Visual editing (in a specialized case): `prerex`

Bob Tennent

Abstract

It is sometimes desirable and straightforward to support visual editing for LaTeX; this article describes one such case — course prerequisite charts, supported by the (`v`)`prerex` programs.

1 Introduction

One of the most frequently asked questions by LaTeX beginners is whether a graphical interface "like a word processor" is available. Most readers of this article know how to respond: we emphasize logical structure and we point out the availability of LaTeX-friendly text editors, LaTeX development environments, `preview-latex`[1] and, in recalcitrant cases, Scientific Word,[2] or the LyX[3] or TeXmacs[4] "document processors". Not as often mentioned is the difficulty of parsing *arbitrary* TeX documents; it's been said that only TeX can process TeX.

In this article, I discuss the design of a system that *allows* (but doesn't require) a form of visual editing of a *specialized* LaTeX environment for "prerequisite charts".

2 Prerequisite charts

A prerequisite chart gives an attractive graphical presentation of courses in a program (or set of related programs), organized by terms or years, linked by pre- and co-requisite arrows (directed edges), and, when possible, supplemented by timetable information; Figure 1 on page 286 is a small example. Realistic examples may be found at `http://www.cs.queensu.ca/students/undergraduate/prerequisites`.

Some notable properties of these charts:

- Each course box is sized to just enclose the text within it, with uniform standard margins.
- Each arrow between courses is oriented from box centre to box centre, rather than from/to standard "connection points" on the box outlines.
- The arrows are "clipped" by the course boxes, but the arrowheads abut the target box exactly.

These desirable properties are not easily achieved using conventional drawing software, no matter how "user-friendly" it purports to be.

In contrast, the use of LaTeX to produce these charts provides complete flexibility as well as professional quality.

- Text within a course box may be partitioned into regions with varying characteristics. For example, the course code and the timetable information on the first line of course boxes are in a smaller font than the course name. The latter is centered and the former are left- and right-justified, respectively. Arbitrary LaTeX formatting can be used for the text.
- Any available fonts may be used. The TeX typesetting engine takes advantage of kerns and ligatures in the fonts.
- Line thickness for boxes may be varied; in the example diagram, heavier boxes (and bold-face text) are used to indicate that a course is "required" in the program, rather than an option.
- Different styles of connectors can be used, for example to distinguish prerequisites, co-requisites, and recommended prerequisites.
- Various sizes or shapes of course boxes may be used, for example to distinguish between half and full courses.
- Graphic images such as logos can be imported.
- Colours and hyperlinks to on-line course descriptions or calendars are possible.

As an example, the chart in Figure 1 is produced by the LaTeX code in Figure 2. A conventional two-dimensional Cartesian coordinate system is used to specify the locations of diagram elements. The origin (where $x = 0$ and $y = 0$) is at the lower-left corner of the diagram. The coordinates of boxes are those of its centre point; an arrow is described by the coordinates of the centre points of its source and target boxes. The order of commands is not significant except that the commands for the source and target boxes of an arrow should precede the command for the arrow.

The `prerex` package[5] currently uses `pgf`[6] and other standard packages to implement the `chart` environment and a specified set of commands within that environment. Some implementation details:

- The half-course boxes are assigned a minimum height to give a more uniform appearance to horizontal rows of such boxes.
- Arrows with a small height are always drawn straight (using a specialized and simpler macro) unless a non-zero curvature is explicitly requested.
- A wider white edge is drawn below every arrow to improve the appearance of crossing arrows.

[1] `http://preview-latex.sourceforge.net/`
[2] `http://www.sciword.demon.co.uk`
[3] `http://www.lyx.org/`
[4] `http://www.texmacs.org/`
[5] `http://www.ctan.org/pkg/prerex`
[6] `http://www.ctan.org/pkg/pgf`

3 The `prerex` editor

It is certainly possible, though rather tedious and error-prone, to use a conventional text editor to create and revise such descriptions, with some operations, such as global or partial "shifts" of chart elements, being particularly problematic.

The `prerex` editor is a C program that allows chart descriptions to be edited interactively. The editor supports add, remove, cut-and-paste, and edit operations on diagram elements, and vertical or horizontal shifts of: a list of specified elements, all the elements in a rectangular region, or the entire diagram. The edited diagram may be saved, re-processed, and viewed in any PDF viewer, without exiting the editor.

The program reads the source file (possibly for an initial "blank" chart), saving text until the chart environment is found, then parses the chart commands into an internal representation of the diagram. The rest of the source file is saved as text. Macro definitions and calls are not processed or expanded.

When the internal representation of the chart (linked lists of node and arrow data) is available, it is routine to implement editing operations. At any time, the user can ask for a source file to be re-created (using the saved texts and the possibly revised internal representation) and then re-processed. The revised output is available in about four seconds and can be observed in any PDF viewer.

Also observable in most PDF viewers are the coordinates of nodes, or the coordinates of the initial and terminal nodes of arrows, when the cursor is moved over the node or arrow; this is because, during editing, the usual URL associated with nodes is replaced by a special URI containing these coordinates. On initialization, a coordinate grid is generated for the background of the chart in order to facilitate determination of coordinates. If necessary, the user can "escape" from the editor to a shell, for example, to edit the source file with a conventional text editor. A multi-level "undo" command is available. If a box is "cut" and then "pasted" elsewhere, the target or source coordinates of arrows into or out of the box are adjusted automatically, and similarly if nodes are shifted or raised.

The source code for the `prerex` editor is available in the documentation directory of the `prerex` package. The only non-standard library dependence is `readline` (or `libedit`).

4 The `vprerex` interface to `prerex`

Of course, interactive editing with revisions monitored in a PDF viewer is not what is usually meant by *visual* editing. But most of the convenience of visual diagram editing can be obtained by simply allowing coordinates of "selected" diagram elements or background points to be conveyed via the "clipboard" from the PDF viewer to the `prerex` editor command line.

To implement this, a bare-bones open-source PDF viewer was hacked so that, when nodes, arrows or background points are mouse-clicked, the relevant chart coordinates are loaded into the clipboard. For nodes and arrows, the relevant chart coordinates are already available in the special URIs. For background points, it is necessary to transform PDF coordinates into chart coordinates. To allow this transformation, two "anchors" (i.e., virtual nodes) are inserted at the southwest and northeast corners of the chart; from their PDF coordinates and their known chart coordinates, it is possible to compute the chart coordinates of arbitrary clicked points on the chart. All the `vprerex` application does is to start up the `prerex` editor in an `xterm` terminal and the `prerex`-enabled PDF viewer. This naive approach to visual editing is relatively simple to implement and quite pleasant to use.

The source code for the `vprerex` application is available in the documentation directory of the `prerex` package. It depends on the `poppler-qt4` and other Qt-4 libraries.

5 Discussion

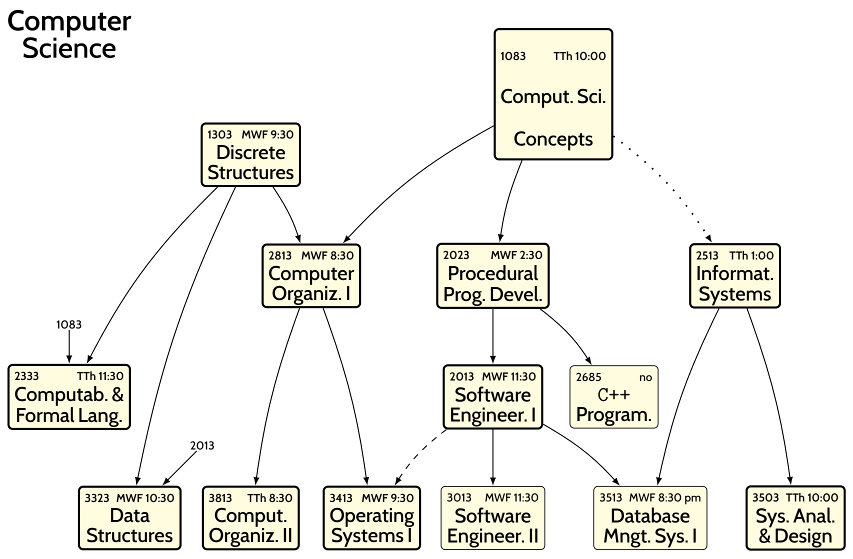
An interactive editor like `prerex` is feasible because it only has to deal with a single very specialized environment and specialized commands within that environment. The "visual editor" `vprerex` is notable for being *unambitious*: it simply puts coordinates into the clipboard for the user to convey to the interactive editor, whereas very ambitious projects such as VorTEX (Visually-oriented TEX) [1] and TEXLite [2] have apparently foundered. Perhaps the approach described here might be applicable to other projects which could benefit from a form of visual editing.

References

[1] Pehong Chen et al. The VorTEX document preparation environment. In *TEX for Scientific Documentation*, volume 236 of *Lecture Notes in Computer Science*, pages 45–54. Springer, 1986.

[2] Igor I. Strokov. A WYSIWYG TEX implementation. *TUGboat*, 20(4):356–359, December 1999. `http://tug.org/TUGboat/tb20-4/tb65strok.pdf`.

◇ Bob Tennent
School of Computing, Queen's University
Kingston, Ontario K7L 3N6 Canada
`rdtennent (at) gmail dot com`
`http://www.ctan.org/pkg/prerex`


Computer Science
1083 TTh 10:00 Comput. Sci. Concepts
1303 MWF 9:30 Discrete Structures
2813 MWF 8:30 Computer Organiz. I
2023 MWF 2:30 Procedural Prog. Devel.
2513 TTh 1:00 Informat. Systems
1083
2333 TTh 11:30 Computab. & Formal Lang.
2013 MWF 11:30 Software Engineer. I
2685 no C++ Program.
2013
3323 MWF 10:30 Data Structures
3813 TTh 8:30 Comput. Organiz. II
3413 MWF 9:30 Operating Systems I
3013 MWF 11:30 Software Engineer. II
3513 MWF 8:30 pm Database Mngt. Sys. I
3503 TTh 10:00 Sys. Anal. & Design

- A solid arrow ⟶ indicates a required prerequisite, a dotted arrow ····▸ indicates a corequisite (to be taken before or concurrently), and a dashed arrow - - -▸ indicates a recommended prerequisite. Core courses are in $\boxed{\textbf{bold}}$ boxes; other courses (i.e., options or prerequisites) are in $\boxed{\text{light}}$ boxes.
- Timetabling abbreviations: M, T, W, Th, F=Mon, Tue, Wed, Thur, Fri, resp.; eve=7:00–9:50 pm; no=not offered.

Figure 1: A `prerex`-formatted prerequisite chart

```
\begin{chart}
  \text 10,50:{\Large Computer\\\Large Science}
  \reqfullcourse 50,45:{1083}{Comput.\,Sci.\\Concepts}{TTh 10:00}
  \reqhalfcourse 25,40:{1303}{Discrete\\Structures}{MWF 9:30}
  \reqhalfcourse 30,30:{2813}{Computer\\Organiz.\,I}{MWF 8:30}
    \prereq 50,45,30,30:
    \prereq 25,40,30,30:
  \reqhalfcourse 45,30:{2023}{Procedural\\Prog.\,Devel.}{MWF 2:30}
    \prereq 50,45,45,30:
  \reqhalfcourse 65,30:{2513}{Informat.\\Systems}{TTh 1:00}
    \coreq 50,45,65,30:
  \mini 10,26:{1083}
  \reqhalfcourse 10,20:{2333}{Computab.\,\&\\Formal\,Lang.}{TTh 11:30}
    \prereq 25,40,10,20:
    \prereq 10,26,10,20:
  \reqhalfcourse 45,20:{2013}{Software\\Engineer.\,I}{MWF 11:30}
    \prereq 45,30,45,20:
  \halfcourse 55,20:{2685}{\texttt{C++}\\Program.}{no}
    \prereq 45,30,55,20:
  \mini 21,16:{2013}
  \reqhalfcourse 15,10:{3323}{Data\\Structures}{MWF 10:30}
    \prereq 25,40,15,10:
    \prereq 21,16,15,10:
  \reqhalfcourse 25,10:{3813}{Comput.\\Organiz.\,II}{TTh 8:30}
    \prereq 30,30,25,10:
  \reqhalfcourse 35,10:{3413}{Operating\\Systems\,I}{MWF 9:30}
    \prereq 30,30,35,10:
    \recomm 45,20,35,10:
  \halfcourse 45,10:{3013}{Software\\Engineer.\,II}{MWF 11:30}
    \prereq 45,20,45,10:
  \halfcourse 58,10:{3513}{Database\\Mngt.\,Sys.\,I}{MWF 8:30 pm}
    \prereq 65,30,58,10:
    \prereq 45,20,58,10:
  \reqhalfcourse 70,10:{3503}{Sys.\,Anal.\\\&\,Design}{TTh 10:00}
    \prereq 65,30,70,10:
\end{chart}
```

Figure 2: LaTeX source for the prerequisite chart in Fig. 1