From Graph Rewriting to Logic Programming

Ivan Lanese and Ugo Montanari
Dipartimento di Informatica, Università di Pisa
e-mail:{lanese,ugo}@di.unipi.it

Models are an useful approach to the study of systems in general and of distributed systems in particular. We are interested especially in models for network aware programming. Among these models there are some which, for their generality, can be used as metamodels, thus allowing one to compare and relate the other models by analyzing them in a common framework.
We will study one of them, graph rewriting, which utilizes hypergraphs to model networks. A hypergraph is composed by hyperedges, which represent machines or processes, and nodes, which stand for ports of communication.
We have chosen a distributed approach to network reconfiguration because this simplifies the implementations. Our approach [2, 3, 4] is called synchronized hyperedge replacement (SHR) and allows a rule to rewrite only a single edge (local reconfiguration) but it combines many rules in a global rewriting by imposing synchronization constraints on common nodes.
This approach allows also to model mobility, which is very important in order to model aspects of the real world such as mobile computers or telephones or Java applets. In particular we use name mobility, where a name can be transmitted from a place to another. Another important aspect is the concept of hiding which is modelled in SHR with an hiding operator that defines the scope of names in order to model restricted access to local information.
Our main contribution is to analyze the relations between SHR and logic programming. This is interesting because at a first glance they are quite different models, but in fact they are analogous for many aspects, and finding relations between them allows many new useful insights about both models. Moreover this allows to exploit the work that has been done in the logic programming field in the world of mobile distributed computing.
To achieve this aim, we do not focus on the traditional view of logic programming for search or theorem proving, but on a view of logic programming [1] seen as an interactive goal rewriting system. In fact we are also interested in partial computations and not only in refutations and in their results. The idea is to understand logical predicates as (possibly open) interacting agents,

with computed substitutions as observations and whose local evolution is coordinated by the unification engine.
We will show that the fundamental mechanism of computation of SHR is essentially the same of logic programming. However between the two formalisms there are some differences. We then need to define a restriction of usual logic programming, which we call synchronized logic programming, in order to preserve only the computations meaningful for SHR.
We will show that there is a straightforward one-to-one correspondence between synchronized logic programming and SHR without the restriction operator.
Furthermore the mapping can be extended by introducing a restriction operator into logic programming. This extension is important because it fills a gap between logic programming and network models, but it is also very natural, because the restriction operator clarifies some aspects of handling of names in traditional logic programming.
A simple application of our mapping is the possibility to build an implementation of SHR through logic programming. To this end we have built a toy meta-interpreter for synchronized logic programming with restriction, which is essentially a simulator of an SHR system while no other implementations of SHR are available by now.

References

[1] Roberto Bruni, Ugo Montanari, and Francesca Rossi. An interactive semantics of logic programming. *Theory and Practice of Logic Programming*, 1(6):647–690, 2001.

[2] GianLuigi Ferrari, Ugo Montanari, and Emilio Tuosto. A lts semantics of ambients via graph synchronization with mobility. In Simona Ronchi Della Rocca Antonio Restivo and Luca Roversi, editors, *Proc. of ICTCS'01*, volume 2202 of *LNCS*, pages 1–16. Springer, October 2001.

[3] Dan Hirsch, Paola Inverardi, and Ugo Montanari. Reconfiguration of software architecture styles with name mobility. In *Proceedings of the 4th International Conference on Coordination Languages and Models, Coordination 2000*, volume 1906 of *Lecture Notes in Computer Science*, February 2000.

[4] Barbara König and Ugo Montanari. Observational equivalence for synchronized graph rewriting. In Springer Verlag, editor, *Proc. TACS'01*, LNCS, 2001.