Linear Ordered Graph Grammars for Mobile Systems *

Andrea Corradini[1], Ugo Montanari[1] and Leila Ribeiro[2]

[1] Dipartimento di Informática, Universitá di Pisa,
Pisa, Italy
andrea,ugo@di.unipi.it,
[2] Instituto de Informática, Universidade Federal do Rio Grande do Sul,
Porto Alegre, Brazil
leila@inf.ufrgs.br

Abstract. In this short paper we outline a special kind of graph grammars, called linear ordered graph grammars, that can be used to describe distributed systems with mobility and object-based systems. It is possible to model such grammars and their semantics in terms of tiles making explicit the aspects of interactivity and compositionality, that are of great importance in distributed systems.

1 Introduction

Wide area network programming is important and difficult to cope with using ad hoc techniques. In particular, testing in such environments has shown to be very inefficient due to the high dynamicity and possibility of failures of the environment. Thus, formal techniques to assure the correct behavior of the system are even more needed in this framework.

Graph transformation systems [9, 3] have been used in a wide variety of applications, and can provide suggestive and technically adequate models of computation, semantic foundations and verification methods. Considering that our aim is to be able to model mobility of processes within a system, a natural way of representing distributed systems graphically is to model communication channels as nodes (abstracting out the physical network), agents/processes/ambients as (hyper)arcs and ports as tentacles (for example, the modeling of π-calculus agents using graph transformations defined in [7]). If we want to have a more object-oriented view, we could see objects as nodes and messages as (hyper)arcs, as it was done in [2]. What is important to notice is that in both modelings, mobility involves the creation and passing of nodes, while the process/message interpretation of arcs requires resource consciousness ("linearity" in Girard sense). In the graph transformation modelings discussed above, this means that neither rules nor matches for rules may identify arcs (because they model resources). Actually, if we do not have this linearity, or resource consciousness, property, the construction of a suitable compositional true concurrency semantics (based on concatenable processes, for example) is not possible.

We propose to specialize the existing single pushout (SPO) approach to graph transformations [4] to a kind of grammars that can be used to model both distributed systems with mobility and object-based systems. These grammars will be called *linear ordered graph grammars*. The order property is achieved

* This work was partially supported by the projects AGILE (FET project), COMETA (MIUR project), IQ-Mobile (CNPq and CNR) and ForMOS (CNPq and Fapergs).

by imposing a total ordering on nodes and arcs, and the linearity property by allowing only injective morphisms on items that represent resources (arcs).

Interactivity and abstract semantics via observations are key features of communicating systems. Compositionality is also very important for large distributed systems. Tiles have these as their main aims, and previous work has shown their feasibility [5]. In addition, they exhibit an algebraic flavor (tiles have a straightforward axiomatization as monoidal double categories) which may allow for universal constructions, compositionality and refinement in the classical style of algebraic semantics [6]. We show that the representation of linear ordered graph grammars - and of the corresponding SPO graph transformations - within tile logic is simple and natural. In this way, we provide a basis for an algebraic theory of graph grammars where observations and interactions are in the foreground. This view is not usually emphasized in the DPO/SPO literature, although it is a very important aspect of the kind of systems we are interested in modeling.

2 Outline of Work

In this short paper we cannot give a precise definition of linear ordered graph grammars, of their SPO derivations and of their compositional algebraic semantics based on tiles. We refer for that to [8]. Roughly speaking, *ordered graphs* are typed hypergraphs where both nodes and (hyper)arcs are totally ordered. Furthermore, *linear* rewriting rules for ordered graphs neither erase nodes nor fuse arcs, but they can create and fuse nodes and create and erase arcs. Finally, SPO derivation steps require the matching morphism to be injective on arcs.

To construct the tile model associated to a linear ordered graph grammar one first builds a tile rewrite system where the horizontal arrows (which represent states) are the arrows of the gs-monoidal theory freely generated by a signature suitably derived from the graph types. Moreover, the vertical arrows (which represent observations) are also freely generated (by an empty signature). Finally, basic tiles correspond directly to the rules of the grammar. One-step derivations of the grammar are characterized as corresponding to certain tiles generated by the tile rewrite system.

The motivation for translating linear ordered graph grammars into tile rewrite systems is that, differently from graph grammars, both tile rewrite systems and tile models for linear ordered graph grammars can be specified algebraically in a straightforward way. Also, a specification morphism can be associated to the operational semantics. This property automatically defines a universal, functorial construction for the latter which guarantees preservation of colimits, i.e. compositionality of the semantics. These highly desirable constructions are not available for existing graph transformation systems.

More in detail, the main aspects of our approach are:

• Types in graphs correspond to signatures in tiles: typed graphs correspond to gs-monoidal arrows of a restricted type on a signature. Graph isomorphism is directly represented by gs-monoidal axioms. Moreover, the set of all arrows has a natural meaning in terms of open graphs.

• Graph morphisms of the special kind we need are obtained by choosing the vertical structure of tiles adequately. It is immediately apparent the correspondence between the vertical "wire structure" (again a variation of symmetric monoidal categories) and the properties of morphisms.

• A SPO can be represented by a horizontal tile construction. Again, the pushout property becomes concrete in terms of certain tiles which can be constructed starting from a finite number of auxiliary tiles.

• The effect of a derivation can be represented by a (flat) tile. We expect that canonical derivations are exactly represented by tiles equipped with their proof terms (in the Curry-Howard style). Note that we did not restrict to injective rules (as it is done in almost all true concurrency models for graph grammars).
• The observable effect of a derivation (which is not usually emphasized in the SPO/DPO literature) is very important in the interactive interpretation of our models of computations. It represents the creation/fusions of channels and creation/termination of processes which took place in the computation. Typically, a useful way of connecting systems is via composition of their states, that can be done through shared channels. Therefore it is important to have information about these channels in interfaces (that is, make them observable) to obtain a compositional semantics.
• The model would be more satisfactory introducing also restriction/hiding of channels and processes, which would be straightforward in the tile model. This would correspond to new and interesting classes of graphs, making even more explicit the advantage of establishing a connection between graph transformations and tiles.
• Here the vertical tile structure only represents graph morphism. However we believe possible to use it also to model graph components which are employed only during transactions to exchange resources and synchronization informations. Tiles without this kind of components in the observations would correspond to fully committed transactions. This construction, if actually possible, would extend to graph transformation systems the approach based on *zero-safe nets* [1] where two kinds of places, *stable* and *zero* places, represent ordinary components and components only present in transactions respectively.

References

1. R. Bruni and U. Montanari, *Zero-Safe Nets: Comparing the Collective and Individual Token Approaches*, Information and Computation, Vol. 156, 2000, pp. 46–89.
2. F. Dotti and L. Ribeiro, *Specification of mobile code systems using graph grammars*, Formal Methods for Open Object-based Systems IV, Kluwer Academic Publishers, 2000, pp.45–64.
3. H. Ehrig, G. Engels, H.-J. Kreowski, and G. Rozenberg, editors, *Handbook of Graph Grammars and Computing by Graph Transformation, Volume 2: Applications, Languages and Tools*, World Scientific, 1999.
4. H. Ehrig, R. Heckel, M. Korff, M. Lwe, L. Ribeiro, A. Wagner and A. Corradini, *Algebraic approaches to graph transformation II: Single pushout approach and comparison with double pushout approach*, in [9], pp. 247–312.
5. F. Gadducci and U. Montanari, *Comparing Logics for Rewriting: Rewriting Logic, Action Calculi and Tile Logic*, TCS, Vol. 285, Issue 2, 28 August 2002, Pages 319-358.
6. J. Meseguer and U. Montanari. *Mapping Tile Logic into Rewriting Logic*, Springer, LNCS 1376, 1998, pp. 62–91.
7. U. Montanari, M. Pistore and F. Rossi, *Modeling concurrent, mobile and coordinated systems via graph transformations*, The Handbook of Graph Grammars, vol. 3: Concurrency, Parallelism and Distribution, World Scientific, 1999, pp. 189–268.
8. U. Montanari and L. Ribeiro, *Linear Ordered Graph Grammars and Their Algebraic Foundations*, in: Andrea Corradini, Hartmut Ehrig, Hans-Jörg Kreowski and Grzegorz Rozenberg, Eds., Graph Transformation, ICGT 2002, Springer LNCS 2505, pp. 317-333.
9. G. Rozenberg (editor), *The Handbook of Graph Grammars, vol. 1: Foundations*, World Scientific, 1997.