Port Graphs, Rules and Strategies for Dynamic Data Analytics -Extended Abstract

Hélène Kirchner

Port Graphs, Rules and Strategies for Dynamic Data Analytics - Extended Abstract

Hélène Kirchner[1]

1 **Inria**
France
helene.kirchner@inria.fr

Abstract

In the context of understanding, planning and anticipating the behaviour of complex systems, such as biological networks or social networks, this paper proposes port graphs, rules and strategies, combined in strategic rewrite programs, as foundational ingredients for interactive and visual programming and shows how they can contribute to dynamic data analytics.

1998 ACM Subject Classification D.3.3 Language Constructs and Features (E.2)

Keywords and phrases Graphs, Rewrite Rules, Strategic Rewriting

Digital Object Identifier 10.4230/LIPIcs.RTA.2015.x

1 Introduction

Understanding, planning and anticipating the behaviour of complex systems, such as biological networks or social networks, raise a number of theoretical and practical challenges. Their complexity comes from heterogeneity of components, from their dynamics, their increasing number, or from the data deluge they generate or manage. Handling this complexity requires languages with a high-level of expressivity and with modular constructs. But this also needs powerful visualization tools to represent data and their dynamic evolution, analysis of different alternatives, parameter tuning. Capability of re-playing or backtracking are important concerns to address.

Three ingredients contribute to address these challenges: port graphs provide a powerful representation of data, rules capture their evolution and provide high-level prototyping mechanism, strategy makes the control and choices explicit. They are combined in the concept of strategic rewrite programs whose logical and semantic background is well-understood. The interactive PORGY environment, yet in-progress, illustrates what can be done and the actual visualization challenges, through an application to the study of propagation in social networks.

2 Data and Graphs

Graph formalisms are useful to easily describe complex structures in an intuitive way, like UML diagrams, proof representation, micro-processors design, or workflows. Port graphs [1] are a general class of labelled graphs that have been used to model a variety of complex data, such as biological networks or social networks. Intuitively, a port graph is a graph where nodes have explicit connection points called ports. Edges are undirected, exclusively attached to ports and two ports may be connected by more than one edge. Nodes, ports and edges are labelled by a set of properties. For instance, an edge may be associated with a state (such as used or marked) and a node may have for instance a name, a colour and a user-defined function as properties. Properties may be used to define the behaviour of the modelled system and for visualization purposes. Thanks to these features, port graphs provide a rich structure able to model many kinds of data and processes.

3 Rewrite rule programming

Rewriting has to be understood here in a broad sense: rewriting transforms syntactic structures that may be words, terms, propositions, dags, graphs, geometric objects like segments, and in general any kind of structured objects. Transformations are expressed with rules, built on the same syntax but with an additional set of variables and with a binder $\Rightarrow$, relating the left and the right-hand side of the rule. Optionally, a condition or constraint restricts the set of values allowed for the variables.

In this rewriting process, there are many possible choices: the rule itself, the position(s) in the structure, the matching homomorphism(s). For instance, one may choose to apply a rule concurrently at all disjoint positions where it matches, or using matching modulo an equational theory like associativity-commutativity.

Rewriting Logic [8] and Rewriting Calculus [5] have contributed to establish rewriting as a model of computation accounting for concurrency, parallelism, communication, and interaction. It also has good properties as a metalogical framework for representing logics.

Graph transformations have many applications in specification, programming, and simulation tools and several languages and tools are based on this formalism. The dynamics of a complex system modeled by a port graph can then be specified using graph rewriting rules.

4 Control via Strategies

While rules describe local transformations, strategies describe the control of rule application. Strategy is an explicit concept in sequential path-building games, in automated deduction and reasoning systems and more generally are used to express complex designs for control in modeling, proof search, program transformation, SAT solving. In these domains, deterministic rule-based computations or deductions are often not sufficient to capture complex computations or proof developments. Strategies provide the formal mechanism needed, for instance, to sequentialize the search for different solutions, to check context conditions, to request user input to instantiate variables, to process subgoals in a particular order, etc.

Several approaches exist to describe strategies: a proof term expressed in rewriting logic; a ρ-term in rewriting calculus; a subset of paths in a derivation tree ; a partial function that associates to a reduction-in-progress, the possible next steps in the reduction sequence ; positional strategies that choose where rules are allowed to apply.

A few strategy languages have been designed in the rewriting community, in particular in ELAN [4], Stratego [11], Maude [6] or Tom [2]. Common constructs with some variants are emerging from these proposals.

5 Strategic Programming

Port graphs, rules and strategies are combined in the concept of strategic programs. A *strategic rewrite program* consists of a finite set of rewrite rules $\mathcal{R}$, a strategy expression S, built from $\mathcal{R}$ using a strategy language, and a given structure G.

There are several ways to describe the operational semantics of a programming language. Due to the fact that rewriting logic is reflexive, it is tempting to describe the operational semantics of a strategy language with a set of rewrite rules. This has been done for instance for ELAN [3], Maude [6] and PORGY [1] at least. Another way by defining a transition relation on configurations using semantic rules in the SOS style is given in [7].

As presented in [9], it is expected from a strategy language and its operational semantics to satisfy the properties of correctness and completeness w.r.t. rewriting derivations.

6 The PORGY environment

PORGY [1, 7] is a visual environment that allows users to define port graphs and port graph rewrite rules, and to apply the rewrite rules in an interactive way, or via the use of strategies. To control the application of rewriting rules, PORGY provides a strategy language. A distinctive feature of PORGY's language is that it allows users to define strategies using not only operators to combine graph rewriting rules but also operators to define the location in the target graph where rules should, or should not, apply. Users can create graph rewriting derivations and specify graph traversals using the language primitives to select rewriting rules and the subgraphs where the rules apply.

In order to support the various tasks involved in the study of a port graph rewriting system, the system provides facilities :
- to offer different views on each component of the rewriting system: the current graph being rewritten, the derivation tree, the rules and the strategy, with drag-and-drop mechanisms to apply rules and strategies on a given state.
- to explore a derivation tree with all possible derivations,
- to perform on-demand reduction using a strategy language which permits to restrict or guide the reductions,
- to track the reduction throughout the whole tree,
- to navigate in the tree, for instance, backtracking and exploring different branches,
- to plot the evolution of a chosen parameter (a specific element in the port graph structure) along a derivation. The system supports synchronisation between the different views: selecting points on the plot view triggers the selection of the corresponding nodes in the trace tree. Such a mechanism obviously helps to track properties of the output graph along the rewriting process. These features have been successfully applied to propose a visual analytics approach to compare propagation models in social networks in [10].

7 Conclusion

Although first results are promising, this approach raises numerous challenges in different domains; let un mention some of them. Considering the huge amount of data to manage, knowledge representation in a structured way should allow better efficient mining, reasoning and inference. For pattern matching on big graphs with millions of nodes, fast or fuzzy matching can be explored as well as exploiting the graph structure. Graph rewriting to model big graphs evolution has to be adapted to their increasing sizes, both for efficient concurrent application of rules and for adapting the granularity of transformation steps to the property of interest. Strategies should help in this respect. From the point of view of strategies and strategic rewrite programs, properties of confluence, termination, or completeness for rewriting under strategies have been already addressed. But other properties of strategies such as fairness or loop-freeness could be worth-fully explored by making connections between different communities (functional programming, proof theory, verification, game theory,...). Finally, at all levels of graphs, rules and strategies, as well as for visualisation of processes evolution, structuration is a challenge and modular constructs for composability have to be studied in a way coherent between all these levels.

Acknowledgements The results presented here are based on joint work on ELAN and Tom languages designed in the Protheo team from 1990 to 2008, and on PORGY since 2008. I am grateful to José Meseguer and to the members of the Protheo and the PORGY teams, for many inspiring discussions on the topics of this paper.

References

1 Oana Andrei, Maribel Fernandez, Hélène Kirchner, Guy Melançon, Olivier Namet, and Bruno Pinaud. PORGY: Strategy-Driven Interactive Transformation of Graphs. In Rachid Echahed, editor, *TERMGRAPH, 6th Int. Workshop on Computing with Terms and Graphs*, volume 48 of *Electronic Proceedings in Theoretical Computer Science (EPTCS)*, pages 54–68, 2011.

2 Emilie Balland, Paul Brauner, Radu Kopetz, Pierre-Etienne Moreau, and Antoine Reilles. Tom: Piggybacking Rewriting on Java. In Franz Baader, editor, *Proceedings of the 18th Conference on Rewriting Techniques and Applications*, volume 4533 of *Lecture Notes in Computer Science*, pages 36–47. Springer, 2007.

3 Peter Borovanský, Claude Kirchner, Hélène Kirchner, and Pierre-Etienne Moreau. ELAN from a rewriting logic point of view. *Theoretical Computer Science*, 2(285):155–185, July 2002.

4 Peter Borovanský, Claude Kirchner, Hélène Kirchner, Pierre-Etienne Moreau, and Christophe Ringeissen. An overview of ELAN. *Electr. Notes Theor. Comput. Sci.*, 15:55–70, 1998.

5 Horatiu Cirstea and Claude Kirchner. The rewriting calculus — Part I *and* II. *Logic Journal of the Interest Group in Pure and Applied Logics*, 9(3):427–498, May 2001.

6 Manuel Clavel, Francisco Durán, Steven Eker, Patrick Lincoln, Narciso Martí-Oliet, José Meseguer, and Carolyn Talcott. *All About Maude - A High-Performance Logical Framework, How to Specify, Program and Verify Systems in Rewriting Logic*, volume 4350. Springer, 2007.

7 Maribel Fernández, Hélène Kirchner, and Olivier Namet. Strategic port graph rewriting: an interactive modelling and analysis framework. In Alberto Lluch Lafuente Dragan Bosnacki, Stefan Edelkamp and Anton Wij, editors, *Proceedings 3rd Workshop on GRAPH Inspection and Traversal Engineering (GRAPHITE 2014), Grenoble, France, 5th April 2014*, volume 159 of *Electronic Proceedings in Theoretical Computer Science*, pages 15–29, 2014.

8 Narciso Martí-Oliet and José Meseguer. Rewriting logic as a logical and semantic framework. In J. Meseguer, editor, *Electronic Notes in Theoretical Computer Science*, volume 4. Elsevier Science Publishers, 2000.

9 Narciso Martí-Oliet, José Meseguer, and Alberto Verdejo. A rewriting semantics for Maude strategies. *Electronic Notes in Theoretical Computer Science*, 238(3):227–247, 2008.

10 Jason Vallet, Hélène Kirchner, Guy Melançon, Olivier Namet, and Bruno Pinaud. A visual analytics approach to compare propagation models in social networks. In Arend Rensink and Eduardo Zambon, editors, *Graphs as Models*, volume 181 of *Electronic Proceedings in Theoretical Computer Science (EPTCS)*, 2015.

11 Eelco Visser. Stratego: A language for program transformation based on rewriting strategies. System description of Stratego 0.5. In A. Middeldorp, editor, *Rewriting Techniques and Applications (RTA'01)*, volume 2051 of *Lecture Notes in Computer Science*, pages 357–361. Springer, May 2001.