Typesetting Attribute-Value Matrices Under LaTeX

Author
Andrew Lincoln Burrow
albcorp@gmail.com

Abstract
The AlbAVM package provides a single alb-avm LaTeX package to typeset typed feature structures and inequated typed feature structures in attribute-value matrix (AVM) notation. This support is provided by an environment and associated markup commands. In marking up AVM material, the resulting LaTeX structure is isomorphic to the AVM structure. The markup is placed in the alb namespace. The package is supported by an emacs lisp file customising AUCTeX to automate the insertion of typed feature structures by a depth-first traversal of the AVM structure.

Version Information
Revision
Date

1 Introduction

The `alb-avm` LaTeX package is designed to typeset typed feature structures and inequated typed feature structures in attribute-value matrix (AVM) notation. The package is supported by an emacs lisp file customising AUCTeX to automate the insertion of typed feature structures.

Section 2 covers the use of the `albAvm` environment and supporting commands. The examples also demonstrate a source code layout that is known to work well with AUCTeX. Section 3 describes how the commands are quickly entered under AUCTeX. This is the best way to enter the AVM commands since it handles tags and source indentation automatically.

2 Using the Environments and Commands

Each topmost feature structure is represented by an instance of the `albAvm` environment. Within such an environment the `albAvmType` command generates type labels, and the `albAvmFeat` command generates feature-value pairs. You must not nest `albAvm` environments as nesting and tag numbering is automated within the `albAvm` environment. Instead, the `albAvmFeat` and `albAvmTag` commands each take a second arguments which is rendered as a nested substructure.

2.1 Simple Feature Structures

The most compact typed feature structure is a simple type. It is the only form expressed without nesting. This simple form is just an `albAvmType` command contained within an `albAvm` environment, viz:

```
\begin{albAvm}
  \albAvmType{universal}
\end{albAvm}
```

which yields

$$\left[\ \textbf{universal}\ \right]$$

Feature-values requires nesting. This is achieved by placing the contents of the feature-value as the second argument to the `albAvmFeat` command. Note that an `albAvm`-like environment is automatically wrapped around the second argument to `albAvmFeat`. For example,

```
\begin{albAvm}
  \albAvmType{child}
  \albAvmFeat{FATHER}{%
    \albAvmType{human}
    }
  \albAvmFeat{MOTHER}{%
    \albAvmType{human}
    }
\end{albAvm}
```

yields

$$\left[\begin{array}{l} \textbf{child} \\ \text{FATHER} : \left[\ \textbf{human}\ \right] \\ \text{MOTHER} : \left[\ \textbf{human}\ \right] \end{array}\right]$$

Also, deeper nesting proceeds in the obvious way, so that

```
\begin{albAvm}
  \albAvmType{child}
  \albAvmFeat{FATHER}{%
    \albAvmType{human}
    \albAvmFeat{MOTHER}{%
      \albAvmType{human}
      }
    \albAvmFeat{NAME}{%
      \albAvmType{name}
      }
    }
  \albAvmFeat{MOTHER}{%
    \albAvmType{human}
    }
\end{albAvm}
```

yields

$$\left[\begin{array}{l} \textbf{child} \\ \text{FATHER} : \left[\begin{array}{l} \textbf{human} \\ \text{MOTHER} : \left[\ \textbf{human}\ \right] \\ \text{NAME} : \left[\ \textbf{name}\ \right] \end{array}\right] \\ \text{MOTHER} : \left[\ \textbf{human}\ \right] \end{array}\right]$$

2.2 Feature Structures with Structure Sharing

The `alb-avm` commands also provide for substructure tagging. This is not automated by LaTeX counters: automatic tags implemented by extending the LaTeX mechanism for labels and cross references is beyond the scope of this implementation. Instead tags must be explicitly entered. For a substructure to be tagged, place the tags as the first argument to the `albAvmTag` command and place the contents of the substructure as the second argument. The `albAvmTag` command occurs in place of the substructure. Likewise, to refer to a substructure place the substructure's tag as the only argument to the `albAvmRef` command. This command also occurs in place of the substructure. For example, the liar sentence is entered as

```
\begin{albAvm}
  \albAvmTag{1}{%
    \albAvmType{false}
    \albAvmFeat{ARG}{%
      \albAvmRef{1}
      }
    }
\end{albAvm}
```

which yields

$$\left[\begin{array}{l} \boxed{1}\ \textbf{false} \\ \text{ARG} : \left[\ \boxed{1}\ \right] \end{array}\right]$$

2.3 Inequated Feature Structures

Finally, provision is included for inequated feature structures. Inequated feature structures are also entered using the `albAvm` environment. The initial part of the environment's body is entered like a feature structure without inequations, excepting that all substructures referred to in the inequations must be tagged. The remainder of the environment contains a comma separated list of inequations. Each inequation is entered with the `albAvmIneqtn` command. This command takes two arguments, which are simply the tags of the substructures to be inequated. For example,

```
\begin{albAvm}
  \albAvmType{house}
  \albAvmFeat{BEDROOM}{%
    \albAvmTag{1}{%
      \albAvmType{room}
      }
    }
  \albAvmFeat{KITCHEN}{%
    \albAvmTag{2}{%
      \albAvmType{room}
      }
    }
  \albAvmFeat{DININGROOM}{%
    \albAvmTag{3}{%
      \albAvmType{room}
      }
    }
  \albAvmFeat{LIVINGROOM}{%
    \albAvmTag{4}{%
      \albAvmType{room}
      }
    }
  \albAvmFeat{BATHROOM}{%
    \albAvmTag{5}{%
      \albAvmType{room}
      }
    }
  \albAvmIneqtn{1}{2}, \albAvmIneqtn{1}{5},
  \albAvmIneqtn{2}{5}, \albAvmIneqtn{3}{5},
  \albAvmIneqtn{4}{5}
\end{albAvmIneq}
```

yields

$$\left[\begin{array}{l}\textbf{house}\\ \text{BEDROOM}:\left[\,\boxed{1}\ \textbf{room}\,\right]\\ \text{KITCHEN}:\left[\,\boxed{2}\ \textbf{room}\,\right]\\ \text{DININGROOM}:\left[\,\boxed{3}\ \textbf{room}\,\right]\\ \text{LIVINGROOM}:\left[\,\boxed{4}\ \textbf{room}\,\right]\\ \text{BATHROOM}:\left[\,\boxed{5}\ \textbf{room}\,\right]\\ \boxed{1}\not\leftrightarrow\boxed{2},\boxed{1}\not\leftrightarrow\boxed{5},\boxed{2}\not\leftrightarrow\boxed{5},\boxed{3}\not\leftrightarrow\boxed{5},\boxed{4}\not\leftrightarrow\boxed{5}\end{array}\right]$$

3 AUCTeX Customisations

Under AUCTeX the file `alb-avm.el` is automatically loaded (subject to certain AUCTeX configuration options). This customises AUCTeX to automate the entry of AVM notation through a collection of mutually recursive functions that interrogate the user for AVM input.

Automatic entry of AVM notation is triggered when an `albAvm` environment is inserted by the `LaTeX-environment` command. By default AUCTeX binds `LaTeX-environment` to the keys `C-c C-e`. Automatic entry corresponds to a depth-first walk of the feature structure. At each substructure the user is prompted for a description of the substructure. The prompt contains the current feature path to the substructure.

The first question about a substructure concerns its tag. The history list contains the tags entered for the feature structure. If no value is entered, then the tag is omitted; if the tag matches an existing tag, then the substructure is represented by a simple reference to the existing substructure; otherwise, the substructure is tagged with the value.

Given a substructure has been allocated a new tag or no tag at all, it must be explicitly constructed. In this case the second question about a substructure concerns its type. The history list contains the types entered during the current editing session. If no value is entered, then the substructure defaults to `universal` with the no feature-values; otherwise, the type is recorded and feature-values collected. Feature-values are collected while non-empty strings are returned for the features. The history list contains the features entered during the current editing session. Recursion occurs in the entry of the value part of the feature-values because each feature-value is itself a substructure.

To prevent confusion while recurring into the substructures the prompt strings are prefixed by the current path. In addition, the source code is incrementally entered with balanced parenthesis and indentation so that it offers a useful cue to the substructure being defined.

As an exercise attempt to enter this feature structure:

$$\left[\begin{array}{l}\textbf{child}\\ \text{FATHER}:\left[\begin{array}{l}\textbf{human}\\ \text{MOTHER}:\left[\,\textbf{human}\,\right]\\ \text{NAME}:\left[\,\textbf{name}\,\right]\end{array}\right]\\ \text{MOTHER}:\left[\,\textbf{human}\,\right]\end{array}\right]$$