The breakurl package[*]

Vilar Camara Neto
neto@dcc.ufmg.br

July 15, 2008

1 Introduction

The hyperref package brings a lot of interesting tools to "boost" documents produced by LaTeX. For instance, a PDF file can have clickable links to references, section headings, URLs, etc.

Generating a link to a URL may be a concern if it stands near the end of a line of text. When one uses pdfLaTeX to directly generate a PDF document, there's no problem: the driver can break a link across more than one line. However, the `dvips` driver (used when one prefers the LaTeX → DVI → PostScript → PDF path), because of internal reasons, can't issue line breaks in the middle of a link. Sometimes this turns into a serious aesthetic problem, generating largely underfull/overfull paragraphs; sometimes the final effect is so bad that the link can't even fit inside the physical page limits.

To overcome that `dvips` limitation, the breakurl package was designed with a simple solution: it provides a command called `\burl` (it stands for "breakable URL"). Instead of generating one long, atomic link, this command breaks it into small pieces, allowing line breaks between them. Each sequence of pieces that stand together in one line are converted to a hot (clickable) link in the PDF document.

2 How to use it

At the preamble, just put a `\usepackage{breakurl}` somewhere *after* the `\usepackage{hyperref}`. The `\burl` command is defined and, by default, the package also turns the `\url` command into a synonym of `\burl`. This might come in handy, for example, if you use BibTeX, your `.bib`-file has lots of `\url` commands and you don't want to replace them by `\burl`. If, for some reason, you want to preserve the original behavior of `\url` (i.e., it creates an unbreakable link), you must supply the `preserveurlmacro` option to the package (see Section 2.1).

[*]This document corresponds to breakurl v1.23, dated 2008/07/15.

In the middle of the document, the syntax of `\burl` (and its synonym `\url`) is the same as the original `\url`: `\burl{`⟨*URL*⟩`}`, where ⟨*URL*⟩ is, of course, the address to point to. You don't need to care (escape) about special characters like `%`, `&`, `_`, and so on.

Another handy command is `\burlalt{`⟨*ActualURL*⟩`}{`⟨*DisplayedURL*⟩`}`, where ⟨*ActualURL*⟩ is the actual link and ⟨*DisplayedURL*⟩ is the link text to be displayed in the document. For consistency, `\urlalt` is a synonym of `\burlalt`, unless the `preserveurlmacro` package option is specified.[1]

The default behavior of the package is to break the link after any sequence of these characters:

'`:`' (colon)	'`/`' (slash)	'`.`' (dot)
'`?`' (question mark)	'`#`' (hash)	'`&`' (ampersand)
'`_`' (underline)	'`,`' (comma)	'`;`' (semicolon)
'`!`' (exclamation mark)		

You can add '`-`' (hyphen) to this list (see below), but read why I decided to keep it out of the default list.

Remember that breaks are allowed *after* a sequence of these characters, so a link starting with `http://` will never break before the second slash.

Also note that I decided not to include the '`-`' (hyphen) character in this default list. It's to avoid a possible confusion when someone encounters a break after a hyphen, e.g.:

> Please visit the page at `http://internet-`
> `page.com`, which shows...

Here comes the doubt: the author is pointing to `http://internet-page.com` or to `http://internetpage.com`? The breakurl package *never* adds a hyphen when a link is broken across lines — so, the first choice would be the right one —, but we can't assume that the reader knows this rule; so, I decided to disallow breaks after hyphens. Nevertheless, if you want to overcome my decision, use the `hyphenbreaks` option:

```
\usepackage[hyphenbreaks]{breakurl}
```

2.1 Package options

When using the `\usepackage` command, you can give some options to customize the package behavior. Possible options are explained below:

- `hyphenbreaks`

  Instructs the package to allow line breaks after hyphens.

[1] The `\burlalt` command resembles `\hyperref`'s `\href`, but since it works in a different manner I decided not to call it "`\bhref`".

- `preserveurlmacro`

  Instructs the package to leave the `\url` command exactly as it was before the package inclusion. Also, `\urlalt` isn't defined as a synonym of `\burlalt`. In either case (i.e., using `preserveurlmacro` or not), the breakable link is available via the `\burl` command.

- `vertfit=`⟨*criterion*⟩

  Estabilishes how the link rectangle's height (and depth) will behave against the corresponding URL text's vertical range. There are three options for ⟨*criterion*⟩: `local` makes each rectangle fit tightly to its text's vertical range. This means that each line of a link broken across lines can have a rectangle with different vertical sizes. `global` first calculates the height (and depth) to enclose the entire link and preserves the measures, so the link maintains the vertical size across lines. `strut` goes even further and ensures that the rectangle's vertical range corresponds to `\strut`. With this option, rectangles in adjacent lines can overlap. The default is `vertfit=local`.

2.2 Additional comments

As stated in the introduction, the breakurl is designed for those compiling documents via LaTeX, not pdfLaTeX. In the latter case, the package doesn't (re)define the `\url` command: it only defines `\burl` to be a synonym of whatever `\url` is defined (e.g., via `url` or hyperref packages). Of course, `\burl` may behave differently compared to (non-pdf)LaTeX, because then the system will use other rules to make line breaks, spacing, etc.

Also, this package was not designed to nor tested against other drivers: it's compatible with dvips only.

2.3 Changelog

(presented in reverse chronological order)

v1.23 `\hypersetup` now works anywhere.

v1.22 Corrected blank lines appearing inside tables.

v1.21 `\burlalt` and the synonym `\urlalt` now work with pdflatex. Also, there are a couple of bug fixes (thank you again, Heiko).

v1.20 An update was needed because hyperref's internals were changed. (Thanks Heiko for sending the correction patch.) Troubleshooting now includes a note about `\sloppy`.

v1.10 A new command, `\burlalt` (and the synonym `\urlalt`), allows one to specify different values for actual and displayed link.

v1.01 Fixed a bug that was happening when a link is split into more than one page.

v1.00 The `\UrlLeft` and `\UrlRight` (defined and explained in the `url` package) are now partially supported. By "partially" I mean: although the original (`url.sty`'s) documentation allows defining `\UrlLeft` as a command with one argument (things such `\def\UrlLeft#1\UrlRight{`*do things with #1*`}`, this isn't expected to work with breakurl. Please use only the basic definition, e.g.: `\def\UrlLeft{<url:\ } \def\UrlRight{>}`.

v0.04 Corrected a bug that prevented URLs to be in color, in despite of hyperref's `colorlinks` and `urlcolor` options. Added an error message if `vertfit` parameter is invalid.

v0.03 The package was tested against `pdfeTeX` engine (which may be the default for some `teTeX` distributions). Introduced a new package option, `vertfit`.

v0.02 The main issue of the initial release — the odd-looking sequence of small links in the same line, if one uses hyperref's link borders — was resolved: now the package generates only one rectangle per line. Also, breaks after hyphens, which weren't allowed in the previous release, are now a users' option. Finally, the package can be used with pdfLaTeX (in this case, `\burl` is defined to be a synonym of the original `\url` command).

v0.01 Initial release.

2.4 Troubleshooting

Here comes a few notes about known issues:

- I received some comments saying that in some cases breakurl destroys the formatting of the document: the left/right margins aren't respected, justification becomes weird, etc. In all these cases, the problems were corrected when other packages were upgraded, notabily `xkeyval`.

- If your compilation issues the following error:

  ```
  ! Undefined control sequence.
  <argument> \headerps@out ...
  ```

  then you need to specify the `dvips` driver as an option to the hyperref package, e.g.:

  ```
  \usepackage[dvips]{hyperref}
  ```

  However, this is related to old versions of hyperref: the package is able to automatically determine the driver in current versions. It's probabily better to update your LaTeX system.

- If everything compiles but sometimes URLs still don't respect the right margin, don't blame the package yet :-) . Roughly speaking, by default the right margin is a limit to be respected "only if word spacing is okay", so it may be ignored even when URLs aren't used. Check the following paragraph:

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Proin zzzzzzzzzzzzzzzzzzzzzzzz...

To overcome this (and make right margins a hard limit) use the command `\sloppy`, preferably before `\begin{document}`. This makes the previous paragraph look like:

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Proin zzzzzzzzzzzzzzzzzzzzzzzz...

As a drawback word spacing becomes terrible, but now the text is kept inside designed margins. You should decide what looks better.

2.5 Acknowledgments

Thanks to Hendri Adriaens, Donald Arseneau, Michael Friendly, Morten Høgholm, David Le Kim, Damian Menscher, Tristan Miller, Heiko Oberdiek, Christoph Schiller, Xiaotian Sun, Michael Toews, David Tulloh, Adrian Vogel, and Jinsong Zhao for suggestions, bug reports, comments, and corrections. A special thanks to the participants of `comp.text.tex` newsgroups for their constant effort to help hundreds of people in the beautiful world of TEX/LATEX.

3 Source code

This section describes the `breakurl.sty` source code.

The breakurl requires some packages, so let's include them:

```
\RequirePackage{xkeyval}
\RequirePackage{ifpdf}
```

Is the document being processed by pdfLATEX? (Actually, is there a PDF file being directly generated?) Then, well, this package doesn't apply: let's just define `\burl` to call the default `\url`.

```
\ifpdf
  % Dummy package options
  \DeclareOptionX{preserveurlmacro}{}
  \DeclareOptionX{hyphenbreaks}{}
  \DeclareOptionX{vertfit}{}
  \ProcessOptionsX\relax

  \PackageWarning{breakurl}{%
  You are using breakurl while processing via pdflatex.\MessageBreak
  \string\burl\space will be just a synonym of \string\url.\MessageBreak}
  \DeclareRobustCommand{\burl}{\url}
  \DeclareRobustCommand*{\burlalt}{\hyper@normalise\burl@alt}
  \def\burl@alt#1#2{\hyper@linkurl{\Hurl{#1}}{#2}}
  \expandafter\endinput
\fi
```

Since breakurl is an extension to hyperref, let's complain loudly if the latter was not yet loaded:

```
\@ifpackageloaded{hyperref}{}{%
  \PackageError{breakurl}{The breakurl depends on hyperref package}%
  {I can't do anything. Please type X <return>, edit the source file%
  \MessageBreak
  and add \string\usepackage\string{hyperref\string} before
  \string\usepackage\string{breakurl\string}.}
  \endinput
}
```

The package options are handled by `\newif`s, which are declared and initialised:

```
\newif\if@preserveurlmacro\@preserveurlmacrofalse
\newif\if@burl@fitstrut\@burl@fitstrutfalse
\newif\if@burl@fitglobal\@burl@fitglobalfalse
```

`\burl@toks` The breakurl package uses a token list to store characters and tokens until a break point is reached:

```
\newtoks\burl@toks
```

`\burl@charlist` `\burl@defifstructure` The following support routines are designed to build the conditional structure that is the kernel of `\burl`: comparing each incoming character with the list of "breakable" characters and taking decisions on that. This conditional structure is built by `\burl@defifstructure` — which is called only at the end of package loading, because the character list (stored in `\burl@charlist`) can be modified by the `hyphenbreaks` option.

```
\def\burl@charlist{}
\def\burl@addtocharlist#1{%
  \expandafter\gdef\expandafter\burl@charlist\expandafter{%
    \burl@charlist #1}%
}

\bgroup
\catcode`\&=12\relax
\hyper@normalise\burl@addtocharlist{:/.?#&_,;!}
\egroup

\def\burl@growmif#1{%
  \expandafter\def\expandafter\burl@mif\expandafter{%
    \burl@mif\def\burl@ttt{#1}\ifx\burl@ttt\@nextchar\@burl@breakabletrue\else
  }%
}
\def\burl@growmfi{%
  \expandafter\def\expandafter\burl@mfi\expandafter{\burl@mfi\fi}%
}
\def\burl@melse{%
  \if@burl@breakable\burl@flush\linebreak[0]\@burl@breakablefalse\fi
  \expandafter\expandafter\expandafter\burl@toks
    \expandafter\expandafter\expandafter{%
    \expandafter\the\expandafter\burl@toks\@nextchar}%
}
```

```
\def\burl@defifstructure{%
  \def\burl@mif{}%
  \def\burl@mfi{}%
  \expandafter\@tfor\expandafter\@nextchar\expandafter:\expandafter=%
    \burl@charlist\do{%
    \expandafter\burl@growmif\@nextchar
    \burl@growmfi
  }%
}

\AtEndOfPackage{\burl@defifstructure}
```

The package options are declared and handled as follows:

```
\def\burl@setvertfit#1{%
  \lowercase{\def\burl@temp{#1}}%
  \def\burl@opt{local}\ifx\burl@temp\burl@opt
    \@burl@fitstrutfalse\@burl@fitglobalfalse
  \else\def\burl@opt{strut}\ifx\burl@temp\burl@opt
    \@burl@fitstruttrue\@burl@fitglobalfalse
  \else\def\burl@opt{global}\ifx\burl@temp\burl@opt
    \@burl@fitstrutfalse\@burl@fitglobaltrue
  \else
    \PackageWarning{breakurl}{Unrecognized vertfit option `\burl@temp'.%
    \MessageBreak
    Adopting default `local'}
    \@burl@fitstrutfalse\@burl@fitglobalfalse
  \fi\fi\fi
}

\DeclareOptionX{preserveurlmacro}{\@preserveurlmacrotrue}
\DeclareOptionX{hyphenbreaks}{\bgroup
  \catcode`\&=12\relax\hyper@normalise\burl@addtocharlist{-}%
  \egroup}
\DeclareOptionX{vertfit}[local]{\burl@setvertfit{#1}}

\ProcessOptionsX\relax
```

These supporting routines are modified versions of those found in the hyperref package. They were adapted to allow a link to be progressively built, i.e., when we say "put a link rectangle here", the package will decide if this will be made.

```
\def\burl@hyper@linkurl#1#2{%
  \begingroup
    \hyper@chars
    \burl@condpdflink{#1}%
  \endgroup
}

\def\burl@condpdflink#1{%
  \literalps@out{
    /burl@bordercolor {\@urlbordercolor} def
  /burl@border {\@pdfborder} def
}%
\if@burl@fitstrut
  \sbox\pdf@box{#1\strut}%
\else\if@burl@fitglobal
  \sbox\pdf@box{\burl@url}%
\else
  \sbox\pdf@box{#1}%
\fi\fi
\dimen@\ht\pdf@box\dimen@ii\dp\pdf@box
\sbox\pdf@box{#1}%
\ifdim\dimen@ii=\z@
  \literalps@out{BU.SS}%
\else
  \lower\dimen@ii\hbox{\literalps@out{BU.SS}}%
\fi
\ifHy@breaklinks\unhbox\else\box\fi\pdf@box
\ifdim\dimen@=\z@
  \literalps@out{BU.SE}%
\else
  \raise\dimen@\hbox{\literalps@out{BU.SE}}%
\fi
\pdf@addtoksx{H.B}%
}
```

`\burl`

`\burl` prepares the catcodes (via `\hyper@normalise`) and calls the `\burl@` macro, which does the actual work.

```
\DeclareRobustCommand*{\burl}{%
  \leavevmode
  \begingroup
  \let\hyper@linkurl=\burl@hyper@linkurl
  \catcode`\&=12\relax
  \hyper@normalise\burl@
}
```

`\burlalt`

`\burlalt` does the same as `\burl`, but calls another macro (`\burl@alt`) to read two following arguments instead of only one.

```
\DeclareRobustCommand*{\burlalt}{%
  \begingroup
  \let\hyper@linkurl=\burl@hyper@linkurl
  \catcode`\&=12\relax
  \hyper@normalise\burl@alt
}
```

`\burl@`
`\burl@alt`
`\burl@@alt`

`\burl@` `{`⟨*URL*⟩`}` just eats the next argument to define the URL address and the link to be displayed. Both are used by `\burl@doit`.

`\burl@alt` `{`⟨*ActualURL*⟩`}` and `\burl@@alt` `{`⟨*DisplayedURL*⟩`}` work together to eat the two arguments (the actual URL to point to and the link text to be displayed). Again, both are used by `\burl@doit`.

```
\newif\if@burl@breakable

\bgroup
\catcode`\&=12\relax
\gdef\burl@#1{%
  \def\burl@url{#1}%
  \def\burl@urltext{#1}%
  \burl@doit
}

\gdef\burl@alt#1{%
  \def\burl@url{#1}%
  \hyper@normalise\burl@@alt
}
\gdef\burl@@alt#1{%
  \def\burl@urltext{#1}%
  \burl@doit
}
```

`\burl@doit` `\burl@doit` works much like hyperref's `\url@` macro (actually, this code macro was borrowed and adapted from the original `\url@`): it builds a series of links, allowing line breaks between them. The characters are accumulated and eventually flushed via the `\burl@flush` macro.

Support for `\UrlLeft`/`\UrlRight`: The `\UrlRight` is emptied until the very last flush (when it is restored). The `\UrlLeft` is emptied after the first flush. So, any string defined in those macros are meant to be displayed only before the first piece and after the last one, which (of course) is what we expect to happen. Unfortunately, breaking doesn't happen inside those strings, since they're not rendered verbatim (and so they aren't processed inside the breaking mechanism).

```
\gdef\burl@doit{%
  \burl@toks{}%
  \let\burl@UrlRight=\UrlRight
  \let\UrlRight=\empty
  \@ifundefined{@urlcolor}{\Hy@colorlink\@linkcolor}{\Hy@colorlink\@urlcolor}%
  \@burl@breakablefalse
  \expandafter\@tfor\expandafter\@nextchar\expandafter:\expandafter=%
    \burl@urltext\do{%
    \expandafter\burl@mif\expandafter\burl@melse\burl@mfi
    \if@burl@breakable
      \expandafter\expandafter\expandafter\burl@toks
        \expandafter\expandafter\expandafter{%
        \expandafter\the\expandafter\burl@toks\@nextchar}%
    \fi
  }%
  \let\UrlRight=\burl@UrlRight
  \burl@flush
  \literalps@out{BU.E}%
  \Hy@endcolorlink
  \endgroup
}
\egroup
```

\burl@flush This macro flushes the characters accumulated during the `\burl@` processing, creating a link to the URL.

```
\def\the@burl@toks{\the\burl@toks}

\def\burl@flush{%
  \expandafter\def\expandafter\burl@toks@def\expandafter{\the\burl@toks}%
  \literalps@out{/BU.L (\burl@url) def}%
  \hyper@linkurl{\expandafter\Hurl\expandafter{\burl@toks@def}}{\burl@url}%
  \global\burl@toks{}%
  \let\UrlLeft=\empty
}%
```

Now the synonyms `\url` and `\urlalt` are (re)defined, unless the `preserveurlmacro` option is given.

```
\if@preserveurlmacro\else\let\url\burl\let\urlalt\burlalt\fi
```

Internally, the package works as follows: each link segment (i.e., a list of non-breakable characters followed by breakable characters) ends with a PDF command that checks if the line ends here. If this check is true, then (and only then) the PDF link rectangle is built, embracing all link segments of this line.

To make that work, we need some code to work at the PostScript processing level. The supporting routines to do so are introduced in the PS dictionary initialization block via specials. Each routine is explained below.

The variables used here are: `burl@stx` and `burl@endx`, which defines the link's horizontal range; `burl@boty` and `burl@topy`, which defines the link's vertical range; `burl@llx`, `burl@lly`, `burl@urx`, and `burl@ury`, which define the bounding box of the current link segment (they resemble the hyperref's `pdf@llx`–`pdf@ury` counterparts); and `BU.L`, which holds the target URL.

```
\AtBeginDvi{%
  \headerps@out{%
    /burl@stx null def
```

`BU.S` is called whenever a link begins:

```
    /BU.S {
      /burl@stx null def
    } def
```

`BU.SS` is called whenever a link segment begins:

```
    /BU.SS {
      currentpoint
      /burl@lly exch def
      /burl@llx exch def
      burl@stx null ne {burl@endx burl@llx ne {BU.FL BU.S} if} if
      burl@stx null eq {
        burl@llx dup /burl@stx exch def /burl@endx exch def
        burl@lly dup /burl@boty exch def /burl@topy exch def
    } if
    burl@lly burl@boty gt {/burl@boty burl@lly def} if
  } def
```

BU.SE is called whenever a link segment ends:

```
  /BU.SE {
    currentpoint
    /burl@ury exch def
    dup /burl@urx exch def /burl@endx exch def
    burl@ury burl@topy lt {/burl@topy burl@ury def} if
  } def
```

BU.SE is called whenever the entire link ends:

```
  /BU.E {
    BU.FL
  } def
```

BU.FL is called to conditionally flush the group of link segments that we have so far. This is meant to be called at each line break:

```
  /BU.FL {
    burl@stx null ne {BU.DF} if
  } def
```

BU.DF is the routine to actually put the link rectangle in the PDF file:

```
  /BU.DF {
    BU.BB
    [ /H /I /Border [burl@border] /Color [burl@bordercolor]
    /Action << /Subtype /URI /URI BU.L >> /Subtype /Link BU.B /ANN pdfmark
    /burl@stx null def
  } def
```

BU.FF adds margins to the calculated tight rectangle:

```
  /BU.BB {
    burl@stx HyperBorder sub /burl@stx exch def
    burl@endx HyperBorder add /burl@endx exch def
    burl@boty HyperBorder add /burl@boty exch def
    burl@topy HyperBorder sub /burl@topy exch def
  } def
```

BU.B converts the coordinates into a rectangle:

```
  /BU.B {
    /Rect[burl@stx burl@boty burl@endx burl@topy]
  } def
```

Finally, we must redefine eop, which is called just when the page ends, to handle links that are split into more than one page. (eop-hook isn't the right place to do so, since this hook is called after the dictionaries were reverted to a previous state, vanishing the rectangle coordinates.)

```
  /eop where {
    begin
```

```
      /@ldeopburl /eop load def
      /eop { SDict begin BU.FL end @ldeopburl } def
      end
    } {
      /eop { SDict begin BU.FL end } def
    } ifelse
  }%
}
```