Use of equation parsers in forest information systems

Vesa, L.

North Karelia Polytechnic, Väisälänkatu 4, FIN-80160 Joensuu, Finland, Lauri.Vesa@ncp.fi

Abstract

A parser is a piece of software which takes a string and makes a syntactic analysis of it. An equation parser is a special type of parser which takes an equation as a string, parses and solves it, returning the result as a value. Currently available equation parsers can handle logical, mathematical, trigonometrical and statistical functions and they can solve very large, complex expressions with numerous levels of parentheses. Moreover, many of these parsers support variables.

Nowadays, there are equation parsers for all the most common programming languages. In the Windows™ environment, an equation parser can be very easily embeded in other programs as an add-on control (*e.g.* as a dynamic link library).

Rigorous testing of any parser with complex mathematical equations is an essential phase, when deciding on applicability of it. Moreover, the function of contants and variables has to be tested to avoid erroneus results. Many evaluation versions of equation parsers are obtainable via the Internet.

A reliable equation parser is an invaluable tool to use when programming workable systems for many purposes where equations are required. From the developers point of view, use of equation parsers makes code writing easier: if formulae are needed, these can be just written into relational database tables as they are in books. By using equation databases and parsing, one can in principle build up a forest information system which can be used in Finland as well as in Wales. The main problem is to ensure that tree species names and their equations are properly written into relational database tables and the order of computing is just the same. Practical uses for equation parsing can be found in application development, research and education.

Keywords: Equation parser, Forest information systems, Programming

1 Introduction

A parser is a piece of software which takes a string and makes a syntactic analysis of it. Parsers have long been studied in programming science and have been applied in syntantic and semantic parsing of languages: the language may be a so-called natural language or a formal language, such as a programming language. An equation parser is a special type of parser which takes an equation as a string, parses and solves it, returning the result either as a value or a string. Currently available equation parsers can handle logical, mathematical, trigonometrical and statistical functions; and can solve very large, complex expressions with numerous levels of parentheses. Moreover, many of these parsers support variables. The most common use of parsers can be found in programming compilers; equation parsers are also found in spreadsheet applications, where a cell value is solved by parser.

In forestry information systems many different kinds of mathematical equations are applied by writing algorithms straight into the programme code. Because similar model structures may be appropriate e.g. for different tree species, the most common method has been to write separate parameter files in which the values of parameters are stored. But this method allows change of the entire model form only if the form of the new model in known in advance (and this requires lot of *if-else* statements in the code). An equation parser will substitute all these computing rules.

2 Testing of equation parser

Nowadays, there are equation parsers for all the most common programming languages (e.g. C, MS Visual Basic and Borland Delphi). However, rigorous testing of any parser with complex mathematical equations is an essential phase, when deciding on its applicability. A dealer's promises concerning functionality of an equation parser are not always realistic: for example trigonometric or statistical functions embeded in complex mathematical equations do not always work as promised. Therefore, a programmer should always first test the candidate parsers. A sensible starting place is to study the order of presedence of the operands. The common order in computing is the following (in Pascal language, see Saikkonen & Voipio 1981):

```
NOT
^
*, /, DIV, MOD, AND
+, -, OR
=, <>, >, <, <=, >=
```

where

```
^ = power
DIV = division (integer part)
MOD = division (fraction part)
```

Moreover, the function of contants and variables has to be tested to avoid erroneus results. One possible source of error is that so-called double precision numbers (16 bits) which are rounded to fewer decimals (8 bit numbers). The function with upper and lower case letters in variable names should also be studied. Here

is an example of function of two commercial equation parsers.

Parser 1

Input string: ln(2)
Result: 0.693147181

Input string: ln(2) + ln(3)
Result: #ERROR

Parser 2

d = 20.0 (diameter, cm)
h = 23.1 (height, m)

Input string: 0.036089 * ***d*** ^ 2.01395 * 0.99676 ^ ***d*** * ***h*** ^ 2.07025 *(***h*** - 1.3) ^ (-1.07209)
Result: 181783811.92 (wrong)

Input string: 0.036089 * (***d*** ^ 2.01395) * (0.99676 ^ ***d***) *(***h*** ^ 2.07025) *((***h*** - 1.3) ^ (-1.07209))
Result: 344.72 (right)

In the previous example parser 1 could not solve the equation, in which there was more than one funtion call and parser 2 made the calculations in an unconventional order, corrected only when the exponentials were separated by parentheses. However, if these kind of errors are found, the programmer of equation parser is usually very willing to fix them.

3 Equations in relational database

Equation can be written into a relational database table as a string nearly the same way as they are written in books. The variable names must be fixed, and the values of the parameters are written into the table (Table 1).

If the user of an application is to be allowed to make changes to model database, then the checking of syntactics must be carried out before computation. Moreover, the allowed limits of variables can also be stored into the database table, if the equation in use supports logical functions (Table 2).

VolumeModel2: Taulukko		
Code	**Equation**	**ModelSource**
2	$0.022927*(d^{1.91505})*(099146^{d})*h^{2.82541})*((h-1.3)^{(-1.53547)})$	Laasasenaho 1982
3	$0.011197(d^{2.10253})*(0.986^{d})*(h^{3.98519})*((h-1.3)^{(-2.659)})$	Laasasenaho 1982
1	$0.036089*(d^{2.01395})*(0.99676^{d})*(h^{2.07025})*((h-1.3)^{(-1.07209)})$	Laasasenaho 1982

Table 1. Equations in relational database table. This table includes single tree volume functions for Scots pine (code 1), Norway spruce (2) and birch (3) (Laasasenaho 1982).

HeightModel: Taulukko				
Code	**Equation**	**Y_dependent**	**ModelSource**	**Rules**
1	$1.3+d*d/((A+B*d)^2)$	$d/((h-1.3)^{0.5})$	Näslund	AND(h > 1.3, d > 0)
2	1.3+A*d+B*d*d	(h-1.3)/d	Polynomic	AND(h > 1.3, d > 0)

Table 2. Example of checking rules in database (field *Rules*). This table includes tree height curve equations.

In Table 2 the parameter values are omitted, because this table is used as a tool for solving a tree height curve. The parameter values are computed by using simple linear regression formulae

$$\hat{Y} = A + B \times \boldsymbol{d}$$

where

$\hat{Y}$ = predicted value of the dependent variable
A, B = parameters
d = tree diameter

The dependent value is solved first by equation parser for each tree in the group of sample trees, then the regression analysis is carried out and values obtained for the parameters. After this the diameter value and parameters are inputed to the equation (in Table 2, field *Equation*) which will give out height value of a tree. This result is also solved by the equation parser.

Equation parsing can be easily applied to many static and dynamic models used in forestry applications. More programming is needed, if special jump command to different equations are needed when for instance maximum limits are exceeded. In the Windows (TM) environment, an equation parser can be very easily embeded in other programs as an add-on control (e.g. as a dynamic link library or ActiveX control). Many evaluation versions of equation parsers are obtainable via the Internet.

4 Conclusions

A reliable equation parser is an invaluable tool to use when making workable computing systems for many purposes where equations are required. From the developer's point of view, use of equation parsers makes code writing easier: if formulae are needed, these can be written into relational database tables just as they are in books. By using equation databases and parsing, one can in principle build up a forest information system which can be used in countries as diverse as Finland and China. The main problem is to ensure that tree species names and their equations are properly written into relational database tables and that the order of computing is still correct. Practical uses for equation parsing can be found in application development, research and education and the use of equation parsing is likely to make forestry applications more universal.

Acknowledgements

I am grateful to Mr. Tom Jenkins, University of Wales, Bangor, for heuristic syntantic parsing and commenting this paper.

References

Laasasenaho, J. 1982. Taper curve and volume functions for pine, spruce and birch. Comm. Inst. For. Fenn. 108. 74 p.

Saarenmaa, H. & Kaila, E. 1990. Tietokoneavusteinen päätöksenteko metsätaloudessa. Summary: Computer-aided decision making in forestry. Folia Forestalia 757. 34 p.

Saukkonen, H. & Voipio, R. 1981. Johdatus ohjelmointiin Pascal-kielen avulla. Teknillinen korkeakoulu, tietojenkäsittelyopin laboratorio. OTADATA. 284 p.