Lindenmayer Systems, Coalgebraically
Short Paper

Baltasar Trancón y Widemann[1] and Joost Winter[2]

[1] Ecological Modelling, University of Bayreuth, Germany
[2] CWI, Amsterdam, Netherlands

1 Introduction

Lindenmayer systems, or L-systems, are a formal model of structural growth due to the theoretical botanist Lindenmayer in 1968, in the wake of research on Chomsky grammars in computer science. They have been immensely influential as a conceptual tool for botany, as well as an application domain for computer-based modelling [2]. Like grammars, L-systems come in various classes of complexity, with optional nondeterminism, context-sensitivity or parametrization. Unlike grammars, L-systems consider only derivations where all symbols are rewritten *in parallel*, that is, they are models of decentral, vegetative growth.

Context-free grammars have been given a coalgebraic treatment [5], extending earlier work on coalgebaic representations of regular languages and automata (see e.g. [3]). In [5], grammars in Greibach normal form (representing the sequential nature of application of rules, as opposed to the parallel rewriting of L-systems) are regarded as coalgebras over the functor $2 \times \left(\mathcal{P}(-)^*\right)^A$, which can then be extended to $2 \times (-)^A$-coalgebras by means of the generalized powerset construction.

2 Coalgebraic L-Systems

The classical presentation of L-systems follows the tradition of Chomsky grammars, and is strictly syntactic. Here we suggest an alternative, semantical perspective on L-systems. The key idea is to represent the single-step rules of an L-system as a finite coalgebra for a monadic functor on the category of sets. The Kleisli extension then yields a function that can be iterated to formalize multi-step derivations.

The simplest class of deterministic context-free L-systems without terminal symbols is modelled by the list functor and its standard monadic structure. Common extensions such as terminals, nondeterminism and probabilism can be added modularly by composing the list functor with the coproduct with a constant set, the covariant finitary powerset functor and the finitely supported distribution functor, respectively. Each of these comes with a standard monadic structure. In general, the composition of monads is not a monad, but for all pairs of the functors in question, distributive laws can be given that make for a composite monadic structure consistent with the traditional semantics of L-systems.

Parametric L-systems can be incorporated using a different, but equally semantical technique. The traditional syntactical approach is to hoist parameter datatypes, guard predicates and operators onto the finite set of basic symbols. Instead, we suggest to relax the condition of finiteness on the carriers of coalgebras, and merely require that they have a finite homomorphic image. Thus, the kernel quotient of that homomorphism can be seen as a finite collection of symbols, and the internal structure of each quotient class as parametrization.

Context-sensitive L-systems are a more complicated matter. As for the Chomsky grammar case, no obvious coalgebraic presentation is available. We conjecture that a bialgebraic approach is promising, by analogy to the bialgebraic semantics of cellular automata [4].

3 Conclusion

The coalgebraic presentation of L-systems is concise, elegant and natural. Some typical problems regarding L-systems reappear as standard coalgebraic notions in disguise, while other problems even become apparent only in coalgebraic form. As an example of the former effect, consider the vast subject of botanical reasoning with L-systems, namely the classification of branching structures and organ placement on higher plants (*phyllotaxis*): it can be understood as an instance of bisimulation. As an example of the latter effect, consider the definition of probabilistic L-systems in the definitive resource [2]: productions are weighted with probabilities, and it is obviously implied that parallel rewriting steps be *stochastically independent*, but an explicit statement has simply been forgotten. Such an oversight is not possible in the coalgebraic form; stochastic independence is exactly the content of the distributive law between lists and distributions.

L-systems as a model class are very easily understood and appealing to intuition. They showcase basic concepts of coalgebraic modelling in such a way that thay could be a useful pedagogic example in the introductory teaching of coalgebra.

References

1. Corradini, A., Klin, B., Cîrstea, C. (eds.): Proceedings 4th International Conference on Algebra and Coalgebra (CALCO 2011), Lecture Notes in Computer Science, vol. 6859. Springer (2011)
2. Prusinkiewicz, P., Lindenmayer, A.: The Algorithmic Beauty of Plants. Springer (1990)
3. Rutten, J.: Automata and coinduction (an exercise in coalgebra). Tech. Rep. SEN-R9803, CWI, Amsterdam (1998)
4. Trancón y Widemann, B., Hauhs, M.: Distributive-law semantics for cellular automata and agent-based models. In: Corradini et al. [1], pp. 344–358
5. Winter, J., Bonsangue, M.M., Rutten, J.: Context-free languages, coalgebraically. In: Corradini et al. [1], pp. 359–376