Integrated grammar representation of genes, metabolites and morphology: The example of hordeomorphs

*Gerhard Buck-Sorlin**,**, *Ole Kniemeyer** and *Winfried Kurth**

* **Brandenburgische Technische Universität Cottbus, Department of Computer Science, Chair for Practical Computer Science/Graphics Systems, P.O. Box 101344, D-03013 Cottbus, Germany, `http://www-gs.informatik.tu-cottbus.de/~wwwgs`**

****Institute of Plant Genetics and Crop Plant Research, Department of Cytogenetics, Corrensstr. 3, D-06466 Gatersleben, Germany, `http://taxon.ipk-gatersleben.de`**

Introduction

The **specification of biological models** in high-level **formal languages**, tailored to the purposes of the life-sciences, is highly desirable for the sake of transparency, compatibility and interfacing of models. In the field of ontogenetic development of the structure of plants, the formalism of **L-systems** has been proven to be able to capture essential aspects of growth and architecture, with applications (amongst others) in:

- herbaceous plants [10]
- agronomical crops [3, 1]
- trees and tree stands [7].

L-systems have been interfaced with process-oriented programs to specify functional-structural plant models as well [5, 9, 8].

However, **L-systems have still some drawbacks** – not so much concerning their theoretical power, but with respect to transparency and simple use when complex, multi-level plant models including functional aspects and/or genetic control are required.

One critical feature:
L-systems operate basically on *strings*, which have to be translated into 3D-structures (representing plants or plant communities), the latter being the actual objects of modelling (see Fig. 1, left part).

⇒ Our new development:

- The concept of **Relational Growth Grammars** (**RGG**)
- corresponding new language **XL** (eXtended L-Systems)

RGG = **graph grammar** formalism

nodes: objects in the sense of object-oriented programming,
- can be associated with Java classes
- generalize the symbols in classical L-system strings

edges: can represent arbitrary, user-defined relations
- generalize the sequential order of symbols in strings

Hence the extra description level of strings can be dispensed of in the rewriting process (Fig. 1, right part) – we use strings only for writing down the rules.

Advantages:

- **Complex relationships** such as genotype-phenotype relations can now be represented with the same simplicity as a topological neighbourhood in classical L-systems,
- the same holds for **multiscalar** plant descriptions [4],
- arbitrary sorts of **context** can easily be defined,
- the representation of **networks**, including feed-back loops, is possible in our formalism in an intuitive way (as graphs),
- the interface between rule-based model description and **procedural modelling** becomes more elegant by incorporating Java classes (as nodes) and scripts in the rules,
- **strings**, **trees** and **multisets** are **subcases** of our graph data structure [6], thus our RGG have at least the same descriptive power as the rewriting systems operating on these restricted structures.

Our example of virtual barley ears ("*hordeomorphs*") will demonstrate some of these features.

Representation of genetic processes

Demonstration at the example of **crossing-over** (*co*):
Probability of a *co* event between two loci = a function of the distance between the two genes
The closer the alleles of certain genes are located to each other on the same chromosome, the higher are their chances of being transmitted together to the same organism. Genes which are located on different chromosomes segregate independently.
Specification of mating and alignment of 2 strings representing the genome by a stochastic RGG rule: Fig. 2
(borders between chromosomes are simulated by recombination probabilities of 50 %)
Effect:

- crosswise exchange of the successor relations
- recombination at places determined according to given probabilities
- automatic update of the "containment" edges (connecting hierarchical levels)

Different mutation types can be encoded equally easily with stochastic RGG rules.

Representation of regulatory networks and dynamic morphology

Morphology = spatio-temporal manifestation of the effects of dynamic morphogenetic processes, based on regulatory networks of genes, transcription factors and metabolites, all of them to a varying degree influenced by environmental factors.
With our approach, we can represent the common situation that the **genome** is present in every part of the organism. Let a specific relation "x *–contains–>* g" depict the presence and accessibility of the genome string g in organ x. An RGG rule describing one developmental step at the macroscopic level, e.g. the formation of an internode from an apical meristem, can then have the following form (the parentheses (* *) denote a context in the graph which must be present in order for the rule to be applied):

Meristem(t) (* *–contains–>* g:*Genome* *)
→ **F**(internodesize(t) / (a + b*g[2]), ...) ... *Meristem*(t + Δt)

F = a new internode object (similar to the use of **F** in L-systems) with size depending on meristem age t (mediated by a predefined function "internodesize") and on the allele at locus 2 of genome g: **Direct control** of morphological features by the genes.
In reality, gene effects are **mediated by metabolites** (transcription factors, hormones), the latter representable by graph nodes as well, with their **concentrations** as numerical attributes. Using "activator" and "inhibitor" arcs between these nodes, the dynamics of the **regulatory network** can be simulated in discrete time steps by an RGG. Morphogenesis can be controlled by the levels of certain metabolites exceeding threshold values.

Environmental factors can be taken into account by relations putting nodes into an environmental context, e.g. geometric neighbourhood or light distribution. Applicability of rules controlling metabolite concentrations or (more directly) meristem behaviour can then depend upon the status of a context defined with the help of these relations.

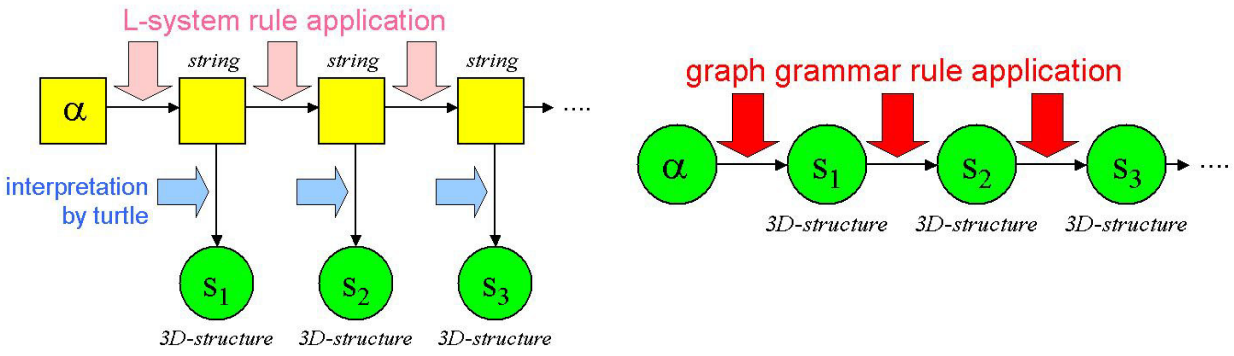

Figure 1: Functioning of a classical L-system (left), compared with a relational growth grammar (right). α is the start symbol (axiom). The developmental steps of a plant or plant community are represented by 3D-structures s_1, s_2, ...

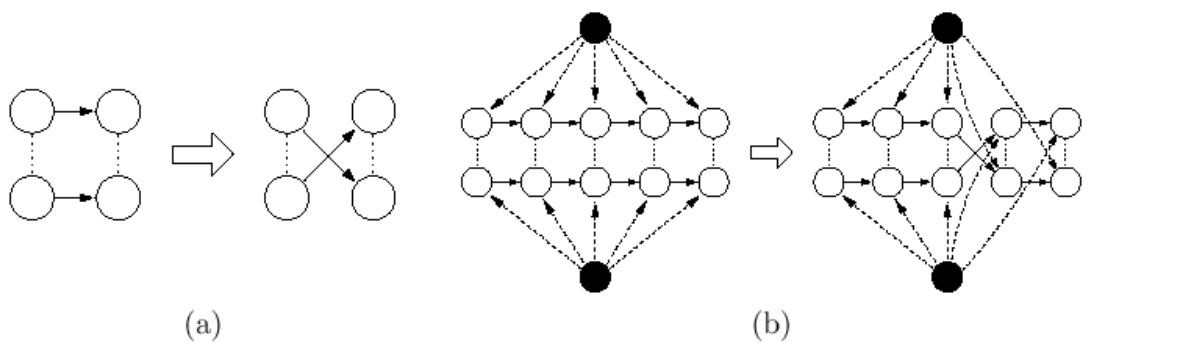

Figure 2: RGG rule representing crossing-over (a) and its application to a pair of strings (b). Continuous edges denote the "successor" relation, dashed edges containment, and point-edges alignment. In (b), the containment edges have been updated.

The example of "Hordeomorphs"

"Hordeomorphs":
- virtual creatures resembling **barley ears** (*Hordeum vulgare*)
- based on R. Dawkins' "**biomorphs**" [2] (XL-implementation: [6])

Rules imitate genetic operations **mutation**, **selection**, **asexual reproduction** and **sexual reproduction** (with suboperations "mating of chromosomes" and "crossing-over"). Rules account for:

- diploid genome of barley,
- genes spread over several chromosomes,
- different recombinatorial distances of genes to each other

Morphology based on earlier L-system with virtual genes [1].
Genes with two or three alleles determine colour, shape and biometry of ear organs:

- *vrs1* (number of spikelet rows)
- *lks2* (awn length)
- *Zeo* (ear density)
- *Blp* (lemma and pericarp colour)
- *glo-b* (globe-shaped grain)

Only *vrs1* and *Zeo* are located on the same chromosome (2H): their mutual recombination probability = 20% (approximated value for our model)
remaining genes segregate independently (probability of 50%)

The model is invoked using the new software tool **GroIMP** [6].
First, five different hordeomorphs are drawn on the output screen, representing the **visible phenotypes** of five different genomes. Initial variability is caused by "**mutate**" rule modifying the axiomatic genome.
The user then **selects** one or two **phenotypes** by clicking on a hordeomorph. The creation of the next generation is achieved by either of two rules "selectOne" or "selectTwo" (the latter containing the crossing-over functionality described above).
Also, rules "reproduce" and "mutate" are invoked:
"reproduce" ensures that a new population is produced out of the genome of the one or the combined two parents
"mutate" changes the allelic values of the genomes according to a given mutation probability.
Sample output of the model: see Fig. 3.

The model presented here could in the future be used as a "breeder's tool" (given adapted and extended developmental rules) in that it can use as input the recombination distances of an arbitrary set of marker genes and QTLs. The plant breeder could then specify the size of the output population and – among other boundary conditions – some "ideotype", i.e. ideal phenotype to be aimed at through the process of breeding. The model would then ideally compute potential "breeding path(s)" and the number of steps required to achieve this goal, as well as potential difficulties and bottlenecks (i.e. with target genes being located too closely to undesired genes).

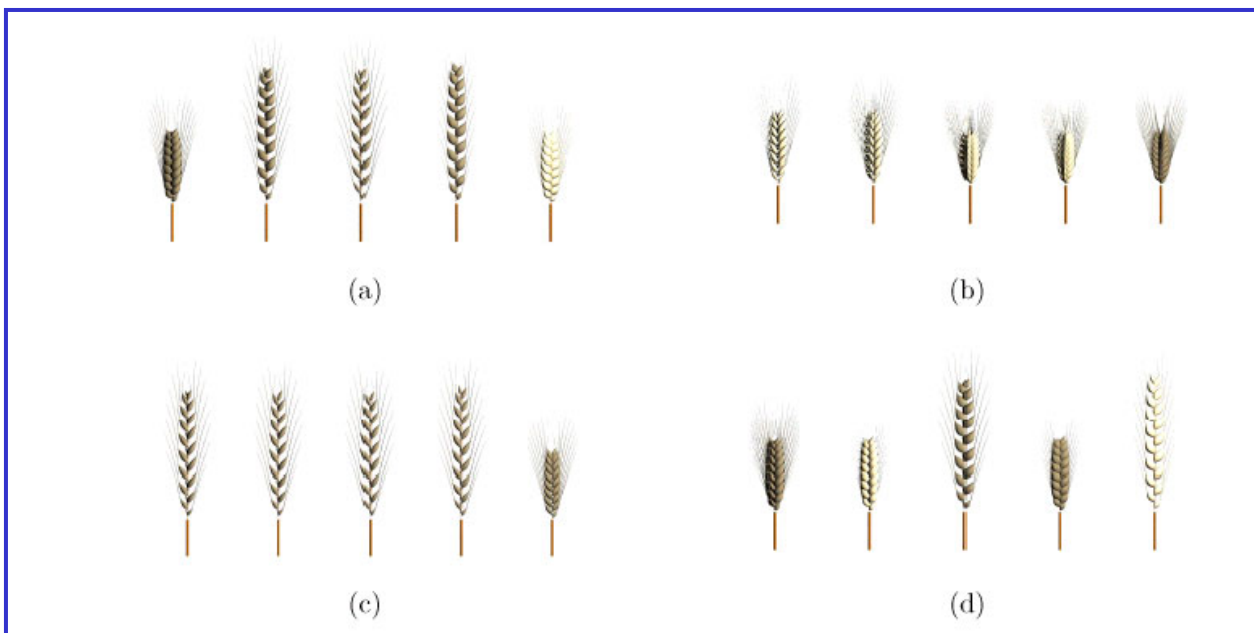

Figure 3: Sample output of the Hordeomorphs model. The first step (before user selection) is shown for four different initial diploid genomes. In (d), the mutation rate has been increased eightfold compared to (a) - (c).

References

[1] Buck-Sorlin, G.H., Bachmann, K.: Simulating the morphology of barley spike phenotypes using genotype information. Agronomie: Plant Genetics and Breeding 20 (2000) 691-702.
[2] Dawkins, R.: The Blind Watchmaker. Longman, Harlow (1986).
[3] Hanan, J.: Virtual plants – integrating architectural and physiological models. Environmental Modelling and Software 12 (1997) 35-42.
[4] Godin, C., Caraglio, Y.: A multiscale model of plant topological structures. J. Theor.Biol. 191 (1998) 1-46.
[5] Karwowski, R., Prusinkiewicz, P.: Design and implementation of the L+C modeling language. Electronic Notes in Theoretical Computer Science 86 (2), http://www.elsevier.nl/gej-ng/31/29/23/135/47/31/86.2.009.pdf (3 Sept. 2003).
[6] Kniemeyer, O., Buck-Sorlin, G.H., Kurth, W.: Representation of genotype and phenotype in a coherent framework based on extended L-Systems. In: W. Banzhaf et al. (eds.): Advances in Artificial Life. Lecture Notes in Artificial Intelligence 2801(2003) 625-634. Springer-Verlag, Berlin, Heidelberg.
[7] Lane, B., Prusinkiewicz, P.: Generating spatial distributions for multilevel models of plant communities. Proceedings of Graphics Interface 2002, Calgary (Can.), May 27-29 (2002) 69-80.
[8] Perttunen, J., Sievänen, R.: Incorporating Lindenmayer systems for architectural development in a functional-structural tree model. Preprint submitted to Elsevier Science (30 Sept. 2003).
[9] Prusinkiewicz, P., Hanan, J., Mech, R.: An L-system-based plant modeling language. - In: M. Nagl, A. Schürr, M. Münch (eds.): Proceedings of the International Workshop AGTIVE'99. LNCS 1779, Springer, Berlin (2000) 395-410.
[10] Prusinkiewicz, P., Lindenmayer, A.: The Algorithmic Beauty of Plants. Springer-Verlag, New York (1990).