Coding system for AUT-QE

de Bruijn, N.G.

Published: 01/01/1970

Document Version
Publisher’s PDF, also known as Version of Record (includes final page, issue and volume numbers)

Please check the document version of this publication:

• A submitted manuscript is the author's version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher's website.
• The final author version and the galley proof are versions of the publication after peer review.
• The final published version features the final layout of the paper including the volume, issue and page numbers.

Link to publication

Citation for published version (APA):

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

• Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
• You may not further distribute the material or use it for any profit-making activity or commercial gain
• You may freely distribute the URL identifying the publication in the public portal

Take down policy
If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

Download date: 29. Dec. 2018
Coding system for AUT-QE.

by N.G. de Bruijn.

The expressions and categories to be stored are all of the form \( \text{EXPRESSION} \), as defined in the following syntax. The notion is a slight extension of those explained in [2] and [3].

The basic symbols are:

\[
\begin{align*}
\text{type} & | \text{genre} | , | [ | ] | \{ | \} | ( | )
\end{align*}
\]

and, furthermore, the elements of the sets \(<\text{variable}>\), \(<\text{constant}>\) and \(<\text{dummy variable}>\). These three sets are disjoint; \(<\text{variable}>\) and \(<\text{constant}>\) contain positive integers only; \(<\text{dummy variable}>\) contains integers \(<-1000\) only.

The notions \(<\text{EXPRESSION}>\) and \(<\text{EXPRESSION string}>\) are defined by:

\[
<\text{EXPRESSION string}> := <\text{EXPRESSION}> \mid <\text{EXPRESSION string}> , <\text{EXPRESSION}>
\]

\[
<\text{EXPRESSION}> := \text{type} \mid \text{genre} \mid <\text{constant}> \mid <\text{variable}> \mid <\text{dummy variable}> \mid ( <\text{EXPRESSION string}> ) \mid \{ <\text{EXPRESSION}> \}<\text{EXPRESSION}> \mid [ <\text{dummy variable}> , <\text{EXPRESSION}> ] <\text{EXPRESSION}>
\]

There are three arrays in which the information about \(<\text{EXPRESSION}>s\) and \(<\text{EXPRESSION string}>s\) is stored: list1[1:P], list2[1:P], list3[1:P].

Every integer \(k\) \((1 \leq k \leq P)\) refers to an \(<\text{EXPRESSION string}>\). In our present discussion we shall denote this string by \(\Omega_k\) (metalingual symbol). If \(\Omega_k\) has the form \(\Lambda_h\), \(\Lambda\) (where \(\Lambda\) is an \(<\text{EXPRESSION}>\)) then we have list \(1[k] = h\); if \(\Omega_k\) has the form \(\Lambda\), where \(\Lambda\) is an \(<\text{EXPRESSION}>\), we have list \(1[k] = 0\). The information about \(\Lambda\) is stored in list \(2[k]\) and list \(3[k]\).

If \(\Lambda = \text{type}\) then list \(2[k] = 0\), list \(3[k] = -1000\).
If \(\Lambda = \text{genre}\) then list \(2[k] = 0\), list \(3[k] = -2000\).
If \(\Lambda = c\), where \(c \in <\text{constant}>\), then list \(2[k] = c\), list \(3[k] = 0\).
If \(\Lambda = x\), where \(x \in <\text{variable}>\), or \(x \in <\text{dummy variable}>\), then list \(2[k] = x\), list \(3[k] = -5000\) or \(-4000\).

The entry \(-4000\) should not be used if \(\Omega_k\) is not an indicator string (\(\Omega_k\) is certainly no indicator string if \(x\) is a dummy variable).

If \(\Lambda\) has the form \(c(<\text{EXPRESSION string}>),\) and if that \(<\text{EXPRESSION string}>\) is \(\Omega_h\), then list \(2[k] = c\), list \(3[k] = h\).
If \(\Lambda\) has the form \(\{\Lambda_1, \Lambda_2\}\), and if \(\Omega_h\) is the \(<\text{EXPRESSION string}>\) \(\Lambda_1, \Lambda_2\)
(this string consists of just two expressions), then list \(2[k] = -12\), list \(3[k] = h\).
If \( \Lambda \) has the form \([t, \Lambda_1] \Lambda_2\), and if \( \Omega_h \) is the expression string \( \Lambda_1, \Lambda_2 \), then
\[
\text{list2}[k] = t, \quad \text{list3}[k] = h.
\]

Note that the above system is obtained from the one in [1] for expressions of the form \(<\text{constant}> <\text{expression string}>\) if we add the following conventions:

<table>
<thead>
<tr>
<th>type</th>
<th>genre</th>
<th>( \Omega )</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>( 0(\Omega_{-1000}) )</td>
</tr>
<tr>
<td></td>
<td></td>
<td>( 0(\Omega_{-2000}) )</td>
</tr>
<tr>
<td></td>
<td></td>
<td>( c(\Omega_0) )</td>
</tr>
<tr>
<td></td>
<td></td>
<td>( x(\Omega_{-4000}) ) or ( x(\Omega_{-5000}) )</td>
</tr>
<tr>
<td></td>
<td></td>
<td>( t(\Omega_{-5000}) )</td>
</tr>
<tr>
<td>( {\Lambda_1} \Lambda_2 )</td>
<td></td>
<td>( -12(\Lambda_1, \Lambda_2) )</td>
</tr>
<tr>
<td>( [t, \Lambda_1] \Lambda_2 )</td>
<td></td>
<td>( t(\Lambda_1, \Lambda_2) )</td>
</tr>
</tbody>
</table>

We did not put the empty string into our syntax. Nevertheless we consider the empty string occasionally, and we give it list number 0, i.e. \( \Omega_0 \) represents the empty string.

We remind the reader of the definition of indicator string. An indicator string is either the empty string or a string of variables (satisfying the condition that the indicator string of the last variable is obtained by taking that last entry away). In the non-empty case it can, of course, be considered as an EXPRESSION string and will be stored as such.

The contents of a book are stored in three arrays: \( \text{indstr}[1:m], \text{middle}[1:m], \text{cat}[1:m] \).

If \( 1 \leq n \leq m \), and if the indicator string of the \( n \)-th line of the book is \( \Omega_k \), then \( \text{indstr}[n] = k \).

If the middle part of the \( n \)-th line is an EXPRESSION \( \Lambda \), and if \( \Omega_k \) is the string consisting of the single entry \( \Lambda \), then \( \text{middle}[n] = k \). (Note that \( \text{list1}[k] = 0 \) in this case.)

If the middle part of the \( n \)-th line is \( \text{PN} \), then \( \text{middle}[n] = -1 \).

If the middle part of the \( n \)-th line is \( \text{EB} \), and if \( \Omega_k \) is the extended indicator string of that line (i.e. the indicator string followed by \( n \)) then \( \text{middle}[n] = -100 -k \).

If the middle part of the \( n \)-th line is not \( \text{EB} \), and if \( \text{cat}[n] = k \), then \( \Omega_k \) is the EXPRESSION string consisting of just one entry, viz. the category part of the \( n \)-th line. (Whence \( \text{list1}[k] = 0 \) in this case.) If, however, the middle part is \( \text{EB} \), then \( \Omega_k \) is the category string of the extended indicator string of
that line. (If \( x_1, \ldots, x_j, n \) is the extended indicator string, then this category string is \( \Gamma_1, \ldots, \Gamma_j, \Gamma_{j+1} \), forming the categories of \( x_1, \ldots, x_j, n \), respectively.) Note that this difference between EB or non-EB applies to list\([\text{cat[n]}]\) only.

References.

