</code>	49, 103, 105, 163, 194, 255, 273, 297, 310, 339, 347, 641, 1245
<code>\empty</code>	1763	<code>\iftabu@long</code> ..	49, 655, 996, 1162, 1215, 1254
<code>\endarray</code>	540, 640, 1163, 1223, 1254, 1289, 2265, 2287, 2409, 2410	<code>\iftabu@measuring</code>	49, 583, 871, 1037, 1043, 1100, 1303, 1357, 1386
<code>\endlinechar</code>	100, 402, 1134	<code>\iftabu@negcoef</code>	49, 1205, 1959
<code>\endlongtable</code>	550	<code>\iftabu@siunitx</code>	49, 1006
<code>\endlongtabu</code>	550, 553	<code>\iftabu@spread</code> 49, 594, 605, 874, 1200, 1298, 1534	
<code>\endtabu</code>	539, 1593	<code>\iftabuscantokens</code>	58, 849, 1134, 1161, 1219, 1228, 1247
<code>\endtabular</code>	541	<code>\iftrue</code>	55
<code>\errmessage</code>	1313	<code>\ignorespaces</code> ..	256, 275, 317, 362, 1856, 1890
<code>\everycr</code>	614, 634, 1240, 1241, 1244	<code>\immediate</code>	144, 146
<code>\everypar</code> 554, 634, 646, 647, 1273, 1599, 2323, 2325		<code>\in@</code>	335, 337, 338, 2072, 2074, 2409
<code>\everyrow</code>	17, 234, 386, 729, 764, 2412	<code>\in@false</code>	427
<code>\extracolsep</code>	1073	<code>\in@true</code>	426
<code>\extrarowdepth</code> 39, 111, 116, 131, 169, 171, 173, 178, 180, 182, 185, 186, 692, 740, 1514, 1648		<code>\indent</code>	602
<code>\extrarowheight</code> ..	110, 115, 126, 169, 171, 178, 180, 182, 185, 186, 691, 739, 1513, 1647	J	
<code>\extrarowsep</code>	13, 162, 385	<code>\justifying</code>	632, 926, 2150, 2152, 2244
<code>\extratabsurround</code>	385, 738, 1512, 1699, 1700, 1729, 1745	L	
F			
<code>\firstline</code>	15, 621, 1701, 1727, 1736	<code>\lastbox</code>	648
<code>\firstline</code>	622	<code>\lasthline</code>	621, 1702, 1740, 1749
<code>\footnotetext</code>	2193, 2197	<code>\lastline</code>	622
<code>\futurelet</code>	62, 156, 560, 576, 895, 952, 1683, 1703, 1723, 1790, 1853	<code>\lastskip</code>	2170, 2175, 2176
<code>\FV@@@CheckEnd</code>	1918	<code>\lccode</code>	446
<code>\FV@@CheckEnd</code>	1917, 1926	<code>\leaders</code>	439, 1760, 1769, 1772, 1801
<code>\FV@CheckEnd</code>	1916	<code>\leftmargin</code>	1936
<code>\FV@DefineCheckEnd</code>	1908, 1924, 2391	<code>\leftskip</code>	2126
<code>\FV@EndScanning</code>	1919	<code>\linegoal</code>	1418, 2331
<code>\FV@EnvironName</code>	1920, 1921, 1923	<code>\linegoal_␣(package_␣option)</code>	2328
<code>\FV@LeftListFrame</code>	1939	<code>\linewidth</code>	610, 644, 1126, 1415, 1416, 1417, 1937, 2331, 2445
		<code>\longtable</code>	550, 2384
		<code>\longtabu</code>	76, 548, 557, 2385
		<code>\lowercase</code>	447, 1480
		<code>\LT@@hline</code>	625
		<code>\LT@bchunk</code>	1089, 1094
		<code>\LT@cols</code>	1759, 1768, 1771
		<code>\LT@echunk</code>	1234
		<code>\LT@get@widths</code>	1235

<code>\LT@next</code>	1752, 1753, 1763, 1774	<code>\setbox</code>	109, 123, 527, 531, 648, 820, 1003, 1216, 1225, 1234, 1284, 1541, 1601, 1608, 1654, 2249, 2256
<code>\LT@p@ftntext</code>	2320	<code>\setlength</code>	877, 2321
<code>\LT@startpbox</code>	554	<code>\span</code>	1784, 1785
<code>\LTchunksize</code>	551	<code>\stretch</code>	2156, 2158
M			
<code>\message</code>	2342, 2352, 2356	<code>\strutbox</code>	110, 111, 115, 116, 126, 131, 691, 692, 1647, 1648
<code>\minrowclearance</code>	744, 1518	T	
<code>\multicolumn</code>	72, 636, 1198	<code>\tabcolsep</code>	735
<code>\multispan</code> ...	844, 847, 1198, 1759, 1768, 1771	<code>\tabu</code>	75, <u>539</u> , 1583
N			
<code>\narrowragged</code>	1544	<code>\tabu@C@cellalign</code> 677, 711, 1655, 1669, 2026, 2042, 2428
<code>\NC@find</code>	590, 797, 840, 853, 860, 861, 865, 1017, 1020, 1029, 1033, 1068, 1523	<code>\tabu@C@cellalign@save</code>	677, 711
<code>\NC@rewrite@S</code>	862	<code>\tabu@C@celleft</code> 679, 713, 1657, 1671, 2024, 2039, 2430
<code>\newcount</code>	21, 22, 23, 24, 27, 28	<code>\tabu@C@celleft@save</code>	679, 713
<code>\newdimen</code> ..	30, 31, 32, 33, 38, 39, 40, 41, 42	<code>\tabu@C@cellralign</code> 678, 712, 1656, 1670, 2027, 2043, 2429
<code>\newsavebox</code>	45, 46, 47, 48	<code>\tabu@C@cellralign@save</code>	678, 712
<code>\newskip</code>	2019	<code>\tabu@C@cellright</code> 680, 714, 1658, 1672, 2025, 2040, 2431
<code>\newtabulinestyle</code>	16, <u>263</u> , 2442	<code>\tabu@C@cellright@save</code>	680, 714
<code>\newtoks</code>	43, 44	<code>\tabu@C@DBG</code>	122, 2345
<code>\noalign</code>	103, 189, 220, 247, 363, 367, 1238, 1245, 1652, 1667, 1681, 1695, 1727, 1732, 1740, 1742, 1756, 1765, 1773, 1778, 1820, 1947, 2020	<code>\tabu@C@everycr</code>	634, 1244
<code>\noindent</code>	1619, 1665, 1666	<code>\tabu@C@everypar</code>	634
<code>\numexpr</code> 79, 162, 193, 1456, 1457, 1458, 1832, 2054		<code>\tabu@C@rowfontreset</code> 682, 716, 2028, 2038, 2044, 2427
O			
<code>\obeyspaces</code>	69, 411, 1134	<code>\tabu@C@rowfontreset@save</code>	682, 716
<code>\omit</code>	1728, 1743, 1783, 1784, 1785	<code>\tabu@C@spixiii</code>	71, 1170
P			
<code>\PackageError</code>	380, 549, 558, 867, 2016	<code>\tabu@C@tabudecimal</code>	1871
<code>\PackageInfo</code>	2013	<code>\tabu@C@thevline</code>	1659, 1673
<code>\PackageWarning</code> 151, 496, 935, 942, 2009, 2285, 2415	<code>\tabu@C@adl@fix</code>	640, 2312, 2317, 2382
<code>\parfillskip</code>	1544, 2127	<code>\tabu@C@afterendpar</code>	606, 726, 731, 1051
<code>\pdfadjustspacing</code>	1545	<code>\tabu@C@aftergroupcleanup</code>	656, 657, 659, 661, 1078, 1083, 1090, 1094, 1129
<code>\pdfelapsedtime</code>	1454, 2437, 2438	<code>\tabu@C@align</code>	590, 591, 1967, 1975, 1995, 1997, 2002, 2004
<code>\penalty</code>	1756, 1765, 1773	<code>\tabu@C@aligndefault</code>	587, 608, 610, 1508, 1967, 1975, 1978, 1997, 2004, 2008
<code>\preamble</code>	20, 988, 994, 1012, 1998	<code>\tabu@C@aligndefault@text</code> ...	1995, 2002, 2008
<code>\preamble_(private_column_type)</code>	988	<code>\tabu@C@alloc</code>	21, 684, 722, 763
<code>\prepnext@tok</code>	635, 2078	<code>\tabu@C@alloc@</code>	29, 603
<code>\prevdepth</code>	1625	<code>\tabu@C@alloc@save</code>	684, 763
<code>\protected@xdef</code>	2203, 2212	<code>\tabu@C@arc@G</code> 322, 663, 664, 697, 698, 752, 2422	
R			
<code>\RaggedLeft</code>	631, 925, 2144, 2146, 2242	<code>\tabu@C@arc@Gsave</code>	663, 697
<code>\raggedleft</code> ...	629, 921, 925, 2115, 2117, 2236	<code>\tabu@C@arc@L</code> 315, 664, 747, 752, 759, 1519, 2372	
<code>\RaggedRight</code>	631, 923, 2132, 2134, 2243	<code>\tabu@C@arith</code>	1299, <u>1332</u> , 1405
<code>\raggedright</code> .	629, 919, 923, 2103, 2105, 2237	<code>\tabu@C@arith@negcoef</code>	1315, 1317, 1331
<code>\rem@pt</code>	1443, 1447, 1450	<code>\tabu@C@arithnegcoef</code> ...	<u>1314</u> , 1342, 1359, 1402
<code>\remove@to@nnil</code>	90	<code>\tabu@C@arstrut</code>	<u>109</u> , 2349
<code>\resetcolorseries</code>	353, 365, 370	<code>\tabu@C@arstrutbox</code>	45, 624
<code>\rightskip</code>	2126	<code>\tabu@C@AtBeginDocument</code> ..	2365, 2366, 2413
<code>\romannumeral</code>	1482, 1499, 1795	<code>\tabu@C@begin</code>	560
<code>\rowcolor</code>	355, 367, 372, 1128	<code>\tabu@C@begin@color@b@x</code>	2256, 2258
<code>\rowfont</code>	18, 632, <u>2019</u>	<code>\tabu@C@begin@box</code>	2249, 2250
S			
<code>\save@decl</code>	137	<code>\tabu@C@box</code> ...	45, 1216, 1223, 1225, 1231, 1343, 1344, 1385, 1403, 1427, 1436, 1541, 1556, 1558, 1559, 1563, 1564, 1570, 1601, 1608, 1613, 1614, 1615, 1626, 1627, 1633
<code>\savetabu</code> ...	19, 746, 1126, 1527, <u>1947</u> , 2014	<code>\tabu@C@bufferX</code>	1262, 1268, 1269, 1296, 1306, 1310, 1378, 1388
<code>\scantokens</code> 59, 100, 412, 416, 1086, 1095, 1233		<code>\tabu@C@C@extra</code>	162, 182, 186, 190, 191
<code>\scantokens_(package_option)</code>	<u>2333</u>	<code>\tabu@C@C@l@r</code>	83, 87, 90, 489, 490

<code>\tabu@C@linesep</code>	193, 213, 217, 221, 222	<code>\tabu@docline@evr</code>	250, 1829
<code>\tabu@c@loff</code>	404, 420, 444, 489	<code>\tabu@doclineafter</code>	1829, 1830
<code>\tabu@c@lon</code>	404, 418, 419, 444, 490	<code>\tabu@docolor@b@x</code>	2255, 2263
<code>\tabu@cell@align</code>	2093, 2102, 2108, 2114, 2120, 2131, 2137, 2143, 2149	<code>\tabu@dp</code>	1463, 1466, 1472, 1474, 1618, 1630, 1644, 1648
<code>\tabu@cell@C</code>	2136, 2141, 2407	<code>\tabu@dpdef</code>	1618, 1630, 1638, 1643
<code>\tabu@cell@c</code>	2101, 2407	<code>\tabu@drsc@G</code>	323, 665, 666, 699, 700, 753, 2423
<code>\tabu@cell@J</code>	2148, 2153, 2408	<code>\tabu@drsc@Gsave</code>	665, 699
<code>\tabu@cell@j</code>	2101, 2408	<code>\tabu@drsc@L</code>	316, 666, 748, 753, 760, 1520, 2372
<code>\tabu@cell@L</code>	2130, 2135, 2405	<code>\tabu@elapsedtime</code>	726, 1082, 1093, 1102, 1197
<code>\tabu@cell@l</code>	2101, 2405	<code>\tabu@end@envir</code>	1160, 1590, 1591, 1594
<code>\tabu@cell@R</code>	2142, 2147, 2406	<code>\tabu@endarray</code>	640, 2268, 2273, 2280, 2410
<code>\tabu@cell@r</code>	2101, 2406	<code>\tabu@endcolor@b@x</code>	2259, 2260, 2264
<code>\tabu@cellfont</code>	2041	<code>\tabu@endenvir</code>	1162, 1163, 1166, 1258, 1261, 1266, 1289
<code>\tabu@celllalign</code>	161, 673, 707, 1655, 1660, 1669, 2026, 2042, 2086, 2089, 2094	<code>\tabu@endfbox</code>	2251, 2252
<code>\tabu@celllalign@def</code>	161, 1106, 1111, 1115, 1192, 1273, 1277, 2091, 2095	<code>\tabu@endofcollect</code>	1156
<code>\tabu@celllalign@save</code>	673, 707	<code>\tabu@endoftrials</code>	1183, 1187, 1246
<code>\tabu@cellleft</code>	158, 675, 709, 1657, 1663, 1671, 1889, 2024, 2031, 2039, 2087, 2089, 2098, 2099, 2103, 2109, 2115, 2121, 2132, 2138, 2144, 2150	<code>\tabu@endpbox</code>	615, 638
<code>\tabu@cellleft@save</code>	675, 709	<code>\tabu@endpboxmeasure</code>	1546, 1554
<code>\tabu@cellralign</code>	674, 708, 1107, 1111, 1115, 1193, 1273, 1278, 1656, 1661, 1670, 2027, 2043, 2083, 2089, 2096, 2097	<code>\tabu@endquick</code>	1587, 1589
<code>\tabu@cellralign@save</code>	674, 708	<code>\tabu@endrewrite</code>	1028, 1032
<code>\tabu@cellright</code>	158, 676, 710, 1658, 1662, 1672, 2025, 2040, 2083, 2089	<code>\tabu@endrewritemulticolumn</code>	834, 842
<code>\tabu@cellright@save</code>	676, 710	<code>\tabu@endtabu</code>	1590, 1597
<code>\tabu@cellrightfalse</code>	2052, 2057, 2065	<code>\tabu@endtabus</code>	1591, 1598
<code>\tabu@cellrighttrue</code>	2062	<code>\tabu@endtrial</code>	1213, 1291
<code>\tabu@cellskip</code>	2019, 2157, 2170, 2171, 2172	<code>\tabu@err</code>	1292, 1293, 1294, 1313
<code>\tabu@cellspacepatch</code>	1547, 1552	<code>\tabu@everycr</code>	1240, 1245
<code>\tabu@Centering</code>	630, 2241	<code>\tabu@everyr@w</code>	241, 244, 260
<code>\tabu@centering</code>	628, 2235	<code>\tabu@everyrow</code>	245, 1245
<code>\tabu@clarstrut</code>	690, 728	<code>\tabu@everyrow@bgroup</code>	103, 234, 272, 283, 327
<code>\tabu@clckpt</code>	1139, 1218, 1227, 1280	<code>\tabu@everyrow@egroup</code>	103, 259, 281, 325, 377
<code>\tabu@cleanup</code>	654	<code>\tabu@everyrowfalse</code>	56, 246, 1666
<code>\tabu@ccline</code>	1778, 1779, 1791	<code>\tabu@everyrowtrue</code>	55, 696, 758, 845, 2077, 2436
<code>\tabu@cclinearg</code>	1781, 1792, 1807	<code>\tabu@evr</code>	261, 262, 1106, 1111, 1115, 1192, 1203, 1272, 1276
<code>\tabu@cclinebox</code>	1796, 1797, 1798, 1799, 1800, 1808	<code>\tabu@evr@G</code>	257, 671, 672, 706, 729, 756, 2424
<code>\tabu@cclineleads</code>	1801, 1802, 1803, 1808, 1809	<code>\tabu@evr@Gsave</code>	671, 706
<code>\tabu@cclinescan</code>	1702, 1707	<code>\tabu@evr@L</code>	256, 672, 749, 756, 764, 1521
<code>\tabu@closetrialsgroup</code>	1246	<code>\tabu@evrh@C@k</code>	252, 261
<code>\tabu@cnt</code>	21, 947, 949, 1001, 1023, 1028, 1122, 1265, 1271, 1340, 1341, 1358, 1378, 1459, 1460, 1489	<code>\tabu@evrstartstop</code>	235, 237, 238, 240, 243
<code>\tabu@collectbody</code>	598, 1135, 1156, 1587	<code>\tabu@extr@</code>	176, 177, 184
<code>\tabu@color</code>	507, 624, 2036, 2394, 2399	<code>\tabu@extra</code>	166, 168, 175
<code>\tabu@color@b@x</code>	633, 2254, 2257	<code>\tabu@extrac@lsep</code>	1076
<code>\tabu@colortblalignments</code>	2185, 2190, 2368	<code>\tabu@extracolsep</code>	1056, 1063
<code>\tabu@colortblfalse</code>	2369	<code>\tabu@fancyvrb</code>	1912, 1925, 2392
<code>\tabu@colortbltrue</code>	2367	<code>\tabu@fbox</code>	633, 2249
<code>\tabu@commaXIII</code>	411, 448	<code>\tabu@finalstrut</code>	139, 1036, 1274
<code>\tabu@DBG</code>	125, 129, 1728, 1743, 2345, 2348	<code>\tabu@firstline</code>	621, 1676
<code>\tabu@DBG@arstrut</code>	122, 2346	<code>\tabu@firstlinecorrection</code>	1676, 1677, 1725
<code>\tabu@debug</code>	1560, 1616, 1628, 2352, 2353	<code>\tabu@firstline</code>	622, 1676
<code>\tabu@decimal</code>	1849, 1903	<code>\tabu@firstspace</code>	87, 88
<code>\tabu@definestyle</code>	399, 402	<code>\tabu@fix@arrayright</code>	2265, 2290, 2411
<code>\tabu@DELTA</code>	30, 1332, 1334, 1337, 1344, 1346, 1347, 1349, 1355, 1360, 1366, 1367, 1369, 1373, 1430, 1431	<code>\tabu@flush</code>	2104, 2109, 2110, 2115, 2133, 2138, 2139, 2144, 2154, 2164
<code>\tabu@docline</code>	250, 1826, 1828	<code>\tabu@footnotetext</code>	2191
		<code>\tabu@footnotes</code>	43, 620, 725, 2192, 2194, 2196, 2209, 2216
		<code>\tabu@footnotetext</code>	627, 2191, 2389
		<code>\tabu@FV@CheckEnd</code>	1916, 1926
		<code>\tabu@FV@DefineCheckEnd</code>	1912
		<code>\tabu@FV@ListProcessLine</code>	1934
		<code>\tabu@G@extra</code>	182, 186, 188, 189, 190, 192
		<code>\tabu@G@linesep</code>	213, 217, 219, 220, 221, 223
		<code>\tabu@GenericError</code>	149, 1125

<code>\tabu@get@decimal</code>	1865, 1893	<code>\tabu@leavevmodecolor</code>	2036, 2395, 2396, 2397, 2400
<code>\tabu@get@decimalspace</code>	1868, 1895	<code>\tabu@lefttok</code>	2060, 2086, 2088
<code>\tabu@getc@l@r</code>	483, 485, 495	<code>\tabu@linegoalt@rget</code>	576, 577, 581
<code>\tabu@getcolor</code>	417, <u>481</u>	<code>\tabu@linegoaltarget</code>	575, 576
<code>\tabu@getdecimal</code>	1859, <u>1872</u>	<code>\tabu@lines</code>	<u>798</u> , 835, 1004
<code>\tabu@getdecimal@</code>	1850, 1853, 1859	<code>\tabu@lines@</code>	801
<code>\tabu@getdecimal@ignorespaces</code>	1850, 1855, 1864	<code>\tabu@linesep</code>	197, 199, 206
<code>\tabu@getline</code>	266, 272, <u>388</u> , 804, 1804	<code>\tabu@linestyle@</code>	268, 2426, 2442
<code>\tabu@getparam</code>	406, 407, 463, 476, <u>480</u>	<code>\tabu@longfalse</code>	539
<code>\tabu@gettargget</code>	573, 574, 575	<code>\tabu@longpream</code>	996, <u>1077</u>
<code>\tabu@Gextra</code>	163, 188	<code>\tabu@longtrial</code>	<u>1212</u>
<code>\tabu@givespace</code>	1332, 1339, 1348	<code>\tabu@longtrue</code>	548
<code>\tabu@Glinesep</code>	194, 219	<code>\tabu@ls@</code>	276, 279, 702, 761, 803, 813, 1793
<code>\tabu@gobblespace</code>	60, 568, 1066, 1861	<code>\tabu@ls@G</code>	278, 279, 667, 668, 701, 702, 754, 2426
<code>\tabu@gobbletoken</code>	60, 166, 169, 171, 178, 180, 197, 200, 202, 209, 211	<code>\tabu@ls@Gsave</code>	667, 701
<code>\tabu@gobbleX</code>	60, 1162, 1164	<code>\tabu@ls@L</code>	275, 276, 668, 749, 754, 761, 1522
<code>\tabu@gobblex</code>	62, 63, 67	<code>\tabu@LT@@chline</code>	625, <u>1751</u>
<code>\tabu@Grestore</code>	190, 221, <u>224</u>	<code>\tabu@LT@startpbox</code>	554, <u>2318</u>
<code>\tabu@Gsave</code>	182, 186, 213, 217, <u>224</u>	<code>\tabu@ltx@verb</code>	1910, 1911
<code>\tabu@halignto</code>	573, 574, 584, 585, 873, 1048, 1267, 1280	<code>\tabu@maybesiunitx</code>	155, 2094, 2377
<code>\tabu@header</code>	1361, 1425, 1451, 1489	<code>\tabu@measuring</code>	1194, 1209, <u>1212</u>
<code>\tabu@hfil</code>	2165, 2183, 2187	<code>\tabu@measuringfalse</code>	619, 1002, 1353, 1371, 1989
<code>\tabu@hfill</code>	2166, 2188	<code>\tabu@measuringtrue</code>	872, 1356, 1537
<code>\tabu@hfuzz</code>	620, 1144, 1346, 1366, 1369, 1430, 1453	<code>\tabu@message</code>	878, 879, 1341, 1345, 1361, 1377, 1390, 1392, 1397, 1455, 1526, 2356, 2360
<code>\tabu@highprioritycolumns</code>	792, 794, 1008	<code>\tabu@message@arith</code>	1345, <u>1425</u>
<code>\tabu@hleads</code>	45, 511	<code>\tabu@message@defaulttarget</code>	878, <u>1413</u>
<code>\tabu@hline</code>	623, 1680, 1684	<code>\tabu@message@endpboxmeasure</code>	1560, 1575, 1580
<code>\tabu@hlineAZ</code>	1676, 1677, 1678, 1679, <u>1695</u>	<code>\tabu@message@etime</code>	1197, <u>1454</u> , 2439
<code>\tabu@hlineAZsurround</code>	1697, 1699, 1705	<code>\tabu@message@negcoef</code>	<u>1442</u>
<code>\tabu@hlinecorrection</code>	1696, 1714	<code>\tabu@message@reached</code>	1377, <u>1451</u>
<code>\tabu@hlinescan</code>	1701, 1706	<code>\tabu@message@save</code>	733, <u>1478</u> , 1983, 1986, 2357
<code>\tabu@hsize</code>	968, <u>972</u> , 999, 1336, 1532, 1533, 1538	<code>\tabu@message@spreadarith</code>	1390, <u>1434</u>
<code>\tabu@hskip</code>	2167, 2189	<code>\tabu@message@target</code>	879, 1423
<code>\tabu@ht</code>	1462, 1464, 1469, 1617, 1629, 1642, 1647	<code>\tabu@message@verticalsp</code>	<u>1461</u> , 1616, 1628
<code>\tabu@htdef</code>	1617, 1629, 1638, 1641	<code>\tabu@mkarstrut</code>	119, 689, 720, 2346, 2349
<code>\tabu@Hy@footnotetext</code>	2214, 2219, 2232	<code>\tabu@mkarstrut@save</code>	689, 720
<code>\tabu@Hy@ftntext</code>	<u>2199</u> , 2389	<code>\tabu@mkpreamblebuffer</code>	815, 842, 1002, 1038, 1044, 1060, 1078, 1090, 1120, 1130, 2076
<code>\tabu@Hy@ftntxt</code>	2199, 2205, 2207, 2218	<code>\tabu@modulo</code>	<u>60</u> , 352, 369
<code>\tabu@Hy@xfootnote</code>	<u>2199</u> , 2390	<code>\tabu@msg@align</code>	1481, 1482, 1484, 1498
<code>\tabu@if@envir</code>	73, 75	<code>\tabu@msgalign</code>	1362, 1363, 1364, 1367, 1426, 1427, 1428, 1431, 1435, 1436, 1437, 1438, 1439, 1465, 1467, 1470, 1475, 1481
<code>\tabu@ifcolorleavevmode</code>	2036	<code>\tabu@multicolumn</code>	636, 843
<code>\tabu@ifenvir</code>	<u>60</u> , 843	<code>\tabu@multispan</code>	1665, 1784, 1785, 1831, 1832, 1835
<code>\tabu@ignorehfil</code>	2104, 2110, 2116, 2133, 2139, 2145, <u>2181</u>	<code>\tabu@n@Gextra</code>	164, 189
<code>\tabu@immediate</code>	144, 147	<code>\tabu@n@Glinesep</code>	195, 220
<code>\tabu@indent</code>	<u>603</u>	<code>\tabu@naturalX</code>	<u>30</u> , 686, 717, 1203, 1563, 1565, 1566
<code>\tabu@init</code>	<u>603</u>	<code>\tabu@naturalX@save</code>	686, 717
<code>\tabu@justify</code>	922, 926, 2121, 2123, 2125, 2128	<code>\tabu@naturalXmax</code>	688, 718, 1201, 1202, 1383, 1385, 1437, 1565, 1566, 2433
<code>\tabu@justifying</code>	632, 2244	<code>\tabu@naturalXmax@save</code>	688, 718
<code>\tabu@keeps</code>	808, 810, 817	<code>\tabu@naturalXmin</code>	687, 719, 1202, 1384, 1385, 1438, 1567, 1568, 2432
<code>\tabu@lasthline</code>	621, <u>1676</u>	<code>\tabu@naturalXmin@save</code>	687, 719
<code>\tabu@lasthlinecorrection</code>	1678, 1679, <u>1738</u>	<code>\tabu@nbcols</code>	<u>21</u> , 1238, 1507, 1665, 1836, 1840, 1841, 1842, 1974, 2050, 2055, 2076
<code>\tabu@lastline</code>	622, <u>1676</u>	<code>\tabu@NC@list</code>	613, 838, 1009, 1010, 1024, 1079, 1091
<code>\tabu@lastn@@p</code>	2072, 2073	<code>\tabu@negcoeffalse</code>	544, 555
<code>\tabu@lastnoop</code>	2062, 2072, 2080	<code>\tabu@negcoeftrue</code>	961, 1150
<code>\tabu@lastspace</code>	88, 89		
<code>\tabu@lasttry</code>	1132		
<code>\tabu@l@textwoe</code>	2233, 2238, 2401		
<code>\tabu@LEADERS</code>	501, <u>506</u>		
<code>\tabu@leaders</code>	499, 500, 503, 819, 1810, 1813		
<code>\tabu@leaders@G</code>	500, 522		
<code>\tabu@leadersstyle</code>	434, <u>499</u>		
<code>\tabu@leads</code>	511, 518, 527, 531, 532, 535		

<code>\tabu@nested</code>	21 , 350 , 353 , 354 , 355 , 364 , 365 , 366 , 367 , 370 , 371 , 372 , 604 , 722 , 724 , 1039 , 1040 , 1050 , 1099 , 1185 , 1186 , 1270 , 1283 , 1302 , 1414 , 1422 , 1445 , 1490 , 1573 , 1574 , 1641 , 1642 , 1643 , 1644 , 1996 , 2003	<code>\tabu@rewritemulticolumn</code>	833
<code>\tabu@nestedmeasure</code>	583 , 593 , 601	<code>\tabu@rewritevline</code>	795 , 796 , 799
<code>\tabu@new@columntype</code>	769 , 774 , 777 , 788	<code>\tabu@rewriteX</code>	852 , 870 , 984
<code>\tabu@newcolumnntype</code>	768 , 833 , 995 , 1019 , 1022	<code>\tabu@rewriteX@Ss</code>	855 , 857 , 866
<code>\tabu@newlinestyle</code>	264 , 266 , 271	<code>\tabu@rewriteXrestore</code>	882 , 984
<code>\tabu@nextlineAZ</code>	1695	<code>\tabu@rewritten</code>	797 , 806 , 807 , 853 , 865 , 883 , 888 , 890 , 942 , 944 , 960 , 966 , 968 , 970
<code>\tabu@nocolor</code>	153 , 507 , 1128 , 2399	<code>\tabu@Rextra</code>	188 , 189 , 190
<code>\tabu@noeverycr</code>	1241 , 1243	<code>\tabu@righttok</code>	2058 , 2066 , 2081 , 2085
<code>\tabu@nohfil</code>	2181 , 2182 , 2184 , 2186	<code>\tabu@Rlinesep</code>	219 , 220 , 221
<code>\tabu@nolongtabu</code>	558 , 2385	<code>\tabu@row@font</code>	2021 , 2022
<code>\tabu@normalcrbackslash</code>	2246 , 2248	<code>\tabu@rowc@lors</code>	328 , 329
<code>\tabu@norowcolor</code>	153 , 1128	<code>\tabu@rowcolors</code>	327 , 328 , 336
<code>\tabu@nosiunitx</code>	853 , 856 , 2378	<code>\tabu@rowcolorseries</code>	335 , 337 , 378
<code>\tabu@nowrite</code>	142 , 1125	<code>\tabu@rowcolorserieserror</code>	343 , 380 , 383
<code>\tabu@noxfootnote</code>	152	<code>\tabu@rowfont</code>	632 , 2020
<code>\tabu@noxfootnotes</code>	142	<code>\tabu@rowfont@reset</code>	2030 , 2037
<code>\tabu@off</code>	30 , 403 , 421 , 423 , 424 , 425 , 436 , 489	<code>\tabu@rowfontreset</code>	248 , 681 , 683 , 715 , 2028 , 2030 , 2038 , 2045 , 2434
<code>\tabu@offiii</code>	446	<code>\tabu@rowfontreset@save</code>	681 , 715
<code>\tabu@offxiii</code>	468 , 471 , 477	<code>\tabu@rule@arc@</code>	308 , 309 , 326
<code>\tabu@ofxiii</code>	446	<code>\tabu@rule@drsc@</code>	290 , 291 , 295 , 296 , 307
<code>\tabu@on</code>	30 , 403 , 421 , 422 , 423 , 425 , 435	<code>\tabu@rulearc</code>	287 , 306 , 308
<code>\tabu@onxiii</code>	446	<code>\tabu@rulecolor</code>	283 , 284 , 288
<code>\tabu@oXIII</code>	411 , 447	<code>\tabu@ruledrsc</code>	285 , 289
<code>\tabu@oxiii</code>	447 , 450	<code>\tabu@ruledrsc@</code>	294 , 295
<code>\tabu@pdftimer</code>	2358 , 2438 , 2439	<code>\tabu@rulesstyle</code>	437 , 499 , 1501
<code>\tabu@pream</code>	997 , 1077	<code>\tabu@sanitizearg</code>	93 , 267 , 344 , 345 , 392 , 1948
<code>\tabu@preamble</code>	1966 , 1981 , 1999 , 2000	<code>\tabu@save</code>	1953 , 1957 , 1984
<code>\tabu@prepnext@tok</code>	635 , 2046	<code>\tabu@save@decl</code>	134
<code>\tabu@prepnext@tokORI</code>	635 , 2070	<code>\tabu@savecounters</code>	1116 , 1137 , 1140 , 1217 , 1218 , 1226 , 1227
<code>\tabu@prev</code>	1017 , 1023 , 1026	<code>\tabu@samed@</code>	751 , 1958
<code>\tabu@printdecimal</code>	1896 , 1899 , 1905	<code>\tabu@samedecl</code>	136 , 138 , 836 , 1015
<code>\tabu@private@columntype</code>	779 , 786 , 789	<code>\tabu@samedparams</code>	733 , 746 , 757 , 1504 , 1965 , 1968 , 1976
<code>\tabu@privatecolumns</code>	787 , 790 , 837 , 1016	<code>\tabu@samedpream</code>	590 , 1503 , 1962 , 1965 , 1969 , 1977
<code>\tabu@privatecolumnntype</code>	778 , 795 , 799 , 852 , 980 , 988	<code>\tabu@samedpreamble</code>	1081 , 1092 , 1963
<code>\tabu@pt</code>	1446 , 1448 , 1479 , 1480	<code>\tabu@saveerr</code>	981 , 983 , 989 , 991 , 2013
<code>\tabu@pushbegins</code>	1157 , 1177	<code>\tabu@savels</code>	816 , 828 , 839 , 1000
<code>\tabu@quick</code>	1583 , 1586 , 1588 , 1596	<code>\tabu@savewarn</code>	1949 , 1950 , 2013
<code>\tabu@quickend</code>	1189 , 1287	<code>\tabu@savewd</code>	1960 , 1961 , 2012
<code>\tabu@quickrule</code>	598 , 602 , 1288	<code>\tabu@scandecimal</code>	1851 , 1853 , 1854 , 1861 , 1866 , 1869
<code>\tabu@RaggedLeft</code>	631 , 2242	<code>\tabu@select</code>	1087 , 1096 , 1098
<code>\tabu@raggedleft</code>	629 , 2236	<code>\tabu@setcleanup</code>	617 , 654
<code>\tabu@RaggedRight</code>	631 , 2243	<code>\tabu@seteverycr</code>	1103 , 1108 , 1112 , 1216 , 1225 , 1239 , 1282
<code>\tabu@raggedright</code>	629 , 2237	<code>\tabu@setextra</code>	169 , 171 , 176 , 178 , 180
<code>\tabu@raggedtwoe</code>	2239 , 2245 , 2403	<code>\tabu@setextrasep</code>	173 , 185 , 187
<code>\tabu@rc@</code>	249 , 339 , 340 , 348 , 356 , 361 , 362 , 368 , 373 , 374 , 379 , 704 , 762	<code>\tabu@setlinesep</code>	204 , 216 , 218
<code>\tabu@rc@G</code>	340 , 374 , 669 , 670 , 703 , 704 , 755 , 2425	<code>\tabu@setreset</code>	617 , 732
<code>\tabu@rc@Gsave</code>	669 , 703	<code>\tabu@sets@p</code>	207 , 208 , 215
<code>\tabu@rc@L</code>	339 , 362 , 670 , 755 , 762	<code>\tabu@sets@ve</code>	766 , 767
<code>\tabu@rearstrut</code>	109 , 253 , 693 , 1636 , 1667	<code>\tabu@setsave</code>	747 , 748 , 749 , 766
<code>\tabu@removehfil</code>	2103 , 2109 , 2115 , 2132 , 2138 , 2144 , 2165	<code>\tabu@setsep</code>	200 , 202 , 207 , 209 , 211
<code>\tabu@rescan</code>	58 , 849 , 1219 , 1228 , 1251	<code>\tabu@setstrategy</code>	1117 , 1121
<code>\tabu@reset</code>	684 , 732 , 830 , 1239 , 1636	<code>\tabu@settarget</code>	544 , 555 , 560
<code>\tabu@resetls</code>	831 , 832	<code>\tabu@setup</code>	589 , 603
<code>\tabu@restored</code>	612 , 1951 , 1952	<code>\tabu@shorttrial</code>	1215 , 1216 , 1224
<code>\tabu@restoreeverycr</code>	1243 , 1244	<code>\tabu@siunitx</code>	852 , 854 , 2378
<code>\tabu@rewritefirst</code>	591 , 613 , 995	<code>\tabu@siunitxerror</code>	863 , 867 , 869
<code>\tabu@rewritelast</code>	1014 , 1019	<code>\tabu@siunitxfalse</code>	2379
<code>\tabu@rewritemiddle</code>	1013 , 1017 , 1019	<code>\tabu@siunitxtrue</code>	2376

<code>\tabu@skipdecimal</code>	1854, 1860, 1862, 1894	<code>\tabu@usetabu</code>	1054, 1080, 1528, 1970, 1980, 1986, 1990, 1992
<code>\tabu@spread</code>	570, 574	<code>\tabu@verb</code>	1907, <u>1910</u>
<code>\tabu@spread@arith</code>	1383, 1408, 1412	<code>\tabu@verbatim</code>	59, <u>1906</u>
<code>\tabu@spreadarith</code>	1298, <u>1382</u>	<code>\tabu@verticaldynamicadjustment</code>	1272, <u>1645</u>
<code>\tabu@spreadfalse</code>	544, 555, 1989	<code>\tabu@verticalinit</code>	1106, 1192, 1276, <u>1635</u>
<code>\tabu@spreadheader</code>	1434, 1493	<code>\tabu@verticalmeasure</code>	1106, 1192, 1277, <u>1599</u>
<code>\tabu@spreadtarget</code>	<u>30</u> , 874, 875, 1046, 1387, 1391, 1396, 1435, 1440	<code>\tabu@verticalsp@lcr</code>	1602, <u>1612</u>
<code>\tabu@spreadtrue</code>	574, 595	<code>\tabu@verticalsp@pmb</code>	1609, <u>1621</u>
<code>\tabu@spxiii</code>	70, 461, 474	<code>\tabu@verticalspacing</code>	1107, 1193, 1278, 1602, 1609
<code>\tabu@start</code>	<u>21</u> , 235, 240, 331, 333, 334, 357, 1776, 1783, 1784, 1838, 1840, 1845, 1846	<code>\tabu@vleads</code>	<u>45</u> , 518
<code>\tabu@start@stop</code>	1836, 1837, 1847	<code>\tabu@vlinearg</code>	796, <u>802</u>
<code>\tabu@startpbox</code>	614, 637	<code>\tabu@warn</code>	151
<code>\tabu@startpboxmeasure</code>	1204, 1206, <u>1531</u>	<code>\tabu@warn@cellspace</code>	2386, 2414, 2420
<code>\tabu@startpboxquick</code>	1207, 1549, <u>1581</u>	<code>\tabu@warn@usetabu</code>	1991, 2009
<code>\tabu@startstop</code>	1779, <u>1836</u>	<code>\tabu@warncolour</code>	492, 496, 498
<code>\tabu@starttime</code>	1456, 1457, 1458, 2438	<code>\tabu@wd</code>	974, 975, 1321, 1325, 1326, 1410, 1444, 1558, 1574, 1578, 1579, 2012
<code>\tabu@starttimer</code>	619, 2358, 2362	<code>\tabu@wddef</code>	1153, 1334, 1336, 1399, 1524, 1559, 1572, 1960, 2012
<code>\tabu@stop</code>	<u>21</u> , 235, 238, 1777, 1780, 1785, 1839, 1840, 1841, 1842, 1843, 1846	<code>\tabu@WRITE</code>	145, 148, 150, 151
<code>\tabu@stoptime</code>	1454, 1456, 1457, 1458	<code>\tabu@write</code>	143, 147
<code>\tabu@strategy</code>	1135, <u>1181</u> , 1311	<code>\tabu@X</code>	68, 1171
<code>\tabu@strippt</code>	964, 1479, 1482, 1499	<code>\tabu@Xalign</code>	919, 920, 921, 922, 923, 924, 925, 926, 934, 940
<code>\tabu@strtrim</code>	<u>80</u> , 97, 289, 486	<code>\tabu@Xarg</code>	870, <u>884</u>
<code>\tabu@tabarray</code>	616, 639	<code>\tabu@Xc@ef</code>	947, 949, 951
<code>\tabu@tabu@</code>	587, 589, 592	<code>\tabu@xcline</code>	1790, 1818, 1827
<code>\tabu@tabudecimal</code>	630, 1848, 1852, 1871	<code>\tabu@Xcoef</code>	928, 930, 946, 948
<code>\tabu@tabular</code>	541, 546	<code>\tabu@Xcoefs</code>	965, 1000, 1047, 1049, 1142, 1315, 1348, 1383, 1960
<code>\tabu@target</code>	<u>30</u> , 561, 575, 584, 585, 594, 602, 605, 874, 876, 877, 1039, 1041, 1045, 1048, 1055, 1080, 1141, 1144, 1151, 1185, 1264, 1288, 1308, 1321, 1326, 1344, 1347, 1360, 1364, 1380, 1387, 1388, 1391, 1393, 1396, 1397, 1421, 1424, 1428, 1439, 1446, 1450, 1505, 1539, 1540, 1542, 1570, 1972, 1991, 1993	<code>\tabu@Xcol</code>	<u>21</u> , 885, 964, 1001, 1148, 1538, 1558, 1559, 1576, 1578, 1579
<code>\tabu@textbar</code>	<u>100</u> , 283, 835, 1004	<code>\tabu@Xd@sp</code>	952, 953, 958
<code>\tabu@thebody</code>	<u>43</u> , 1083, 1084, 1086, 1094, 1095, 1159, 1221, 1230, 1253, 1257	<code>\tabu@Xdisp</code>	886, 954, 967
<code>\tabu@thehleaders</code>	513, 1814	<code>\tabu@xfootnote</code>	628, 1127, 2198, 2390
<code>\tabu@thehline</code>	1786, 1812, 1817	<code>\tabu@xfootnotetext</code>	627, 2195
<code>\tabu@thehrule</code>	504, 1811, 1815	<code>\tabu@xhline</code>	623, 1683, 1685, 1694
<code>\tabu@theleaders</code>	513, 520, 533, 537	<code>\tabu@xhlineAZ</code>	<u>1695</u>
<code>\tabu@theparam</code>	483, 489	<code>\tabu@Xinit</code>	1142, 1148, 1155
<code>\tabu@therule</code> 525, 528, 529, 530, 532, 534, 536		<code>\tabu@Xlcr</code>	885, 935, 937, 938, 967
<code>\tabu@thestyle</code> 270, 275, 278, 393, 395, 396, 398, 432, 433, 442, 803, 807, 808, 1781, 1793, 1810		<code>\tabu@Xm@th</code>	929, 952
<code>\tabu@thetarget</code>	612, 649, 877, 1416, 1417	<code>\tabu@Xmath</code>	886, 953, 967, 969
<code>\tabu@thevleaders</code>	520, 821	<code>\tabu@Xparse</code>	<u>884</u>
<code>\tabu@thevline</code> 806, 818, 827, 1659, 1664, 1673		<code>\tabu@Xparsespace</code>	896, 927
<code>\tabu@thevrule</code>	504, 825	<code>\tabu@Xrewritten</code>	893, <u>959</u>
<code>\tabu@thick</code>	<u>30</u> , 403, 429, 430, 434, 438, 439, 1706, 1707, 1719, 1720, 1721	<code>\tabu@Xsum</code>	<u>38</u> , 1142, 1148, 1334, 1355, 1431
<code>\tabu@titles</code>	1341, 1490	<code>\tabu@Xtest</code>	895, 897, 933
<code>\tabu@to</code>	569, 573	<code>\tabu@Xtype</code>	916, 917, 918, 941, 945
<code>\tabu@tr@cing</code>	2339, 2341, 2364	<code>\tabu@xline</code>	<u>16</u> , 1700, 1709, 1713, <u>1776</u>
<code>\tabu@tracing</code>	2336, 2337, 2338, 2340	<code>\tabu@xline@scan</code>	1700, 1708
<code>\tabu@tracing@save</code>	2357, 2361	<code>\tabucolumn</code>	<u>791</u>
<code>\tabu@tracinglines</code>	1788, 2342, 2343	<code>\tabucolX</code>	<u>30</u> , 888, 890, 949, 961, 962, 964, 974, 975, 977, 1040, 1141, 1142, 1145, 1152, 1186, 1263, 1307, 1319, 1321, 1323, 1352, 1355, 1362, 1373, 1379, 1388, 1404, 1426, 1431, 1446, 1448, 1506, 1539, 1540, 1577, 1973
<code>\tabu@trial</code>	<u>1212</u>	<code>\tabucolX@init</code>	1131, 1141, 1147, 1401
<code>\tabu@trialh@k</code>	140, 141, 1122	<code>\tabudecimal</code>	<u>21</u> , 630, <u>1848</u>
<code>\tabu@trimspaces</code>	84, 87, 91	<code>\tabudefaulttarget</code>	610, 612, 644, 650, 1126, 1415, 1418, 2331, 2445
<code>\tabu@trivlist</code>	626, 2247	<code>\tabuDisableCommands</code>	<u>140</u> , 379
		<code>\tabuendlongtrial</code>	1231, 1233, 1236
		<code>\tabular</code>	546

