Logic–based Genetic Programming with Definite Clause Translation Grammars

Brian J. Ross
Brock University, Dept. of Computer Science
St. Catharines, Ontario, Canada L2S 3A1
bross@cosc.brocku.ca
(905) 688-5550 ext. 4284

DCTG-GP is a genetic programming system that uses definite clause translation grammars. A definite clause translation grammar (DCTG) is a logical version of an attribute grammar, which permits the BNF–style specification of context–free grammars (Abramson and Dahl 1989). The attribute portion of the DCTG permits multiple semantic definitions to be included within the syntactic definition of the grammar. The benefits of this for genetic programming are two–fold. Firstly, context–sensitive information pertinent to a problem being solved can be incorporated into the language definition, which can aid search efficiency. Secondly, the semantic component can conveniently define the complete computational semantics of a language, rather than requiring a separate interpreter or compiler.

In genetic programming, a variety of tree shapes and sizes in the initial population is conducive to evolution, as it enriches the genetic diversity required for latter generations. This can be difficult to ensure when using grammars. When grammar rules are randomly selected for generation of a program, there is no inherent information regarding whether a selected rule terminates too soon or is recursive. To overcome this, DCTG-GP uses a grammar pre–processor that analyzes DCTG grammar rules for such characteristics as whether rules are recursive or not, and the minimal depths of subtrees generated by them. This information is required for the effective generation of program trees of desired depths and shapes during population initialization and program mutation. Other work using grammars for GP has not done this, and so there is much less control over the shape of trees during initialization.

DCTG-GP encodes program trees as Prolog data structures. The semantic components are only interpreted if and when the user requires their execution. The internal tree representation contains information regarding the names of nonterminals used to generate the nodes. This information is used during reproduction. For example, selected subtrees for crossover have the same nonterminals roots in order to preserve grammatical correctness after exchange. Furthermore, if there are semantic goals to execute, they are interpreted after the exchange to ensure the semantic integrity of the children.

The DCTG-GP system is largely inspired by the LOGENPRO system (Wong and Leung 1997). LOGENPRO uses definite clause grammars, which are a subset of DCTGs. LOGENPRO also permits the encoding of semantic constraints within the language definition, which are encoded as embedded calls to logic programming code. An advantage of DCTG-GP in this regard is its use of attribute information. Much more complex semantic constraints can be incorporated into the grammar with considerable clarity, including entire interpreters for the language. The motivation for definite clause translation grammars is their ability to encode complex syntactic and semantic information together in one unified framework.

DCTG-GP is programmed in SICStus Prolog 3.7 on Silicon Graphics and Windows platforms. It is currently being used successfully in experiments in stochastic pattern recognition.

Acknowledgement: This research is supported by NSERC operating grant 138467-1998.

References

Abramson, H. and V. Dahl (1989). *Logic grammars.* Springer-Verlag.

Wong, M.L. and K.S. Leung (1997). Evolutionary Program Induction Directed by Logic Grammars. *Evolutionary Computation* **5**(2), 143–180.