

t a b u

Flexible \LaTeX tabulars

FC

2010/11/18 – version 1.3

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 than `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 modifications¹ which are loaded only inside a `tabu` environment.

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

Contents

| | | |
|----------|--|-----------|
| 1 | The <code>tabu</code> environment | 3 |
| 1.1 | <code>tabu</code> , <code>tabu to</code> and <code>tabu spread</code> | 3 |
| 1.2 | <code>tabu X</code> columns | 3 |
| 1.2.1 | <i>X columns with “<code>tabu spread</code>”</i> | |
| 1.2.2 | <i>Multicolumn in <code>tabu</code></i> | |
| 1.2.3 | <i>Nesting <code>tabus</code> with X columns</i> | |
| 1.3 | Lines inside <code>tabu</code> | 6 |
| 1.3.1 | <i>First important remarks</i> | |
| 1.3.2 | <i>Vertical lines: <code> </code> has an optional parameter</i> | |
| 1.3.3 | <i>More style for horizontal lines: <code>\tabucline</code></i> | |
| 1.4 | <code>tabu</code> in math mode | 8 |
| 1.5 | Verbatim inside <code>tabu</code> with X columns | 8 |
| 1.6 | Printing numbers inside <code>tabu</code> with <code>numprint</code> and <code>siunitx</code> | 9 |
| 1.6.1 | <i>Just make it easy !</i> | |
| 1.6.2 | <i>You should know how it works...</i> | |
| 1.7 | <code>delarray</code> shortcuts | 10 |
| 2 | Save the <code>tabu</code> preamble and X columns widths: <code>\savetabu</code>, <code>\usetabu</code> | 10 |
| 3 | Change the font and the alignment in one row: <code>\rowfont</code> | 11 |
| 4 | Differences between <code>tabu</code>, <code>tabular</code> and <code>tabularx</code> | 11 |
| 4.1 | Paragraph indentation | 11 |
| 4.2 | Custom environments | 12 |
| 4.3 | Inversion of tokens | 12 |
| 4.4 | Improved process for rewriting columns | 12 |
| 5 | The package options | 13 |
| 5.1 | The <code>debugshow</code> package option | 13 |
| 5.2 | The <code>linegoal</code> package option | 13 |
| 5.3 | The <code>delarray</code> package option | 14 |
| 5.4 | The <code>light</code> package option | 14 |
| 6 | Corrections of some bugs (available only inside <code>tabu</code>) | 14 |
| 6.1 | Correction for <code>colortbl</code> and <code>arydshln</code> : compatibility with <code>delarray</code> | 14 |
| 6.2 | Correction for <code>arydshln</code> : <code>@</code> 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 | <code>\tabu</code> and <code>\endtabu</code> | 19 |
| 7.4 | Flow chart of expansion | 21 |
| | 7.4.1 General case: <i>tabu</i> , <i>tabu to</i> and <i>tabu spread</i> | |
| | 7.4.2 <i>tabu to</i> with <i>X</i> column | |
| | 7.4.3 <i>tabu spread</i> with <i>X</i> column | |
| 7.5 | <i>tabu</i> <i>X</i> column definition | 23 |
| 7.6 | Before trials to reach the target | 25 |
| | 7.6.1 Trial setup after <code>\@mkpream</code> | |
| | 7.6.2 Collecting the <i>tabu</i> body: required for <i>X</i> columns | |
| 7.7 | Measuring the <i>tabu</i> | 27 |
| | 7.7.1 One trial after another | |
| | 7.7.2 The arithmetic of <i>X</i> columns | |
| | 7.7.3 Reporting in the <i>.log</i> file (<i>debugshow</i> option) | |
| | 7.7.4 Measuring the natural width for <i>tabu spread</i> | |
| 7.8 | Lines inside <i>tabu</i> | 29 |
| | 7.8.1 Vertical lines | |
| | 7.8.2 Horizontal lines: <code>\tabucline</code> | |
| 7.9 | Verbatim inside <i>tabu</i> with <i>X</i> columns | 34 |
| 7.10 | Numbers in <i>tabu</i> | 34 |
| | 7.10.1 <code>\tabudecimal</code> | |
| 7.11 | <code>\savetabu</code> | 35 |
| 7.12 | <code>\rowfont</code> | 36 |
| | 7.12.1 Setting font and alignment specification | |
| | 7.12.2 Preparing stuff to be able to use <code>\rowfont</code> | |
| | 7.12.3 Neutralisation of glues and alignment modification | |
| 7.13 | Utilities | 41 |
| | 7.13.1 <i>tabu</i> <code>\fbox</code> | |
| | 7.13.2 <code>\centering</code> , <code>\raggedright</code> , <code>\raggedleft</code> | |
| 7.14 | Corrections | 41 |
| | 7.14.1 <i>delarray</i> compatibility fix for <i>colortbl</i> and <i>arydshln</i> | |
| | 7.14.2 <i>arydshln</i> @ columns | |
| | 7.14.3 <i>arydshln</i> and empty <i>p</i> columns | |
| 8 | History | 44 |
| | [2010/11/18 v1.3] | 44 |
| | [2010/11/15 v1.2] | 44 |
| | [2010/10/28 v1.1] | 44 |
| 9 | Index | 44 |

1 The tabu environment

1.1 tabu, tabu to and tabu spread

```
\begin{tabu}[\langle pos \rangle]{\langle tabular preamble \rangle}
\begin{tabu} to \langle dimen \rangle[\langle pos \rangle]{\langle tabular preamble \rangle}
\begin{tabu} spread \langle dimen \rangle[\langle pos \rangle]{\langle tabular preamble \rangle}
```

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 \langle dimen \rangle` 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 `\langle dimen \rangle`.

`\begin{tabu} spread \langle dimen \rangle` does a tabular whose width is `\langle dimen \rangle` wider than its natural width. `@{\extracolsep{0pt plus 1fil}}` is inserted by default if `\langle dimen \rangle > 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 $\frac{5}{7}$ 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 T_EX: 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: <code>>{\$}X<{\$}</code> | |

- `tabu X` columns can be spanned with `\multicolumn` 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 ε -T_EX `\dimexpr` instead of T_EX 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 pdfT_EX 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 <code>tabu spread</code> behaves as if you didn't give it a target. The result of this example is the same as if one had written <code>\begin{tabu}to\linewidth.</code> | Sherlock Holmes |
| And a <code>\multicolumn</code> cell, just to see | | |

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

Multicolumn in tabu

`\tabuphantomline`

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

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

| | |
|-------|-------|
| Hello | World |
|-------|-------|

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

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

| | |
|-------|-------|
| Hello | World |
|-------|-------|

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

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

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 `\newcolumnntype` 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.

[1pt on 1.5pt off 2pt]

[1.5pt]

default

[on 2pt red]

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

`\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:

| | | | | | | | | | | | | | | | | | | | | |
|---|-------|-------|-------|--|-------|-------|--|-------|-------|-------|--|-------|-------|--|-------|-------|-------|--|-------|-------|
| <table><tr><td>Hello</td><td>world</td></tr><tr><td colspan="2"><hr/></td></tr><tr><td>Hello</td><td>world</td></tr></table> <code>\tabucline</code> | Hello | world | <hr/> | | Hello | world | <table><tr><td>Hello</td><td>world</td></tr><tr><td colspan="2"><hr/></td></tr><tr><td>Hello</td><td>world</td></tr></table> <code>\tabucline*</code> | Hello | world | <hr/> | | Hello | world | <table><tr><td>Hello</td><td>world</td></tr><tr><td colspan="2"><hr/></td></tr><tr><td>Hello</td><td>world</td></tr></table> <code>\firsthline</code> and <code>\hline</code> and <code>\lasthline</code> | Hello | world | <hr/> | | Hello | world |
| Hello | world | | | | | | | | | | | | | | | | | | | |
| <hr/> | | | | | | | | | | | | | | | | | | | | |
| Hello | world | | | | | | | | | | | | | | | | | | | |
| Hello | world | | | | | | | | | | | | | | | | | | | |
| <hr/> | | | | | | | | | | | | | | | | | | | | |
| Hello | world | | | | | | | | | | | | | | | | | | | |
| Hello | world | | | | | | | | | | | | | | | | | | | |
| <hr/> | | | | | | | | | | | | | | | | | | | | |
| Hello | world | | | | | | | | | | | | | | | | | | | |

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 |
|--|--|
| <code>\tabulinestyle{myline=0.4pt on 1.5pt off 1pt}</code> | <code>\tabucline[myline]{-}</code> |
| <code>\tabulinestyle{myleader=\hbox{\$\scriptstyle\star\$}}</code> | <code>\tabucline[myleader]{1-3}</code> |

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

| | | | | | | | | | | | | | | | | | | | |
|---|------------------|--------------------------|---|------|-----|--|--|--------------------|--|---|------|--------|--------------|-----|-----|-----------|---------------|--|--|
| <table><tr><td>Dashed or dotted</td></tr><tr><td>And below is the default</td></tr></table> | Dashed or dotted | And below is the default | <table><tr><td>Dash</td><td>Gap</td></tr><tr><td> </td><td></td></tr><tr><td colspan="2">This one was thick</td></tr></table> | Dash | Gap | | | This one was thick | | <table><tr><td>Dash</td><td>Gap...</td><td>...or leader</td></tr><tr><td>⌘--</td><td>⌘--</td><td>⌘--⌘--⌘--</td></tr><tr><td colspan="3">Interesting ?</td></tr></table> | Dash | Gap... | ...or leader | ⌘-- | ⌘-- | ⌘--⌘--⌘-- | Interesting ? | | |
| Dashed or dotted | | | | | | | | | | | | | | | | | | | |
| And below is the default | | | | | | | | | | | | | | | | | | | |
| Dash | Gap | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | | |
| This one was thick | | | | | | | | | | | | | | | | | | | |
| Dash | Gap... | ...or leader | | | | | | | | | | | | | | | | | |
| ⌘-- | ⌘-- | ⌘--⌘--⌘-- | | | | | | | | | | | | | | | | | |
| 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.

$$\left\{\begin{array}{l} \operatorname{div} \vec{E} = \frac{\rho}{\epsilon_0} \\ \operatorname{div} \vec{B} = 0 \\ \operatorname{rot} \vec{E} = -\frac{\partial \vec{B}}{\partial t} \\ \operatorname{rot} \vec{B} = \mu_0 \vec{j} + \mu_0 \epsilon_0 \frac{\partial \vec{E}}{\partial t} \end{array}\right.$$

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 |
|---|---|---|
| $\begin{array}{cc} \alpha & \beta \\ \gamma & \delta \end{array}$ | $\begin{array}{cc} \alpha & \beta \\ \gamma & \delta \end{array}$ | $\begin{array}{cc} \alpha & \beta \\ \gamma & \delta \end{array}$ |

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

Vertical lines inside the tabular preamble gives:

$$\begin{array}{cc} \alpha & \beta \\ \gamma & \delta \end{array}$$

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 -T_EX `\scantokens` primitive and is defined only inside the `tabu` environment. `\verbatim` has some other limitations you must know:

| | |
|------------------|---|
| <code>^^@</code> | can be used to get a carriage return inside <code>\verbatim</code> |
| <code>^</code> | consequently <code>^</code> is a normal subscript character |
| <code>{ }</code> | curly braces must be equilibrated |
| <code>%</code> | The percent character <code>%</code> is for commentaries: the content of the environment is collected before <code>\verbatim</code> 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 Opt{|[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 exponents. The scanner stops as soon as something else than a number, blank space, comma, dot, +, –, `e`, `E` is found, and even if it is a macro that contains a number.

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

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:

| |
|--|
| $\left(\begin{array}{c} \text{This is \texttt{tabu} with \texttt{delarray} shortcut for parenthesis} \\ \text{around} \end{array} \right)$ |
| $\left(\begin{array}{c} \text{This is \texttt{tabularx} with \texttt{delarray} shortcut for parenthesis} \\ \text{thesis around} \end{array} \right)$ |
| $\left\{ \begin{array}{c} \text{This is \texttt{tabu} with \texttt{delarray} shortcut for curly} \\ \text{brackets around} \end{array} \right\}$ |
| $\left\{ \begin{array}{c} \text{This is \texttt{tabularx} with \texttt{delarray} shortcut for curly} \\ \text{brackets around} \end{array} \right\}$ |

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 & \texttt{tabu} & package \\ \tabucline*[on1pt]-
\end{tabu}
```

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

```
\begin{tabu}{\usetabu{mytabu}}
\multicolumn{3}{c}{This is tabu} & package
\end{tabu}
```

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

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

| | | |
|--------------------------|--|----------------|
| <code>l</code> = left | or for <code>ragged2e</code> settings: | <code>L</code> |
| <code>c</code> = center | | <code>C</code> |
| <code>r</code> = right | | <code>R</code> |
| <code>j</code> = justify | | <code>J</code> |

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

```
\begin{tabu}{|X|X|}
\rowfont[c]\bfseries
This & Is
\end{tabu}
```

| | |
|------|---------|
| This | Is |
| tabu | package |

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 L^AT_EX default: `\parindent = 20pt`.

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

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

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

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

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 rewriting columns

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

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

The process of rewriting 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**:

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

it is not recommanded no nest such constructions like:

```
\newcolumnntype{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: $410.67296\text{pt} / 2 = 205.33649\text{pt}$ and then:
 $X = 386.67296\text{pt} - 205.33649\text{pt} = 181.33647\text{pt}$
- 2) The second attempt lead to a **tabu** width of 386.67294pt: the target is reached.
 The final width of each **X** column is the product of **tabu X** by its width coefficient.

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 pdfT_EX, otherwise, a warning is displayed and the **linegoal** option has no effect: the default target remains **\linewidth**. **\linegoal** works with pdfT_EX 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}[2010/11/18 v1.3 - flexible LaTeX tabulars (FC)]
4 \RequirePackage{array}[2008/09/09]
```

Then minimal catcode ascertaining 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}
```



```
43 \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` compatibility with `delarray` shortcuts available inside `tabu`: requirement for this fix is checked by `\tabu@fix@arrayright`.

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

```
44 \AtBeginDocument{%
45   \@ifundefined{NC@rewrite@X}{\newcolumnntype{X}{}}{}% new column X if not exists
46   \expandafter\in@\expandafter\@arrayright\expandafter{\endarray}%
47   \ifin@ \let\tabu@endarray\endarray
48   \else \tabu@fix@arrayright \fi % <fix for colortbl & arydshln (delarray)>
49   \@ifpackageloaded{colortbl} \tabu@colortbltrue \tabu@colortblfalse
50 }
```

7.2 Some constants

`\tabu@cnt` Used 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`”.

```
51 \newcount\tabu@cnt
52 \newcount\tabu@nbcols
53 \newcount\tabu@X@cols
54 \newcount\tabu@start
55 \newcount\tabu@stop
```

`\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
56 \newdimen\tabu@target
57 \newdimen\tabu@spreadtarget
58 \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.

```
59 \newdimen\tabucolX
60 \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 switch 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 `\firstline`).

```
61 \newif\iftabu@measuring
62 \newif\iftabu@spread
63 \newif\iftabu@nested
64 \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.

```

65 \newsavebox\tabu@box
66 \chardef\tabu@arstrutbox=\@arstrutbox
67 \let\tabu@vcenter \vcenter

```

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

```

68 \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}`

```

69 \newcommand\tabu@NC@rewrite@X[1] [] {\tabu@rewrite@X{#1}%
70                                     \let\@halignto \relax
71                                     \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.

```

72 \expandafter\def\csname NC@rewrite@\string\usetabu\endcsname#1{%
73   \ifx\#1\\\tabu@saveerr{}\else
74     \ifundefined{tabu@saved@\string#1}
75       {\tabu@saveerr{#1}}
76       {\let\tabu@rewrite@X \tabu@rewrite@Xrestore
77        \def\tabu@temp{\xdef\tabu@usetabu{%
78          \col@sep \the\col@sep\relax
79          \arrayrulewidth \the\arrayrulewidth\relax
80          \@tempdima \ht\@arstrutbox \advance\@tempdima -\extrarowheight
81          \extrarowheight \the\extrarowheight\relax
82          \advance\@tempdima \extrarowheight
83          \ht\@arstrutbox \@tempdima
84          \extratabsurround \the\extratabsurround\relax}}%
85          \doublerulesep \the\doublerulesep\relax}}%
86        \aftergroup \tabu@usetabu
87        \csname tabu@saved@\string#1\expandafter\endcsname}%
88   \fi
89 }% \NC@rewrite@usetabu
90 \expandafter\def\csname NC@find@\string\usetabu\endcsname#1\usetabu{\NC@{#1}}

```

`\tabu@rewritelfirst` (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@rewritelfirst` 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).

```

91 \expandafter\def\csname NC@rewrite@\string\tabu@rewritelfirst\endcsname{%
92   \ifx\tabu@lines\relax
93     \NC@list\expandafter{\expandafter\NC@do \expandafter\usetabu
94                           \NC@do X\NC@do\tabu@rewritelast}%
95   \else
96     \NC@list\expandafter{\expandafter\NC@do \expandafter\usetabu
97                           \expandafter\NC@do \expandafter|\tabu@NC@list
98                           \NC@do X\NC@do\tabu@rewritelast}%
99     \tabu@lines % defines NC@rewrite@| for tabu only (inside @mkpream group)
100   \fi
101   \let\save@decl \tabu@save@decl % inversion of tokens in text mode
102   \let\NC@rewrite@X \tabu@NC@rewrite@X

```

```

103 \aftergroup \tabu@global@temp
104 \NC@find \tabu@rewritelast
105 }% \NC@rewrite@\tabu@rewritefirst
106 \expandafter\def\csname NC@find@\string\tabu@rewritefirst\endcsname
107 \def\tabu@rewritefirst{\NC@{#1}}
108 \def\tabu@rewritefirst{%
109 \edef\tabu@NC@list{\the\NC@list}%
110 \NC@list{\NC@do \tabu@rewritefirst \NC@do *}%
111 }% \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.

```

112 \expandafter\def\csname NC@rewrite@\string\tabu@rewritelast\endcsname{%
113 \global\NC@list\expandafter{\tabu@NC@list}%
114 \futurelet \tabu@temp \tabu@rewritelast
115 }% \NC@rewrite@\tabu@rewritelast
116 \expandafter\def\csname NC@find@\string\tabu@rewritelast\endcsname
117 \def\tabu@rewritelast{\NC@{#1}}
118 \def\tabu@rewritelast{%
119 \ifx \@halignto\relax % found a X column
120 \let\tabu@temp \@empty
121 %\aftergroup \tabu@addphantomline
122 \ifx \tabu@halignto\relax % X column measure
123 \aftergroup \tabu@prep@TRIAL
124 \else \aftergroup \tabu@setup@everycr
125 \fi
126 \else
127 \ifdim \tabu@target=z@ \let\tabu@temp \@empty
128 \else
129 \if |\noexpand\tabu@temp \def\tabu@temp{@{\extracolsep\@flushglue}}\else
130 \if !\noexpand\tabu@temp \def\tabu@temp{@{\extracolsep\@flushglue}}\else
131 \def\tabu@temp{!\extracolsep\@flushglue}\fi\fi
132 \fi
133 \aftergroup\tabu@setup@everycr
134 \fi
135 \let\@halignto \tabu@halignto
136 \expandafter\NC@find \tabu@temp
137 }% \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.

```

138 %%\newtoks\tabu@everycr@tok
139 \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.)

```

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

```

```

143         \afterassignment\let\count@
144 }% \tabu@nowrite
145 \let\tabu@write\write

```

7.3 \tabu and \endtabu

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

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

```

146 \def\tabu{%
147     \ifmmode \def\endtabu{\endarray}%
148     \else    \def\endtabu{\endtabular}\fi
149     \tabu@setup \tabu@settarget
150 }% \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.

```

151 \def\tabu@setup{%
152     \let\adl@xarraydashrule \tabu@adl@xarraydashrule % <fix> arydshln
153     \let\adl@act@endpbox \tabu@adl@act@endpbox        % <fix> arydshln
154     \let\adl@act@@endpbox \tabu@adl@act@endpbox       % <fix> arydshln
155     \let\@preamerror \@preamerr                      % <fix> arydshln
156     \let\endarray \tabu@endarray                      % <fix> colortbl & arydshln (delarray)
157     \let\tabu@global@temp \@empty \let\tabu@global@X \@empty
158     \ifx\verbatim \tabu@sanitizetext
159         \tabu@nestedtrue
160         \def\tabu@aligndefault{t}\def\tabudefaulttarget{\linewidth}%
161     \else \tabu@nestedfalse
162         \def\tabu@aligndefault{c}%
163         \global\tabu@footnotes{}%
164         \aftergroup\the\aftergroup\tabu@footnotes
165         \ifdim\parindent>\z@ \ifx\linewidth\tabudefaulttarget
166             \everypar\expandafter{%                    % correction for indentation
167                 \the\everypar\everypar\expandafter{\the\everypar}%
168                 \setbox\z@=\lastbox
169                 \ifdim\wd\z@>\z@ \advance\linewidth -\wd\z@\fi
170                 \box\z@
171             }%
172         \fi\fi
173     \fi
174     \let\@footnotetext \tabu@footnotetext
175     \let\@xfootnotenext\tabu@xfootnotetext
176     \let\centering \tabu@centering
177     \let\raggedright \tabu@raggedright
178     \let\raggedleft \tabu@raggedleft
179     \let\@trivlist \tabu@trivlist                    %<restore \\=\@centercr inside lists>
180     \let\tabudecimal \tabu@tabudecimal
181     \let\verbatim \tabu@sanitizetext
182     \let\fbbox \tabu@fbbox
183     \let\prepnext@tok \tabu@prepnext@tok             % <for rowfont and tabucline>
184     \let\rowfont \tabu@rowfont
185     \everycr{}\let\tabu@everycr \everycr
186     \tabu@spreadfalse \tabu@measuringfalse
187     \edef\tabu@hfuzz{\ifdim\hfuzz<.1\p@ .1\p@\else\the\hfuzz\fi}%
188     \tabu@rewritefirst

```

```

189 }% \tabu@setup
190 \def\tabu@save@decl{% no inversion on tokens when not in math mode
191   \ifcat$\d@llarend
192     \toks\count@\expandafter\expandafter\expandafter{%
193       \expandafter\@nextchar \the\toks\count@}%
194   \else
195     \toks\count@\expandafter\expandafter\expandafter{%
196       \expandafter\the\expandafter\toks\expandafter\count@\@nextchar}%
197   \fi
198 }% \tabu@save@decl

```

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

```

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

```

`\tabu@footnotes`

```

237 \newtoks\tabu@footnotes
238 \long\def\tabu@footnotetext#1{%
239   \edef\@tempa{\the\tabu@footnotes
240     \noexpand\footnotetext [\the\c@name c@\@mpfn\endcsname]}%
241   \global\tabu@footnotes\expandafter{\@tempa{#1}}}%
242 \long\def\tabu@xfootnotetext[#1]#2{%

```

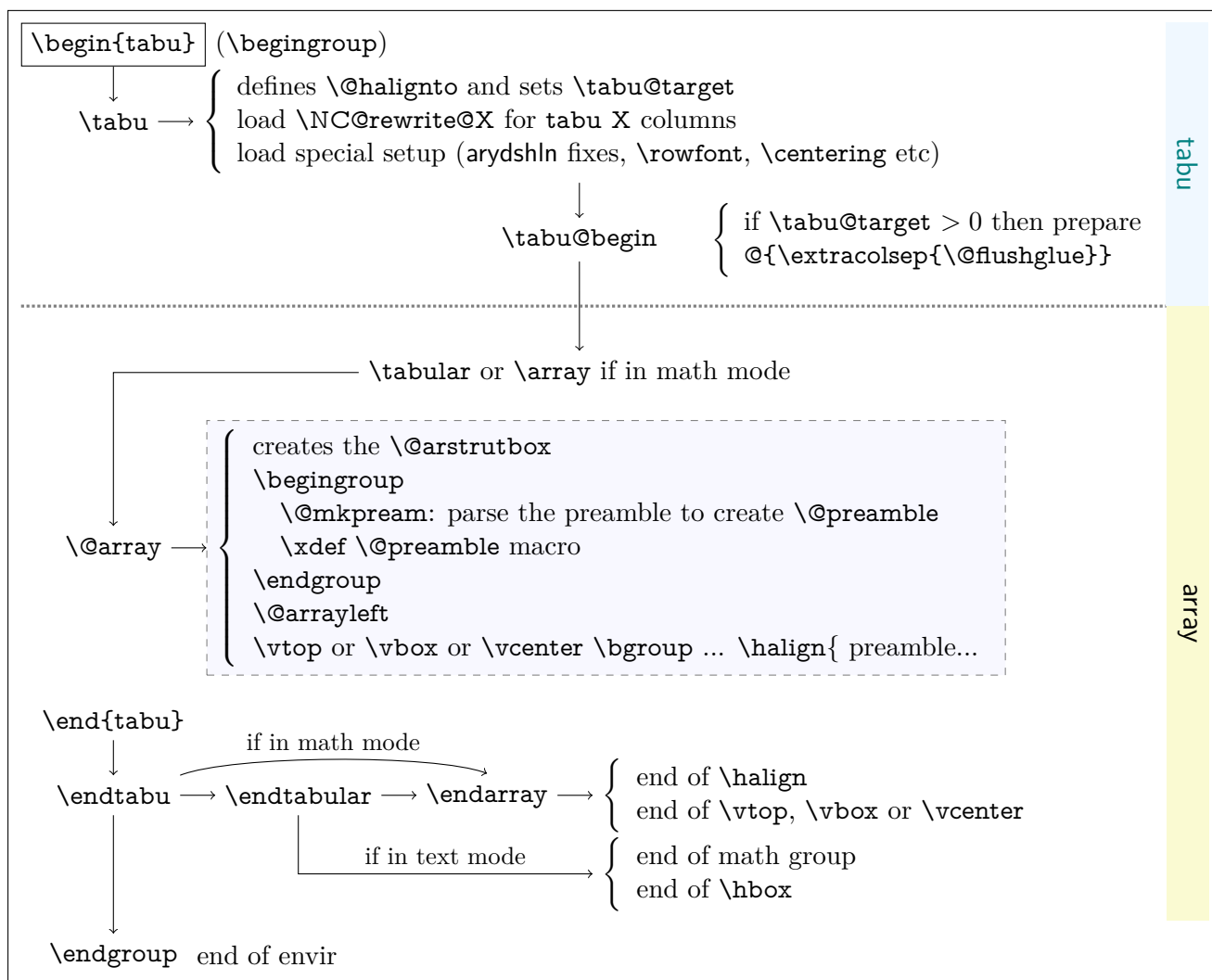
```

243 \global\tabu@footnotes\expandafter{\the\tabu@footnotes
244 \footnotetext[{#1}]{#2}}

```

7.4 Flow chart of expansion

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



tabu to with X column

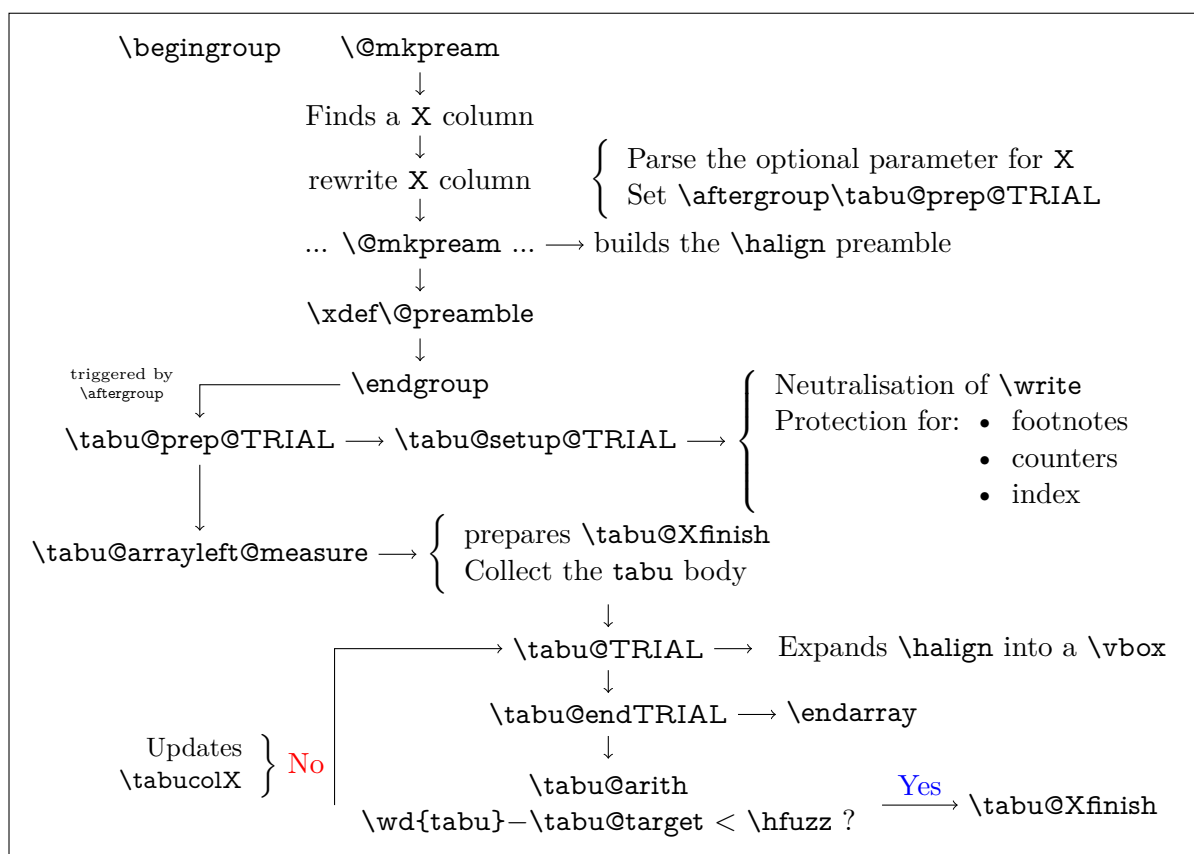
The important part of the job is made inside the dashed box above: `\@mkpream` expands the columns definitions, which can be user defined. Hopefully, it does its job inside a group, therefore a user-column can set a macro to be expanded `\aftergroup`. This implementation allows much modifications in 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 beginning 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 T_EX mechanism of expansion inside `\halign` (T_EX 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:

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

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

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

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

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`.

```
245 \def\tabu@rewrite@X#1{\tabu@Xarg{#1}%
246   \iftabu@spread \tabu@rewrite@Xspread
247   \else          \tabu@rewrite@Xto
248   \fi
249 }% \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`.

```
250 \def\tabu@rewrite@Xnested#1{\in@ ${#1}\ifin@
251   \def\tabucolX@spec{>{$}p{\tabucolX}<{$}}\else
252   \def\tabucolX@spec{p{\tabucolX}}\fi}
253 \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`.

```
254 \def\tabu@rewrite@Xto{%
255   \iftabu@measuring % not the first X column found in preamble
256     \xdef\tabu@global@X {\tabu@global@X
257       \advance\tabu@X@sum \the\tabu@X@sum\relax}%
258   \else % first X column found in preamble
259     \tabu@measuringtrue
260     \ifdim\tabu@target=\z@
261       \setlength \tabu@target \tabu@defaulttarget
262     \fi
263     \xdef\tabu@global@X{%
264       \tabu@X@sum \the\tabu@X@sum\relax
265       \tabu@target \the\tabu@target\relax}%
266     \let\tabu@halignto \relax
267   \fi
268 }% \tabu@rewrite@Xto
```

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

```
269 \def\tabu@rewrite@Xspread{% tabu spread with X columns: we need to store each coef
270   \iftabu@measuring % not the first X column found in preamble
271     \advance\tabu@X@cols \@ne
272     \expandafter\let\csname tabu@X\the\tabu@X@cols\endcsname \relax
273     \xdef\tabu@global@X {\tabu@global@X
274       \advance\tabu@X@cols \@ne
275       \advance\tabu@X@sum \the\tabu@X@sum\relax
276       \def\csname tabu@X\the\tabu@X@cols\endcsname{\strip@pt\tabu@X@sum}}%
277   \else % first X column found in preamble
278     \tabu@measuringtrue
279     \tabu@X@cols \@ne \tabu@spreadtarget=\tabu@target
280     \setlength \tabu@target \tabu@defaulttarget
```

```

281 \expandafter\let\csname tabu@X\the\tabu@X@cols\endcsname \relax
282 \xdef\tabu@global@X {%
283   \tabu@X@cols \one
284   \tabu@X@sum \the\tabu@X@sum\relax
285   \tabu@target \the\tabu@target\relax
286   \tabu@spreadtarget \the\tabu@spreadtarget\relax
287   \def\csname tabu@X\the\tabu@X@cols\endcsname{\strip@pt\tabu@X@sum}}%
288 \let\tabu@halignto \relax
289 \fi
290 }% \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`

```

291 \def\tabu@Xarg#1{%
292   \ifx\|#1\% <shortcut when no option>
293     \tabu@X@sum \p@
294     \def\tabucolX@spec{p{\tabucolX}}%
295     \edef\tabu@temp{p{\the\tabucolX}}% <required for \usetabu>
296   \else
297     \tabu@X@sum \z@
298     \let\tabucolX@align \@empty
299     \let\tabucolX@spec \@empty
300     \let\tabu@Xmath \relax
301     \tabu@Xparse {}#1\relax\@nnil
302   \fi
303 }% \tabu@Xarg
304 \def\tabu@Xparse#1{\futurelet\@let@token\tabu@Xtest}
305 \expandafter\def\expandafter\tabu@Xparsespace\space{\tabu@Xparse{}}
306 \def\tabu@Xtest{%
307   \ifcase \ifx \@nnil\@let@token \z@ \else
308     \ifx \relax\@let@token \m@ne\else
309     \if ,\@let@token \m@ne\else
310     \if p\@let@token 1\else
311     \if m\@let@token 2\else
312     \if b\@let@token 3\else
313     \if l\@let@token 4\else
314     \if c\@let@token 5\else
315     \if r\@let@token 6\else
316     \if .\@let@token 7\else
317     \ifx \@sptoken\@let@token 8\else
318     \if L\@let@token 9\else
319     \if C\@let@token 10\else
320     \if R\@let@token 11\else
321     \ifcat $\@let@token 12\else
322     13\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\relax
323   \tabucolX@do \expandafter\remove@to@nnil
324   \or \tabu@Xcolspec{p}%
325   \or \tabu@Xcolspec{m}%
326   \or \tabu@Xcolspec{b}%
327   \or \tabu@Xalign{>{\raggedright}}%
328   \or \tabu@Xalign{>{\centering}}%
329   \or \tabu@Xalign{>{\raggedleft}}%
330   \or \expandafter\tabu@Xcoef
331   \or \expandafter\tabu@Xparsespace
332   \or \tabu@Xalign{>{\RaggedRight}}%

```

```

333 \or \tabu@Xalign{>{\Centering}}%
334 \or \tabu@Xalign{>{\RaggedLeft}}%
335 \or \let\tabu@Xmath=$\expandafter\tabu@Xparse
336 \or \tabu@Xcoef{}%
337 \else \expandafter\tabu@Xparse
338 \fi
339 }% \tabu@Xtest
340 \def\tabu@Xalign#1{%
341 \ifx \tabucolX@align\@empty \else \PackageWarning{tabu}
342 {Duplicate horizontal alignment specification}\fi
343 \def\tabucolX@align{#1}\expandafter\tabu@Xparse
344 }% \tabu@Xalign
345 \def\tabu@Xcolspec#1{%
346 \ifx\tabucolX@spec\@empty\else \PackageWarning{tabu}
347 {Duplicate vertical alignment specification}\fi
348 \def\tabucolX@spec{#1}\expandafter\tabu@Xparse
349 }% \tabu@Xcolspec
350 \def\tabu@Xcoef#1{\def\@tempa{#1}%
351 \afterassignment\tabu@Xc@ef \tabu@cnt
352 }% \tabu@Xcoef
353 \def\tabu@Xc@ef{%
354 \advance\tabu@X@sum \@tempa\the\tabu@cnt\p@
355 \tabu@Xparse{}%
356 }% \tabu@Xc@ef
357 \def\tabucolX@do{%
358 \ifx\tabucolX@spec\@empty \def\tabucolX@spec{p}\fi
359 \ifdim \tabu@X@sum=\z@ \tabu@X@sum \p@\fi
360 \begingroup
361 \edef\tabucolX@spec##1{\tabucolX@spec{##1}}%
362 \edef\tabu@temp{\dimexpr\strip@pt\tabu@X@sum \tabucolX \relax}%
363 \toks@ \expandafter{\tabucolX@align}%
364 \edef\x{%
365 \def\noexpand\tabu@temp{%
366 \ifx \tabu@Xmath\relax\else >{\$}\fi
367 \the\toks@ \tabucolX@spec{\the\tabu@temp}%
368 \ifx \tabu@Xmath\relax\else <{\$}\fi}%
369 \def\noexpand\tabucolX@spec{%
370 \ifx \tabu@Xmath\relax\else >{\$}\fi
371 \the\toks@ \tabucolX@spec{\tabu@temp}%
372 \ifx \tabu@Xmath\relax\else <{\$}\fi}%
373 }\expandafter\endgroup\x
374 }% \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 \@halign 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.

```

375 \def\tabu@prep@TRIAL{%
376 \let\tabu@savdpreamble \@preamble
377 \def\savetabu {\tabu@saveX}%
378 \begingroup\iffalse{\fi\ifnum0='}\fi
379 \tabu@setup@TRIAL

```

```

380 \iftabu@spread % first trial only
381 \advance\tabu@X@cols \@ne
382 \expandafter\let\csname tabu@X\the\tabu@X@cols\endcsname \@undefined
383 \tabu@X@cols \z@ \tabu@naturalX \z@ \tabucolX \z@
384 \let\tabu@naturalX@max \z@
385 \let\tabu@naturalX@min \z@
386 \let\tabu@startpboxORI \@startpbox
387 \let\@startpbox \tabu@spread@startpbox
388 \else
389 \tabucolX = \tabu@target
390 \fi
391 \tabu@cnt \z@ % number of trials
392 \def\tabu@lasttry{\m@ne\p@}%
393 \tabu@arrayleft@measure
394 }% \tabu@prep@TRIAL

395 \def\tabu@setup@TRIAL{\tabu@global@X
396 \let\tabu@rewrite@X \tabu@rewrite@Xnested
397 \def\tabudefaulttarget{\linewidth}%
398 \def\@elt##1{\global\value{##1}\the\value{##1}\relax}%
399 \xdef\tabu@global@X {\cl@ckpt}\let\@elt \relax
400 \hbadness\@M \hfuzz\maxdimen
401 \let\hbadness \@tempcnta
402 \let\hfuzz \@tempdima
403 \let\write \tabu@nowrite
404 \let\@footnotetext\@gobble
405 \let\tabu@saveX \@gobble
406 \tabu@TRIAL@hook
407 }% \tabu@setup@TRIAL

```

\tabuDisableCommands

```

408 \let\tabu@TRIAL@hook\@empty
409 \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**.

```

410 \def\tabu@arrayleft@measure#1\bgroup{%
411 \def\tabu@Xfinish{\ifnum0='{ \fi\iffalse}\fi\endgroup\tabu@global@X #1\bgroup}%
412 \toks@{%
413 \tabu@setup@everycr
414 \let\@preamble\tabu@savedpreamble}% <required for multicolumn>
415 \tabu@collect
416 }% \tabu@arrayleft@measure

```

Collecting the tabu body: required for X columns

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

\tabu@collectbody

```

417 \def\tabu@collect{\catcode'\^^@=13\def\tabu@stack{b}\tabu@collectbody}
418 \long\def\tabu@collectbody#1\end#2{%
419 \edef\tabu@stack{\tabu@pushbegins #1\begin\end\expandafter\@gobble\tabu@stack}%
420 \ifx\tabu@stack\@empty
421 \toks@\expandafter{\the\toks@#1}\def\tabu@endenvir{\end{#2}}%
422 \expandafter\tabu@TRIAL
423 \else
424 \toks@\expandafter{\the\toks@#1\end{#2}}%
425 \expandafter\tabu@collectbody
426 \fi
427 }% \tabu@collectbody

```

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

```
429 \def\tabu@TRIAL{\setbox\tabu@box \hbox\bgroup $\@arrayleft
430   \if c\tabu@align\vcenter\else\vbox\fi\bgroup \the\toks@
431                                     \tabu@endTRIAL}% constant
432 \def\tabu@endTRIAL{\endarray$\egroup
433   \iftabu@spread \tabu@spreadfalse
434     \let\@startpbox \tabu@startpboxORI
435     \tabu@spreadarith % <only once>
436   \else \tabu@arith
437   \fi
438   \iftabu@measuring \tabu@measuringfalse
439     \expandafter \tabu@TRIAL % <continue trials>
440   \else
441     \toks@\expandafter\expandafter\expandafter{%
442       \the\expandafter\toks@ \tabu@endenvir}%
443     \expandafter \tabu@Xfinish \the\toks@ % <we are then!>
444   \fi
445 }% \tabu@endTRIAL
```

The arithmetic of X columns

`\tabu@arith` Algorithms.
`\tabu@spreadarith`

```
446 \def\tabu@arith{%
447   \advance\tabu@cnt \@ne
448   %% \showbox\tabu@box
449   \ifdim \tabu@lasttry=\wd\tabu@box
450     \tabu@message{\tabu@header Reached minimum width. Backing up}%
451   \else
452     \edef\tabu@lasttry{\the\wd\tabu@box}%
453     \tabu@message{\tabu@message@arith}%
454     \ifdim \dimexpr \wd \tabu@box - \tabu@target < \tabu@hfuzz
455       \tabu@message{\tabu@message@reached}%
456       \xdef\tabu@global@X {\tabu@target\the\tabu@target\relax \tabu@global@X
457                                     \tabucolX \the\tabucolX\relax}%
458     \else
459       \advance\tabucolX -\dimexpr(\wd\tabu@box-\tabu@target-\tabu@X@sum/65536)
460                                     *65536/\tabu@X@sum\relax
461       \ifdim \tabucolX<\z@
462         {\let\write\tabu@write\PackageWarning\tabu}
463         {X Columns too narrow (table too wide)\MessageBreak}}%
464         \tabucolX=1em%
465       \else \tabu@measuringtrue
466       \fi
467     \fi
468   \fi
469 }% \tabu@arith
470 \def\tabu@spreadarith{%
471   \tabu@naturalX \tabu@naturalX@max\relax
```


Measuring the natural width for tabu spread

`\tabu@spread@startpbox` 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 further trials, the standard scheme for X column is used.

```

523 \def\tabu@spread@startpbox#1{\bgroup
524   \in@{\tabucolX}{#1}%
525   \ifin@
526     \let\vss\relax\let\vfil\vss\let\vfill\vss\let\vskip\@tempskipa
527     \hbox to\tabu@target\bgroup\aftergroup\tabu@spread@endhbox\hfil
528   \else
529     \expandafter\@gobble\tabu@startpboxORI{#1}% \@gobble \bgroup
530   \fi
531 }% \tabu@spread@startpbox
532 \def\tabu@spread@endhbox{%
533   \ifnum\badness>\@M %<text does not fit>
534     \global\tabu@naturalX \tabu@target
535   \else
536     \setbox\tabu@box\lastbox
537     \setbox\tabu@box\hbox{\unhbox\tabu@box}%
538     \global\advance\tabu@X@cols \@ne
539     \ifcsname tabu@X\the\tabu@X@cols\endcsname
540       \global\advance\tabu@naturalX \wd\tabu@box
541     \else
542       \global\tabu@X@cols \@ne
543       \global\tabu@naturalX \wd\tabu@box
544     \fi
545     \ifdim \tabu@naturalX@max<\tabu@naturalX
546       \xdef\tabu@naturalX@max{\the\tabu@naturalX}%
547     \fi
548     \ifdim \tabu@naturalX@min<\dimexpr \wd\tabu@box * \tabu@X@sum /
549       (65536*\csname tabu@X\the\tabu@X@cols\endcsname)\relax
550       \xdef\tabu@naturalX@min{\the\dimexpr \wd\tabu@box * \tabu@X@sum /
551         (65536*\csname tabu@X\the\tabu@X@cols\endcsname)\relax}%
552     \fi
553   \fi
554   \box\tabu@box
555   \egroup % end of \vtop (measure)
556 }% \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.

```

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

```

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

```

560 \def\tabuvline@arg#1{%
561   \ifx\#1\\% <shortcut when no option>
562     \def\tabu@temp{\tabu@vline\arrayrulewidth}%
563   \else \futurelet \tabu@temp \tabuvline@argi #1\p@\p@\@nnil \tabuvline@argiii

```



```

564 \fi
565 }% \tabuvline@arg
566 \def\tabuvline@argi{%
567 \let\tabu@color \@empty
568 \ifcat A\noexpand\tabu@temp
569 \@tempdima\arrayrulewidth
570 \expandafter\tabu@getlinecolor
571 \else \expandafter\tabuvline@argii
572 \fi
573 }% \tabuvline@argi
574 \def\tabuvline@argii#1\@nnil{%
575 \tabu@maybeicolor \@tempdima #1 \@nnil
576 }% \tabuvline@argii
577 \def\tabuvline@argiii{%
578 \edef\tabu@temp{!\color@begingroup\tabu@color
579 \noexpand\tabu@vline{\the\@tempdima}\color@endgroup}}%
580 }% \tabuvline@argiii

```

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

```

581 {
582 \expandafter\let\csname NC@find@|\endcsname\relax
583 \expandafter\let\csname NC@rewrite@|\endcsname\relax
584 \xdef\tabu@lines{%
585 \let\csname NC@find@|\endcsname \noexpand\tabuvlines@find
586 \let\csname NC@rewrite@|\endcsname \noexpand\tabuvline@rewrite
587 }% \tabu@vline
588 }
589 \def\tabuvlines@find#1|{\NC@{#1}}

```

Horizontal lines: `\tabucline`

`\tabu@setup@everycr`

```

590 \def\tabu@setup@everycr{%
591 \tabu@firstcline true
592 \let\tabu@first@ccline \z@
593 \def\tabu@restoreeverycr {\let\everycr \tabu@everycr}%
594 \def\everycr{\afterassignment\tabu@restoreeverycr \@temptokena}% <for ialign>
595 \tabu@everycr\expandafter{\the\tabu@everycr
596 \noalign{%
597 \ifcase\tabu@first@ccline
598 \global\let\tabu@first@ccline \tw@
599 \or\global\let\tabu@first@ccline \tw@
600 \or\global\tabu@firstcline false
601 \global\let\tabu@first@ccline \m@ne
602 \fi}%
603 \tabu@rowfontreset
604 }%
605 }% \tabu@setup@everycr

```

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

```

606 \def\tabuphantomline{\crrc\noalign{%
607 \global\let\@arstrutbox \voidb@x
608 \iftabu@colortbl
609 \xdef\tabu@minrowclearance{\the\minrowclearance}%
610 \global\minrowclearance \z@
611 \fi
612 \global\let\vcenter \vbox

```

```

613 \iftabu@nested\vspace*{\dimexpr\extrarowheight-\ht\@arstrutbox}\fi
614 \toks@{\}\count@\@ne
615 \@whilenum\count@<\tabu@nbc\do{\advance\count@\@ne
616 \toks@\expandafter{\the\toks@&}}}%
617 \toks@\expandafter{\the\toks@
618 \cr\noalign{%
619 \global\let\@arstrutbox \tabu@arstrutbox
620 \global\let\vcenter \tabu@vcenter
621 \iftabu@colortbl
622 \global\minrowclearance \tabu@minrowclearance\relax\fi
623 \global\let\tabu@first@cline \@ne}}}%
624 \expandafter\the\toks@
625 }% \tabuphantomline

```

`\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.

```

626 \def\tabu@linedash{4pt}% <default value>
627 \def\tabu@dashgap{4pt}% <default value>
628 \def\tabucline{\noalign{\ifnum0='}\fi\@ifstar
629 {\@tempswatrue\tabu@cline}
630 {\@tempswafalse\tabu@cline}}}%
631 \newcommand*\tabu@cline[2][{}]{%
632 \tabu@startstop{#2}\tabu@getlinespec{#1}\@multicnt\@ne
633 \ifnum\tabu@start>\tabu@stop \the\toks@
634 \else\ifx\tabu@xleaders\relax\the\toks@
635 \else
636 \toks@{\ifnum0='{\\fi}\noalign{\global\let\@arstrutbox \voidb@x}}}%
637 %% \iftabu@firstcline\if\tabu@align t\vskip-\ht\@arstrutbox\fi\fi
638 \@whilenum\@multicnt<\tabu@start\do{\advance\@multicnt\@ne
639 \toks@\expandafter{\the\toks@ &}}}%
640 \loop
641 \toks@\expandafter{\the\toks@ \omit }%
642 \ifnum \@multicnt=\@ne
643 \expandafter \tabu@add \csname tabu@tok@\the\@multicnt L\endcsname \@nil
644 \fi
645 \ifcase 0\if@tempswa\else\ifnum \@multicnt<\tabu@stop 1\fi\fi\relax
646 \toks@\expandafter{\the\toks@ \color@begingroup}%
647 \toks@\expandafter\expandafter\expandafter{\expandafter\the
648 \expandafter\toks@ \tabu@xleaders \color@endgroup}%
649 \fi
650 \ifcase 0\if@tempswa\else\ifnum \@multicnt<\tabu@stop 1\fi\fi %
651 \expandafter \tabu@add \csname tabu@tok@\the\@multicnt R\endcsname \@nil
652 \fi
653 \ifnum\@multicnt<\tabu@stop
654 \advance\@multicnt\@ne
655 \if@tempswa \toks@\expandafter{\the\toks@ &}}%
656 \else \toks@\expandafter{\the\toks@ \span}\fi
657 \repeat
658 \@whilenum\@multicnt<\tabu@nbc\do{\advance\@multicnt\@ne
659 \toks@\expandafter{\the\toks@ &}}}%
660 \toks@\expandafter{\the\toks@ \cr\noalign{%
661 \global\let\@arstrutbox \tabu@arstrutbox}}}%
662 %% \message{LIGNE AJOUTEE:\the\toks@}%
663 \the\toks@

```

```

664 \fi\fi
665 }% \tabucline
666 \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.

```

667 \def\tabu@startstop#1{\tabu@start@stop #1\relax 1-\tabu@nbcolls\@nnil}
668 \def\tabu@start@stop #1-#2\@nnil{%
669 \@defaultunits \tabu@start \number0#1\relax\@nnil
670 \@defaultunits \tabu@stop \number0#2\relax\@nnil
671 \ifnum\tabu@start>\tabu@nbcolls\tabu@start=\tabu@nbcolls\else
672 \ifnum\tabu@start>\z@\else \tabu@start=\@ne \fi\fi
673 \ifnum\tabu@stop>\tabu@nbcolls \tabu@stop\tabu@nbcolls \else
674 \ifnum\tabu@stop>\z@\else \tabu@stop=\tabu@nbcolls \fi\fi
675 }% \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).

```

676 \def\tabu@getlinespec#1{\let\tabu@xleaders \relax
677 \@defaultunits \let\@tempa=#1 \relax\@nnil
678 \ifx\@tempa\relax \let\tabu@xleaders \tabu@defaultleaders\else
679 \ifx\@tempa\hbox \tabu@defleaders{#1}\else
680 \ifx\@tempa\box \tabu@defleaders{#1}\else
681 \ifx\@tempa\copy \tabu@defleaders{#1}\else
682 \ifcsname tabu@line@style@\detokenize{#1}\endcsname
683 \csname tabu@line@style@\detokenize{#1}\endcsname
684 \else \tabu@getline{#1}\p@ on0pt off0pt\fi\fi\fi\fi\fi
685 }% \tabu@getlinespec
686 \def\tabu@defleaders#1{%
687 \def\tabu@xleaders{\xleaders\hbox{\lower.5\extrarowheight#1}\tabu@leaderfill}}
688 \def\tabucline@warn#1{\PackageWarning{tabu}
689 {Undefined line syle: #1
690 \MessageBreak Using default line style instead}}%
691 \let\tabu@xleaders \tabu@defaultleaders
692 }% \tabucline@warn

```

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

```

693 \def\tabu@getline#1{\tabu@lineon #1 \@nil on\tabu@linedash \p@ \@nil\@nnil{#1}}
694 \def\tabu@maybecolor{\ifx\tabu@color\@empty \afterassignment\tabu@colortest
695 \else \@defaultunits\fi}
696 \def\tabu@lineon #1on#2\@nil#3\@nnil#4{%
697 \let\tabu@color \@empty
698 \@defaultunits \let\@tempa=#1 \relax\@nnil
699 \ifcase 0\ifx o\@tempa 1\else\ifcat A\noexpand\@tempa 2\fi\fi\relax
700 \tabu@maybecolor \@tempdima #1\arrayrulewidth \p@\@nnil
701 \tabu@maybecolor \@tempdimb #2\p@ \@nnil
702 \tabu@lineoff #1 on#2 off\tabu@dashgap \p@ \@nnil
703 \or\@tempdima \arrayrulewidth
704 \tabu@maybecolor \@tempdimb #2\p@ \@nnil
705 \tabu@lineoff #1 on#2 off\tabu@dashgap \p@ \@nnil
706 \else
707 \tabu@maybecolor \@tempdima \arrayrulewidth #1\p@\@nnil
708 \ifx\tabu@color\@empty \tabucline@warn{#4}\else
709 \@tempdimb \z@
710 \tabu@lineoff off0pt \p@\@nnil \fi
711 \fi
712 }% \tabu@lineon

```

```

713 \def\tabu@lineoff #1off#2\@nnil{%
714   \tabu@maybecolor \@tempdimc #2\p@ \@nnil
715   \ifdim \@tempdimb=\z@
716     \ifdim \@tempdimc>\z@ \@tempdimb \tabu@dashgap\relax\fi\fi
717   \ifdim \@tempdimc=\z@
718     \ifdim \@tempdimb>\z@ \@tempdimc \tabu@linedash\relax\fi\fi
719   \ifdim \@tempdima<\z@ \else
720   \ifdim \@tempdimb<\z@ \else
721   \ifdim \@tempdimc<\z@ \else
722     \edef \tabu@xleaders{\tabu@color\xleaders
723       \ifdim\@tempdimc>\z@
724         \hbox\bgroup \kern\the\dimexpr\@tempdimc/2\relax\fi
725         \noexpand\iftabu@firstcline
726         \vrule depth\dimexpr\the\@tempdima
727           \ifdim\@tempdimb>\z@ width\the\@tempdimb\fi
728         \noexpand\else
729         \vrule height\dimexpr-\extrarowheight+\the\@tempdima
730           depth\dimexpr\extrarowheight
731           \ifdim\@tempdimb>\z@ width\the\@tempdimb\fi
732         \noexpand\fi
733       \ifdim\@tempdimc>\z@
734         \kern\the\dimexpr\@tempdimc/2\egroup\fi
735       \tabu@leaderfill}%
736   \fi\fi\fi
737 }% \tabu@lineoff

```

\tabu@colortest

```

738 \def\tabu@colortest{\futurelet\tabu@temp\tabu@linecolor}
739 \def\tabu@linecolor{%
740   \ifcase 0\if ,\noexpand\tabu@temp\else
741     \ifx\relax\tabu@temp\else
742     \ifx \@sptoken\tabu@temp1\else
743     \ifcat A\noexpand\tabu@temp2\else
744     3\fi\fi\fi\fi\relax
745   \def\tabu@next##1{\futurelet\tabu@temp\tabu@linecolor}%
746   \or\def\tabu@next{\tabu@gobblespace{\futurelet\tabu@temp\tabu@linecolor}}}%
747   \or\let\tabu@next\tabu@getlinecolor
748   \else\expandafter\remove@to@nnil
749   \fi \tabu@next
750 }% \tabu@linecolor
751 \def\tabu@getlinecolor#1\p@{%
752   \edef\@tempa{\zap@space #1 \@empty}%
753   \ifcsname\string\color@\@tempa\endcsname
754     \edef\tabu@color{\noexpand\noexpand\noexpand\color{\@tempa}}% \set@color
755   \fi\remove@to@nnil
756 }% \tabu@getlinecolor

```

\tabulinestyle \tabulinestyle{style=spec.}

```

757 \def\tabulinestyle#1{\@for\@tempa:=#1\do{\expandafter\tabu@linestyle\@tempa==\@nil}}
758 \def\tabu@linestyle#1=#2=#3\@nil{%
759   \begingroup \tabu@getlinespec {#2}\expandafter\gdef
760     \csname tabu@line@style@\detokenize{#1}\expandafter\endcsname
761     \expandafter{\expandafter\def\expandafter\tabu@xleaders
762       \expandafter{\tabu@xleaders}}}%
763   \endgroup
764 }% \tabu@linestyle
765 \expandafter\def \csname tabu@line@style@\endcsname {%
766   \let\tabu@xleaders \tabu@defaultleaders}%

```

```

767 \def\tabu@defaultleaders{\leaders
768   \iftabu@firstcline
769     \vrule depth \arrayrulewidth
770   \else
771     \vrule height\dimexpr-\extrarowheight+\arrayrulewidth
772           depth \extrarowheight
773   \fi
774   \tabu@leaderfill}
775 \let\tabu@leaderfill \hfil

```

7.9 Verbatim inside tabu with X columns

`\tabu@sanitizetext`

```

776 {\catcode32=13\relax\catcode'\^^@=13\relax
777 \gdef\tabu@verb{\@sanitize\makeatletter\catcode'\^=7\edef\^{\string^}%
778 \catcode32=13\let =\ \catcode'\^^@=13\def^^@{\par}\endlinechar\m@ne}%
779 }
780 \newcommand\tabu@sanitizetext[1][\ttfamily]{\begingroup
781   \tabu@verb #1\tabu@s@nitizetext}
782 \long\def\tabu@s@nitizetext#1{\@makeother\{\@makeother\}%
783   \everyeof{\noexpand}\scantokens{#1}\endgroup}

```

7.10 Numbers in tabu

`\tabudecimal`

`\tabudecimal`

```

784 \def\tabu@tabudecimal#1{%
785   \def\tabu@decimal{#1}\@temptokena{}%
786   \let\tabu@getdecimal@ \tabu@getdecimal@ignorespaces
787   \tabu@scandecimal
788 }% \tabudecimal
789 \def\tabu@scandecimal{\futurelet \tabu@temp \tabu@getdecimal@}
790 \def\tabu@skipdecimal#1{#1\tabu@scandecimal}
791 \def\tabu@getdecimal@ignorespaces{%
792   \ifcase 0\ifx\tabu@temp\ignorespaces\else
793     \ifx\tabu@temp\@sptoken1\else
794       2\fi\fi\relax
795   \let\tabu@getdecimal@ \tabu@getdecimal
796   \expandafter\tabu@skipdecimal
797   \or \expandafter\tabu@gobblespace\expandafter\tabu@scandecimal
798   \else \expandafter\tabu@skipdecimal
799   \fi
800 }% \tabu@getdecimal@ignorespaces
801 \def\tabu@get@decimal#1{\@temptokena\expandafter{\the\@temptokena #1}%
802   \tabu@scandecimal}
803 \def\do#1{%
804   \def\tabu@get@decimalspace#1{%
805     \@temptokena\expandafter{\the\@temptokena #1}\tabu@scandecimal}%
806 }\do{ }

```

`\tabu@getdecimal`

```

807 \def\tabu@getdecimal{%
808   \ifcase 0\ifx 0\tabu@temp\else
809     \ifx 1\tabu@temp\else
810     \ifx 2\tabu@temp\else
811     \ifx 3\tabu@temp\else

```

```

812         \ifx 4\tabu@temp\else
813         \ifx 5\tabu@temp\else
814         \ifx 6\tabu@temp\else
815         \ifx 7\tabu@temp\else
816         \ifx 8\tabu@temp\else
817         \ifx 9\tabu@temp\else
818         \ifx .\tabu@temp\else
819         \ifx ,\tabu@temp\else
820         \ifx -\tabu@temp\else
821         \ifx +\tabu@temp\else
822         \ifx e\tabu@temp\else
823         \ifx E\tabu@temp\else
824         \ifx\tabu@celleft\tabu@temp1\else
825         \ifx\ignorespaces\tabu@temp1\else
826         \ifx\@sptoken\tabu@temp2\else
827         3\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\relax
828         \expandafter\tabu@get@decimal
829     \or \expandafter\tabu@skipdecimal
830     \or \expandafter\tabu@get@decimalspace
831     \else\expandafter\tabu@printdecimal
832     \fi
833 }% \tabu@getdecimal
834 \def\tabu@printdecimal{%
835     \edef\tabu@temp{\the\@temptokena}%
836     \ifx\tabu@temp@empty\else
837     \ifx\tabu@temp\space\else
838         \expandafter\tabu@decimal\expandafter{\the\@temptokena}%
839     \fi\fi
840 }% \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`.

```

841 \newcommand*\savetabu[1]{\noalign{%
842     \ifx\#1\\\tabu@savewarn{}\{The tabu will not be saved\}\else
843     \@ifundefined{tabu@save@\string#1}\{\tabu@savewarn{#1}\{Overwriting\}\}%
844     {\toks@ \expandafter{\tabu@saved}%
845     \expandafter\xdef\csname tabu@saved@\string#1\endcsname{%
846         \tabu@target\the\tabu@target\relax
847         \the\toks@}}%
848     \fi}%
849 }% \savetabu

850 \def\tabu@saveX#1{\noalign{%
851     \ifx\#1\\\tabu@savewarn{}\{The tabu will not be saved\}\else
852     \@ifundefined{tabu@saved@\string#1}\{\tabu@savewarn{#1}\{Overwriting\}\}%
853     {\toks@ \expandafter{\tabu@saved}%
854     \expandafter\xdef\csname tabu@saved@\string#1\endcsname{%
855         \tabucolX\the\tabucolX\relax
856         \tabu@target\the\tabu@target\relax
857         \the\toks@}}%
858     \fi}%
859 }% \tabu@saveX

```

\tabu@savewarn Info for overwriting when `\savetabu` is used.

\tabu@saveerr

Error if `\usetabu` is called with an unknown argument.

```

860 \def\tabu@savewarn#1#2{\PackageInfo{tabu}
861   {User-name ‘#1’ already used for \string\savetabu
862   \MessageBreak #2}}%
863 \def\tabu@saveerr#1{\PackageError{tabu}
864   {User-name ‘#1’ is unknown for \string\usetabu
865   \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@celleft`, `\tabu@cellright` and `\tabu@cellralign` which have been placed on purpose into the user-defined tokens inserted in any preamble by the array package.

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

`\tabu@celleft` contains the font-modification information.

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

```

866 \def\tabu@rowfont{\noalign{\ifnum0='}\fi\tabu@row@font}
867 \newcommand*\tabu@row@font[2][]{%
868   \global\let\tabu@rowfontreset \tabu@rowfont@reset
869   \global\let\tabu@celleft \tabu@celleft
870   \global\let\tabu@cellright \tabu@cellright
871   \gdef\tabu@cellfont{#2}%
872   \ifcsname tabu@cell@#1\endcsname % row alignment
873     \csname tabu@cell@#1\endcsname \fi
874   \toks@{\expandafter {\tabu@celleft\tabu@cellfont}}% inside \noalign group ok
875   \xdef\tabu@celleft {\the\toks@}%
876   \ifnum0='}\fi}% end of noalign group
877 }% \rowfont

```

`\tabu@rowfont@reset` This macro resets `\tabu@celllalign`, `\tabu@celleft`, `\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.

```

878 \def\tabu@rowfont@reset{%
879   \noalign{%
880     \global\let\tabu@rowfontreset \@empty
881     \global\let\tabu@celleft \tabu@celleft
882     \global\let\tabu@cellright \tabu@cellright
883     \global\let\tabu@cellfont \@empty
884     \global\let\tabu@celllalign \@empty
885     \global\let\tabu@cellralign \@empty
886   }%
887 }% \tabu@rowfont@reset
888 \let\tabu@rowfontreset \@empty

```


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@celleft`, `\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.

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

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

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

either: $i = \text{count@} = \text{@tempcnta}$: 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@celleft` to this token (`\toks<i>`). This token is finished and will not change afterwards.

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

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

```

889 \newif\iftabu@cellright
890 \AtBeginDocument{\let\array@prepnext@tok \prepnext@tok}% original definition
891 \def\tabu@prepnext@tok{%
892   \ifnum \count@<\z@ % <first initialisation>
893     \@tempcnta \@M % <not initialized by array.sty>
894     \tabu@nbc\z@
895     \gdef\tabu@global@temp{\tabu@nbc\z@}%
896     \expandafter\let\csname tabu@tok@1L\endcsname \relax
897     \tabu@cellrightfalse
898   \else
899     \ifcase \numexpr \count@-\@tempcnta \relax % (case 0): prev. token is left
900       \advance \tabu@nbc\@ne
901       \expandafter\let\csname tabu@tok@\the\tabu@nbc R\endcsname \relax
902       \expandafter\gdef\expandafter\tabu@global@temp\expandafter{%
903         \tabu@global@temp \advance\tabu@nbc\@ne}%
904       \iftabu@cellright % before-previous token is right and is finished
905         \tabu@cellrightfalse % <only once>
906         \tabu@savetok R\tabu@preptokenright
907       \fi

```

```

908     \ifnum \tabu@nbcolls=\@ne
909         \tabu@savetok L% LEFT token is always empty unless >{...}
910     \fi
911     \tabu@preptokenleft
912 \or                                     % (case 1) previous token is right
913     \tabu@savetok R\tabu@cellrighttrue
914 \else % special column: do not change the token
915     \ifnum \tabu@nbcolls>\z@ %special column: always on the right of normal one
916         \tabu@savetok R%
917     \else % unless this is the very first column (\tabu@nbcolls=0)
918         \advance\tabu@nbcolls\@ne \tabu@savetok L\advance\tabu@nbcolls\m@ne
919     \fi
920     \iftabu@cellright % before-previous token is right
921         \tabu@cellrightfalse
922         \tabu@preptokenright
923     \fi
924 \fi % \ifcase
925 \fi
926 \array@prepnext@tok
927 }% \tabu@prepnext@tok
928 \def\tabu@preptokenright{%
929     \advance \count@ \m@ne
930     \toks\count@\expandafter {\the\toks\count@ \tabu@cellright \tabu@celllralign}%
931     \advance \count@ \@ne
932 }% \tabu@preptokenright
933 \def\tabu@preptokenleft{\toks\count@\expandafter {\expandafter\tabu@celllalign
934                                     \the\toks\count@ \tabu@cellleft}%
935 }% \tabu@preptokenleft
936 \def\tabu@savetok#1{\begingroup
937     \expandafter\tabu@savetok\csname tabu@tok@\the\tabu@nbcolls #1\endcsname
938 }% \tabu@savetok
939 \def\tabu@savetok#1{%
940     \@temptokena\toks\count@
941     \ifx#1\relax\else \@temptokena\expandafter\expandafter
942         \expandafter{\expandafter#1\the\@temptokena}%
943     \fi
944     \@temptokena\expandafter{\expandafter\def\expandafter#1\expandafter{%
945                                     \the\@temptokena}}%
946     \toks@\expandafter\expandafter\expandafter{\expandafter\tabu@global@temp
947                                     \the\@temptokena}%
948     \xdef\tabu@global@temp{\the\toks@}%
949     \expandafter\endgroup \the\@temptokena
950 }% \tabu@savetok

```

Neutralisation of glues and alignment modification

| | |
|---|---|
| <pre> \tabu@cellleft \tabu@celllalign \tabu@cellright \tabu@celllralign \tabu@cellfont </pre> | <pre> First initialisation to \@empty. 951 \let\tabu@cellleft\@empty 952 \let\tabu@cellright\@empty 953 \def\tabu@celllalign{\tabu@cellleft}% row font spec. applies to pre-column material 954 \let\tabu@celllralign\@empty 955 \let\tabu@cellfont\@empty </pre> |
| <pre> \tabu@cell@l \tabu@cell@c \tabu@cell@r \tabu@cell@j </pre> | <pre> 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. 956 \def\tabu@cell@l{% force alignment to left </pre> |

```

957 \gdef\tabu@celllalign{\tabu@removehfil
958 \raggedright\arraybackslash
959 \tabu@cellleft}%
960 \gdef\tabu@cellralign{\tabu@flush1\tabu@ignorehfil}%
961 \toks@{\expandafter{\tabu@cellleft\raggedright\arraybackslash}}% local
962 \xdef\tabu@cellleft{\the\toks@}%
963 }% \tabu@cell@l
964 \def\tabu@cell@c{% force alignment to center
965 \gdef\tabu@celllalign{\tabu@removehfil
966 \centering\arraybackslash
967 \tabu@flush{.5}\tabu@cellleft}%
968 \gdef\tabu@cellralign{\tabu@flush{.5}\tabu@ignorehfil}%
969 \toks@{\expandafter{\tabu@cellleft\centering\arraybackslash}}% local
970 \xdef\tabu@cellleft{\the\toks@}%
971 }% \tabu@cell@c
972 \def\tabu@cell@r{% force alignment to right
973 \gdef\tabu@celllalign{\tabu@removehfil
974 \raggedleft\arraybackslash
975 \tabu@flush1\tabu@cellleft}%
976 \gdef\tabu@cellralign{\tabu@ignorehfil}%
977 \toks@{\expandafter{\tabu@cellleft\raggedleft\arraybackslash}}% local
978 \xdef\tabu@cellleft{\the\toks@}%
979 }% \tabu@cell@r
980 \def\tabu@cell@j{% force justification (for p, m, b columns)
981 \gdef\tabu@celllalign{\tabu@justify\tabu@cellleft}%
982 \global\let\tabu@cellralign\@empty
983 \toks@{\expandafter{\tabu@cellleft\tabu@justify}}% local (noalign grp)
984 \xdef\tabu@cellleft{\the\toks@}%
985 }% \tabu@cell@j
986 \def\tabu@justify{%
987 \leftskip\z@skip \@rightskip\leftskip \rightskip\@rightskip
988 \parfillskip\@flushglue
989 }% \tabu@justify
990 %% ragged2e settings
991 \def\tabu@cell@L{% force alignment to left (ragged2e)
992 \gdef\tabu@celllalign{\tabu@removehfil
993 \RaggedRight\arraybackslash
994 \tabu@cellleft}%
995 \gdef\tabu@cellralign{\tabu@flush1\tabu@ignorehfil}%
996 \toks@{\expandafter{\tabu@cellleft\RaggedRight\arraybackslash}}%
997 \xdef\tabu@cellleft{\the\toks@}%
998 }% \tabu@cell@L
999 \def\tabu@cell@C{% force alignment to center (ragged2e)
1000 \gdef\tabu@celllalign{\tabu@removehfil
1001 \Centering\arraybackslash
1002 \tabu@flush{.5}\tabu@cellleft}%
1003 \gdef\tabu@cellralign{\tabu@flush{.5}\tabu@ignorehfil}%
1004 \toks@{\expandafter{\tabu@cellleft\Centering\arraybackslash}}%
1005 \xdef\tabu@cellleft{\the\toks@}%
1006 }% \tabu@cell@C
1007 \def\tabu@cell@R{% force alignment to right (ragged2e)
1008 \gdef\tabu@celllalign{\tabu@removehfil
1009 \RaggedLeft\arraybackslash
1010 \tabu@flush1\tabu@cellleft}%
1011 \gdef\tabu@cellralign{\tabu@ignorehfil}%
1012 \toks@{\expandafter{\tabu@cellleft\RaggedLeft\arraybackslash}}%
1013 \xdef\tabu@cellleft{\the\toks@}%
1014 }% \tabu@cell@R

```

```

1015 \def\tabu@cell@J{% force justification (ragged2e)
1016   \gdef\tabu@celllalign{\justifying\arraybackslash\tabu@cellleft}%
1017   \global\let\tabu@cellralign\@empty
1018   \toks@\expandafter{\tabu@cellleft\justifying\arraybackslash}%
1019   \xdef\tabu@cellleft{\the\toks@}%
1020 }% \tabu@cell@J
1021 \def\tabu@flush#1{%
1022   \iftabu@colortbl      % colortbl uses \hfill rather than \hfil
1023     \hskip \ifnum\currentgrouptype>13 \stretch{#1}%
1024     \else\ifdim#1pt<1pt \tabu@cellskip
1025     \else \stretch{#1}
1026     \fi\fi \relax
1027   \else                  % array.sty
1028     \ifnum \currentgrouptype>13\relax
1029       \hfil \hskip1sp
1030     \fi
1031   \fi
1032 }% \tabu@flush
1033 \AtBeginDocument{%
1034   \ifpackageloaded{ragged2e}
1035   {}
1036   {\let\tabu@cell@L \tabu@cell@l
1037    \let\tabu@cell@R \tabu@cell@r
1038    \let\tabu@cell@C \tabu@cell@c
1039    \let\tabu@cell@J \tabu@cell@j
1040   }%
1041 }% 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.

```

1042 \newskip\tabu@cellskip
1043 \let\tabu@hfil\hfil
1044 \let\tabu@hfill\hfill
1045 \let\tabu@hskip\hskip
1046 \def\tabu@removehfil{%
1047   \iftabu@colortbl
1048     \unkern \tabu@cellskip = \lastskip
1049     \ifnum\gluestretchorder\tabu@cellskip = \tw@ \hskip-\tabu@cellskip
1050     \else \tabu@cellskip = \z@skip
1051     \fi
1052   \else
1053     \ifdim\lastskip=1sp\unskip\fi
1054     \ifnum\gluestretchorder\lastskip = \@ne
1055       \hfilneg % \hfilneg for array.sty but not for colortbl...
1056     \fi
1057   \fi
1058 }% \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.

```

1059 \def\tabu@ignorehfil{%
1060   \aftergroup\tabu@nohfil
1061 }% \tabu@ignorehfil
1062 \def\tabu@nohfil{% \hfil -> do nothing + restore original \hfil
1063   \def\hfil{\let\hfil\tabu@hfil}% local to (alignment template) group
1064 }% \tabu@nohfil
1065 \AtBeginDocument{%
1066   \ifpackageloaded{colortbl}

```

```

1067   {%
1068   \def\tabu@nohfil{%
1069       \def\hfil{\let\hfil\tabu@hfil}% local to (alignment template) group
1070       \def\hfill{\let\hfill\tabu@hfill}% (colortbl uses \hfill) pfff...
1071       \def\hskip##1\relax{\let\hskip\tabu@hskip}}% local
1072   }% @ifpackageloaded colortbl
1073   {%
1074 }% AtBeginDocument

```

7.13 Utilities

tabu \fbox

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

```

1075 \def\tabu@fbox{%
1076     \leavevmode
1077     \let\color@bgroup\bgroup
1078     \def\color@egroup{\endgraf\egroup}%
1079     \afterassignment\tabu@begin@fbox
1080     \setbox\@tempboxa \hbox
1081 }% \tabu@fbox
1082 \def\tabu@begin@fbox{\color@bgroup\kern\fboxsep\aftergroup\tabu@end@fbox}
1083 \def\tabu@end@fbox{\kern\fboxsep\color@egroup\@framebex\relax}

```

\centering, \raggedright, \raggedleft

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

```

1084 \expandafter\def\expandafter\tabu@centering\expandafter{%
1085                                     \centering\arraybackslash}
1086 \expandafter\def\expandafter\tabu@raggedleft\expandafter{%
1087                                     \raggedleft\arraybackslash}
1088 \expandafter\def\expandafter\tabu@raggedright\expandafter{%
1089                                     \raggedright\arraybackslash}
1090 \AtBeginDocument{%
1091 \expandafter\def\expandafter\tabu@trivlist\expandafter{%
1092     \expandafter\let\expandafter\\expandafter\@centercr\@trivlist}%
1093 }%

```

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.

```

1094 \def\tabu@fix@arrayright{%
1095     \@ifpackageloaded{arydshln}
1096     {%
1097     \@ifpackageloaded{colortbl}
1098     {% colortbl + arydshln
1099     \def\tabu@endarray{%
1100         \adl@endarray \egroup \adl@arrayrestore \CT@end \egroup %<original>
1101         \@arrayright      % <FC>

```

```

1102      \gdef\@preamble{}}% <FC>
1103      }}% \endarray
1104      {%% arydshln / no colortbl
1105      \def\tabu@endarray{%
1106      \adl@endarray \egroup \adl@arrayrestore \egroup %<original>
1107      \@arrayright % <FC>
1108      \gdef\@preamble{}}% <FC>
1109      }}% \endarray
1110  }%
1111  {%
1112  \@ifpackageloaded{colortbl}
1113  {%% colortbl / no arydshln
1114  \def\tabu@endarray{%
1115  \crrc \egroup \egroup
1116  \@arrayright % <FC>
1117  \gdef\@preamble{}}\CT@end
1118  }}%
1119  {\PackageWarning{tabu}
1120   {\string\@arrayright\space is missing from the
1121   \MessageBreak definition of \string\endarray.
1122   \MessageBreak Comptability with delarray.sty is broken.}}%
1123  }%
1124 }% \tabu@fix@arrayright

```

arydshln @ columns

```

1125 \def\tabu@adl@xarraydashrule#1#2#3{%
1126   \ifnum\@lastchclass=\adl@class@start\else
1127   \ifnum\@lastchclass=\@ne\else
1128   \ifnum\@lastchclass=5 \else % <FC> @-arg (class 5) and !-arg (class 1)
1129     \adl@leftrulefalse \fi\fi % must be treated the same
1130   \fi
1131   \ifadl@zwvrule\else \ifadl@inactive\else
1132     \@addtopreamble{\vrule\@width\arrayrulewidth
1133     \@height\z@ \@depth\z@}\fi \fi
1134   \ifadl@leftrule
1135     \@addtopreamble{\adl@vlineL{\CT@arc@}{\adl@dashgapcolor}%
1136     {\number#1}#3}%
1137   \else \@addtopreamble{\adl@vlineR{\CT@arc@}{\adl@dashgapcolor}%
1138     {\number#2}#3}%
1139   \fi
1140 }% \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).

```

1141 \def\tabu@adl@act@endpbox{%
1142   \unskip \ifhmode \nobreak \fi \@finalstrut \@arstrutbox
1143   \egroup

```

```
1144 \adl@colhtdp \box\adl@box \hfil
1145 }% \tabu@adl@act@endpbox

1146 </package>
```


8 History

[2010/11/18 v1.3]

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

[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.

9 Index

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

| Symbols | |
|--|--|
| <code>\@arrayleft</code> | 429 |
| <code>\@arrayright</code> | 46, 1101, 1107, 1116, 1120 |
| <code>\@arstrutbox</code> | 66, 80, 83, 607, 613, 619, 636, 637, 661, 1142 |
| <code>\@centercr</code> | 179, 1092 |
| <code>\@defaultunits</code> | 669, 670, 677, 695, 698 |
| <code>\@elt</code> | 398, 399 |
| <code>\@finalstrut</code> | 1142 |
| <code>\@flushglue</code> | 129, 130, 131, 988 |
| <code>\@footnotetext</code> | 174, 404 |
| <code>\@halignto</code> | 70, 119, 135, 214, 215, 218, 219 |
| <code>\@let@token</code> | 199, 201, 202, 203, 204, 205, 304, 307, 308, 309, 310, 311, 312, 313, 314, 315, 316, 317, 318, 319, 320, 321 |
| <code>\@m</code> | 511 |
| <code>\@makeother</code> | 782 |
| <code>\@mpfn</code> | 240 |
| <code>\@multicnt</code> | 632, 638, 642, 643, 645, 650, 651, 653, 654, 658 |
| <code>\@preamble</code> | 376, 414, 1102, 1108, 1117 |
| <code>\@preamerr</code> | 155 |
| <code>\@preamerror</code> | 155 |
| <code>\@rightskip</code> | 987 |
| <code>\@sanitize</code> | 777 |
| <code>\@sptoken</code> | 203, 317, 742, 793, 826 |
| <code>\@tempcnta</code> | 401, 893, 899 |
| <code>\@tempdimb</code> | 701, 704, 709, 715, 716, 718, 720, 727, 731 |
| <code>\@tempdimc</code> | 714, 716, 717, 718, 721, 723, 724, 733, 734 |
| <code>\@tempskipa</code> | 526 |
| <code>\@temptokena</code> | 594, 785, 801, 805, 835, 838, 940, 941, 942, 944, 945, 947, 949 |
| <code>\@trivlist</code> | 179, 1092 |
| <code>\@undefined</code> | 5, 382 |
| <code>\@whilenum</code> | 615, 638, 658 |
| <code>\@xfootnotenext</code> | 175 |
| <code>\{</code> | 782 |
| <code>\}</code> | 782 |
| <code>\^</code> | 417, 776, 777, 778 |
| <code>_</code> | 778 |
| A | |
| <code>\adl@act@@endpbox</code> | 154 |
| <code>\adl@act@endpbox</code> | 153 |
| <code>\adl@arrayrestore</code> | 1100, 1106 |
| <code>\adl@box</code> | 1144 |

| | | | |
|--|----------------------------|--|----------------------------|
| <code>\adl@class@start</code> | 1126 | F | |
| <code>\adl@colhtdp</code> | 1144 | <code>\footnotetext</code> | 240, 244 |
| <code>\adl@dashgapcolor</code> | 1135, 1137 | <code>\futurelet</code> 114, 199, 304, 563, 738, 745, 746, 789 | |
| <code>\adl@endarray</code> | 1100, 1106 | G | |
| <code>\adl@leftrulefalse</code> | 1129 | <code>\gluestretchorder</code> | 1049, 1054 |
| <code>\adl@vlineL</code> | 1135 | H | |
| <code>\adl@vlineR</code> | 1137 | <code>\hbadness</code> | 400, 401 |
| <code>\adl@xarraydashrule</code> | 152 | <code>\hfilneg</code> | 1055 |
| <code>\afterassignment</code> 142, 143, 216, 351, 594, 694, 1079 | | <code>\hfuzz</code> | 187, 400, 402 |
| <code>\aftergroup</code> | 86, | <code>\ht</code> | 80, 83, 613, 637 |
| 103, 121, 123, 124, 133, 164, 527, 1060, 1082 | | I | |
| <code>\array</code> | 233 | <code>\ifadl@inactive</code> | 1131 |
| <code>\array@prepnext@tok</code> | 890, 926 | <code>\ifadl@leftrule</code> | 1134 |
| <code>\arraybackslash</code> | 958, 961, | <code>\ifadl@zwvrule</code> | 1131 |
| 966, 969, 974, 977, 993, 996, 1001, 1004, | | <code>\ifcase</code> | 201, 307, |
| 1009, 1012, 1016, 1018, 1085, 1087, 1089 | | 597, 645, 650, 699, 740, 792, 808, 899, 924 | |
| <code>\arraycolsep</code> | 225 | <code>\ifcat</code> | 191, 321, 568, 699, 743 |
| <code>\arrayrulewidth</code> | 79, | <code>\ifcsname</code> | 539, 682, 753, 872 |
| 227, 562, 569, 700, 703, 707, 769, 771, 1132 | | <code>\ifdim</code> | 127, 165, 169, 187, |
| <code>\AtEndOfPackage</code> | 5, 23, 27, 30 | 218, 260, 359, 449, 454, 461, 474, 476, | |
| B | | 480, 484, 545, 548, 715, 716, 717, 718, | |
| <code>\badness</code> | 533 | 719, 720, 721, 723, 727, 731, 733, 1024, 1053 | |
| <code>\begin</code> | 3 | <code>\iffalse</code> | 378, 411 |
| C | | <code>\ifin@</code> | 47, 250, 525 |
| <code>\catcode</code> | 10, 12, 417, 776, 777, 778 | <code>\ifnum</code> . . . 378, 411, 509, 510, 511, 514, 517, | |
| <code>\Centering</code> | 333, 1001, 1004 | 533, 628, 633, 636, 642, 645, 650, 653, | |
| <code>\centering</code> | 176, 328, 966, 969, 1085 | 671, 672, 673, 674, 866, 876, 892, 908, | |
| <code>\chardef</code> | 66 | 915, 1023, 1028, 1049, 1054, 1126, 1127, 1128 | |
| <code>\cl@ckpt</code> | 399 | <code>\iftabu@cellright</code> | 889, 904, 920 |
| <code>\color</code> | 754 | <code>\iftabu@colortbl</code> . . . 138, 608, 621, 1022, 1047 | |
| <code>\color@</code> | 753 | <code>\iftabu@firstcline</code> | 61, 637, 725, 768 |
| <code>\color@begingroup</code> | 578, 646 | <code>\iftabu@measuring</code> | 61, 255, 270, 438 |
| <code>\color@bgroup</code> | 1077, 1082 | <code>\iftabu@nested</code> | 61, 613 |
| <code>\color@egroup</code> | 1078, 1083 | <code>\iftabu@spread</code> | 61, 246, 380, 433 |
| <code>\color@endgroup</code> | 579, 648 | <code>\ignorespaces</code> | 792, 825 |
| <code>\copy</code> | 681 | <code>\in@</code> | 46, 250, 524 |
| <code>\crrr</code> | 606, 1115 | J | |
| <code>\currentgrouptype</code> | 1023, 1028 | <code>\justifying</code> | 1016, 1018 |
| D | | L | |
| <code>\d@llarend</code> | 191 | <code>\lastbox</code> | 168, 536 |
| <code>\delarray_(package_option)</code> | 26 | <code>\lastskip</code> | 1048, 1053, 1054 |
| <code>\detokenize</code> | 682, 683, 760 | <code>\leaders</code> | 767 |
| <code>\doublerulesep</code> | 85, 230 | <code>\leftskip</code> | 987 |
| E | | <code>\light_(package_option)</code> | 29 |
| <code>\endarray</code> 46, 47, 147, 156, 432, 1103, 1109, 1121 | | <code>\linegoal</code> | 24 |
| <code>\endlinechar</code> | 778 | <code>\linegoal_(package_option)</code> | 22 |
| <code>\endtabu</code> | 146 | <code>\linewidth</code> | 20, 24, 160, 165, 169, 397 |
| <code>\endtabular</code> | 148 | <code>\loop</code> | 640 |
| <code>\everycr</code> | 185, 593, 594 | <code>\lower</code> | 687 |
| <code>\everyeof</code> | 783 | M | |
| <code>\everypar</code> | 166, 167 | <code>\maxdimen</code> | 400 |
| <code>\extracolsep</code> | 129, 130, 131 | <code>\message</code> | 19, 662 |
| <code>\extrarowheight</code> | 80, | <code>\minrowclearance</code> | 609, 610, 622 |
| 81, 82, 228, 613, 687, 729, 730, 771, 772 | | | |
| <code>\extratabsurround</code> | 84, 229 | | |

| | |
|---|---|
| N | |
| <code>\NC@find</code> | 71, 104, 136, 232, 558 |
| <code>\NC@rewrite@X</code> | 102 |
| <code>\newcolumnntype</code> | 45 |
| <code>\newcount</code> | 51, 52, 53, 54, 55 |
| <code>\newdimen</code> | 56, 57, 58, 59, 60 |
| <code>\newsavebox</code> | 65 |
| <code>\newskip</code> | 1042 |
| <code>\newtoks</code> | 138, 237 |
| <code>\noalign</code> | 596, 606, 618, 628, 636, 660, 841, 850, 866, 874, 879 |
| <code>\numexpr</code> | 899 |
| O | |
| <code>\omit</code> | 641 |
| P | |
| <code>\PackageError</code> | 39, 41, 863 |
| <code>\PackageInfo</code> | 860 |
| <code>\PackageWarning</code> | 341, 346, 462, 688, 1119 |
| <code>\parfillskip</code> | 988 |
| <code>\prepnxt@tok</code> | 31, 183, 890 |
| <code>\ProcessOptions</code> | 43 |
| R | |
| <code>\RaggedLeft</code> | 334, 1009, 1012 |
| <code>\raggedleft</code> | 178, 329, 974, 977, 1087 |
| <code>\RaggedRight</code> | 332, 993, 996 |
| <code>\raggedright</code> | 177, 327, 958, 961, 1089 |
| <code>\remove@to@nnil</code> | 323, 748, 755 |
| <code>\repeat</code> | 657 |
| <code>\rightskip</code> | 987 |
| <code>\rowfont</code> | 11, 40, 184, 866 |
| S | |
| <code>\save@decl</code> | 32, 101 |
| <code>\savetabu</code> | 10, 377, 841, 861 |
| <code>\scantokens</code> | 783 |
| <code>\set@color</code> | 754 |
| <code>\setbox</code> | 168, 429, 536, 537, 1080 |
| <code>\setlength</code> | 261, 280 |
| <code>\span</code> | 656 |
| <code>\stretch</code> | 1023, 1025 |
| <code>\strip@pt</code> | 276, 287, 362 |
| T | |
| <code>\tabcolsep</code> | 226 |
| <code>\tabu</code> | 146 |
| <code>\tabu@@begin</code> | 220, 222, 236 |
| <code>\tabu@@cellleft</code> | 869, 881 |
| <code>\tabu@@cellright</code> | 870, 882 |
| <code>\tabu@add</code> | 643, 651, 666 |
| <code>\tabu@addphantomline</code> | 121 |
| <code>\tabu@adl@act@endpbox</code> | 153, 154, 1141, 1145 |
| <code>\tabu@adl@xarraydashrule</code> | 152, 1125, 1140 |
| <code>\tabu@align</code> | 223, 430, 637 |
| <code>\tabu@aligndefault</code> | 160, 162, 220 |
| <code>\tabu@arith</code> | 436, 446 |
| <code>\tabu@arrayleft@measure</code> | 393, 410 |
| <code>\tabu@arstrutbox</code> | 66, 619, 661 |
| <code>\tabu@begin</code> | 199 |
| <code>\tabu@begin@fbox</code> | 1079, 1082 |
| <code>\tabu@box</code> | 65, 429, 448, 449, 452, 454, 459, 474, 475, 484, 494, 497, 501, 536, 537, 540, 543, 548, 550, 554 |
| <code>\tabu@cell@C</code> | 999, 1006, 1038 |
| <code>\tabu@cell@c</code> | 956 |
| <code>\tabu@cell@J</code> | 1015, 1020, 1039 |
| <code>\tabu@cell@j</code> | 956 |
| <code>\tabu@cell@L</code> | 991, 998, 1036 |
| <code>\tabu@cell@l</code> | 956 |
| <code>\tabu@cell@R</code> | 1007, 1014, 1037 |
| <code>\tabu@cell@r</code> | 956 |
| <code>\tabu@cellfont</code> | 871, 874, 883, 951 |
| <code>\tabu@celllalign</code> | 884, 933, 951, 957, 965, 973, 981, 992, 1000, 1008 |
| <code>\tabu@celllalign</code> | 1016 |
| <code>\tabu@celleft</code> | 824, 869, 874, 875, 881, 934, 951, 959, 961, 962, 967, 969, 970, 975, 977, 978, 981, 983, 984, 994, 996, 997, 1002, 1004, 1005, 1010, 1012, 1013, 1016, 1018, 1019 |
| <code>\tabu@cellralign</code> | 885, 930, 951, 960, 968, 976, 982, 995, 1003, 1011, 1017 |
| <code>\tabu@cellright</code> | 870, 882, 930, 951 |
| <code>\tabu@cellrightfalse</code> | 897, 905, 921 |
| <code>\tabu@cellrighttrue</code> | 913 |
| <code>\tabu@cellskip</code> | 1024, 1042, 1048, 1049, 1050 |
| <code>\tabu@centering</code> | 176, 1084 |
| <code>\tabu@ccline</code> | 629, 630, 631 |
| <code>\tabu@cnt</code> | 51, 351, 354, 391, 447, 514, 517 |
| <code>\tabu@collect</code> | 415, 417 |
| <code>\tabu@collectbody</code> | 417 |
| <code>\tabu@color</code> | 567, 578, 694, 697, 708, 722, 754 |
| <code>\tabu@colortblfalse</code> | 49 |
| <code>\tabu@colortbltrue</code> | 49 |
| <code>\tabu@colortest</code> | 694, 738 |
| <code>\tabu@dashgap</code> | 627, 702, 705, 716 |
| <code>\tabu@decimal</code> | 785, 838 |
| <code>\tabu@defaultleaders</code> | 678, 691, 766, 767 |
| <code>\tabu@defleaders</code> | 679, 680, 681, 686 |
| <code>\tabu@end@fbox</code> | 1082, 1083 |
| <code>\tabu@endarray</code> | 47, 156, 1099, 1105, 1114 |
| <code>\tabu@endenvir</code> | 421, 442 |
| <code>\tabu@endTRIAL</code> | 429 |
| <code>\tabu@everycr</code> | 185, 593, 595 |
| <code>\tabu@everycr@tok</code> | 138 |
| <code>\tabu@fbox</code> | 182, 1075, 1081 |
| <code>\tabu@first@ccline</code> | 592, 597, 598, 599, 601, 623 |
| <code>\tabu@firstccline</code> | 35 |
| <code>\tabu@firstcclinefalse</code> | 600 |
| <code>\tabu@firstcclinetrue</code> | 591 |
| <code>\tabu@fix@arrayright</code> | 48, 1094, 1124 |
| <code>\tabu@flush</code> | 960, 967, 968, 975, 995, 1002, 1003, 1010, 1021, 1032 |
| <code>\tabu@footnotes</code> | 163, 164, 237 |
| <code>\tabu@footnotetext</code> | 174, 238 |
| <code>\tabu@get@decimal</code> | 801, 828 |
| <code>\tabu@get@decimalspace</code> | 804, 830 |
| <code>\tabu@getdecimal</code> | 795, 807 |

| | | | |
|--|---|--|---|
| <code>\tabu@getdecimal@</code> | 786, 789, 795 | <code>\tabu@printdecimal</code> | 831, 834, 840 |
| <code>\tabu@getdecimal@ignorespaces</code> | 786, 791, 800 | <code>\tabu@pushbegins</code> | 419, 428 |
| <code>\tabu@getline</code> | 684, 693 | <code>\tabu@raggedleft</code> | 178, 1086 |
| <code>\tabu@getlinecolor</code> | 570, 747, 751, 756 | <code>\tabu@raggedright</code> | 177, 1088 |
| <code>\tabu@getlinespec</code> | 632, 676, 759 | <code>\tabu@removehfil</code> | |
| <code>\tabu@gettarget</code> | 214, 215, 216 | | 957, 965, 973, 992, 1000, 1008, 1042 |
| <code>\tabu@global@temp</code> | | <code>\tabu@restoreeverycr</code> | 593, 594 |
| | 103, 157, 895, 902, 903, 946, 948 | <code>\tabu@rewrite@X</code> | 69, 76, 245, 396 |
| <code>\tabu@global@X</code> | 157, | <code>\tabu@rewrite@Xnested</code> | 250, 396 |
| | 256, 263, 273, 282, 395, 399, 411, 456 | <code>\tabu@rewrite@Xrestore</code> | 76, 250 |
| <code>\tabu@gobblespace</code> | 68, 208, 746, 797 | <code>\tabu@rewrite@Xspread</code> | 246, 269 |
| <code>\tabu@halignto</code> | 122, 135, 219, 232, 266, 288, 488 | <code>\tabu@rewrite@Xto</code> | 247, 254 |
| <code>\tabu@header</code> | 450, 492, 504, 513 | <code>\tabu@rewritefirst</code> | 91, |
| <code>\tabu@hfil</code> | 1043, 1063, 1069 | | 105, 106, 107, 108, 110, 111, 188, 235 |
| <code>\tabu@hfill</code> | 1044, 1070 | <code>\tabu@rewritefirst_(new_column_type)</code> .. | 91 |
| <code>\tabu@hfuzz</code> | 187, 454, 506 | <code>\tabu@rewritelast</code> | 94, |
| <code>\tabu@hskip</code> | 1045, 1071 | | 98, 104, 112, 114, 115, 116, 117, 118, 137 |
| <code>\tabu@ignorehfil</code> | | <code>\tabu@rewritelast_(new_column_type)</code> .. | 112 |
| | 960, 968, 976, 995, 1003, 1011, 1059 | <code>\tabu@row@font</code> | 866, 867 |
| <code>\tabu@justify</code> | 981, 983, 986, 989 | <code>\tabu@rowfont</code> | 33, 184, 866 |
| <code>\tabu@lasttry</code> | 392, 449, 452 | <code>\tabu@rowfont@reset</code> | 868, 878 |
| <code>\tabu@leaderfill</code> | 687, 735, 774, 775 | <code>\tabu@rowfontreset</code> | 603, 868, 880, 888 |
| <code>\tabu@linecolor</code> | 738, 739, 745, 746, 750 | <code>\tabu@sanitizetext</code> | 158, 181, 776 |
| <code>\tabu@linedash</code> | 626, 693, 718 | <code>\tabu@save@decl</code> | 32, 101, 190, 198 |
| <code>\tabu@lineoff</code> | 702, 705, 710, 713, 737 | <code>\tabu@savd</code> | 224, 231, 844, 853 |
| <code>\tabu@lineon</code> | 693, 696, 712 | <code>\tabu@savdpreamble</code> | 376, 414 |
| <code>\tabu@lines</code> | 36, 92, 99, 581 | <code>\tabu@saveerr</code> | 73, 75, 860 |
| <code>\tabu@linestyle</code> | 757, 758, 764 | <code>\tabu@savetok</code> | 937, 939 |
| <code>\tabu@maybecolor</code> | 575, 694, 700, 701, 704, 707, 714 | <code>\tabu@savetok</code> | 906, 909, 913, 916, 918, 936, 938, 950 |
| <code>\tabu@measuringfalse</code> | 186, 438 | <code>\tabu@savewarn</code> | 842, 843, 851, 852, 860 |
| <code>\tabu@measuringtrue</code> | 259, 278, 465, 490 | <code>\tabu@saveX</code> | 377, 405, 841 |
| <code>\tabu@message</code> | 18, | <code>\tabu@scandecimal</code> | 787, 789, 790, 797, 802, 805 |
| | 19, 450, 453, 455, 473, 479, 480, 483, 484 | <code>\tabu@settarget</code> | 149, 199 |
| <code>\tabu@message@arith</code> | 453, 492 | <code>\tabu@setup</code> | 149, 151 |
| <code>\tabu@message@reached</code> | 455, 504 | <code>\tabu@setup@everycr</code> | 124, 133, 413, 590 |
| <code>\tabu@message@spread</code> | 473, 499 | <code>\tabu@setup@TRIAL</code> | 379, 395, 407 |
| <code>\tabu@minrowclearance</code> | 609, 622 | <code>\tabu@skipdecimal</code> | 790, 796, 798, 829 |
| <code>\tabu@msg@align</code> | 479, 483, 493, 494, | <code>\tabu@spread</code> | 210, 215 |
| | 495, 496, 497, 500, 501, 502, 503, 505, 508 | <code>\tabu@spread@endhbox</code> | 523 |
| <code>\tabu@naturalX</code> | 56, | <code>\tabu@spread@startpbox</code> | 387, 523 |
| | 383, 471, 475, 503, 534, 540, 543, 545, 546 | <code>\tabu@spreadarith</code> | 435, 446 |
| <code>\tabu@naturalX@max</code> | 384, 471, 545, 546 | <code>\tabu@spreadfalse</code> | 186, 433 |
| <code>\tabu@naturalX@min</code> | 385, 472, 548, 550 | <code>\tabu@spreadheader</code> | 499, 518 |
| <code>\tabu@nbcols</code> | 51, | <code>\tabu@spreadtarget</code> | 56, |
| | 615, 658, 667, 671, 673, 674, 894, 895, | | 279, 286, 474, 475, 476, 477, 480, 500 |
| | 900, 901, 903, 908, 915, 917, 918, 937 | <code>\tabu@spreadtrue</code> | 215 |
| <code>\tabu@NC@list</code> | 97, 109, 113 | <code>\tabu@start</code> | 54, 633, 638, 669, 671, 672 |
| <code>\tabu@NC@rewrite@X</code> | 69, 102 | <code>\tabu@start@stop</code> | 667, 668, 675 |
| <code>\tabu@nestedfalse</code> | 161 | <code>\tabu@startpboxORI</code> | 386, 434, 529 |
| <code>\tabu@nestedtrue</code> | 159 | <code>\tabu@startstop</code> | 632, 667 |
| <code>\tabu@next</code> | 745, 746, 747, 749 | <code>\tabu@stop</code> | 55, 633, 645, 650, 653, 670, 673, 674 |
| <code>\tabu@nocline</code> | 34, 41 | <code>\tabu@tabudecimal</code> | 180, 784 |
| <code>\tabu@nohfil</code> | 1060, 1062, 1064, 1068 | <code>\tabu@target</code> .. | 56, 127, 200, 216, 218, 232, |
| <code>\tabu@norowfont</code> | 33, 39 | | 260, 261, 265, 279, 280, 285, 389, 454, |
| <code>\tabu@nowrite</code> | 140, 403 | | 456, 459, 474, 476, 477, 479, 480, 483, |
| <code>\tabu@prep@TRIAL</code> | 123, 375 | | 484, 488, 489, 495, 497, 527, 534, 846, 856 |
| <code>\tabu@prepnext@tok</code> | 31, 183, 889 | <code>\tabu@temp</code> | 77, 114, 120, |
| <code>\tabu@preptokenleft</code> | 911, 933, 935 | | 127, 129, 130, 131, 136, 232, 253, 295, |
| <code>\tabu@preptokenright</code> | 906, 922, 928, 932 | | 362, 365, 367, 371, 558, 562, 563, 568, |

UVWXZ