The Journal of Open Source Software

BiGGer: A Model Transformation Tool written in Java for Bigraph Rewriting in GrGen.NET

Dominik Grzelak [1,2¶]

1 Chair of Software Technology, Technische Universität Dresden, Germany **2** The Centre for Tactile Internet with Human-in-the-Loop (CeTI), Germany ¶ Corresponding author

DOI: 10.21105/joss.06491

Software

- Review
- Repository
- Archive

Editor: Vissarion Fisikopoulos

Reviewers:

- @abhishektiwari
- @idoby

Submitted: 09 February 2024
Published: 11 June 2024

Summary

Graphs are well-studied mathematical structures that have diverse applications in fields such as computer science, chemistry, biology and social sciences. In this respect, graph rewriting is a powerful technique that allows for the manipulation of graph structures through the application of so-called *graph transformation rules*.[1] In other words, graph rewriting techniques elevate static graphs to the concept of time-varying graphs.

BiGGer is a Java library that implements a novel approach to graph rewriting for *bigraphs*, as devised by Robin Milner (Milner, 2009), using the graph transformation tool *GrGen.NET* (Geiß et al., 2006).[2] Bigraphs provide a compositional framework to model graph structures with two semantic dimensions that can be reconfigured by rules.

With regard to BiGGer, bigraphs are specified using a meta-modeling approach (Grzelak, 2023; Kehrer et al., 2016) that is based on the EMOF standard (ISO/IEC 19508:2014, 2014), which is common in the software engineering sciences, where this library is most useful. Grounding bigraphs on metamodels facilitate its construction via a universal and platform-agnostic language.

Ultimately, this library transforms EMOF-complaint bigraphs into multigraphs that GrGen.NET can visualize and execute. Furthermore, BiGGer is also shipped as a command-line tool for using the functionality via the terminal for experimentation. The most challenging aspect was the accurate translation of bigraphical rules to SPO-based rules, given that bigraphs and GrGen.NET employ distinct approaches regarding graph rewriting.[3]

Statement of Need

Primary Purpose

The tool is built upon the formal principles of bigraphs, allowing for the intuitive and expressive representation of dynamic graph structures equipped with two semantic dimensions. In the domain of bigraph rewriting, the Java library and command-line utility BiGGer provide a highly efficient solution by streamlining the intricacies linked to model conversions and system simulation when bigraphs and GrGen.NET are utilized in conjunction. The fundamental objective of BiGGer is to establish an alternative and resilient setting for bigraph matching and rewriting. This will enable the simulation of state transitions in software applications, models, and systems, as well as in any other field where graph-structured data possesses inherent advantages over simple data structures such as scalars, lists, or pointers.

[1] The terms graph rewriting and graph transformation are used interchangeably.
[2] https://grgen.de/
[3] For an analysis of the interrelated aspects of these approaches, refer to (Ehrig, 2004; Milner, 2005).

Efficiency

When considering the manipulation of bigraphs through rules, the central elements that require implementation are bigraph matching and rewriting. This issue is mathematically known as the *subgraph isomorphism problem*, a general variant of the graph isomorphism problem. Until now, it is unknown whether the test for graph isomorphism can be solved in polynomial time or whether it is NP-complete. Subgraph isomorphism is computationally more complex than graph isomorphism and is known to be NP-complete due to the combinatorial nature of finding subpatterns in graphs. Thus, graph rewriting is NP-complete (Bacci et al., 2014).

However, bigraphs require the matching and rewriting of two substructures. Specifically, one substructure of a bigraph is a *forest*, which is relevant to the problem of locating forests in trees (Bacci et al., 2014), while the other substructure is a *hypergraph*, which is associated with the subhypergraph matching problem. To date, practical bigraph matching primarily exists as a CSP implementation (Miculan & Peressotti, 2014), or SAT-based algorithm (Sevegnani et al., 2010) in the literature.

GrGen.NET provides a universal framework written in C# implementing the so-called Single-Pushout (SPO) approach of graph rewriting for multigraphs (see, for example, (Ehrig et al., 2006; Jakumeit et al., 2023)). The choice to use GrGen.NET instead of other common graph rewrite systems in the software domain (e.g., AGG, PROGRES, FUJABA) is its execution performance with respect to matching and rewriting (refer to the benchmark described in (Geiß & Kroll, 2007), which was introduced in (Varro et al., 2005) first). Specifically, GrGen.NET's pattern matching engine employs some techniques that increase the practical execution performance of subgraph matching, which is an NP-complete problem but is at least one order of magnitude faster than comparable candidates (Jakumeit et al., 2023).

Key Features

The tool implements a unidirectional transformation from bigraph models to GrGen.NET models, and generates several files that GrGen.NET can execute and visualize. These include GrGen.NET's graph metamodel (`*.gm`), the graph model conforming to the metamodel (`*.grs`), rules (`*.grg`), and a script that configures the initial graph state and defines a simple rule execution strategy (`*.grs`). In turn this means that BiGGer is able to process the signature, bigraph, rules and tracking maps specified in the Eclipse EMF Ecore format (Steinberg et al., 2008), which implements the EMOF standard (ISO/IEC 19508:2014, 2014). Refer to www.bigraphs.org on how to model EMOF-compliant bigraphs in Java, visually or via the dedicated domain-specific language (Grzelak, 2021).

The functionality of BiGGer is offered via different interfaces suited for users of different backgrounds: programmatically in Java via the API, or by using the command-line tool. Developers and students benefit from the integration of BiGGer with other software systems relying on bigraph rewriting via an object-oriented API. BiGGer supports the efficient execution of transformation rules on large bigraphs currently available in the literature.

Moreover, BiGGer is shipped with sample bigraphs and rules and detailed instructions, which are essential in facilitating a better understanding of BiGGer's capabilities and usage.

State of the Field

There are few practical solutions concerned with the bigraph matching problem for various kinds of bigraphs; a non-exhaustive presentation is the following. For *binding bigraphs* (i.e., links have local scopes) by an inductive characterization of matching (Birkedal et al., 2007; Damgaard et al., 2013); for *directed bigraphs* (which subsume *pure bigraphs*) (Bacci et al., 2009); for *bigraphs with sharing* (i.e., the place graph is a directed acyclic graph) by using a SAT-based algorithm (Sevegnani & Calder, 2016); for the pure case by (Miculan & Peressotti,

2014) as a CSP, and further an adapted reduction of the problem for directed bigraphs to a CSP (Chiapperini et al., 2020); and the work in (Gassara et al., 2019), which proposes a toolchain for bigraph matching that is conceptually most similar to our approach but not actively developed anymore and not as efficient as GrGen.NET.

Acknowledgements

Funded by the German Research Foundation (DFG, Deutsche Forschungsgemeinschaft) as part of Germany's Excellence Strategy – EXC 2050/1 – Project ID 390696704 – Cluster of Excellence "Centre for Tactile Internet with Human-in-the-Loop" (CeTI) of Technische Universität Dresden.

References

Bacci, G., Grohmann, D., & Miculan, M. (2009). DBtk: A Toolkit for Directed Bigraphs. *Proceedings of the 3rd International Conference on Algebra and Coalgebra in Computer Science*, 413–422. https://doi.org/10.1007/978-3-642-03741-2_28

Bacci, G., Miculan, M., & Rizzi, R. (2014). Finding a Forest in a Tree. In M. Maffei & E. Tuosto (Eds.), *Trustworthy Global Computing* (pp. 17–33). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-662-45917-1_2

Birkedal, L., Damgaard, T. C., Glenstrup, A. J., & Milner, R. (2007). Matching of Bigraphs. *Electronic Notes in Theoretical Computer Science*, *175*(4), 3–19. https://doi.org/10.1016/j.entcs.2007.04.013

Chiapperini, A., Miculan, M., & Peressotti, M. (2020). Computing Embeddings of Directed Bigraphs. In F. Gadducci & T. Kehrer (Eds.), *Graph Transformation* (pp. 38–56). Springer International Publishing. https://doi.org/10.1007/978-3-030-51372-6_3

Damgaard, T. C., Glenstrup, A. J., Birkedal, L., & Milner, R. (2013). An inductive characterization of matching in binding bigraphs. *Formal Aspects of Computing*, *25*(2), 257–288. https://doi.org/10.1007/s00165-011-0184-5

Ehrig, H. (2004). Bigraphs meet double pushouts. In *Current Trends in Theoretical Computer Science* (Vol. 1–0, pp. 27–40). WORLD SCIENTIFIC. https://doi.org/10.1142/9789812562494_0038

Ehrig, H., Ehrig, K., Prange, U., & Taentzer, G. (2006). *Fundamentals of Algebraic Graph Transformation*. Springer-Verlag. https://doi.org/10.1007/3-540-31188-2

Gassara, A., Bouassida Rodriguez, I., Jmaiel, M., & Drira, K. (2019). Executing bigraphical reactive systems. *Discrete Applied Mathematics*, *253*, 73–92. https://doi.org/10.1016/j.dam.2018.07.006

Geiß, R., Batz, G. V., Grund, D., Hack, S., & Szalkowski, A. (2006). GrGen: A Fast SPO-Based Graph Rewriting Tool. In A. Corradini, H. Ehrig, U. Montanari, L. Ribeiro, & G. Rozenberg (Eds.), *Graph Transformations* (pp. 383–397). Springer Berlin Heidelberg. https://doi.org/10.1007/11841883_27

Geiß, R., & Kroll, M. (2007). *On improvements of the Varró Benchmark for graph transformation tools* (No. 7; Interner Bericht. Fakultät Für Informatik, Universität Karlsruhe, Vol. 2007). Universität Karlsruhe (TH). https://doi.org/10.5445/IR/1000007369

Grzelak, D. (2021). *Bigraphical Domain-specific Language (BDSL): User Manual* [Report]. Technische Universität Dresden. https://nbn-resolving.org/urn:nbn:de:bsz:14-qucosa2-752170

Grzelak, D. (2023). *Bigraph Ecore Metamodel (BEM): An EMOF-Compliant Specification for*

Bigraphs. https://zenodo.org/records/10043063

ISO/IEC 19508:2014. (2014). *ISO/IEC 19508:2014 Information technology - Object Management Group Meta Object Facility (MOF) Core* [ISO/IEC Standard]. International Organization for Standardization (ISO). https://www.iso.org/standard/61844.html

Jakumeit, E., Blomer, J., & Geiß, R. (2023). *The GrGen.NET User Manual: Refers to GrGen.NET Release 6.7*. https://www.grgen.de/GrGenNET-Manual.pdf

Kehrer, T., Tsigkanos, C., & Ghezzi, C. (2016). An EMOF-Compliant Abstract Syntax for Bigraphs. *Electronic Proceedings in Theoretical Computer Science*, *231*, 16–30. https://doi.org/10.4204/EPTCS.231.2

Miculan, M., & Peressotti, M. (2014). *A CSP implementation of the bigraph embedding problem*. http://arxiv.org/abs/1412.1042

Milner, R. (2005). Embeddings and Contexts for Link Graphs. In H.-J. Kreowski, U. Montanari, F. Orejas, G. Rozenberg, & G. Taentzer (Eds.), *Formal Methods in Software and Systems Modeling: Essays Dedicated to Hartmut Ehrig on the Occasion of His 60th Birthday* (pp. 343–351). Springer. https://doi.org/10.1007/978-3-540-31847-7_20

Milner, R. (2009). *The Space and Motion of Communicating Agents* (1st ed.). Cambridge University Press. ISBN: 978-0-521-73833-0

Sevegnani, M., & Calder, M. (2016). BigraphER: Rewriting and Analysis Engine for Bigraphs. In S. Chaudhuri & A. Farzan (Eds.), *28th International Conference on Computer Aided Verification* (Vol. 9780, pp. 494–501). Springer International Publishing. https://doi.org/10.1007/978-3-319-41540-6_27

Sevegnani, M., Unsworth, C., & Calder, M. (2010). *A SAT based algorithm for the matching problem in bigraphs with sharing* (Technical Report TR-2010-311; p. 27). Department of Computing Science, University of Glasgow.

Steinberg, D., Budinsky, F., & Paternostro, M. (2008). *EMF: Eclipse Modeling Framework* (2nd Revised edition). Addison-Wesley Professional. ISBN: 978-0-321-33188-5

Varro, G., Schurr, A., & Varro, D. (2005). Benchmarking for Graph Transformation. *2005 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC'05)*, 79–88. https://doi.org/10.1109/VLHCC.2005.23