TUG 2009 abstracts

Editor's note: Many of the conference presentations are available at `http://www.river-valley.tv` in video form, thanks to Kaveh Bazargan and River Valley Technologies.

Jin-Hwan Cho

Extended DVI formats and DVIasm

DVIasm is a TEX utility program that is designed for editing DVI files with three steps: disassembling, editing, and assembling.

The first stage of DVIasm [1] supported the standard DVI file format as in DVItype and DTL, but in a much more flexible way than those utilities. In the second stage [2], DVIasm made it possible to handle two-byte characters, CJK and Unicode characters. The extended DVI formats generated by Omega, Japanese pTEX were all supported, as well as ordinary LATEX packages with subfont scheme such as CJK-LATEX and Korean *ko*.TEX. The final stage of DVIasm confronts the two advanced TEX engines, LuaTEX and XETEX, both of which can handle OpenType and TrueType files in a direct way without TFM files.

In this talk we introduce DVIasm with a few interesting applications to the TEX world, and discuss how DVIasm handles the extended DVI formats generated by LuaTEX and XETEX.

[1] Jin-Hwan Cho, *Hacking DVI files: Birth of DVIasm*, *The PracTEX Journal* (2007), no. 1, and *TUGboat* **28** (2007), no. 2, 210–217. `http://tug.org/TUGboat/Articles/tb28-2/tb89cho.pdf`.

[2] Jin-Hwan Cho, *Handling Two-Byte Characters with DVIasm*, The Asian Journal of TEX **2** (2008), no. 1, 63–68. `http://ajt.ktug.kr/assets/2008/5/1/0201cho.pdf`.

Paulo Ney de Souza

The MSP TEX production system: Restoration of TUGboat

Mathematical Sciences Publishers will begin to offer *TUGboat* to its subscribers (primarily academic libraries) soon. This talk will discuss the challenges in bringing all thirty volumes of *TUGboat* into MSP's electronic delivery system via its web site, `http://www.mathscipub.org`.

Idris Hamid

Dynamic Arabic: Towards the philosopher's stone of Arabic-script typography

We discuss the present status of the Oriental TEX project, particularly the problem of Arabic-script microtypography. This includes glyph substitution and hz parameterization.

Idris Hamid

Arabic typography: Past, present, and TEX

Accommodating the classical Arabic script to print typography has always been a challenge. In this talk we go over some of the history of this effort — including TEX-based solutions — with a view to providing a backdrop to the Oriental TEX Project and its progress.

Morten Høgholm

Consolidation of expl3

The expl3 language used as the foundation of LATEX3 has gone through a consolidation phase where almost each and every concept has been questioned, taken apart and put back, sometimes in the same form as it was and sometimes in radically different forms. We will go through some of the most interesting changes and highlight the areas where special effort has been made to ensure simple and natural interfaces.

Morten Høgholm

Next steps for breqn

The next version of the breqn package for automatic line breaking of displayed equations is underway. We will discuss different areas of math typesetting: some things breqn handles well, some areas have room for improvement, and some areas are simply not covered. We will spend some time talking about the technical challenges posed by these requirements.

Chris Rowley

TEX-free LATEX, an overview & Standards for LATEX documents and processors or *Whither LATEX? (The language)*

The LATEX is the message. As some of you will be aware, and all should be, LATEX code, possibly with some variations, extensions or simplifications, has for a long time been used, raw and unprocessed, as a lingua franca for communicating mathematics via text files in computers. [I have even seen it used on napkins and coffee tables.]

This has led to a proliferation of LATEX-like input systems for mathematical information and this in turn produced a reluctance by users of maths notation to adopt any other type of input. However, much of this math input is not intended (primarily) to ever be input to a TEX machine. (It may get swallowed by a TEX-like system after, for example, some copy/paste actions.)

More recently, systems are being developed to produce whole LaTeX-encoded documents that are to be processed by systems such as OMDoc or LaTeXML and so will not necessarily ever pass through a TeX-like engine. Systems such as PlasTeX also belong in this category, despite using TeX as a helper utility in their implementation.

A very recent discovery surprised me more than a little: that many systems in the maths world are not only able to produce LaTeX output (e.g., computer algebra packages) but, currently at least, have LaTeX maths as their only or primary output! This is because: it is wanted or preferred by mathematicians; it is widely accepted by other mathematical software; or simply that nothing else is known to be available for a consistent and familiar encoding of maths notation.

A more sophisticated reason put forward for the increasing ubiquity of LaTeX is that if you are looking for a user-friendly and flexible editor for structured documents, then there are no rivals to the various environments available for the production and editing of LaTeX documents (such as `auctex+(x)emacs`).

Standards. What *standards?* It would be possible to make an exhaustive list of everything that is allowed to appear in a Standard Basic LaTeX document. But that would be both tedious, uncheckable and ignored.

It is currently much easier to pin down which parts of the LaTeX language are accepted by the various non-TeX-like processors of LaTeX (from the first part). Also, there are corpora that can be automatically studied to produce definitions of the subsets actually used by various communities.

Amongst those who handle mathematics in computers there is a growing demand to analyse these de facto standards, at least for LaTeX-math, and to produce reference standards in this area. These would be used to compare systems and communities and make recommendations for usage. This could possibly lead to some more formal standards and, most importantly, extension mechanisms so that, for example, general-purpose parsers can be used to read such code.

TUG Institutional Members

American Mathematical Society,
Providence, *Rhode Island*

Aware Software, Inc.,
Midland Park, *New Jersey*

Banca d'Italia,
Roma, *Italy*

Center for Computing Sciences,
Bowie, *Maryland*

Certicom Corp.,
Mississauga, *Ontario*, *Canada*

CSTUG, *Praha*, *Czech Republic*

Florida State University,
School of Computational Science and Information Technology,
Tallahassee, *Florida*

IBM Corporation,
T J Watson Research Center,
Yorktown, *New York*

Institute for Defense Analyses,
Center for Communications Research,
Princeton, *New Jersey*

MacKichan Software, Inc.,
Washington/New Mexico, *USA*

Marquette University,
Department of Mathematics, Statistics and Computer Science,
Milwaukee, *Wisconsin*

Masaryk University,
Faculty of Informatics,
Brno, *Czech Republic*

Moravian College,
Department of Mathematics and Computer Science,
Bethlehem, *Pennsylvania*

MOSEK ApS,
Copenhagen, *Denmark*

New York University,
Academic Computing Facility,
New York, *New York*

Princeton University,
Department of Mathematics,
Princeton, *New Jersey*

Springer-Verlag Heidelberg,
Heidelberg, *Germany*

Stanford University,
Computer Science Department,
Stanford, *California*

Stockholm University,
Department of Mathematics,
Stockholm, *Sweden*

University College, Cork,
Computer Centre,
Cork, *Ireland*

University of Delaware,
Computing and Network Services,
Newark, *Delaware*

Université Laval,
Ste-Foy, *Québec*, *Canada*

University of Oslo,
Institute of Informatics,
Blindern, *Oslo*, *Norway*