

Drawing medical pedigree trees with \TeX and PSTricks

Boris Veytsman

Computational Materials Science Center, MS 5A2

George Mason University

Fairfax, VA 22030

borisv (at) lk dot net

Leila Akhmadeeva

Bashkir State Medical University

3 Lenina Str. Ufa, 450077, Russia

leila_ufa (at) mail dot ru

Abstract

Medical pedigrees look like genealogical trees, but also have certain interesting features. Usually they are drawn by hand by medical geneticists. This is a cumbersome and time-consuming process. Freely available programs for drawing genealogies are not fully suitable for this task because of the special format of medical pedigrees.

We discuss a package for drawing pedigrees based on PSTricks. The information is input by geneticists in a spreadsheet; a Perl program extracts it and calls \TeX to produce the final output.

1 Introduction

A medical pedigree is a very important tool for clinicians, genetic researchers and medical educators. As noted by Bennett, Steinhaus, Uhrich, O’Sullivan, Resta, Lochner-Doyle, Markei, Vincent, and Hamanishi, 1995, “The construction of an accurate family pedigree is a fundamental component of a clinical genetic evaluation and of human genetic research.” Regrettably, at present most geneticists make pedigrees manually. There are several programs for making pedigrees (see the list at <http://www.kumc.edu/gec/prof/genecomp.html#pedigree>), but they are rather expensive, lack multi-language support and the quality of typesetting is wanting. Maybe this is why they are not in wide use.

We tried to write a program suitable for constructing complex pedigrees from genetics data in English, Russian and possibly other languages. \TeX and PSTricks (Van Zandt, 1993) seemed to be a natural choice. First, the quality of \TeX typesetting is unsurpassed. Pedigrees combine graphical and textual data, therefore a package like PSTricks is handy to produce them. There are good facilities for multi-language support in \TeX , therefore we could easily deploy them. \TeX code can be easily generated by other programs, which is another advantage of implementing the drawing as a \TeX file.

The idea to draw genealogical trees with \TeX and PSTricks is by no means new. One can enjoy the beautiful trees at <http://www.tug.org/PSTricks/main.cgi?file=Examples/Genealogy/genealogy>.

The problem is, medical pedigrees differ from genealogical trees. They might not even be trees from the mathematical point of view: any marriage between relatives adds a cycle to the graph of relationships. Even if there are no such marriages, pedigrees are not layered rooted trees in the terminology of graph theory (Di Battista, Eades, Tamassia, and Tollis, 1999). The difference is the following. Layered rooted trees have an “oldest” node (global ancestor). This node has no ancestors. It has descendants, and each of them is an ancestor for its own layered rooted tree. On the other hand, a geneticist drawing a pedigree is interested both in the male and female ancestors of the patient, as well as in the male and female ancestors of *them*, etc. Therefore a pedigree might have several “local ancestors”: nodes that do not have ancestors. This makes the problem of drawing pedigrees quite interesting.

We divided the problem of drawing pedigrees into two parts. First, we developed a set of PSTricks macros (Veytsman and Akhmadeeva, 2006a) to draw (almost) any pedigree. They can be used “manually”, i.e., the user can put nodes at any place on the

canvas and make any connections between them, using `\rput`, `\psmatrix` or even `\pstree` (if the pedigree is in fact a layered tree). Second, we developed a Perl program (Veytsman and Akhmadeeva, 2006b) that can process relationship data in the CSV (comma separated values) format and output T_EX code for an important subset of pedigrees: when there is no inbreeding, and only the people having common genes with the primary patient (*proband*) are shown. Both these parts are discussed below.

2 PSTricks macros

From the perspective of graph theory a pedigree is a collection of nodes and connections. There are three main kinds of nodes in our macros: `\PstPerson` to show persons, `\PstAbortion` to show terminated pregnancies, and a *marriage node*,¹ which is implemented as a simple `\pnode`.² Each node except `\PstAbortion` may have *descendants*: children of this person or of this marriage. Each node except the marriage node has exactly one *ancestor node*: a person or a marriage. Additionally a marriage node has a male and a female spouse.

The nodes may have a number of properties: they might be affected by a disease (or not), they might be deceased, they are male or female. These properties are shown by optional arguments to the corresponding command. Each node drawing command has one obligatory argument: the name of the node, which is used to draw node connections.

The first and simplest connection command is `\PstDescent`, which shows the relationship between an ancestor (a person or marriage node) and descendant. It is implemented internally as an `\ncangle` command. There are special commands to show relationships between twins and their parents, between spouses, infertility of a union or a person, etc.

An extremely complex pedigree is used as an example in the paper of Bennett, Steinhaus, Uhrich, O’Sullivan, Resta, Lochner-Doyle, Markei, Vincent, and Hamanishi (1995). In Figure 1 we reproduce this pedigree. The corresponding code is shown in Figures 2, 3 and 4.

A version of node-drawing commands makes *tree nodes*. They are useful if the pedigree is a tree or can be constructed from a tree by adding several connections. An example of such a pedigree is shown in Figure 5, and the corresponding code in Figure 6. Note that the pedigree on this figure is

¹ For our purposes both official marriages and unofficial unions are loosely referred to as marriages.

² There are also special nodes for special circumstances: *twin node* is used to show the relation between twins, a special node is used to show infertility of person or a marriage, etc.

not a tree from the mathematical point of view due to the marriage between Peter and Joan.

The user manual (Veytsman and Akhmadeeva, 2006a) is distributed with the macros. By tradition, PSTricks-related code works with both L^AT_EX and plain T_EX, so our code is written to work in both these modes too. However, most of testing was done in the L^AT_EX mode only.

3 Perl program

The T_EX code in the examples above is straightforward. Nevertheless, it might be too much to expect from geneticists to write it themselves. Therefore if we want the code to work not only for the authors, but also for medical and science professionals, we must find a way to generate this code from the relationship data automatically. This is the aim of the second part of our project (Veytsman and Akhmadeeva, 2006b).

The idea of the program is to take the data in a simple format, easy for the users to understand (and easy to generate in turn from databases or other sources) and convert them into T_EX code above. The input format was chosen to be a plain text CSV format (originally the acronym meant Comma Separated Values, but it is often used now for any plain text separated data). In this format each person record is a line separated into fields by the symbol “|”. The fields show the name of the person, the dates of birth and death, the genetic condition, etc. An important field is *Id*, a unique identifier of the person. The relations between the persons are set by the fields “Mother” and “Father”, which contain Ids of the parents of the given person. In this way we do not need to specifically show the relations between spouses, siblings, etc.: spouses in our system are the persons who have common children, and siblings are the persons with common parent(s). An example of an input file is shown in Figure 7. Such files are easy to generate by common spreadsheet programs and from databases.

The code generated from the input file in Figure 7 is shown in Figure 8, and the typeset pedigree in Figure 9.

Our program works with a subset of all possible pedigrees. Still, this subset covers most pedigrees used by geneticists. First, we do not include inbreeding unions in the pedigrees, so our graphs are in fact trees. Second, we show only the persons that have common genes with the proband.³ Note that a pedigree that violates these rules may not allow

³ The proband is the first person among the relatives who came to a geneticist; he or she is the primary patient.

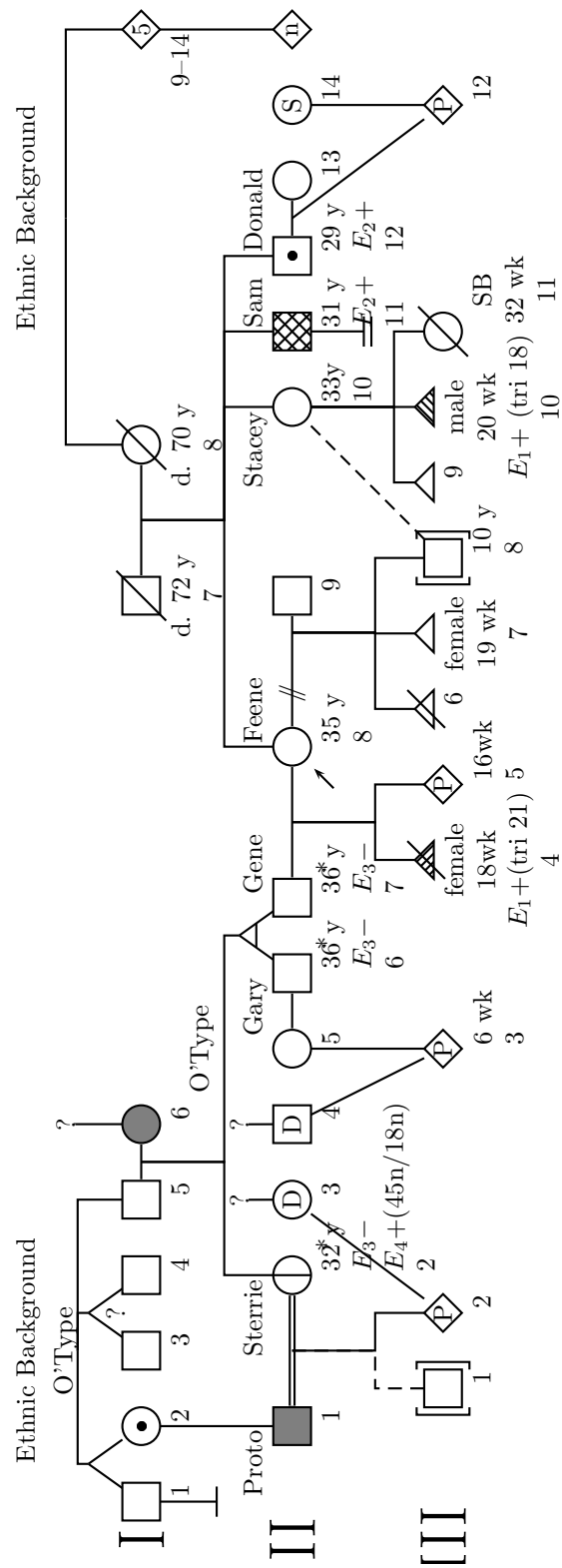


Figure 1: A complex pedigree from Bennett, Steinhaus, Uhrich, O'Sullivan, Resta, Lochner-Doyle, Markei, Vincent, and Hamanishi (1995)

```

\psset{armB=1.1, hatchsep=1.5pt}
\rput(3.5,8){Ethnic Background}
\rput(18.5,8){Ethnic Background}
\rput(3.5,7.5){\rnnode{0Type1}{0'Type}}
\rput(18.5,7.5){\pnnode{0Origin2}}
\rput(6.5,7.5){\rnnode{Quest1}{?}}
\rput(1,6.5){\Huge I}
\rput(1.5,6.5){\pstPerson[male, belowtext=1]{I1}}
\rput(2.5,6.5){\pstPerson[female, obligatory, belowtext=2]{I2}}
\rput(3.5,6.5){\pstPerson[male, belowtext=3]{I3}}
\rput(4.5,6.5){\pstPerson[male, belowtext=4]{I4}}
\rput(5.5,6.5){\pstPerson[male, belowtext=5]{I5}}
\rput(6.5,6.5){\pstPerson[female, affected,
  belowtext=6]{I6}}
\rput(2,7.2){\pnnode{Twins1}}
\rput(4,7.2){\pnnode{Twins2}}
\pstTwins[armB=0]{0Type1}{Twins1}{I1}{I2}
\pstTwins[qzygotic, armB=0, mzlinepos=0.8]{0Type1}{Twins2}{I3}{I4}
\pstDescent[armB=0]{0Type1}{I5}
\pstDescent[armB=0]{Quest1}{I6}
\pstRelationship[descentnode=I5I6]{I5}{I6}
\rput(1.5,5.5){\pstChildless{CI1}}
\ncline{I1}{CI1}
\rput(13.5,6.5){\pstPerson[male, deceased, belowtext=trp=t,
  belowtext=\parbox{2cm}{\centering d. 72 y\7}]{I7}}
\rput(15.5,6.5){\pstPerson[female, deceased, belowtext=trp=t,
  belowtext=\parbox{2cm}{\centering d. 70 y\8}]{I8}}
\pstRelationship[descentnode=I7I8]{I7}{I8}
\rput(21,6.5){\pstPerson[insidetext=5, belowtext=9--14,
  belowtext=rt]{I9}}
\pstDescent[armB=0]{0Origin2}{I8}
\pstDescent[armB=0]{0Origin2}{I9}

```

Figure 2: Code for Figure 1: Generation I

two-dimensional drawing without self-intersections at all.

The biggest problem is that even with these rules the pedigree tree is in the general case not a layered tree, and the algorithm by Reingold and Tilford (Di Battista, Eades, Tamassia, and Tollis, 1999, § 3.1) would not work. We therefore describe a generalization of the Reingold-Tilford algorithm.

First we will summarize the main idea of the Reingold-Tilford algorithm. We draw a tree down in the y direction. The algorithm is recursive. We start from the root of the tree. If it has no descendants, it is easy to draw it. If it has descendants, each descendant is a rooted layered tree. We draw them, recursively applying the algorithm. Then we move them in the horizontal direction as close as possible, and put the root one layer above with an x coordinate in the middle of the descendants.

Now we can generalize this algorithm for our case.

There are two kinds of nodes in the pedigree graph: person nodes and marriage nodes. A node has a *predecessor* and *children*. A marriage node does not have a predecessor, but has *male spouse* and *female spouse* (usually male spouses are on the left and female spouses are on the right on pedigrees). Any node has a *downward tree* of its children, grandchildren, etc. The downward tree may be empty.

Any node in an acyclic graph can be a root. However, in layered trees there is a special root: the one that has no predecessor. Similarly, we will call a *local root* a node that has no predecessor. All marriage nodes are local roots. Some person nodes can be local roots as well.

```

\rput(1,4.5){\Huge II}
\rput(2.5,4.5){\pstPerson[male, affected, belowtext=1,
  abovetext=Proto, abovetextrp=rB]{II1}}
\rput(4.5,4.5){\pstPerson[female, asymptomatic,
  belowtext=\parbox{3cm}{32 y\ $E_3-$\ $E_4+(45n/18n)\2},
  abovetext={Sterrie}, abovetextrp=rB, evaluated]{II2}}
\pstDescent{I2}{II1}\pstDescent{I5I6}{II2}
\pstRelationship[consanguinic, descentnode=II1II2]{II1}{II2}
\rput(5.5,5.2){\rnode{Quest2}{?}}
\rput(5.5,4.5){\pstPerson[female, insidetext=D, belowtext=3]{II3}}
\rput(6.5,5.2){\rnode{Quest3}{?}}
\rput(6.5,4.5){\pstPerson[male, insidetext=D, belowtext=4]{II4}}
\ncline{Quest2}{II3}\ncline{Quest3}{II4}
\rput(7.5,4.5){\pstPerson[female, belowtext=5]{II5}}
\rput(8.5,4.5){\pstPerson[male, abovetext=Gary, abovetextrp=rB,
  belowtext=\parbox{2cm}{36 y\ $E_3-$\6}, evaluated]{II6}}
\rput(9.5,4.5){\pstPerson[male, abovetext={Gene},
  belowtext=\parbox{2cm}{36 y\ $E_3-$\7}, evaluated]{II7}}
\rput(9,5.2){\pnode{Twins3}}
\pstTwins[monozygotic]{I5I6}{Twins3}{II6}{II7}
\pstRelationship{II5}{II6}
\rput(7.5,5.7){0>Type}
\rput(11.5,4.5){\pstPerson[female, proband,
  belowtext=\parbox{1cm}{35 y\8}, abovetext=Feene]{II8}}
\pstRelationship[descentnode=II7II8]{II7}{II8}
\rput(13.5,4.5){\pstPerson[male, belowtext=9]{II9}}
\pstRelationship[broken, descentnode=II8II9,
  descentnodepos=0.85]{II8}{II9}
\rput(16,4.5){\pstPerson[abovetext=Stacey, female,
  abovetextrp=rB, belowtext=\parbox{1cm}{33y\ 10}]{II10}}
\def\affectedstyle{fillstyle=crosshatch}
\rput(17,4.5){\pstPerson[male, affected, abovetext=Sam,
  belowtext=\parbox{3cm}{31 y\ $E_2+$\ 11}, hatchsep=3pt]{II11}}
\rput(17,3.5){\pstChildless[infertile]{C2}}
\ncline{II11}{C2}
\rput(18,4.5){\pstPerson[male, obligatory, abovetext=Donald,
  belowtext=\parbox{3cm}{29 y\ $E_2+$\ 12}]{II12}}
\pstDescent{I7I8}{II8}\pstDescent{I7I8}{II10}
\pstDescent{I7I8}{II11}\pstDescent{I7I8}{II12}
\rput(19,4.5){\pstPerson[female, belowtext=13]{II13}}
\pstRelationship[descentnode=II12II13]{II12}{II13}
\rput(20,4.5){\pstPerson[female, insidetext=S, belowtext=14]{II14}}
\rput(21,4.5){\pstPerson[insidetext=n]{II15}}
\pstDescent{I9}{II15}

```

Figure 3: Code for Figure 1: Generation II

```

\rput(1,2.5){\Huge III}
\rput(3,2.5){\pstPerson[male, adopted, belowtext=1]{III1}}
\rput(4,2.5){\pstPerson[insidetext=P, belowtext=2]{III2}}
\pstDescent[linestyle=dashed]{III1II2}{III1}
\pstDescent{III1II2}{III2}
\ncline{II3}{III2}
\rput(7.5,2.5){\pstPerson[insidetext=P,
  belowtext=\parbox{2cm}{6 wk\3}]{III3}}
\pstDescent{II5}{III3}
\ncline{II4}{III3}
\def\affectedstyle{fillstyle=vlines}
\rput(10,2.5){\pstAbortion[affected,
  belowtext=\parbox{2cm}{\centering
    female\18wk\ $E_1+(\tri 21)\4}, belowtext=trp=t]{III4}}
\rput(11,2.5){\pstPerson[insidetext=P,
  belowtext=\parbox{1cm}{16wk\5}]{III5}}
\pstDescent{II7II8}{III4}
\pstDescent{II7II8}{III5}
\rput(12,2.5){\pstAbortion[belowtext=6]{III6}}
\rput(13,2.5){\pstAbortion[sab, belowtext=trp=t,
  belowtext=\parbox{2cm}{\centering female\19 wk\ 7}]{III7}}
\rput(14,2.5){\pstPerson[adopted, male,
  belowtext=\parbox{1cm}{10 y\ 8}]{III8}}
\pstDescent{II8II9}{III6}
\pstDescent{II8II9}{III7}
\pstDescent{II8II9}{III8}
\ncline[linestyle=dashed]{II10}{III8}
\rput(15,2.5){\pstAbortion[sab, belowtext=9]{III9}}
\def\affectedstyle{fillstyle=hlines}
\rput(16,2.5){\pstAbortion[sab, belowtext=trp=t, affected,
  belowtext=\parbox{2cm}{\centering male\ 20 wk\ $E_1+$
    (\tri 18)\ 10}]{III10}}
\rput(17,2.5){\pstPerson[deceased, female,
  belowtext=\parbox{1cm}{\centering SB\32 wk\ 11}]{III11}}
\pstDescent{II10}{III9}
\pstDescent{II10}{III10}
\pstDescent{II10}{III11}
\rput(20,2.5){\pstPerson[insidetext=P,
  belowtext=12]{III12}}
\pstDescent{II14}{III12}
\ncline{II12II13}{III12}

```

Figure 4: Code for Figure 1: Generation III

Our algorithm is recursive and starts from a local root. Strictly speaking, it can start from any local root, but since medical pedigrees have the special proband mentioned earlier, it makes sense to start from the local root which has the proband in its downward tree.

If this local root is a person node, the pedigree is the layered tree, and the Reingold-Tilford algorithm is sufficient. Therefore we should consider only the

case when the local root is a marriage node. In this case we can typeset the downward tree using the Reingold-Tilford algorithm. The male and female spouses do not belong to this tree. However, each of them belongs to each own subpedigree. We will call them *left subpedigree* and *right subpedigree*. We recursively apply our algorithm to typeset left and right subpedigrees. Then we move the left subpedigree to the right and right subpedigree to the left

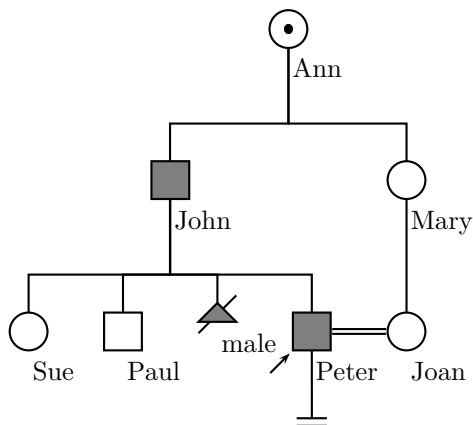


Figure 5: Example of pedigree with tree-making commands

```

\begin{pspicture}(0,1)(7,7)
  \rput(3,4){%
    \pstree{\TpstPerson[female, obligatory, belowtext=Ann]{Ann}}{%
      \def\psedge{\pstDescent}\psset{armB=1}
      \pstree{\TpstPerson[male, affected, belowtext=John]{John}}{%
        \TpstPerson[female, belowtext=Sue]{Sue}
        \TpstPerson[male, belowtext=Paul]{Paul}
        \TpstAbortion[affected, belowtext=male]{A1}
        \pstree[thislevelsep=1.2]{\TpstPerson[male,
          belowtext=Peter, affected, proband]{Peter}}{%
          \def\psedge{\ncline}
          \TpstChildless[infertile]{C1}
        }
      }
    }
  \pstree{\TpstPerson[female, belowtext=Mary]{Mary}}{
    \TpstPerson[female, belowtext=Joan]{Joan}
  }
}
\pstRelationship[consanguinic]{Peter}{Joan}
\end{pspicture}

```

Figure 6: Code producing Figure 5

Id	Name	Sex	DoB	DoD	Mother	Father	Proband	Condition	Comment
P	John Smith	male	1970/02/05		M1	F1	yes	affected	Evaluated 2005/12/01
M1	Mary Smith (Brown)	female	1940/02/05		GM2	GF2		normal	
F1	Bill Smith	male	1938/04/03		GM1	GF1		affected	
GM1	Joan Smith	female	1902/07/01	1975/12/13				asymptomatic	
GF1	Joseph Smith	male	unknown	unknown				normal	
GF2	Jim Brown	male	1905/11/01					normal	
GM2	Lisa Brown	female	1910/03/03					normal	
S1	Rebecca Smith	female	1972/12/25		M1	F1		affected	
S2	Alexander Smith	male	1975/11/12		M1	F1		normal	
A1	Ann Gold (Smith)	female	1941/09/02		GM1	GF1		asymptomatic	Aunt of the proband
C1	Jenny Smith	female	1969/12/03		A1			affected	Cousin of the proband

Figure 7: Example of data file

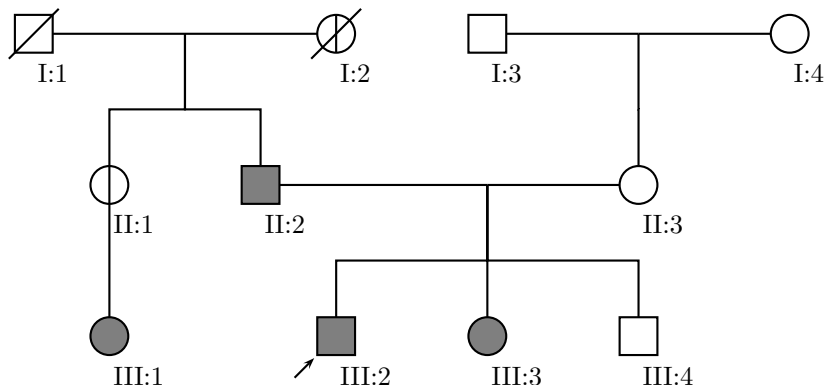
```

\begin{pspicture}(-8,-4)(6,4)
\rput(-6, 2){\pstPerson[male, normal, belowtext=I:1, deceased]{GF1}}
\rput(-4, 2){\pnode{GF1_m_GM1}}
\rput(-2, 2){\pstPerson[female, asymptomatic, belowtext=I:2, deceased]{GM1}}
\rput(0, 2){\pstPerson[male, normal, belowtext=I:3]{GF2}}
\rput(2, 2){\pnode{GF2_m_GM2}}
\rput(4, 2){\pstPerson[female, normal, belowtext=I:4]{GM2}}
\rput(-5, 0){\pstPerson[female, asymptomatic, belowtext=II:1]{A1}}
\rput(-3, 0){\pstPerson[male, affected, belowtext=II:2]{F1}}
\rput(0, 0){\pnode{F1_m_M1}}
\rput(2, 0){\pstPerson[female, normal, belowtext=II:3]{M1}}
\rput(-5, -2){\pstPerson[female, affected, belowtext=III:1]{C1}}
\rput(-2, -2){\pstPerson[male, affected, belowtext=III:2, proband]{P}}
\rput(0, -2){\pstPerson[female, affected, belowtext=III:3]{S1}}
\rput(2, -2){\pstPerson[male, normal, belowtext=III:4]{S2}}
\pstDescent{GF1_m_GM1}{A1}
\pstDescent{GF1_m_GM1}{F1}
\ncline{GF1_m_GM1}{GM1}
\ncline{GF1_m_GM1}{GF1}
\pstDescent{GF2_m_GM2}{M1}
\ncline{GF2_m_GM2}{GM2}
\ncline{GF2_m_GM2}{GF2}
\pstDescent{A1}{C1}
\pstDescent{F1_m_M1}{P}
\pstDescent{F1_m_M1}{S1}
\pstDescent{F1_m_M1}{S2}
\ncline{F1_m_M1}{M1}
\ncline{F1_m_M1}{F1}
\end{pspicture}

\begin{description}
\item[I:1] Joseph Smith; born: unknown; died: unknown.
\item[I:2] Joan Smith; born: 1902/07/01; died: 1975/12/13.
\item[I:3] Jim Brown; born: 1905/11/01.
\item[I:4] Lisa Brown; born: 1910/03/03.
\item[II:1] Ann Gold (Smith); born: 1941/09/02; Aunt of the proband.
\item[II:2] Bill Smith; born: 1938/04/03.
\item[II:3] Mary Smith (Brown); born: 1940/02/05.
\item[III:1] Jenny Smith; born: 1969/12/03; Cousin of the proband.
\item[III:2] John Smith; born: 1970/02/05; Evaluated 2005/12/01.
\item[III:3] Rebecca Smith; born: 1972/12/25.
\item[III:4] Alexander Smith; born: 1975/11/12.
\end{description}

```

Figure 8: Example of program output (data from Figure 7)



- I:1** Joseph Smith; born: unknown; died: unknown.
- I:2** Joan Smith; born: 1902/07/01; died: 1975/12/13.
- I:3** Jim Brown; born: 1905/11/01.
- I:4** Lisa Brown; born: 1910/03/03.
- II:1** Ann Gold (Smith); born: 1941/09/02; Aunt of the proband.
- II:2** Bill Smith; born: 1938/04/03.
- II:3** Mary Smith (Brown); born: 1940/02/05.
- III:1** Jenny Smith; born: 1969/12/03; Cousin of the proband.
- III:2** John Smith; born: 1970/02/05; Evaluated 2005/12/01.
- III:3** Rebecca Smith; born: 1972/12/25.
- III:4** Alexander Smith; born: 1975/11/12.

Figure 9: Example of the typeset pedigree (code from Figure 8)

as far as we can without intersection between them and the downward tree.

This process is shown in Figure 10. Obviously this algorithm converges and leads to typesetting the pedigree without intersections between the subtrees and subpedigrees.

The program is implemented in Perl and can process input in English or Russian, creating a pedigree legend in any of these languages (the Russian examples can be found in the manual (Veytsman and Akhmadeeva, 2006b)). It is quite straightforward to add language modules for any other language or script that \TeX can typeset.

4 Installation notes

A couple of words about the installation of the packages.

The installation of the \TeX part follows the usual guidelines (<http://www.tex.ac.uk/cgi-bin/textfaq2html?label=instpackages>).

The \TeX package depends on a number of other packages, which should be installed on your system. You need fresh versions of `pstricks` and `pst-xkey`. If you want to typeset the documentation you also need `pstricks-add`, but if you are satisfied with the PDF manual provided with the package, you might skip this step.

The Perl part includes the executable, library and manual pages. There is a `Makefile`, and in most cases the command `make install` suffices.

5 Conclusions and future work

Our programs were written mostly as a proof of concept. Surprisingly (or unsurprisingly if we recall the

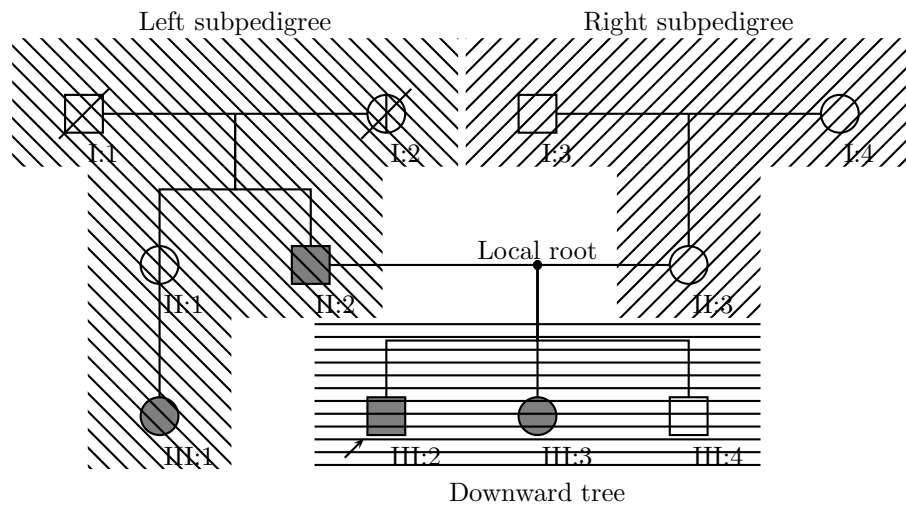


Figure 10: Subpedigrees and downward tree

properties of $\text{T}_{\text{E}}\text{X}$) the typesetting quality of the output is rather high. The next logical step is to make them user-friendly, so any genetic specialist can use them without reading manual.

Since we cannot count on the medical and genetic personnel to have $\text{T}_{\text{E}}\text{X}$ and Perl on their computers, we envision a “Pedigree Live” disk akin to $\text{T}_{\text{E}}\text{X}$ Live: a CD with Perl and a subset of $\text{T}_{\text{E}}\text{X}$ on it, which “just works” after inserting in the computer. This requires creating a cross-platform user interface and the selection of programs and tools to be included on a CD.

Acknowledgements

One of the authors (LA) would like to thank Russian Foundation for Fundamental Research (travel grant 06-04-58811), Russian Federation President Council for Grants Supporting Young Scientists and Flagship Science Schools (grant MD-4245.2006.7) and TUG for support at the Practical $\text{T}_{\text{E}}\text{X}$ 2006 conference.

References

- Bennett, Robin L., K. A. Steinhaus, S. B. Uhrich, C. K. O’Sullivan, R. G. Resta, D. Lochner-Doyle, D. S. Markei, V. Vincent, and J. Hamanishi. “Recommendations for Standardized Human Pedigree Nomenclature”. *Am. J. Hum. Genet.* **56**(3), 745–752, 1995.
- Di Battista, Giuseppe, P. Eades, R. Tamassia, and I. G. Tollis. *Graph Drawing: Algorithms for the Visualization of Graphs*. An Alan R. Apt Book. Prentice Hall, New Jersey, 1999.
- Van Zandt, Timothy. *PSTricks: PostScript Macros for Generic $\text{T}_{\text{E}}\text{X}$* , 1993. <http://tug.org/PSTricks/doc>.
- Veytsman, Boris, and L. Akhmadeeva. *Creating Medical Pedigrees with PSTricks and $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$* , 2006a. <http://ctan.tug.org/tex-archive/graphics/pstricks/contrib/pedigree/pst-pdgr>.
- Veytsman, Boris, and L. Akhmadeeva. *A Program For Automatic Pedigree Construction With pst-pdgr. User Manual and Algorithm Description*, 2006b. <http://ctan.tug.org/tex-archive/graphics/pstricks/contrib/pedigree/pedigree-perl>.