# t a b u

## Flexible LaTeX tabulars

FC

2010/11/15 – version 1.2

**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.

tabu is more flexible that tabular, tabular*, tabularx and array and extends the possibilities. All tabulars in this document were made with the tabu environment. Last (but not least) tabu is more than compatible with any package that provides commands to format tabulars. Indeed, tabu **does not modify any of the macros of array.sty.** All functionalities are obtained with only two modifications[1] which are loaded only inside a tabu environment.

tabu requires ε-TeX, the standard package array.sty and no other package.

## Contents

**1 The tabu environment ..... 3**
- 1.1 tabu, tabu to and tabu spread ..... 3
- 1.2 tabu X columns ..... 3
  - *1.2.1 X columns with "tabu spread"*
  - *1.2.2 Nesting tabus with X columns*
- 1.3 Lines inside tabu ..... 5
  - *1.3.1 First important remarks*
  - *1.3.2 Vertical lines: | has an optional parameter*
  - *1.3.3 More style for horizontal lines: \tabucline*
- 1.4 tabu in math mode ..... 7
- 1.5 Verbatim inside tabu with X columns ..... 8
- 1.6 Printing numbers inside tabu with numprint and siunitx ..... 8
  - *1.6.1 Just make it easy !*
  - *1.6.2 You should know how it works...*
- 1.7 delarray shortcuts ..... 9

**2 Save the tabu preamble and X columns widths: \savetabu, \usetabu ..... 9**

**3 Change the font and the alignment in one row: \rowfont ..... 10**

**4 Differences between tabu, tabular and tabularx ..... 11**
- 4.1 Paragraph indentation ..... 11
- 4.2 Custom environments ..... 11
- 4.3 Inversion of tokens ..... 11
- 4.4 Improved process for rewritting columns ..... 12

**5 The package options ..... 12**
- 5.1 The debugshow package option ..... 12
- 5.2 The linegoal package option ..... 13
- 5.3 The delarray package option ..... 13
- 5.4 The light package option ..... 13

**6 Corrections of some bugs (available only inside tabu) ..... 13**
- 6.1 Correction for colortbl and arydshln: compatibility with delarray ..... 13
- 6.2 Correction for arydshln: @ columns ..... 14

This documentation is produced with the +DocStrip+ utility. Package tabu is used with the linegoal option.

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

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

1. \prepnext@tok and \save@decl. See "The light package option".

**7 Implementation** ..... **15**
7.1 Identification, requirements and options ..... 15
7.2 Some constants ..... 16
7.3 \tabu and \endtabu ..... 19
7.4 Flow chart of expansion ..... 21
*7.4.1 General case: tabu, tabu to and tabu spread*
*7.4.2 tabu to with X column*
*7.4.3 tabu spread with X column*
7.5 tabu X column definition ..... 23
7.6 Before trials to reach the target ..... 26
*7.6.1 Trial setup after \@mkpream*
*7.6.2 Collecting the tabu body: required for X columns*
7.7 Measuring the tabu ..... 27
*7.7.1 One trial after another*
*7.7.2 The arithmetic of X columns*
*7.7.3 Reporting in the .log file (debugshow option)*
*7.7.4 Measuring the natural width for tabu spread*
7.8 Lines inside tabu ..... 30
*7.8.1 Vertical lines*
*7.8.2 Horizontal lines: \tabucline*
7.9 Verbatim inside tabu with X columns ..... 34
7.10 Numbers in tabu ..... 34
*7.10.1 \tabudecimal*
7.11 \savetabu ..... 35
7.12 \rowfont ..... 36
*7.12.1 Setting font and alignment specification*
*7.12.2 Preparing stuff to be able to use \rowfont*
*7.12.3 Neutralisation of glues and alignment modification*
7.13 Utilities ..... 41
*7.13.1 tabu \fbox*
*7.13.2 \centering, \raggedright, \raggedleft*
7.14 Corrections ..... 41
*7.14.1 delarray comptability fix for colortbl and arydshln*
*7.14.2 arydshln @ columns*
*7.14.3 arydshln and empty p columns*

**8 History** ..... **44**
[2010/11/15 v1.2] ..... 44
[2010/10/28 v1.1] ..... 44

# 1 The tabu environment

## 1.1 tabu, tabu to and tabu spread

\begin{tabu}[⟨pos⟩]{⟨tabular preamble⟩}
\begin{tabu} to ⟨dimen⟩[⟨pos⟩]{⟨tabular preamble⟩}
\begin{tabu} spread ⟨dimen⟩[⟨pos⟩]{⟨tabular preamble⟩}

The tabu environment behaves exactly like tabular: the preamble is parsed by the macros in array.sty with no modification in the characters classes. tabu improves tabular and array:

- footnotes and index words are allowed inside tabu
- X columns are also allowed, with an *optional* parameter for the width-coefficient, the alignment (l,r,c or L,R,C for ragged2e settings) and the column type (p,m,b). tabu has also a *default target width* when used with X columns.
- You are used to the tabular environment in text mode, and array 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 introduces optional parameters to columns specifications, providing facilities for modifying columns or line widths, inserting verbatim text...
- tabu is more than compatible with arydshln (for dashed and dotted lines) and colortbl: actually some corrections of those packages are loaded as soon as you enter a tabu environment. Compatibility with delarray, hhline, makecell, booktabs, siunitx, dcolumn, warpcol, *etc.* is fine too. 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 you specify another value for \extracolsep. 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.

## 1.2 tabu X columns

tabu X columns can be viewed as an enhancement of tabularx X columns:

- 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 5/7 of the whole tabular width.

| X[2.5] | 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 TeX: by you...).

| Modifier | Meaning | Default |
|---|---|---|
| `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: `>{$}X<{$}` | |

- `tabu X` columns can be spanned with `\multicolum` without any restriction.
- `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 $\varepsilon$-TeX `\dimexpr` instead of TeX primitives (with round/truncate bias correction).
- Convergence to the target width is optimized: 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 pdfTeX either in `pdf` or `dvi` mode, and package linegoal is loaded. Remember that `\linegoal` uses zref-savepos and `\pdfsavepos`: 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 is 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 London 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.<br>The result of this example is the same as if one had written \begin{tabu}to\linewidth. | **Sherlock Holmes** |
| And a \multicolumn cell, just to see | | |

In the preamble, @{} means that you remove the margin.

### Nesting tabus with X columns

This section should contain some examples but I've no time presently...

## 1.3 Lines inside tabu

### First important remarks

The features provided in this section are quite experimental: they are not generally taken for good typography. You can use tabu 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 tabu does not.

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

### 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 (sorry for that, but I was tired to parse parameters... as for X columns, commas are optional). Example:

| | | |
|---|---|---|
| Hello \| World | The tabu you see on left was made with the code displayed on the right. | \begin {tabu}{|||[5pt]|c|c|||[5pt]|}<br>Hello & World<br>\end {tabu} |
| Hello \| World | The tabu you see on left was made with the code displayed on the right. | \begin {tabu}{|||[5pt,red]|c|c|||[5pt blue]|}<br>Hello & World<br>\end {tabu} |

The \verbatim command available inside tabu with X columns was used: guess in which cells...?

Note that it is always a good idea to protect the optional argument with braces: [{...}]. But it's not necessary because tabu takes care the | token to be rewritten before any other column type (just after ∗ however, for obvious reasons). But if you use the optional argument for the vertical line into a user-defined column type (declared with \newcolumntype for example), you can get an error. In this case, it is compulsory to protect the optional argument by braces. Finally, it's not very often that a user-column type contains a vertical bar...

### More style for horizontal lines: \tabucline

\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*⟩ on⟨*dash*⟩ off⟨*gap*⟩]{⟨*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.

| | | |
|---|---|---|
| | \tabucline[1pt on 1.5pt off 2pt]{1-4} | 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. |
| [1pt on 1.5pt off 2pt] | \tabucline[1.5pt]{-} | draws a horizontal solid line of width 1.5pt between the first and the last column. |
| [1.5pt] | \tabucline{2-} | draws a horizontal solid line of width \arrayrulewidth between the second column and the last one. |
| default | \tabucline[on 2pt red]{-5} | draws a horizontal dashed line between columns 1 and 5 of width \arrayrulewidth. Dashed are 2pt long and gaps width is 4pt (the default). |
| [on 2pt red] | | |

\tabucline∗ [⟨*style or spec.*⟩]{start-end}

For fine tuning, the star form \tabucline∗ can be used to keep the vertical lines that might cross the horizontal line. As a consequence, the content of special @ of ! columns will interrupt the horizontal line either. This might be usefull when \extrarowheight is high. Example:

| | |
|---|---|
| Hello | world |
| Hello | world |

\tabucline

| | |
|---|---|
| Hello | world |
| Hello | world |

\tabucline∗

| | |
|---|---|
| Hello | world |
| Hello | world |

\firsthline and \hline and \lasthline

In each table, \extrarowheight is equal to 8 pt.

\tabuclines tries to put the line in the middle, so that the text is centered in its cell. \tabuclines∗ tries to take care of vertical lines. Well this is not perfect: it works pretty well with simple vertical lines, but not really with double lines, triple lines with colors etc... But it can help in simple cases...

`\tabulinestyle{⟨style=spec., style=spec., style=spec. ...⟩}`

This command defines a line style to be used in the first optional argument of `\tabucline`:

| Define the line style | Use the line style |
|---|---|
| `\tabulinestyle{myline=0.4pt on 1.5pt off 1pt}` | `\tabucline[myline]{-}` |
| `\tabulinestyle{myleader=\hbox{$\scriptstyle\star$}}` | `\tabucline[myleader]{1-3}` |

This way it is easy to change dashed lines into solid ones

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

Now we stop it because things are be- com- ing too ugly and go to have a look at `tabu` in mathematical mode...

## 1.4 `tabu` in math mode

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

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

In this example, the big `tabu` is: `\begin{tabu} to\linewidth{X[1.5]X[r$]}`.

The nested `tabu` (in math mode) uses `delarray` shortcut: its preamble is:

`\begin{tabu}({>{\Gape[8pt]\displaystyle}r>{\displaystyle}l}`.

You immediately see the interest for `tabu` to have a default target.

`\Gape` is a `makecell` macro to adjust the height and depth of the rows.

Horizontal rules are `booktabs` `\toprule`, `\midrule` and `\bottomrule`.

| `array` | `tabu` | `tabu spread 1em` |
|---|---|---|
| $\left\lvert \begin{array}{cc} \alpha & \beta \\ \gamma & \delta \end{array} \right\rvert$ | $\left\lvert \begin{array}{cc} \alpha & \beta \\ \gamma & \delta \end{array} \right\rvert$ | $\left\lvert \begin{array}{cc} \alpha & \beta \\ \gamma & \delta \end{array} \right\rvert$ |

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

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

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

## 1.5 Verbatim inside tabu with X columns

The process of computing X columns widths implies to make "trials": this means that the tabular is first printed in internal mode, inside a box which is measured until the target width is reached. Such trials require to collect the content of the tabu environment. Therefore, the category codes of the characters read in the environment are fixed, and \verb commands are not allowed.

\verbatim{⟨*general text*⟩}

To get round this limitation, tabu provides the command \verbatim which allows to put some pieces of verbatim text inside a tabu: it is based on ε-TeX \scantokens primitive and is defined only inside the tabu environment. \verbatim has some other limitations you must know:

| | |
|---|---|
| ^^@ | can be used to get a carriage return inside \verbatim |
| ^ | consequently ^ is a normal subscript character |
| { } | curly braces must be equilibrated |
| % | The percent character % is for commentaries: the content of the environment is collected before \verbatim does its job, therefore commentaries are removed... |

If you need more verbatim inside a tabu environment:

- you should avoid the use of X columns
- or if you really want to use X columns, you must save your verbatim text before the tabu, for example with the SaveVerbatim environment provided by the package fancyvrb.

## 1.6 Printing numbers inside tabu with numprint and siunitx

### Just make it easy !

tabu 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 developped 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 have to be defined before.

Example with **\numprint**:

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

\begin {tabu}{|[GreenYellow]*2{>{\tabudecimal \usermacro }X[r]|[GreenYellow]}}
 \rowfont [c]{\bf } January & February \\
 12.324 & 745.32 \\
 21.13 & 0 \\
 213.3245 & 12.342 \\
 2143.12 & 324.325 \\
 \end {tabu}
```

| January | February | ... |
|---|---|---|
| 12,32 € | 745,32 € | ... |
| 21,13 € | 0,00 € | ... |
| 213,32 € | 12,34 € | ... |
| 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.

### 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 exponants. 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...

## 1.7 delarray shortcuts

When you enclose your tabular with math delimiters using delarray shortcuts, tabu tries to reach its target for the whole: the tabular and the delimiter(s). You can see the difference:

$$\left(\begin{array}{l}\text{This is tabu with delarray shortcut for parenthesis around}\end{array}\right)$$

$$\left(\begin{array}{l}\text{This is tabularx with delarray shortcut for parenthesis around}\end{array}\right)$$

$$\left\{\begin{array}{l}\text{This is tabu with delarray shortcut for curly brackets around}\end{array}\right\}$$

$$\left\{\begin{array}{l}\text{This is tabularx with delarray shortcut for curly brackets around}\end{array}\right\}$$

With `overfull hboxes (15.8pt and 16.1pt too wide)` for `tabularx`.

# 2 Save the tabu preamble and X columns widths: \savetabu, \usetabu

`\savetabu{`⟨*user-name*⟩`}`

The command `\savetabu` can be used at the end of any line of a `tabu` environment to save the target width (or the spread), the columns specifications (tabular preamble) and the widths of `tabu X` columns (if there is any). This possibility 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.

`\usetabu{⟨user-name⟩}`

`\usetabu` is the complement of `\savetabu`: it can be used 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` locally restores:

- the preamble,
- 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, and `X` columns are directly transformed into `p`, `m` or `b` columns.
- `\tabcolsep` (or `\arraycolsep` in math mode)
- `\arrayrulewidth`, `\doublerulesep`, `\extrarowheight` and `\extratabsurround`

Example:

```
\extrarowheight=5pt\tabcolsep=12pt
\begin{tabu}to .7\linewidth{|XXX|X[c]|} \savetabu{mytabu}   \tabucline*[on1pt]-
 This & is & tabu & package                                  \\ \tabucline*[on1pt]-
\end{tabu}
\begin{tabu}{\usetabu{mytabu}}                               \tabucline*[on1pt]-
\multicolumn3{|c}{This is tabu} & package                   \\ \tabucline*[on1pt]-
\end{tabu}
```

| This | is | tabu | package |
|---|---|---|---|
| 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. `\usetabu` is a help to get several tabulars of exactly the same shape, same target, same preamble.

## 3 Change 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.

# 4 Differences between tabu, tabular and tabularx

## 4.1 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 LaTeX 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

## 4.2 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 and \usetabu for this purpose.

## 4.3 Inversion of tokens

When you typeset the following tabular:

```
\begin{tabular}{|>{\bfseries}>{ before }l<{ 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.

## 4.4 Improved process for rewritting columns

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

```
\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}
```

Inside tabular:

1) Rewrite S: not found because inside {...}
2) Rewrite ∗
3) Rewrite n column defined by package numprint
   Then the 'n' in green is rewritten ⟶ problem

Inside tabu:

1) Rewrite ∗
2) Rewrite | (there is none here)

go back

3) Rewrite ∗
4) Rewrite |
5) Rewrite S
6) Rewrite n ⟶ not found because S rewritten before, according to siunitx definition.

The process of rewritting 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, not rebuilt before each trial as tabularx does...

The process of rewritting is very sensitiv to the order in which columns are actually rewritten. Therefore, if it possible to define a new column type using the X token for use with tabu:

```
\newcolumntype{C}{X[c]}
```

it is not recommanded no nest such constructions like:

```
\newcolumntype{Q}{>{\color{green}}C}
```

In fact, a problem may arise in nested tabus if such a Q column type is defined before the C column type...

Well, just avoid to nest new column type definitions !

# 5 The package options

## 5.1 The debugshow package option

\tracingtabu

With the package option debugshow, tabu will report the widths it computes at each attempt to read the target, when X columns are used. The control sequence \tracingtabu has the same effect as the debugshow option.

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=386.67296pt
   The tabu width (797.34592pt) exceeded the target by 410.67296pt.

Thus X has been updated: 410.67296pt /2 = 205.33649pt and then:
X = 386.67296pt − 205.33649pt = 181.33647pt

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.

## 5.2 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 pdfTeX, otherwise, a warning is displayed and the linegoal option has no effect: the default target remains \linewidth. \linegoal works with pdfTeX 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

## 5.3 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.

## 5.4 The light package option

When you enter a tabu environment, two macros amongst the smallest possible of array.sty are modified: these are \prepnext@tok, which is expanded while \@mkpream builds the \halign preamble and \save@decl to avoid inversion of tokens in text mode (see Inversion of tokens).

Modification of \prepnext@tok is loaded only inside the tabu environment, while the modification of \save@decl is loaded only inside the group in which \@mkpream works: this is very very local to tabu and cannot interfere with any other tabular !

\prepnext@tok could have been loaded inside the \@mkpream group as well, but then \rowfont would not have worked inside array or tabular nested into a tabu. Thus the choice.

As you see, these modifications do no modify tabular, tabular*, tabularx, longtable *etc.etc.etc.*

If for some reason you prefer the original macros of array.sty, then you can load tabu with the "light" option. As a consequence: \rowfont and \tabucline will not be available, and vertical lines | will not have an optional argument for their widths and color. However, tabu X columns will work normally.

# 6 Corrections of some bugs (available only inside tabu)

## 6.1 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.

Those control sequences are used by the delarray package to put parenthesis or bracket around the array.

## 6.2 Correction for arydshln: @ 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 arydshln documentation is solved.

# 7 Implementation

## 7.1 Identification, requirements and options

The package namespace is **tabu@**.

```
1 ⟨*package⟩
2 \NeedsTeXFormat{LaTeX2e}[2005/12/01]
3 \ProvidesPackage{tabu}[2010/11/15 v1.2 - flexible LaTeX tabulars (FC)]
4 \RequirePackage{array}[2008/09/09]
```

Then minimal catcode acertaining for loading tabu in good conditions:

```
5 \AtEndOfPackage{\tabu@AtEnd\let\tabu@AtEnd\@undefined}
6 \let\tabu@AtEnd\@empty
7 \def\TMP@EnsureCode#1#2{%
8   \edef\tabu@AtEnd{%
9     \tabu@AtEnd
10    \catcode#1 \the\catcode#1\relax
11  }%
12  \catcode#1 #2\relax
13 }% \TMP@EnsureCode
14 \TMP@EnsureCode{33}{12} % !
15 \TMP@EnsureCode{124}{12}% |
16 \TMP@EnsureCode{0}{12}%  ^^@
17 \TMP@EnsureCode{36}{3}% $ = math shift
```

`\tracingtabu (debugshow option)` `\tracingtabu` is the same as option debugshow.

```
18 \let\tabu@message\@gobble
19 \def\tracingtabu{\let\tabu@message\message}
20 \let\tabudefaulttarget\linewidth
21 \DeclareOption{debugshow}{\tracingtabu}
```

`linegoal (package option)`

```
22 \DeclareOption{linegoal}{%
23   \AtEndOfPackage{\RequirePackage{linegoal}[2010/10/31]}%
24   \def\tabudefaulttarget{\linegoal}% \linegoal is \linewidth if not pdfTeX
25 }% linegoal option
```

`delarray (package option)`

```
26 \DeclareOption{delarray}{%
27   \AtEndOfPackage{\RequirePackage{delarray}}%
28 }% delarray option
```

`light (package option)`

```
29 \DeclareOption{light}{%
30   \AtEndOfPackage{%
31     \let\tabu@prepnext@tok \prepnext@tok
32     \let\tabu@save@decl \save@decl
33     \let\tabu@rowfont \tabu@norowfont
34     \let\tabucline \tabu@nocline
35     \let\tabu@firstcline \relax
36     \let\tabu@lines \relax
37   }
38 }% light option
39 \def\tabu@norowfont{\PackageError{tabu}
40   {\string\rowfont\space is not available with option 'light'}\@ehd}
41 \def\tabu@nocline{\PackageError{tabu}
42   {\string\tabucline\space is not available with option 'light'}\@ehd}
```

```
\ProcessOptions
```

At Begin Document, we check if a `X` column has already been defined (tabularx) and if not, we define a new column type `X`.

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

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

```
\AtBeginDocument{%
   \@ifundefined{NC@rewrite@X}{\newcolumntype{X}{}}{}% new column X if not exists
   \expandafter\in@\expandafter\@arrayright\expandafter{\endarray}%
   \ifin@ \let\tabu@endarray\endarray
   \else \tabu@fix@arrayright \fi % <fix for colortbl & arydshln (delarray)>
   \@ifpackageloaded{colortbl} \tabu@colortbltrue \tabu@colortblfalse
}
```

## 7.2 Some constants

`\tabu@cnt` Used in in `\tabu@arith` (the number of trials) and `\tabu@prepnext@tok` (for `\rowfont`).

`\tabu@nbcols` A counter that save the number of columns of the `tabu`.

`\tabu@X@cols` Used only when `tabu X` columns are used with "`tabu spread`".

```
\newcount\tabu@cnt
\newcount\tabu@nbcols
\newcount\tabu@X@cols
```

`\tabu@target` Stores the `tabu` target (either "`to`" or "`spread`").

`\tabu@spreadtarget` Used only when `tabu X` columns are used with "`tabu spread`".

`\tabu@naturalX`

```
\newdimen\tabu@target
\newdimen\tabu@spreadtarget
\newdimen\tabu@naturalX
```

`\tabucolX` The dimen corresponding to the preamble token `X[1]`: the standard width of `X` columns.

`\tabu@X@sum` Stores the sum of all width coefficients for `X` columns.

```
\newdimen\tabucolX
\newdimen\tabu@X@sum
```

`\iftabu@measuring` This switch is set to `true` by `\tabu@arith` if the trial did not reached the target.

It is also temporarily set to true when the first `X` column is encountered in the `tabu` preamble, at the time `\@mkpream` scans it to built the `\halign` preamble. The first `X` column found actually triggers some special setup to be expanded before `\halign` (see the flow chart...)

`\iftabu@spread` A switch whether "`tabu spread`" is used or not.

`\iftabu@nested` A swith set at the entry in `tabu` environment: true if `tabu` is nested inside another `tabu`.

`\iftabu@firstcline` A switch to adapt `\tabucline` automatically if it comes first in the `tabu` (similar to `\firsthline`).

```
\newif\iftabu@measuring
\newif\iftabu@spread
\newif\iftabu@nested
\newif\iftabu@firstcline
```

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

It is also used by `\tabucline` to save the `\@arstrutbox` when inserting a horizontal line.

```
\newsavebox\tabu@box
```

`\tabu@gobblespace` A macro which is needed when scanning tokens with `\futurelet`.

```
\def\tabu@gobblespace#1 {#1}
```

`\tabu@NC@rewrite@X` 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}`

```
\newcommand\tabu@NC@rewrite@X[1][]{\tabu@rewrite@X{#1}%
                                   \let\@halignto \relax
                    \expandafter \NC@find \tabucolX@spec}
```

The next part of the definition (`\tabu@rewrite@X`) can be found page 23.

`\usetabu (new column type)` `\usetabu` is defined as a new column type. `\NC@rewrite@\usetabu` is expanded where `\@mkpream` does its job.

```
\expandafter\def\csname NC@rewrite@\string\usetabu\endcsname#1{%
   \ifx\\#1\\\tabu@saveerr{}\else
      \@ifundefined{tabu@saved@\string#1}
         {\tabu@saveerr{#1}}
         {\let\tabu@rewrite@X \tabu@rewrite@Xrestore
         \def\tabu@temp{\xdef\tabu@usetabu{%
            \col@sep \the\col@sep\relax
            \arrayrulewidth \the\arrayrulewidth\relax
            \@tempdima \ht\@arstrutbox \advance\@tempdima -\extrarowheight
            \extrarowheight \the\extrarowheight\relax
            \advance\@tempdima \extrarowheight
            \ht\@arstrutbox \@tempdima
            \extratabsurround \the\extratabsurround\relax
            \doublerulesep \the\doublerulesep\relax}}%
         \aftergroup \tabu@usetabu
         \csname tabu@saved@\string#1\expandafter\endcsname}%
   \fi
}% \NC@rewrite@\usetabu
\expandafter\def\csname NC@find@\string\usetabu\endcsname#1\usetabu{\NC@{#1}}
```

`\tabu@rewritefirst (new column type)` 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@do\tabu@rewritefirst` is added **at the beginning of** `\NC@list` at the entry of a (not nested) `tabu` environment.

This "column type" sets up the new column type `\tabu@rewritelast` which is added to **at the end of** `\NC@list`, and defines the token `X` to be rewritten by `\tabu@NC@rewrite@X` (in case tabularx is used with tabu, this modification of the `X` column occurs only inside the group where `\@mkpream` does its job).

```
\expandafter\def\csname NC@rewrite@\string\tabu@rewritefirst\endcsname{%
   \ifx\tabu@lines\relax
   \NC@list\expandafter{\expandafter\NC@do \expandafter\usetabu
                        \NC@do X\NC@do\tabu@rewritelast}%
   \else
   \NC@list\expandafter{\expandafter\NC@do \expandafter\usetabu
                        \expandafter\NC@do \expandafter|\tabu@NC@list
                        \NC@do X\NC@do\tabu@rewritelast}%
      \tabu@lines % defines NC@rewrite@| for tabu only (inside @mkpream group)
   \fi
   \let\save@decl \tabu@save@decl % inversion of tokens in text mode
   \let\NC@rewrite@X \tabu@NC@rewrite@X
   \aftergroup \tabu@global@temp
   \aftergroup \tabu@firstcline
   \NC@find \tabu@rewritelast
}% NC@rewrite@\tabu@rewritefirst
```

```
\expandafter\def\csname NC@find@\string\tabu@rewritefirst\endcsname
                                              #1\tabu@rewritefirst{\NC@{#1}}
\def\tabu@rewritefirst{%
   \edef\tabu@NC@list{\the\NC@list}%
   \NC@list{\NC@do \tabu@rewritefirst \NC@do *}%
}% \tabu@rewritefirst
```

`\tabu@rewritelast (new column type)` This new column type is rewritten after `X` columns, because it is declared by when the column `\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.

```
\expandafter\def\csname NC@rewrite@\string\tabu@rewritelast\endcsname{%
   \global\NC@list\expandafter{\tabu@NC@list}%
   \futurelet \tabu@temp \tabu@rewritelast
}% \NC@rewrite@\tabu@rewritelast
\expandafter\def\csname NC@find@\string\tabu@rewritelast\endcsname
                                              #1\tabu@rewritelast{\NC@{#1}}
\def\tabu@rewritelast{%
   \ifx \@halignto\relax   % found a X column
      \let\tabu@temp \@empty
      \aftergroup \tabu@addphantomline
   \else
      \ifdim \tabu@target=\z@ \let\tabu@temp \@empty
      \else
         \if |\noexpand\tabu@temp \def\tabu@temp{@{\extracolsep\@flushglue}}\else
         \if !\noexpand\tabu@temp \def\tabu@temp{@{\extracolsep\@flushglue}}\else
                                  \def\tabu@temp{!{\extracolsep\@flushglue}}\fi\fi
      \fi
   \fi
   \let\@halignto \tabu@halignto
   \expandafter\NC@find \tabu@temp
}% \tabu@rewritelast
```

`\tabu@everycr@tok` This token is used to store and restore the content of `\everycr` when `\rowfont` is used.

`\iftabu@colortbl` The switch `\iftabu@colortbl` is used by `\rowfont` when modifying the alignment, because colortbl changes the glue put inside the `\halign` preamble to make standard alignments. This switch is set At Begin Document.

```
\newtoks\tabu@everycr@tok
\newif\iftabu@colortbl
```

`\tabu@nowrite` A trick to fobidd `\write` when a trial is done on the `\halign`. (Copied from D. Carlisle's tabularx code.)

```
\def\tabu@nowrite{% cancels a \write command (tabularx method)
   \begingroup
     \def\let{\afterassignment\endgroup\toks@}%
        \afterassignment\let\count@
}% \tabu@nowrite
\let\tabu@write\write
```

`\tabu@phantomline` 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`...

```
\def\tabu@phantomline{\noalign{%
   \global\everycr{}%
```

```
\global\setbox\tabu@box\box\@arstrutbox
\global\let\tabu@minrowclearance \minrowclearance
\global\let\minrowclearance \z@
\global\let\tabu@temp \vcenter
\global\let\vcenter \vbox
\toks@{}\count@\@ne
\@whilenum\count@<\tabu@nbcols\do{\advance\count@\@ne
                        \toks@\expandafter{\the\toks@&}}%
\toks@\expandafter{\the\toks@
   \cr\noalign{\global\setbox\@arstrutbox\box\tabu@box
      \global\let\minrowclearance \tabu@minrowclearance
      \global\let\vcenter \tabu@temp}}%
\expandafter}\the\toks@
}% \tabu@phantomline
```

## 7.3 \tabu and \endtabu

\tabu `\tabu` is the command of the environment.

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

```
\def\tabu{%
   \ifmmode \def\endtabu{\endarray}%
   \else    \def\endtabu{\endtabular}\fi
   \tabu@setup \tabu@settarget
}% \tabu
```

\tabu@setup This macro sets the `tabu X` column definition at the beginning of the `tabu` environment.

Incidentally, `\tabu@X` (number of `tabu X` columns) and `\tabu@X@sum` (sum of the width-coefs) are reset to 0.

The current value of `\hfuzz` is stored in `\tabu@hfuzz`, with a minimum of $0.1pt$.

```
\def\tabu@setup{%
   \let\adl@xarraydashrule \tabu@adl@xarraydashrule % <fix> arydshln
   \let\adl@act@endpbox \tabu@adl@act@endpbox       % <fix> arydshln
   \let\adl@act@@endpbox \tabu@adl@act@endpbox      % <fix> arydshln
   \let\@preamerror \@preamerr                     % <fix> arydshln
   \let\endarray \tabu@endarray          % <fix> colortbl & arydshln (delarray)
   \let\tabu@global@temp \@empty \let\tabu@global@X \@empty
   \ifx\verbatim \tabu@sanitizetext
         \tabu@nestedtrue
         \def\tabu@aligndefault{t}\def\tabudefaulttarget{\linewidth}%
   \else \tabu@nestedfalse
         \def\tabu@aligndefault{c}%
         \ifdim\parindent>\z@ \ifx\linewidth\tabudefaulttarget
         \everypar\expandafter{%                   % correction for indentation
            \the\everypar\everypar\expandafter{\the\everypar}%
            \setbox\z@=\lastbox
            \ifdim\wd\z@>\z@ \advance\linewidth -\wd\z@\fi
            \box\z@
            }%
         \fi\fi
   \fi
   \let\@footnotetext \tabu@footnotetext
   \let\@xfootnotenext\tabu@xfootnotetext
   \iftabu@nested\else
      \global\tabu@footnotes{}%
      \aftergroup\the\aftergroup\tabu@footnotes
   \fi
```

```
186    \let\centering \tabu@centering
187    \let\raggedright \tabu@raggedright
188    \let\raggedleft \tabu@raggedleft
189    \let\@trivlist \tabu@trivlist           %<restore \\=\@centercr inside lists>
190    \let\tabudecimal \tabu@tabudecimal
191    \let\verbatim \tabu@sanitizetext
192    \let\fbox \tabu@fbox
193    \let\prepnext@tok \tabu@prepnext@tok            % <for rowfont and tabucline>
194    \let\rowfont \tabu@rowfont
195    \tabu@spreadfalse \tabu@measuringfalse
196    \edef\tabu@hfuzz{\ifdim\hfuzz<.1\p@ .1\p@\else\the\hfuzz\fi}%
197    \tabu@rewritefirst
198 }% \tabu@setup
199 \def\tabu@save@decl{% no inversion on tokens when not in math mode
200    \ifcat$\d@llarend
201       \toks\count@\expandafter\expandafter\expandafter{%
202                \expandafter\@nextchar \the\toks\count@}%
203    \else
204       \toks\count@\expandafter\expandafter\expandafter{%
205          \expandafter\the\expandafter\toks\expandafter\count@\@nextchar}%
206    \fi
207 }% \tabu@save@decl
```

\tabu@settarget
\tabu@begin

The macro sets \tabu@target (a dimen) to the value specified for "tabu to" or "tabu spread".

```
208 \def\tabu@settarget{\futurelet\@let@token \tabu@sett@rget}
209 \def\tabu@sett@rget{\tabu@target\z@
210    \ifcase \ifx \bgroup\@let@token 0\else
211            \ifx [\@let@token 0\else
212            \ifx \@sptoken\@let@token 1\else
213            \if t\@let@token 2\else
214            \if s\@let@token 3\else
215            \m@ne\fi\fi\fi\fi\fi\relax
216        \expandafter\tabu@begin
217    \or \expandafter\tabu@gobblespace\expandafter\tabu@settarget
218    \or \expandafter\tabu@to
219    \or \expandafter\tabu@spread
220    \else\expandafter\tabu@begin
221    \fi
222 }% \tabu@sett@rget
223 \def\tabu@to to{\def\@halignto{to}\tabu@gettarget}
224 \def\tabu@spread spread{\tabu@spreadtrue\def\@halignto{spread}\tabu@gettarget}
225 \def\tabu@gettarget{\afterassignment\tabu@begin\tabu@target}
226 \def\tabu@begin#1#{%
227    \edef\@halignto{\ifdim\tabu@target>\z@ \@halignto\the\tabu@target\fi}%
228    \let\tabu@halignto \@halignto
229    \expandafter\@testopt\expandafter\tabu@@begin \tabu@aligndefault #1\@nil
230 }% \tabu@begin
231 \def\tabu@@begin[#1]#2\@nil#3{%
232    \edef\tabu@align{#1}%
233    \edef\tabu@saved{%
234       \ifmmode \col@sep \the\arraycolsep
235       \else    \col@sep \the\tabcolsep     \fi\relax
236       \arrayrulewidth \the\arrayrulewidth\relax
237       \extrarowheight \the\extrarowheight\relax
238       \extratabsurround \the\extratabsurround\relax
239       \doublerulesep \the\doublerulesep\relax}%
240    \expandafter\def\expandafter\tabu@saved\expandafter{\tabu@saved
241             \tabu@temp \edef\tabu@halignto{to\the\tabu@target}\NC@find #3}%
```

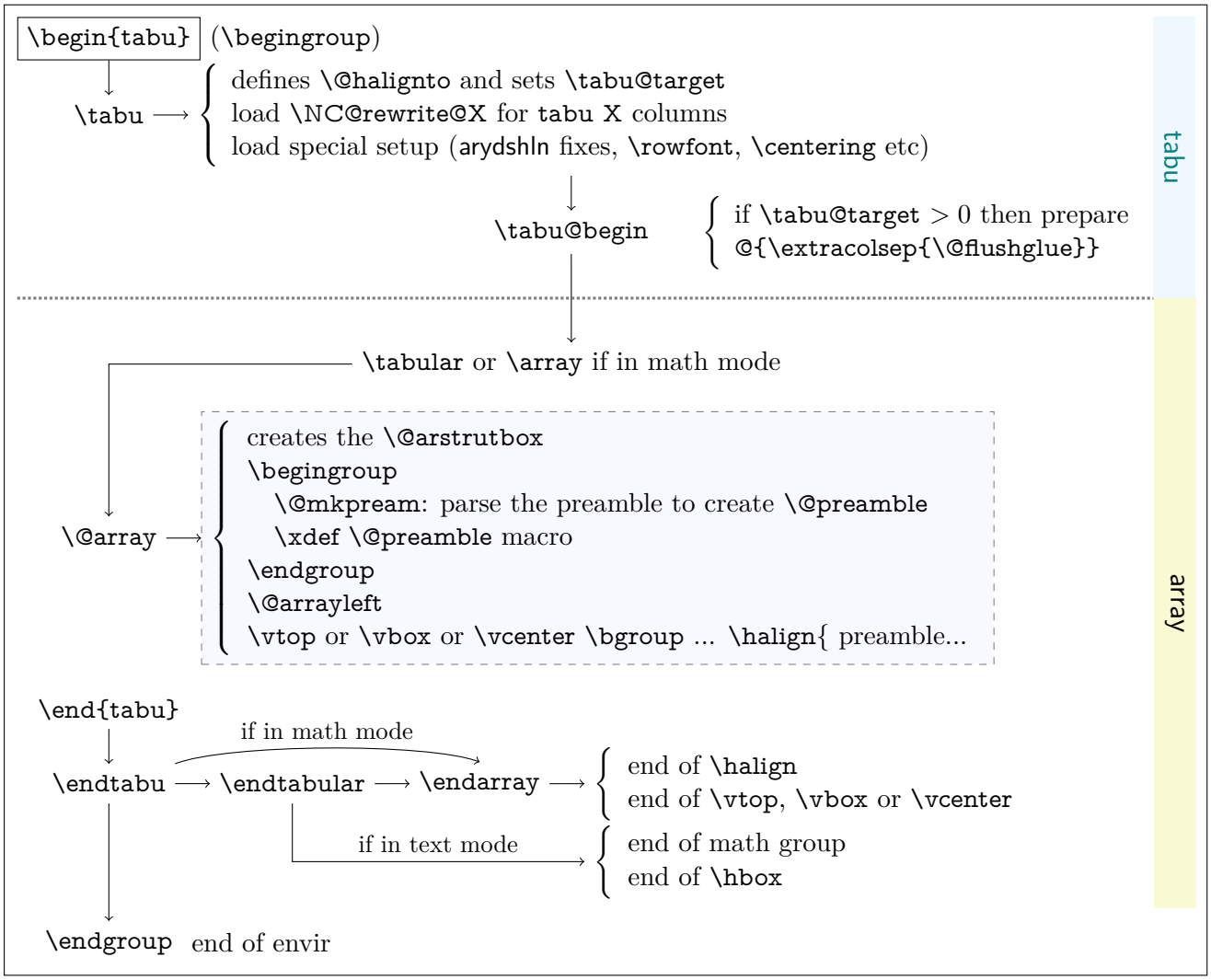
```
    \ifmmode \expandafter\array
    \else    \expandafter\tabular
    \fi      [{#1}]#2{\tabu@rewritefirst #3}%
}% \tabu@@begin
```

\tabu@footnotes

```
\newtoks\tabu@footnotes
\long\def\tabu@footnotetext#1{%
   \edef\@tempa{\the\tabu@footnotes
      \noexpand\footnotetext [\the\csname c@\@mpfn\endcsname]}%
   \global\tabu@footnotes\expandafter{\@tempa{#1}}}%
\long\def\tabu@xfootnotetext[#1]#2{%
  \global\tabu@footnotes\expandafter{\the\tabu@footnotes
                              \footnotetext[{#1}]{#2}}}
```

## 7.4 Flow chart of expansion

### General case: tabu, tabu to and tabu spread



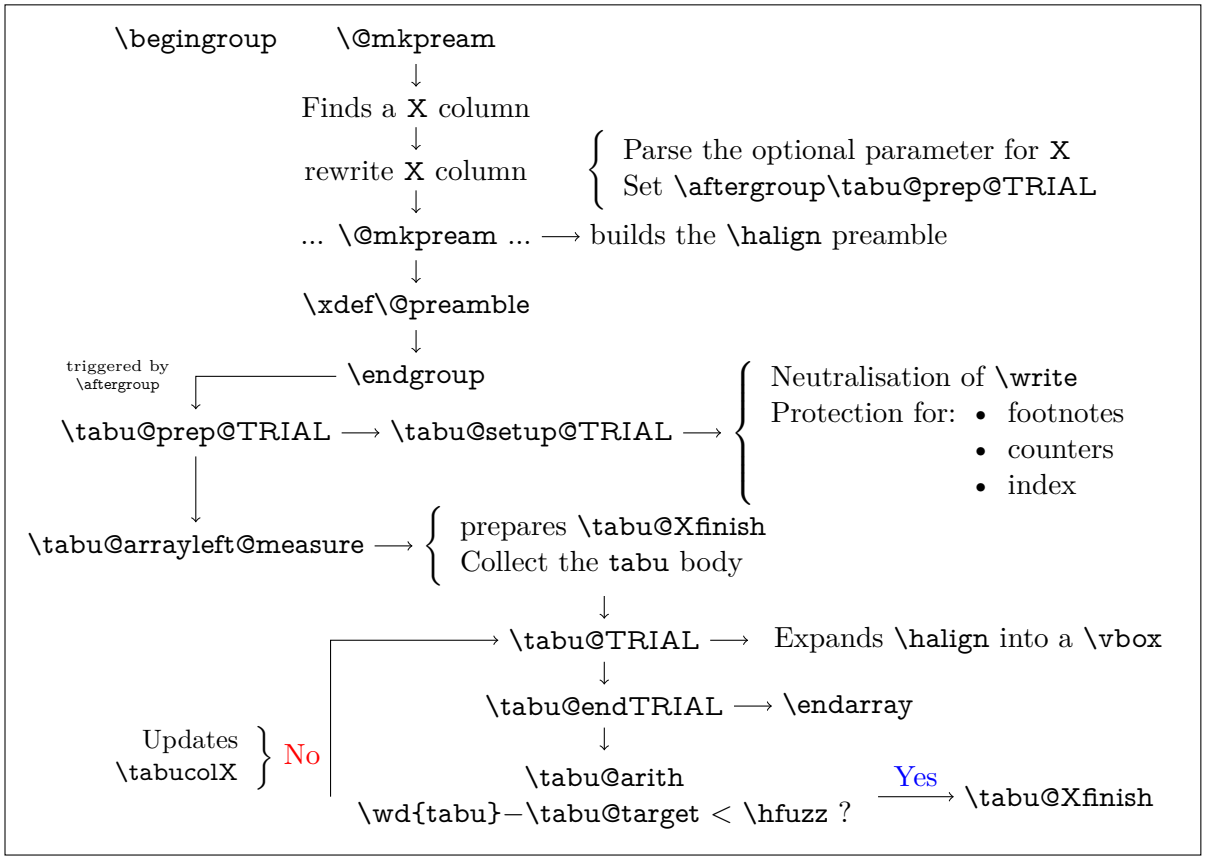
### tabu to with X column

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`.

When a `tabu` `X` column is found in the preamble by `\@mkpream`, `tabu` changes his strategy: the macro `\tabu@prep@TRIAL` is set to be expanded `\aftergroup` that is, just after the preamble (`\@preamble`) has been built. This macro does some setup for `tabu` trials to reach the target with variable `X` column widths and gobbles everything until the next `\bgroup` which corresponds to the `\vtop`, `\vbox` or `\vcenter` for the whole tabular. This part of `\@array` is stored into `\tabu@Xfinish` to be expanded after the last trial gave satisfaction to reach the `tabu` target. Then `\tabu@collect` is expanded to find the end of the `tabu` environment, temporarily storing the environment content into a token register.

The last part of `\@array` until `\halign` is expanded inside a `\vbox` which is stored into the box register `\tabu@box` for measuring purpose. `\halign` ends by `\endarray` which stops the `\tabu@box` as well, and then `\tabu@arith` is expanded to compute the gap between the width of `\tabu@box` and the target, and `\tabucolX` (the dimension that correspond to `X[1]`) is updated accordingly.

The trials are "protected" by `{\ifnum0=‘}\fi` : they occur in a group that will be closed at the very begining of `\tabu@Xfinish`, when the final tabular will be printed actually. This protection is absolutely necessary to be able to collect the environment body in the case of nested `tabu` with `X` columns. This is related to TeX mechanism of expansion inside `\halign` (TeX stops reading when it encounters a & `alignment tab character` and goes backward expanding anything that were not expanded before).



### 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:

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

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

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

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

## 7.5 tabu X column definition

\tabu@rewrite@X This macro is expanded by \@mkpream in case a X column is found.

\tabu@X@sum (a dimen) store the sum of the width coefficients. For the first X column found in the preamble, a special setup occurs:

- the default target (either \linewidth or \linegoal if available) is set if it has not been specified by the user.
- \@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.

```
254 \def\tabu@rewrite@X#1{\tabu@Xarg{#1}%
255   \iftabu@spread \tabu@rewrite@Xspread
256   \else          \tabu@rewrite@Xto
257   \fi
258 }% \tabu@rewrite@X
```

\tabu@rewrite@Xnested This macro replaces \tabu@rewrite@X when tabu makes a trial for X columns.

\tabu@rewrite@Xrestore This macro replaces \tabu@rewrite@X in the case of \usetabu.

```
259 \def\tabu@rewrite@Xnested#1{\def\tabucolX@spec{p{\tabucolX}}}
260 \def\tabu@rewrite@Xrestore#1{\tabu@Xarg{#1}\let\tabucolX@spec\tabu@temp}%
```

\tabu@rewrite@Xto The setup for trial is not the same in case of "tabu to" and "tabu spread".

The important thing is: \aftergroup\tabu@prep@TRIAL.

```
261 \def\tabu@rewrite@Xto{%
262   \iftabu@measuring % not the first X column found in preamble
```

```
263       \xdef\tabu@global@X {\tabu@global@X
264          \advance\tabu@X@sum \the\tabu@X@sum\relax}%
265    \else % first X column found in preamble
266       \tabu@measuringtrue
267       \ifdim\tabu@target=\z@
268          \setlength \tabu@target \tabudefaulttarget
269       \fi
270       \xdef\tabu@global@X{%
271          \tabu@X@sum \the\tabu@X@sum\relax
272          \tabu@target \the\tabu@target\relax}%
273       \let\tabu@halignto \relax
274       \aftergroup \tabu@prep@TRIAL
275    \fi
276 }% \tabu@rewrite@Xto
```

\tabu@rewrite@Xspread This macro is used instead of \tabu@rewrite@Xto for X columns with "tabu spread".

```
277 \def\tabu@rewrite@Xspread{% tabu spread with X columns: we need to store each coef
278    \iftabu@measuring % not the first X column found in preamble
279       \advance\tabu@X@cols \@ne
280       \expandafter\let\csname tabu@X\the\tabu@X@cols\endcsname \relax
281       \xdef\tabu@global@X {\tabu@global@X
282             \advance\tabu@X@cols\@ne
283             \advance\tabu@X@sum \the\tabu@X@sum\relax
284             \def\csname tabu@X\the\tabu@X@cols\endcsname{\strip@pt\tabu@X@sum}}%
285    \else % first X column found in preamble
286       \tabu@measuringtrue
287       \tabu@X@cols \@ne  \tabu@spreadtarget=\tabu@target
288       \setlength \tabu@target \tabudefaulttarget
289       \expandafter\let\csname tabu@X\the\tabu@X@cols\endcsname \relax
290       \xdef\tabu@global@X {%
291          \tabu@X@cols \@ne
292          \tabu@X@sum \the\tabu@X@sum\relax
293          \tabu@target \the\tabu@target\relax
294          \tabu@spreadtarget \the\tabu@spreadtarget\relax
295          \def\csname tabu@X\the\tabu@X@cols\endcsname{\strip@pt\tabu@X@sum}}%
296       \let\tabu@halignto \relax
297       \aftergroup \tabu@prep@TRIAL
298    \fi
299 }% \tabu@rewrite@Xspread
```

\tabu@Xarg A tedious (and fastidious) macro to parse the optional argument of X columns. The aim is to built \tabucolX@spec 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\tabucolX@spec

```
300 \def\tabu@Xarg#1{%
301    \ifx\\#1\\%    <shortcut when no option>
302       \tabu@X@sum \p@
303       \def\tabucolX@spec{p{\tabucolX}}%
304       \edef\tabu@temp{p{\the\tabucolX}}%  <required for \usetabu>
305    \else
306       \tabu@X@sum \z@
307       \let\tabucolX@align \@empty
308       \let\tabucolX@spec \@empty
309       \let\tabu@Xmath \relax
310       \tabu@Xparse {}#1\relax\@nnil
311    \fi
312 }% \tabu@Xarg
```

```
\def\tabu@Xparse#1{\futurelet\@let@token\tabu@Xtest}
\expandafter\def\expandafter\tabu@Xparsespace\space{\tabu@Xparse{}}
\def\tabu@Xtest{%
   \ifcase \ifx \@nnil\@let@token \z@ \else
           \ifx \relax\@let@token \m@ne\else
           \if ,\@let@token \m@ne\else
           \if p\@let@token 1\else
           \if m\@let@token 2\else
           \if b\@let@token 3\else
           \if l\@let@token 4\else
           \if c\@let@token 5\else
           \if r\@let@token 6\else
           \if .\@let@token 7\else
           \ifx \@sptoken\@let@token 8\else
           \if L\@let@token 9\else
           \if C\@let@token 10\else
           \if R\@let@token 11\else
           \ifcat $\@let@token 12\else
           13\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\relax
       \tabucolX@do \expandafter\remove@to@nnil
   \or \tabu@Xcolspec{p}%
   \or \tabu@Xcolspec{m}%
   \or \tabu@Xcolspec{b}%
   \or \tabu@Xalign{>{\raggedright}}%
   \or \tabu@Xalign{>{\centering}}%
   \or \tabu@Xalign{>{\raggedleft}}%
   \or \expandafter\tabu@Xcoef
   \or \expandafter\tabu@Xparsespace
   \or \tabu@Xalign{>{\RaggedRight}}%
   \or \tabu@Xalign{>{\Centering}}%
   \or \tabu@Xalign{>{\RaggedLeft}}%
   \or \let\tabu@Xmath=$\expandafter\tabu@Xparse
   \or \tabu@Xcoef{}%
   \else \expandafter\tabu@Xparse
   \fi
}% \tabu@Xtest
\def\tabu@Xalign#1{%
   \ifx \tabucolX@align\@empty \else \PackageWarning{tabu}
      {Duplicate horizontal alignment specification}\fi
   \def\tabucolX@align{#1}\expandafter\tabu@Xparse
}% \tabu@Xalign
\def\tabu@Xcolspec#1{%
   \ifx\tabucolX@spec\@empty\else \PackageWarning{tabu}
      {Duplicate vertical alignment specification}\fi
   \def\tabucolX@spec{#1}\expandafter\tabu@Xparse
}% \tabu@Xcolspec
\def\tabu@Xcoef#1{\def\@tempa{#1}%
   \afterassignment\tabu@Xc@ef \tabu@cnt
}% \tabu@Xcoef
\def\tabu@Xc@ef{%
   \advance\tabu@X@sum \@tempa\the\tabu@cnt\p@
   \tabu@Xparse{}%
}% \tabu@Xc@ef
\def\tabucolX@do{%
   \ifx\tabucolX@spec\@empty \def\tabucolX@spec{p}\fi
   \ifdim \tabu@X@sum=\z@ \tabu@X@sum \p@\fi
   \begingroup
      \ifx \tabu@Xmath\relax
```

```
371          \edef\tabucolX@spec##1{\tabucolX@spec{##1}}%
372       \else
373          \edef\tabucolX@spec##1{>{$}\tabucolX@spec{##1}<{$}}%
374       \fi
375       \edef\tabu@temp{\dimexpr\strip@pt\tabu@X@sum \tabucolX \relax}%
376       \toks@\expandafter{\tabucolX@align}%
377       \edef\x{%
378          \def\noexpand\tabu@temp{\the\toks@ \tabucolX@spec{\the\tabu@temp}}%
379          \def\noexpand\tabucolX@spec{\the\toks@ \tabucolX@spec{\tabu@temp}}%
380       }\expandafter\endgroup\x
381 }% \tabucolX@do
```

## 7.6 Before trials to reach the target

### Trial setup after \@mkpream

\tabu@prep@TRIAL

\@mkpream does its job inside a semi-simple group. At the end, \@preamble is expanded with \xdef, and the group is left: this triggers the expansion of \tabu@prep@TRIAL set \aftergroup by the first X column encountered in the preamble.

We \let \@halignto to \@empty: it's a measurement, and put some protections. A group is opened with the famous {\ifnum0=`}\fi and control is given to \tabu@arrayleft@measure.

```
382 \def\tabu@prep@TRIAL{%
383    \let\tabu@savedpreamble \@preamble
384    \def\savetabu {\tabu@saveX}%
385    {\ifnum0=`}\fi
386       \tabu@setup@TRIAL
387       \iftabu@spread % first trial only
388          \advance\tabu@X@cols \@ne
389          \expandafter\let\csname tabu@X\the\tabu@X@cols\endcsname \@undefined
390          \tabu@X@cols \z@ \tabu@naturalX \z@ \tabucolX \z@
391          \let\tabu@naturalX@max \z@
392          \let\tabu@naturalX@min \z@
393          \let\tabu@startpboxORI \@startpbox
394          \let\@startpbox \tabu@spread@startpbox
395       \else
396          \tabucolX = \tabu@target
397       \fi
398       \tabu@cnt \z@  % number of trials
399       \def\tabu@lasttry{\m@ne\p@}%
400       \tabu@arrayleft@measure
401 }% \tabu@prep@TRIAL
```

```
402 \def\tabu@setup@TRIAL{\tabu@global@X
403    \let\tabu@rewrite@X \tabu@rewrite@Xnested
404    \def\tabudefaulttarget{\linewidth}%
405    \def\@elt##1{\global\value{##1}\the\value{##1}\relax}%
406    \xdef\tabu@global@X {\cl@@ckpt}\let\@elt \relax
407    \hbadness\@M \hfuzz\maxdimen
408    \let\hbadness \@tempcnta
409    \let\hfuzz \@tempdima
410    \let\write \tabu@nowrite
411    \let\@footnotetext\@gobble
412    \let\tabu@saveX \@gobble
413    \tabu@TRIAL@hook
414 }% \tabu@setup@TRIAL
```

\tabuDisableCommands

```
415 \let\tabu@TRIAL@hook\@empty
```

```
\newcommand\tabuDisableCommands[1]{\g@addto@macro\tabu@TRIAL@hook{#1}}
```

\tabu@arrayleft@measure Measuring the whole tabular occurs just before \@arrayleft. Hence the name of the macro: \tabu@arrayleft@measure.

```
\def\tabu@arrayleft@measure#1\bgroup{%
  \def\tabu@Xfinish{\ifnum0=`{\fi}\tabu@global@X
        \let\@halignto\tabu@halignto \tabu@firstcline  #1\bgroup}%
  \toks@{\let\@preamble\tabu@savedpreamble}% <required for multicolumn>
  \tabu@collect
}% \tabu@arrayleft@measure
```

### Collecting the tabu body: required for X columns

\tabu@collect
\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.

```
\def\tabu@collect{\catcode`\^^@=13\def\tabu@stack{b}\tabu@collectbody}
\long\def\tabu@collectbody#1\end#2{%
   \edef\tabu@stack{\tabu@pushbegins #1\begin\end\expandafter\@gobble\tabu@stack}%
   \ifx\tabu@stack\@empty
      \toks@\expandafter{\the\toks@#1}\def\tabu@endenvir{\end{#2}}%
      \expandafter\tabu@TRIAL
   \else
      \toks@\expandafter{\the\toks@#1\end{#2}}%
      \expandafter\tabu@collectbody
   \fi
}% \tabu@collectbody
\long\def\tabu@pushbegins#1\begin#2{\ifx\end#2\else b\expandafter\tabu@pushbegins\fi}
```

## 7.7 Measuring the tabu

### One trial after another

\tabu@TRIAL
\tabu@endTRIAL

\halign is temporarily expanded inside a \vbox which is captured in \tabu@box.

At the end of the trial, we call \tabu@arith to compute the widths. \tabu@arith exits leaving \iftabu@measuring equal to \iftrue: a further trial is necessary, or equal to \iffalse: the target is reached, \tabu@Xfinish can print the tabu in a last expansion of \halign.

```
\def\tabu@TRIAL{\setbox\tabu@box \hbox\bgroup $\@arrayleft\vbox\bgroup \the\toks@
                                  \tabu@endTRIAL}% constant
\def\tabu@endTRIAL{\endarray$\egroup
   \iftabu@spread \tabu@spreadfalse
      \let\@startpbox \tabu@startpboxORI
      \tabu@spreadarith % <only once>
   \else \tabu@arith
   \fi
   \iftabu@measuring \tabu@measuringfalse
         \expandafter \tabu@TRIAL % <continue trials>
   \else
         \toks@\expandafter\expandafter\expandafter{%
               \the\expandafter\toks@ \tabu@endenvir}%
         \expandafter \tabu@Xfinish \the\toks@ % <we are then!>
   \fi
}% \tabu@endTRIAL
```

## The arithmetic of X columns

\tabu@arith
\tabu@spreadarith

Algorithms.

```
\def\tabu@arith{%
   \advance\tabu@cnt \@ne
   \ifdim \tabu@lasttry=\wd\tabu@box
      \tabu@message{\tabu@header Reached minimum width. Backing up}%
   \else
      \edef\tabu@lasttry{\the\wd\tabu@box}%
      \tabu@message{\tabu@message@arith}%
      \ifdim \dimexpr \wd \tabu@box - \tabu@target <\tabu@hfuzz
         \tabu@message{\tabu@message@reached}%
         \xdef\tabu@global@X {\tabu@target\the\tabu@target\relax \tabu@global@X
                              \tabucolX \the\tabucolX\relax}%
      \else
         \advance\tabucolX -\dimexpr(\wd\tabu@box-\tabu@target-\tabu@X@sum/65536)
                              *65536/\tabu@X@sum\relax
         \ifdim \tabucolX<\z@
            {\let\write\tabu@write\PackageWarning{tabu}
               {X Columns too narrow (table too wide)\MessageBreak}}%
               \tabucolX=1em%
         \else \tabu@measuringtrue
         \fi
      \fi
   \fi
}% \tabu@arith
\def\tabu@spreadarith{%
   \tabu@naturalX \tabu@naturalX@max\relax
   \tabucolX \tabu@naturalX@min\relax
   \tabu@message{\tabu@message@spread}%
   \ifdim \dimexpr \wd\tabu@box + \tabu@spreadtarget <\tabu@target
      \advance\tabu@spreadtarget \dimexpr\wd\tabu@box-\tabu@naturalX+\tabucolX\relax
      \ifdim \tabu@spreadtarget <\tabu@target
         \tabu@target = \tabu@spreadtarget
      \fi
      \tabu@message{\expandafter\tabu@msg@align \the\tabu@target {}{}{}{}{}\@@}%
      \tabu@message{\ifdim \tabu@spreadtarget>\tabu@target
                           (tabu) default target used.\fi}%
   \else
      \tabu@message{\expandafter\tabu@msg@align \the\tabu@target { }{}{}{}{}\@@}
      \tabu@message{(tabu)\ifdim\wd\tabu@box<\tabu@target spread too large\else
                                                 tabu spread is useless here\fi,
                     default target used.}%
   \fi
   \xdef\tabu@halignto {to\the\tabu@target}%
   \tabucolX \tabu@target
   \tabu@measuringtrue
}% \tabu@spreadarith
```

## Reporting in the .log file (debugshow option)

\tabu@message@arith

```
\def\tabu@message@arith{\tabu@header
   \expandafter\tabu@msg@align \the\tabucolX { }{ }{ }{ }{}\@@
   \expandafter\tabu@msg@align \the\wd\tabu@box { }{ }{ }{ }{}\@@
   \expandafter\tabu@msg@align \the\tabu@target { }{ }{ }{ }{}\@@
   \expandafter\tabu@msg@align \the\tabu@X@sum {}{}{}{}{}\@@
   \expandafter\tabu@msg@align \the\dimexpr(\tabu@target-\wd\tabu@box)
                                        *65536/\tabu@X@sum\relax {}{}{}{}{}\@@}
\def\tabu@message@spread{\tabu@spreadheader
   \expandafter\tabu@msg@align \the\tabu@spreadtarget {}{}{}{}{}\@@
   \expandafter\tabu@msg@align \the\wd\tabu@box {}{}{}{}{}\@@
   \expandafter\tabu@msg@align \the\tabucolX {}{}{}{}{}\@@
   \expandafter\tabu@msg@align \expandafter-\the\tabu@naturalX {}{}{}{}{}\@@}
\def\tabu@message@reached{\tabu@header
   \expandafter\tabu@msg@align \the\tabucolX { }{ }{ }{ }{}\@@
   ******** Reached Target : hfuzz=\tabu@hfuzz\space ********^^J}
\def\do#1{%
   \def\tabu@msg@align##1.##2##3##4##5##6##7##8##9\@@{%
   \ifnum##1<10 #1#1#1\else
   \ifnum##1<100 #1#1\else
   \ifnum##1<\@m #1\fi\fi\fi
   ##1.##2##3##4##5##6##7##8#1}%
   \def\tabu@header{%
      \ifnum\tabu@cnt=\@ne (tabu)
       Try#1#1#1#1tabu X#1#1#1#1#1#1tabu Width#1#1#1#1#1Target%
                   #1#1#1#1#1#1#1#1#1Coefs#1#1#1#1#1#1#1Update^^J\fi
       (tabu) \ifnum\tabu@cnt<10 #1\fi\the\tabu@cnt)#1#1}%
   \def\tabu@spreadheader{%
      (tabu) Try#1#1#1#1Spread#1#1#1tabu Width#1#1#1#1#1Min Nat.#1#1#1#1Nat.X%
                                                #1#1#1#1#1#1#1#1#1New Target^^J%
      (tabu)spread}%
}\do{ }
```

## Measuring the natural width for tabu spread

\tabu@spread@startpbox
\tabu@spread@endhbox

When "tabu spread" is used with X columns, the first trial must measure the natural width of the columns. Therefore, the \vbox, \vtop or \vcenter corresponding to the X columns definitions are temporarily changed into \hbox. More precisely, the natural width is measured with \vbox{\hbox{...}}.

For the furthur trials, the standard scheme for X column is used.

```
\def\tabu@spread@startpbox#1{\bgroup
   \in@{\tabucolX}{#1}%
   \ifin@
      \let\vss\relax\let\vfil\vss\let\vfill\vss\let\vskip\@tempskipa
      \hbox to\tabu@target\bgroup\aftergroup\tabu@spread@endhbox\hfil
   \else
      \expandafter\@gobble\tabu@startpboxORI{#1}% \@gobble \bgroup
   \fi
}% \tabu@spread@startpbox
\def\tabu@spread@endhbox{%
   \ifnum\badness>\@M %<text does not fit>
      \global\tabu@naturalX \tabu@target
   \else
      \setbox\tabu@box\lastbox
      \setbox\tabu@box\hbox{\unhbox\tabu@box}%
```

```
      \global\advance\tabu@X@cols \@ne
      \ifcsname tabu@X\the\tabu@X@cols\endcsname
         \global\advance\tabu@naturalX \wd\tabu@box
      \else
         \global\tabu@X@cols \@ne
         \global\tabu@naturalX \wd\tabu@box
      \fi
      \ifdim \tabu@naturalX@max<\tabu@naturalX
         \xdef\tabu@naturalX@max{\the\tabu@naturalX}%
      \fi
      \ifdim \tabu@naturalX@min<\dimexpr \wd\tabu@box * \tabu@X@sum /
                  (65536*\csname tabu@X\the\tabu@X@cols\endcsname)\relax
         \xdef\tabu@naturalX@min{\the\dimexpr \wd\tabu@box * \tabu@X@sum /
                     (65536*\csname tabu@X\the\tabu@X@cols\endcsname)\relax}%
      \fi
   \fi
   \box\tabu@box
   \egroup % end of \vtop (measure)
}% \tabu@spread@endhbox
```

## 7.8 Lines inside tabu

### Vertical lines

`\tabuvline@rewrite` `|` is defined as a new column type (only inside `tabu`) in order to add an optional argument: the width of the rule.

```
\newcommand*\tabuvline@rewrite[1][]{\tabuvline@arg{#1}%
                     \expandafter \NC@find \tabu@temp}
\def\tabu@vline#1{\vrule width#1}
```

`\tabuvline@arg` A tedious (and fastidious) macro to parse the optional argument of `|` vertical lines...

```
\def\tabuvline@arg#1{%
   \ifx\\#1\\%    <shortcut when no option>
      \def\tabu@temp{!{\tabu@vline\arrayrulewidth}}%
   \else \futurelet \tabu@temp \tabuvline@argi #1\p@\p@\@nnil \tabuvline@argiii
   \fi
}% \tabuvline@arg
\def\tabuvline@argi{%
   \let\tabu@color \@empty
   \ifcat A\noexpand\tabu@temp
         \@tempdima\arrayrulewidth
         \expandafter\tabu@getlinecolor
   \else \expandafter\tabuvline@argii
   \fi
}% \tabuvline@argi
\def\tabuvline@argii#1\@nnil{%
   \tabu@maybecolor \@tempdima #1 \@nnil
}% \tabuvline@argii
\def\tabuvline@argiii{%
   \edef\tabu@temp{!{\color@begingroup\tabu@color
                     \noexpand\tabu@vline{\the\@tempdima}\color@endgroup}}%
}% \tabuvline@argiii
```

`\tabu@lines` This macro is used in `\tabu@setup` to "mount" the character `|` as a column type.

```
{
\expandafter\let\csname NC@find@|\endcsname\relax
\expandafter\let\csname NC@rewrite@|\endcsname\relax
\xdef\tabu@lines{%
```

```
    \let\csname NC@find@|\endcsname \noexpand\tabuvlines@find
    \let\csname NC@rewrite@|\endcsname \noexpand\tabuvline@rewrite
}% \tabu@vline
}
\def\tabuvlines@find#1|{\NC@{#1}}
```

### Horizontal lines: \tabucline

\tabu@firstcline

```
\def\tabu@firstcline{%
   \tabu@firstclinetrue
   \let\tabu@everycr \everycr
   \def\tabu@restoreeverycr {\let\everycr\tabu@everycr}%
   \def\everycr{\afterassignment\tabu@restoreeverycr\@temptokena}% <for ialign>
   \tabu@everycr{\noalign{%
      \global\tabu@everycr{\noalign{%
            \global\tabu@everycr{}\global\tabu@firstclinefalse}}%
      \global\let\everycr\tabu@everycr}}%
}% \tabu@firstcline
\def\tabu@addphantomline{\tabu@everycr\expandafter{\the\tabu@everycr
                                                   \tabu@phantomline}}
```

\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. We built a new line to be inserted inside the **tabu**: this new line which contains the rule or leaders, is stored into **\toks@** (inside the **\noalign** group). When leaving the group **\toks@** is expanded.

```
\def\tabu@linedash{4pt}%   <default value>
\def\tabu@dashgap{4pt}%    <default value>
\def\tabucline{\noalign{\ifnum0=`}\fi\@ifstar
   {\@tempswatrue\tabu@cline}
   {\@tempswafalse\tabu@cline}}%
\newcommand*\tabu@cline[2][]{%
   \tabu@startstop{#2}\tabu@getlinespec{#1}\@multicnt\@ne
   \ifnum\tabu@start>\tabu@stop \the\toks@
   \else\ifx\tabu@xleaders\relax\the\toks@
   \else
      \toks@{\ifnum0=`{\fi}\noalign{\global\setbox\tabu@box \box\@arstrutbox}}%
      \iftabu@firstcline\if\tabu@align t\vskip-\ht\@arstrutbox\fi\fi
      \@whilenum\@multicnt<\tabu@start\do{\advance\@multicnt\@ne
                              \toks@\expandafter{\the\toks@ &}}%
      \loop
         \toks@\expandafter{\the\toks@ \omit }%
         \ifnum \@multicnt=\@ne
         \expandafter \tabu@add \csname tabu@tok@\the\@multicnt L\endcsname \@nil
         \fi
         \ifcase 0\if@tempswa\else\ifnum \@multicnt<\tabu@stop 1\fi\fi\relax
            \toks@\expandafter{\the\toks@ \color@begingroup}%
            \toks@\expandafter\expandafter\expandafter{\expandafter\the
                              \expandafter\toks@ \tabu@xleaders \color@endgroup}%
         \fi
         \ifcase 0\if@tempswa\else\ifnum \@multicnt<\tabu@stop 1\fi\fi %
         \expandafter \tabu@add \csname tabu@tok@\the\@multicnt R\endcsname \@nil
         \fi
         \ifnum\@multicnt<\tabu@stop
            \advance\@multicnt\@ne
            \if@tempswa \toks@\expandafter{\the\toks@ &}%
            \else       \toks@\expandafter{\the\toks@ \span}\fi
```

```
      \repeat
      \@whilenum\@multicnt<\tabu@nbcols\do{\advance\@multicnt\@ne
                                \toks@\expandafter{\the\toks@ &}}%
      \toks@\expandafter{\the\toks@ \cr \noalign{%
           \global\tabu@firstclinefalse \global\setbox\@arstrutbox \box\tabu@box}}%
      \the\toks@
   \fi\fi
}% \tabucline
\def\tabu@add #1\@nil{\toks@\expandafter{\the\toks@ #1\ifmmode$\fi}}
```

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

```
\def\tabu@startstop#1{\tabu@start@stop #1\relax 1-\tabu@nbcols\@nnil}
\def\tabu@start@stop #1-#2\@nnil{%
   \countdef\tabu@start100\countdef\tabu@stop101 % <inside noalign group>
   \@defaultunits \tabu@start \number0#1\relax\@nnil
   \@defaultunits \tabu@stop  \number0#2\relax\@nnil
   \ifnum\tabu@start>\tabu@nbcols\tabu@start=\tabu@nbcols\else
      \ifnum\tabu@start>\z@\else \tabu@start=\@ne        \fi\fi
   \ifnum\tabu@stop>\tabu@nbcols \tabu@stop\tabu@nbcols  \else
      \ifnum\tabu@stop>\z@\else  \tabu@stop=\tabu@nbcols \fi\fi
}% \tabu@start@stop
```

\tabu@getlinespec This macro parses the optional argument of \tabucline and check if it's a line specification (then \tabu@getline is expanded) or a \leaders specification (then \tabu@leaders is expanded).

```
\def\tabu@getlinespec#1{\let\tabu@xleaders \relax
   \@defaultunits \let\@tempa=#1 \relax\@nnil
   \ifx\@tempa\relax \let\tabu@xleaders \tabu@defaultleaders\else
   \ifx\@tempa\hbox  \tabu@defleaders{#1}\else
   \ifx\@tempa\box   \tabu@defleaders{#1}\else
   \ifx\@tempa\copy  \tabu@defleaders{#1}\else
   \ifcsname tabu@line@style@\detokenize{#1}\endcsname
      \csname tabu@line@style@\detokenize{#1}\endcsname
   \else \tabu@getline{#1\p@ on0pt off0pt}\fi\fi\fi\fi\fi
}% \tabu@getlinespec
\def\tabu@defleaders#1{%
   \def\tabu@xleaders{\xleaders\hbox{\lower.5\extrarowheight#1}\tabu@leaderfill}}
\def\tabucline@warn#1{\PackageWarning{tabu}
   {Undefined line syle: #1
   \MessageBreak Using default line style instead}%
   \let\tabu@xleaders \tabu@defaultleaders
}% \tabucline@warn
```

\tabu@getline This macro parses the optional argument of \tabucline (or the one of \tabulinestyle) and extract the thickness, the dash an gap specified. Default values assignments are done either.

```
\def\tabu@getline#1{\tabu@lineon #1 \@nil on\tabu@linedash \p@ \@nil\@nnil{#1}}
\def\tabu@maybecolor{\ifx\tabu@color\@empty \afterassignment\tabu@colortest
                     \else \@defaultunits\fi}
\def\tabu@lineon #1on#2\@nil#3\@nnil#4{%
   \let\tabu@color \@empty
   \@defaultunits \let\@tempa=#1 \relax\@nnil
   \ifcase 0\ifx o\@tempa 1\else\ifcat A\noexpand\@tempa 2\fi\fi\relax
      \tabu@maybecolor \@tempdima #1\arrayrulewidth \p@\@nnil
      \tabu@maybecolor \@tempdimb #2\p@ \@nnil
      \tabu@lineoff #1 on#2 off\tabu@dashgap \p@ \@nnil
   \or\@tempdima \arrayrulewidth
      \tabu@maybecolor \@tempdimb #2\p@ \@nnil
      \tabu@lineoff #1 on#2 off\tabu@dashgap \p@ \@nnil
```

```
686    \else
687       \tabu@maybecolor \@tempdima \arrayrulewidth #1\p@\@nnil
688       \ifx\tabu@color\@empty \tabucline@warn{#4}\else
689       \@tempdimb \z@
690       \tabu@lineoff off0pt \p@\@nnil \fi
691    \fi
692 }% \tabu@lineon
693 \def\tabu@lineoff #1off#2\@nnil{%
694    \tabu@maybecolor \@tempdimc #2\p@ \@nnil
695    \ifdim \@tempdimb=\z@
696       \ifdim \@tempdimc>\z@ \@tempdimb \tabu@dashgap\relax\fi\fi
697    \ifdim \@tempdimc=\z@
698       \ifdim \@tempdimb>\z@ \@tempdimc \tabu@linedash\relax\fi\fi
699    \ifdim \@tempdima<\z@ \else
700    \ifdim \@tempdimb<\z@ \else
701    \ifdim \@tempdimc<\z@ \else
702       \edef \tabu@xleaders{\tabu@color\xleaders
703          \ifdim\@tempdimc>\z@
704             \hbox\bgroup \kern\the\dimexpr\@tempdimc/2\relax\fi
705          \noexpand\iftabu@firstcline
706          \vrule depth\dimexpr\the\@tempdima
707                 \ifdim\@tempdimb>\z@ width\the\@tempdimb\fi
708          \noexpand\else
709          \vrule height\dimexpr-\extrarowheight+\the\@tempdima
710                 depth\dimexpr\extrarowheight
711                \ifdim\@tempdimb>\z@ width\the\@tempdimb\fi
712          \noexpand\fi
713          \ifdim\@tempdimc>\z@
714             \kern\the\dimexpr\@tempdimc/2\egroup\fi
715          \tabu@leaderfill}%
716    \fi\fi\fi
717 }% \tabu@lineoff
```

\tabu@colortest

```
718 \def\tabu@colortest{\futurelet\tabu@temp\tabu@linecolor}
719 \def\tabu@linecolor{%
720    \ifcase 0\if ,\noexpand\tabu@temp\else
721             \ifx\relax\tabu@temp\else
722             \ifx \@sptoken\tabu@temp1\else
723             \ifcat A\noexpand\tabu@temp2\else
724             3\fi\fi\fi\fi\relax
725    \def\tabu@next##1{\futurelet\tabu@temp\tabu@linecolor}%
726    \or\def\tabu@next{\tabu@gobblespace{\futurelet\tabu@temp\tabu@linecolor}}%
727    \or\let\tabu@next\tabu@getlinecolor
728    \else\expandafter\remove@to@nnil
729    \fi \tabu@next
730 }% \tabu@linecolor
731 \def\tabu@getlinecolor#1\p@{%
732    \edef\@tempa{\zap@space #1 \@empty}%
733    \ifcsname\string\color@\@tempa\endcsname
734       \edef\tabu@color{\noexpand\noexpand\noexpand\color{\@tempa}}%  \set@color
735    \fi\remove@to@nnil
736 }% \tabu@getlinecolor
```

\tabulinestyle \tabulinestyle{style=spec.}

```
737 \def\tabulinestyle#1{\@for\@tempa:=#1\do{\expandafter\tabu@linestyle\@tempa==\@nil}}
738 \def\tabu@linestyle#1=#2=#3\@nil{%
739    \begingroup \tabu@getlinespec {#2}\expandafter\gdef
```

```
740         \csname tabu@line@style@\detokenize{#1}\expandafter\endcsname
741                     \expandafter{\expandafter\def\expandafter\tabu@xleaders
742                                                 \expandafter{\tabu@xleaders}}%
743   \endgroup
744 }% \tabu@linestyle
745 \expandafter\def \csname tabu@line@style@\endcsname {%
746                                       \let\tabu@xleaders \tabu@defaultleaders}%
747 \def\tabu@defaultleaders{\leaders
748   \iftabu@firstcline
749     \vrule  depth \arrayrulewidth
750   \else
751     \vrule  height\dimexpr-\extrarowheight+\arrayrulewidth
752             depth \extrarowheight
753   \fi
754   \tabu@leaderfill}
755 \let\tabu@leaderfill \hfil
```

## 7.9 Verbatim inside tabu with X columns

\tabu@sanitizetext

```
756 {\catcode32=13\relax\catcode`\^^@=13\relax
757 \gdef\tabu@verb{\@sanitize\makeatletter\catcode`\^=7\edef\^{\string^}%
758 \catcode32=13\let =\ \catcode`\^^@=13\def^^@{\par}\endlinechar\m@ne}%
759 }
760 \newcommand\tabu@sanitizetext[1][\ttfamily]{\begingroup
761                         \tabu@verb #1\tabu@s@nitizetext}
762 \long\def\tabu@s@nitizetext#1{\@makeother\{\@makeother\}%
763             \everyeof{\noexpand}\scantokens{#1}\endgroup}
```

## 7.10 Numbers in tabu

### \tabudecimal

\tabudecimal

```
764 \def\tabu@tabudecimal#1{%
765   \def\tabu@decimal{#1}\@temptokena{}%
766   \let\tabu@getdecimal@ \tabu@getdecimal@ignorespaces
767   \tabu@scandecimal
768 }% \tabudecimal
769 \def\tabu@scandecimal{\futurelet \tabu@temp \tabu@getdecimal@}
770 \def\tabu@skipdecimal#1{#1\tabu@scandecimal}
771 \def\tabu@getdecimal@ignorespaces{%
772   \ifcase 0\ifx\tabu@temp\ignorespaces\else
773             \ifx\tabu@temp\@sptoken1\else
774             2\fi\fi\relax
775     \let\tabu@getdecimal@ \tabu@getdecimal
776     \expandafter\tabu@skipdecimal
777   \or \expandafter\tabu@gobblespace\expandafter\tabu@scandecimal
778   \else \expandafter\tabu@skipdecimal
779   \fi
780 }% \tabu@getdecimal@ignorespaces
781 \def\tabu@get@decimal#1{\@temptokena\expandafter{\the\@temptokena #1}%
782                         \tabu@scandecimal}
783 \def\do#1{%
784   \def\tabu@get@decimalspace#1{%
785     \@temptokena\expandafter{\the\@temptokena #1}\tabu@scandecimal}%
786 }\do{ }
```

\tabu@getdecimal

```
\def\tabu@getdecimal{%
   \ifcase    0\ifx 0\tabu@temp\else
               \ifx 1\tabu@temp\else
               \ifx 2\tabu@temp\else
               \ifx 3\tabu@temp\else
               \ifx 4\tabu@temp\else
               \ifx 5\tabu@temp\else
               \ifx 6\tabu@temp\else
               \ifx 7\tabu@temp\else
               \ifx 8\tabu@temp\else
               \ifx 9\tabu@temp\else
               \ifx .\tabu@temp\else
               \ifx ,\tabu@temp\else
               \ifx -\tabu@temp\else
               \ifx +\tabu@temp\else
               \ifx e\tabu@temp\else
               \ifx E\tabu@temp\else
               \ifx\tabu@cellleft\tabu@temp1\else
               \ifx\ignorespaces\tabu@temp1\else
               \ifx\@sptoken\tabu@temp2\else
            3\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\relax
       \expandafter\tabu@get@decimal
   \or \expandafter\tabu@skipdecimal
   \or \expandafter\tabu@get@decimalspace
   \else\expandafter\tabu@printdecimal
   \fi
}% \tabu@getdecimal
\def\tabu@printdecimal{%
   \edef\tabu@temp{\the\@temptokena}%
   \ifx\tabu@temp\@empty\else
   \ifx\tabu@temp\space\else
      \expandafter\tabu@decimal\expandafter{\the\@temptokena}%
   \fi\fi
}% \tabu@printdecimal
```

## 7.11 \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*⟩.

\tabu@saveX \tabu@saveX replaces \savetabu inside a tabu with X columns. The X columns widths have to be stored, that is, the value in points of \tabucolX.

```
\newcommand*\savetabu[1]{\noalign{%
   \ifx\\#1\\\tabu@savewarn{}{The tabu will not be saved}\else
      \@ifundefined{tabu@save@\string#1}{}{\tabu@savewarn{#1}{Overwritting}}%
      {\toks@\expandafter{\tabu@saved}%
      \expandafter\xdef\csname tabu@saved@\string#1\endcsname{%
         \tabu@target\the\tabu@target\relax
         \the\toks@}}%
   \fi}%
}% \savetabu
```

```
\def\tabu@saveX#1{\noalign{%
   \ifx\\#1\\\tabu@savewarn{}{The tabu will not be saved}\else
      \@ifundefined{tabu@saved@\string#1}{}{\tabu@savewarn{#1}{Overwritting}}%
      {\toks@\expandafter{\tabu@saved}%
      \expandafter\xdef\csname tabu@saved@\string#1\endcsname{%
```

```
835          \tabucolX\the\tabucolX\relax
836          \tabu@target\the\tabu@target\relax
837          \the\toks@}}%
838    \fi}%
839 }% \tabu@saveX
```

\tabu@savewarn Info for overwritting when \savetabu is used.

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

```
840 \def\tabu@savewarn#1#2{\PackageInfo{tabu}
841    {User-name `#1' already used for \string\savetabu
842    \MessageBreak #2}}%
843 \def\tabu@saveerr#1{\PackageError{tabu}
844    {User-name `#1' is unknown for \string\usetabu
845    \MessageBreak I cannot restore an unknown preamble!}\@ehd}
```

## 7.12 \rowfont

### Setting font and alignment specification

\rowfont \rowfont uses the control sequences \tabu@celllalign, \tabu@cellleft, \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 sequence expand to \@empty.

\tabu@cellleft 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@prepnext@tok.

```
846 \def\tabu@rowfont{\noalign{\ifnum0=`}\fi\tabu@row@font}
847 \newcommand*\tabu@row@font[2][]{%
848    \global\tabu@everycr@tok=\everycr
849    \global\let\tabu@@cellleft \tabu@cellleft
850    \global\let\tabu@@cellright \tabu@cellright
851    \gdef\tabu@cellfont{#2}%
852    \ifcsname tabu@cell@#1\endcsname  % row alignment
853          \csname tabu@cell@#1\endcsname \fi
854    \toks@\expandafter {\tabu@cellleft\tabu@cellfont}% inside \noalign group ok
855    \xdef\tabu@cellleft {\the\toks@}%
856    \global\everycr\expandafter {\the\everycr\tabu@rowfont@reset}%
857    \ifnum0=`{\fi}% end of noalign group
858 }% \rowfont
```

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

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

```
859 \def\tabu@rowfont@reset{%
860    \noalign{%
861       \global\let\tabu@cellleft \tabu@@cellleft
862       \global\let\tabu@cellright \tabu@@cellright
863       \global\let\tabu@cellfont \@empty
864       \global\let\tabu@celllalign \@empty
865       \global\let\tabu@cellralign \@empty
866       \global\everycr=\tabu@everycr@tok
867       \global\tabu@everycr@tok{}%
868    }%
869 }% \tabu@rowfont@reset
```

## Preparing stuff to be able to use \rowfont

\tabu@prepnext@tok

\tabu@prepnext@tok will replace \prepnext@tok (in array.sty): its purpose is to add the control sequences \tabu@celllalign, \tabu@cellleft, \tabu@cellright and \tabu@cellralign at the right position in the "preamble" for \halign. Those control sequences are not inserted directly into the preamble, but by the means of the user-tokens placed there by the array package.

The package array defines a macro \prenext@tok to initialize each user-token inserted at both side of each "normal" column. For "special'' @ and ! columns, there is only one token.

$$\underbrace{>\{\backslash bfseries\backslash color\{red\}\}}_{\backslash toks<i>} \quad r \quad \underbrace{<\{\backslash color\{black\}\backslash,\backslash\$\}}_{\backslash toks<i+1>}$$

When a column is inserted in the tabular preamble (\@preamble), the TEX 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 \prepnext@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 \prepnext@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\@ne and initialize the token to prepare (\toks\count@ =\toks< $i+1$ >) to an empty one.

```
\newif\iftabu@cellright
\AtBeginDocument{\let\array@prepnext@tok \prepnext@tok }% original definition
\def\tabu@prepnext@tok{%
   \ifnum \count@<\z@  % <first initialisation>
      \@tempcnta \@M   % <not initialized by array.sty>
      \tabu@nbcols\z@
      \gdef\tabu@global@temp{\tabu@nbcols\z@}%
      \expandafter\let\csname tabu@tok@1L\endcsname \relax
      \tabu@cellrightfalse
   \else
      \ifcase \numexpr \count@-\@tempcnta \relax % (case 0): prev. token is left
         \advance \tabu@nbcols\@ne
         \expandafter\let\csname tabu@tok@\the\tabu@nbcols R\endcsname \relax
         \expandafter\gdef\expandafter\tabu@global@temp\expandafter{%
                           \tabu@global@temp \advance\tabu@nbcols\@ne}%
         \iftabu@cellright % before-previous token is right and is finished
            \tabu@cellrightfalse % <only once>
            \tabu@savetok R\tabu@preptokenright
         \fi
```

```
889         \ifnum \tabu@nbcols=\@ne
890            \tabu@savetok L% LEFT token is always empty unless >{...}
891         \fi
892         \tabu@preptokenleft
893      \or                     % (case 1) previous token is right
894         \tabu@savetok R\tabu@cellrighttrue
895      \else % special column: do not change the token
896         \ifnum \tabu@nbcols>\z@ %special column: always on the right of normal one
897            \tabu@savetok R%
898         \else % unless this is the very first column (\tabu@nbcols=0)
899            \advance\tabu@nbcols\@ne \tabu@savetok L\advance\tabu@nbcols\m@ne
900         \fi
901         \iftabu@cellright    % before-previous token is right
902            \tabu@cellrightfalse
903            \tabu@preptokenright
904         \fi
905      \fi % \ifcase
906   \fi
907   \array@prepnext@tok
908 }% \tabu@prepnext@tok
909 \def\tabu@preptokenright{%
910   \advance \count@ \m@ne
911   \toks\count@\expandafter {\the\toks\count@ \tabu@cellright \tabu@cellralign}%
912   \advance \count@ \@ne
913 }% \tabu@preptokenright
914 \def\tabu@preptokenleft{\toks\count@\expandafter {\expandafter\tabu@celllalign
915                                                   \the\toks\count@ \tabu@cellleft}%
916 }% \tabu@preptokenleft
917 \def\tabu@savetok#1{\begingroup
918   \expandafter\tabu@savet@k\csname tabu@tok@\the\tabu@nbcols #1\endcsname
919 }% \tabu@savetok
920 \def\tabu@savet@k#1{%
921   \@temptokena\toks\count@
922   \ifx#1\relax\else \@temptokena\expandafter\expandafter
923                     \expandafter{\expandafter#1\the\@temptokena}%
924   \fi
925   \@temptokena\expandafter{\expandafter\def\expandafter#1\expandafter{%
926                                                   \the\@temptokena}}%
927   \toks@\expandafter\expandafter\expandafter{\expandafter\tabu@global@temp
928                                                   \the\@temptokena}%
929   \xdef\tabu@global@temp{\the\toks@}%
930   \expandafter\endgroup \the\@temptokena
931 }% \tabu@savetok
```

## Neutralisation of glues and alignment modification

\tabu@cellleft
\tabu@celllalign
\tabu@cellright
\tabu@cellralign
\tabu@cellfont

First initialisation to \@empty.

```
932 \let\tabu@cellleft\@empty
933 \let\tabu@cellright\@empty
934 \def\tabu@celllalign{\tabu@cellleft}% row font spec. applies to pre-column material
935 \let\tabu@cellralign\@empty
936 \let\tabu@cellfont\@empty
```

\tabu@cell@l
\tabu@cell@c
\tabu@cell@r
\tabu@cell@j

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.

```
937 \def\tabu@cell@l{% force alignment to left
```

```
938    \gdef\tabu@celllalign{\tabu@removehfil
939                          \raggedright\arraybackslash
940                          \tabu@cellleft}%
941    \gdef\tabu@cellralign{\tabu@flush1\tabu@ignorehfil}%
942    \toks@\expandafter{\tabu@cellleft\raggedright\arraybackslash}% local
943    \xdef\tabu@cellleft{\the\toks@}%
944 }% \tabu@cell@l
945 \def\tabu@cell@c{% force alignment to center
946    \gdef\tabu@celllalign{\tabu@removehfil
947                          \centering\arraybackslash
948                          \tabu@flush{.5}\tabu@cellleft}%
949    \gdef\tabu@cellralign{\tabu@flush{.5}\tabu@ignorehfil}%
950    \toks@\expandafter{\tabu@cellleft\centering\arraybackslash}% local
951    \xdef\tabu@cellleft{\the\toks@}%
952 }% \tabu@cell@c
953 \def\tabu@cell@r{% force alignment to right
954    \gdef\tabu@celllalign{\tabu@removehfil
955                          \raggedleft\arraybackslash
956                          \tabu@flush1\tabu@cellleft}%
957    \gdef\tabu@cellralign{\tabu@ignorehfil}%
958    \toks@\expandafter{\tabu@cellleft\raggedleft\arraybackslash}% local
959    \xdef\tabu@cellleft{\the\toks@}%
960 }% \tabu@cell@r
961 \def\tabu@cell@j{% force justification (for p, m, b columns)
962    \gdef\tabu@celllalign{\tabu@justify\tabu@cellleft}%
963    \global\let\tabu@cellralign\@empty
964    \toks@\expandafter{\tabu@cellleft\tabu@justify}% local (noalign grp)
965    \xdef\tabu@cellleft{\the\toks@}%
966 }% \tabu@cell@j
967 \def\tabu@justify{%
968    \leftskip\z@skip \@rightskip\leftskip \rightskip\@rightskip
969    \parfillskip\@flushglue
970 }% \tabu@justify
971 %% ragged2e settings
972 \def\tabu@cell@L{% force alignment to left (ragged2e)
973    \gdef\tabu@celllalign{\tabu@removehfil
974                          \RaggedRight\arraybackslash
975                          \tabu@cellleft}%
976    \gdef\tabu@cellralign{\tabu@flush1\tabu@ignorehfil}%
977    \toks@\expandafter{\tabu@cellleft\RaggedRight\arraybackslash}%
978    \xdef\tabu@cellleft{\the\toks@}%
979 }% \tabu@cell@L
980 \def\tabu@cell@C{% force alignment to center (ragged2e)
981    \gdef\tabu@celllalign{\tabu@removehfil
982                          \Centering\arraybackslash
983                          \tabu@flush{.5}\tabu@cellleft}%
984    \gdef\tabu@cellralign{\tabu@flush{.5}\tabu@ignorehfil}%
985    \toks@\expandafter{\tabu@cellleft\Centering\arraybackslash}%
986    \xdef\tabu@cellleft{\the\toks@}%
987 }% \tabu@cell@C
988 \def\tabu@cell@R{% force alignment to right (ragged2e)
989    \gdef\tabu@celllalign{\tabu@removehfil
990                          \RaggedLeft\arraybackslash
991                          \tabu@flush1\tabu@cellleft}%
992    \gdef\tabu@cellralign{\tabu@ignorehfil}%
993    \toks@\expandafter{\tabu@cellleft\RaggedLeft\arraybackslash}%
994    \xdef\tabu@cellleft{\the\toks@}%
995 }% \tabu@cell@R
```

```
\def\tabu@cell@J{% force justification (ragged2e)
   \gdef\tabu@celllallign{\justifying\arraybackslash\tabu@cellleft}%
   \global\let\tabu@cellralign\@empty
   \toks@\expandafter{\tabu@cellleft\justifying\arraybackslash}%
   \xdef\tabu@cellleft{\the\toks@}%
}% \tabu@cell@J
\def\tabu@flush#1{%
   \iftabu@colortbl      % colortbl uses \hfill rather than \hfil
      \hskip \ifnum\currentgrouptype>13 \stretch{#1}%
      \else\ifdim#1pt<1pt \tabu@cellskip
      \else \stretch{#1}
      \fi\fi \relax
   \else                 % array.sty
      \ifnum \currentgrouptype>13\relax
         \hfil \hskip1sp
      \fi
   \fi
}% \tabu@flush
\AtBeginDocument{%
   \@ifpackageloaded{ragged2e}
      {}
      {\let\tabu@cell@L \tabu@cell@l
       \let\tabu@cell@R \tabu@cell@r
       \let\tabu@cell@C \tabu@cell@c
       \let\tabu@cell@J \tabu@cell@j
      }%
}% AtBeginDocument
```

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

```
\newskip\tabu@cellskip
\let\tabu@hfil\hfil
\let\tabu@hfill\hfill
\let\tabu@hskip\hskip
\def\tabu@removehfil{%
   \iftabu@colortbl
      \unkern \tabu@cellskip = \lastskip
      \ifnum\gluestretchorder\tabu@cellskip = \tw@ \hskip-\tabu@cellskip
      \else \tabu@cellskip = \z@skip
      \fi
   \else
      \ifdim\lastskip=1sp\unskip\fi
      \ifnum\gluestretchorder\lastskip = \@ne
         \hfilneg % \hfilneg for array.sty but not for colortbl...
      \fi
   \fi
}% \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.

```
\def\tabu@ignorehfil{%
   \aftergroup\tabu@nohfil
}% \tabu@ignorehfil
\def\tabu@nohfil{% \hfil -> do nothing + restore original \hfil
   \def\hfil{\let\hfil\tabu@hfil}%   local to (alignment template) group
}% \tabu@nohfil
\AtBeginDocument{%
   \@ifpackageloaded{colortbl}
```

```
1048    {%
1049    \def\tabu@nohfil{%
1050       \def\hfil{\let\hfil\tabu@hfil}% local to (alignment template) group
1051       \def\hfill{\let\hfill\tabu@hfill}% (colortbl uses \hfill) pfff...
1052       \def\hskip##1\relax{\let\hskip\tabu@hskip}}% local
1053    }% @ifpackageloaded colortbl
1054    {}%
1055 }% AtBeginDocument
```

## 7.13 Utilities

### tabu \fbox

\tabu@fbox works exactly like LaTeX \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.

```
1056 \def\tabu@fbox{%
1057    \leavevmode
1058    \let\color@bgroup\bgroup
1059    \def\color@egroup{\endgraf\egroup}%
1060    \afterassignment\tabu@begin@fbox
1061       \setbox\@tempboxa \hbox
1062 }% \tabu@fbox
1063 \def\tabu@begin@fbox{\color@bgroup\kern\fboxsep\aftergroup\tabu@end@fbox}
1064 \def\tabu@end@fbox{\kern\fboxsep\color@egroup\@frameb@x\relax}
```

### \centering, \raggedright, \raggedleft

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

```
1065 \expandafter\def\expandafter\tabu@centering\expandafter{%
1066                                                 \centering\arraybackslash}
1067 \expandafter\def\expandafter\tabu@raggedleft\expandafter{%
1068                                                 \raggedleft\arraybackslash}
1069 \expandafter\def\expandafter\tabu@raggedright\expandafter{%
1070                                                 \raggedright\arraybackslash}
1071 \AtBeginDocument{%
1072 \expandafter\def\expandafter\tabu@trivlist\expandafter{%
1073    \expandafter\let\expandafter\\\expandafter\@centercr\@trivlist}%
1074 }%
```

## 7.14 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 implementation.

```
1075 \def\tabu@fix@arrayright{%
1076    \@ifpackageloaded{arydshln}
1077    {%
1078    \@ifpackageloaded{colortbl}
1079       {%% colortbl + arydshln
1080       \def\tabu@endarray{%
1081          \adl@endarray \egroup \adl@arrayrestore \CT@end \egroup %<original>
1082          \@arrayright     % <FC>
```

```
1083          \gdef\@preamble{}% <FC>
1084                  }}% \endarray
1085       {%% arydshln / no colortbl
1086       \def\tabu@endarray{%
1087          \adl@endarray \egroup \adl@arrayrestore \egroup %<original>
1088          \@arrayright     % <FC>
1089          \gdef\@preamble{}% <FC>
1090                  }}% \endarray
1091    }%
1092    {%
1093    \@ifpackageloaded{colortbl}
1094       {%% colortbl / no arydshln
1095       \def\tabu@endarray{%
1096          \crcr \egroup \egroup
1097          \@arrayright  % <FC>
1098          \gdef\@preamble{}\CT@end
1099                  }}%
1100       {\PackageWarning{tabu}
1101          {\string\@arrayright\space is missing from the
1102          \MessageBreak definition of \string\endarray.
1103          \MessageBreak Comptability with delarray.sty is broken.}}%
1104    }%
1105 }% \tabu@fix@arrayright
```

### arydshln @ columns

```
1106 \def\tabu@adl@xarraydashrule#1#2#3{%
1107      \ifnum\@lastchclass=\adl@class@start\else
1108      \ifnum\@lastchclass=\@ne\else
1109      \ifnum\@lastchclass=5 \else % <FC> @-arg (class 5) and !-arg (class 1)
1110              \adl@leftrulefalse \fi\fi           % must be treated the same
1111      \fi
1112      \ifadl@zwvrule\else \ifadl@inactive\else
1113              \@addtopreamble{\vrule\@width\arrayrulewidth
1114                      \@height\z@ \@depth\z@}\fi \fi
1115      \ifadl@leftrule
1116              \@addtopreamble{\adl@vlineL{\CT@arc@}{\adl@dashgapcolor}%
1117                      {\number#1}#3}%
1118      \else   \@addtopreamble{\adl@vlineR{\CT@arc@}{\adl@dashgapcolor}%
1119                      {\number#2}#3}
1120      \fi
1121 }% \tabu@adl@xarraydashrule
```

### arydshln 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 inconditionnally, and \ifhmode applies only to \nobreak (misplaced \fi in arydshln definition).

```
1122 \def\tabu@adl@act@endpbox{%
1123    \unskip \ifhmode \nobreak \fi    \@finalstrut \@arstrutbox
1124    \egroup
```

```
1125    \adl@colhtdp \box\adl@box \hfil
1126 }% \tabu@adl@act@endpbox

1127 ⟨/package⟩
```

# 8 History

## [2010/11/15 v1.2]

- Improvement in parameters parsing for optional parameters (| and \tabucline).
- Modification / optimization in \tabu@prepnext@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.