A living monograph for graph transformation

Nicolas Behr, Russ Harmer

A living monograph for graph transformation

Nicolas Behr[1][0000−0002−8738−5040] and Russ Harmer[2][0000−0002−0817−1029]

[1] Université Paris Cité, CNRS, IRIF
8 Place Aurélie Nemours, Paris Cedex 13, 75205, France
nicolas.behr@irif.fr
[2] Université de Lyon, ENS de Lyon, UCBL, CNRS, LIP
46 allée d'Italie, Lyon Cedex 07, 69364, France
russell.harmer@ens-lyon.fr

Abstract. A preliminary account of the notion of a *living monograph* for the field of graph transformation, and the reasons that led us to it, is given. The advantages of such a system are discussed along with the technical problems that will need to be overcome in order to build it.

Keywords: graph transformation · collaborative mathematics · formalised mathematics

1 Towards a living monograph

In this paper, we outline our notion of *living monograph* that we plan to develop in the context of a new French national research agency project[3]. A key aspect of this project is to provide theoretical and technological tools for the representation and expression of diagrammatic reasoning in the proof assistant Coq [9] with the goal of enabling the user to interact with the proof assistant by manipulating graphical depictions of diagrams.

The notion of living monograph for graph transformation presented in this paper is not a specific deliverable of this project but we anticipate that it will follow as a natural consequence to showcase certain aspects of the project: an online, collaborative system for reading and writing the style of mathematics required for graph transformation that is fully integrated with the possibility of an underlying formalisation in Coq. Such a system could even be built in the absence of full success of the project in providing purely graphical interaction with Coq, although the result would clearly be less ergonomic, and the success of the living monograph itself would depend principally on the degree of support and engagement of the graph transformation community.

In the remainder of this introduction, we explain the reasons that led us to this notion. In the subsequent sections, we explain the conceptual advantages of such a system as well as the specific technical hurdles that we will need to overcome in the course of the project. We conclude with some remarks about the more general significance of these ideas beyond the graph transformation community.

[3] CoREACT: COq-based Rewriting: towards Executable Applied Category Theory (ANR-22-CE48-0015).

1.1 Surveys in the graph transformation literature

The field of graph transformation has a rich and extensive research literature which is generally considered to begin with the work of Pfaltz & Rosenfeld [21] and Schneider [23] in the late 1960s. The early community and literature was notably structured by a series of six quadrennial international workshops, the first of which was held in 1978; the first and third proceedings thereof included significant survey material providing up-to-date, comprehensive entry points to the field in 1978 [10] and 1986 [14] respectively.

The publication in the 1990s of a three-volume handbook [22,12,13] marked a significant maturation of the field; its approximately 1800 pages provided a comprehensive account of the theory and applications of graph transformation after some twenty-five years of active development. In hindsight, the appearance of this monumental text seems to identify the period in time where the field had grown sufficiently to support the emergence of more specialised subfields.

Indeed, in 2002, the workshop series mutated into the biennial conference ICGT (which became annual from 2014 onwards) accompanied by various, more specialised, satellite events. It is noteworthy that, other than an introduction to the field from the perspective of software engineering [5] and a short introduction to its application to DNA computing [17], both included in the 2002 proceedings, there have been no further survey articles in subsequent proceedings.

This is perhaps unsurprising, because this modern style of 'conference plus satellite events' precisely emphasises the rapid dissemination of new research over the production of up-to-date surveys, but we argue that such surveys remain important. Clearly, they serve as entry points to a field for young researchers, although tutorial satellite events at least partially fulfil this need; but they can also serve as a means for a community to retain a certain degree of global cohesion in the face of an ever growing, ever more specialised primary literature.

The final surveys, to date, in the field of graph transformation were published in 2006 and 2020 in the form of traditional research monographs. The first [11] provided a full account of the theory of graph transformation, particularly those developments subsequent to the first volume of the handbook [22], as well as an account of the application of this theory to model transformation. The second [18] gave a comprehensive account of the application of graph transformation to a variety of problems in software engineering, including an updated treatment of model transformation.

1.2 Shortcomings of surveys in an active literature

The above discussion already makes it clear that, beyond a certain point of development, the writing of a comprehensive survey of an entire research field requires too much time and effort to be worthwhile: the two final monographs discussed above both have restricted scope—the first provides full theoretical coverage, for its time, but only a limited treatment of applications—and it seems reasonable to expect that any future such manuscripts will similarly target subfields and/or focus exclusively on developments since a previous survey.

A second aspect, which remained implicit in the above discussion, is that any such survey—no matter how long or comprehensive—will inevitably become out-of-date sooner or later: the developments on the theoretical side, in the decade that followed the production of the first volume of the handbook [22], were thoroughly documented in the later monograph [11]; but, at the time of writing, the wide-ranging theoretical developments in the field since the latter's publication have yet to be treated in the form of a monograph. Similarly, much of the more application-oriented material in the second and third volumes of the handbook [12,13] has never been revisited in such a form. Perhaps the very nature of a traditional survey (be it an article, a handbook chapter or a monograph) is actually rather ill-suited to the purpose it purports to address: even for a well-coordinated team of experts, there is an irreconcilable tension between the sheer amount of time it takes to provide comprehensive coverage and the need for the text to remain up-to-date.

A further problem arises in the form of the errors that inevitably creep into a long manuscript. These generally take the form of incorrect, or more often simply incomplete, proofs that escape the proof-reading process. For example, in graph transformation, the complex diagrammatic reasoning increasingly employed can be difficult to convey in an accurate, yet concise, fashion in a traditional text.

This analysis leads us to envisage the possibility of a system that enables the collaborative writing of mathematics—and, in particular, the diagrammatic reasoning prevalent in graph transformation—in a way that intrinsically supports the formalisation of said mathematics in a proof assistant. By these means, we seek to address the shortcomings identified above: by opening the writing of such a *living monograph* to an entire community, the different members of that community can contribute, according to their specialised knowledge, and as such provide better coverage, and far greater reactivity in the event that updates or corrections become necessary, than any small group of experts; and, by enabling access to the benefits of proof assistants without requiring expertise on the part of the user, the bottleneck of technical proof-reading should be reduced with a concomitant improvement in the quality of the resulting monograph.

In effect, we wish to combine the well-known advantages of a wiki-like system, based on distributed collaboration, for writing down mathematics with those of a proof assistant, that aids and informs the writing process, in order to provide an online resource that can be readily maintained up-to-date and rapidly corrected upon discovery of errors or omissions.

2 Technical challenges of a living monograph

The input format of current proof assistants is one-dimensional and text-based. In order to be able to express higher-dimensional notations, such as categorical diagrams, some kind of encoding is therefore required. For our purposes, this raises two initial questions: (i) how to represent an individual diagram and prove desired properties of that diagram; and (ii) how to combine diagrams so as to perform diagrammatic reasoning that is valid in some desired logic?

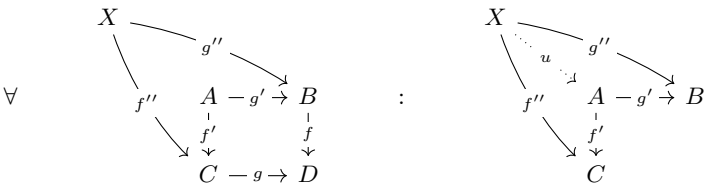
However, these questions still ultimately presuppose a classical interaction with the proof assistant whereby the diagrammatic reasoning, in the head of the user, is only ever 'seen' in encoded form by the system. This allows the proof assistant to verify the correctness and completeness of a proof proposed by the user but does not enable the interactive development of a proof between the user and the machine. Ideally, we would like an augmented notion of interaction with the proof assistant where the user, and potentially the system, can perform diagrammatic reasoning by manipulating graphical depictions of diagrams.

We now examine these questions in greater detail to clarify the technical—and conceptual—obstacles to the construction of a system capable of supporting the diagrammatic reasoning found in graph transformation, e.g. the formalisation of axiomatic settings, such as adhesive categories, the proofs therein of results such as confluence, normalisation or the concurrency theorem and the proofs that concrete settings satisfy those axioms and thus enjoy those results.

2.1 Categorical reasoning in a proof assistant

Diagrams The most natural approach to representing commutative diagrams in a proof assistant consists in encapsulating the objects and arrows of the diagram, together with the required equalities of paths, in a type. Such a representation can subsequently be instantiated into any specific concrete category that we have defined. We can therefore delegate the composition and decomposition, i.e. basic diagram chasing, of such commutative diagrams to the proof assistant.

In order to identify commutative diagrams that satisfy a universal property, we can encapsulate the statement of that universal property together with the commutative diagram into a new type. For example, the property



of *being a pullback square* could be represented in Coq in the following manner:

```
Record Pullback {C: Category} [A B C D: C]
        (g': A~>B) (f: B~>D) (f': A~>C) (g: C~>D) := {
    square_commutes: f∘g' ≈ g∘f';
    pullback_up: ∀ X (g'': X~>B) (f'': X~>C),
        f∘g'' ≈ g∘f'' -> ∃! u: X~>A, f'' ≈ f'∘u ∧ g'' ≈ g'∘u }.
```

We define the *property* of being a pullback square rather than the more usual notion of the *construction* of a pullback span given a starting co-span, since many possible such pullback spans can be constructed. As a result, that latter style of definition is highly inefficient for use with a proof assistant whereas, given the former definition, it would be straightforward to define an operation that constructs a pullback span, given a co-span, using existential quantifiers [4].

This definition is sufficient and convenient for proving typical properties of pullbacks that depend on the manipulation of their universal property, e.g. their uniqueness up to unique isomorphism or the pasting lemma. However, this style of definition forces us to prove uniqueness up to unique isomorphism for every limit (and co-limit) construction individually [4]. An alternative approach is possible, based on defining constructions such as pullbacks directly as (co-)limits, which establishes uniqueness up to unique isomorphism, once and for all, for all such constructions. This has the disadvantage of forcing us to prove equivalence of those definitions with the original ones, but does provide some useful flexibility as different formalisations typically lend themselves to different uses.

Diagrammatic reasoning Much of the basic diagrammatic reasoning used in category theory involves composing and decomposing commutative diagrams in order to propagate properties known to hold in some part of a diagram to another part. For example, the *composition* lemma for pullbacks

$$\begin{array}{ccccc} A & \xrightarrow{h'} & B & \xrightarrow{f'} & E \\ {\scriptstyle g''}\downarrow & PB & {\scriptstyle g'}\downarrow & PB & \downarrow{\scriptstyle g} \\ C & \xrightarrow{h} & D & \xrightarrow{f} & F \end{array} \quad \Longrightarrow \quad \begin{array}{ccc} A & \xrightarrow{f'\circ h'} & E \\ {\scriptstyle g''}\downarrow & PB & \downarrow{\scriptstyle g} \\ C & \xrightarrow{f\circ h} & F \end{array}$$

provides the means to propagate the property of *being a pullback* from the two inner squares to the outer rectangle. Clearly, many other such lemmata exist, e.g. the preservation of monomorphisms by pullback or the stability of final pullback complements under pullbacks, whose proofs—essentially just repeated application of universal properties plus diagram chasing—can be performed in category theory with no additional axioms required.

In a formalisation in a proof assistant, we would clearly not want to inline such proofs in order to justify each successive step in a chain of diagrammatic reasoning. Instead, it would be more natural to formulate these various means of propagating properties as the *reasoning rules* of a basic diagrammatic logic whose soundness would be established, once and for all, by formalising the proof justifying each reasoning rule of the logic. A key early goal of the CoREACT project is precisely to develop such a logical formalism within Coq.

This basic diagrammatic logic would serve as a universal core that could be extended, in many different ways, with additional reasoning rules whose proofs of soundness would rely on additional axioms. For example, in the field of graph transformation, various properties[4] observed to hold in concrete settings—and frequently used in the proofs of important results—have been subsumed into a number of closely-related axiom systems, e.g. adhesive and quasi-adhesive categories [20], $\mathcal{M}$-adhesive categories [11] and rm-quasi-adhesive categories [16] among many others. A key second goal of the CoREACT project is to provide the means to define such extensions of the core logical formalism in Coq so as to enable fully-fledged diagrammatic reasoning for graph transformation as found in [6,11,20] and many others.

[4] e.g. the preservation of monomorphisms under pushout or the stability of pushouts under pullbacks

An interesting aspect of this methodology is that one could formalise such a logic in a proof assistant without necessarily formally proving the soundness of its rules. From this perspective, a collection of reasoning rules could be viewed as an autonomous layer that could, but need not, be connected to an underlying formalisation of category theory (plus some optional axiom system). Of course, such an approach might not make any sense—in the case that the collection of reasoning rules is badly chosen—but, if done sensibly, would provide a *light-weight* approach to formalisation that might be highly appropriate for certain purposes: a user wishing to prove some complicated theorems using a well-chosen collection of reasoning rules would probably be happy to accept them as given to focus exclusively on formalising the higher-level logic of those theorems.

More generally, such a separation of concerns seems to be a pragmatic course of action. On the one hand, the higher-level logic enables the succinct proof of theorems *about* graph transformation; on the other hand, the lower-level logic that determines reasonable collections of reasoning rules, on the basis of the various axiom systems proposed in the literature, has more of a *meta-theoretic* character with the purpose of analysing the relative expressive power of different axiom systems.

Moreover, our recent work on compositional rewriting theories [7] suggests that this separation of concerns could be taken further still. The fibrational framework introduced in that work provides higher-level reasoning *macros* that intrinsically induce and structure entire collections of the lower-level reasoning rules. In the same way that those rules enable the user to focus on higher-level logic, rather than repeated application of low-level universal properties, these reasoning macros seem to enable a further level of succinctness that remains to be investigated.

2.2 Categorical reasoning with a proof assistant

The discussion so far has focussed on questions concerning the representation of diagrams, and of diagrammatic reasoning, in a proof assistant such as Coq. Assuming an adequate resolution of the various technical difficulties identified, this would still require all formalised content to be expressed in the language of the proof assistant. However, in order for a living monograph system to provide a comfortable and ergonomic interface to the proof assistant for the non-expert, it should ideally allow for diagrammatic input and the expression of diagrammatic reasoning in that interface. It would then be the responsibility of the system to translate the input to, and other events in, the interface to the one-dimensional input format of the proof assistant itself.

These aspects of our proposed living monograph are currently dependent on certain tasks of the CoREACT project, but we do not exclude alternative future approaches. The idea is to extend the Coq document model with a *diagram* object of which the jsCoq framework [3] could handle the visualisation and, in concert with the YADE commutative diagram editor, provide a web-based interface to create and manipulate commutative diagrams and generate their underlying Coq representations.

This is a complicated but—we believe—realistic integration task that would provide the first prototype of a system capable of creating a wiki-like document, whose text, diagrams and other mathematical content are provided by users, connected to an underlying formalisation in Coq.

The CoREACT project also seeks to incorporate diagrammatic reasoning into the interface with the aim of providing assistance to the user in *developing* proofs, not merely formalising proofs already worked out on paper. This has two principal aspects: a database of combinations of reasoning steps found in already-formalised proofs for the system to query in order to *suggest* pertinent next steps in a proof; and the use of gestures, in the general sense of proof-by-pointing [8], to enable the user to invoke diagrammatic reasoning steps.

The first of these aspects aims to provide means to ease the cognitive burden of reasoning with the large and complicated diagrams that occur in proofs in graph transformation, e.g. simply identifying that certain squares within such a diagram are pullbacks could suggest plausible next steps to the user—and, in the longer term, enable the system to make pertinent suggestions. The second aspect is already under active investigation in the context of the Actema system, that currently provides a graphical interface for the manipulation of one-dimensional Coq syntax, which will also play a key role in the CoREACT project.

3 Advantages of a living monograph

Our initial motivation to develop this project grew out of the observations made in section 1 concerning the shortcomings of a traditional research monograph in a large, but still dynamic, area of research: the intrinsic tension between providing broad coverage, of maintaining the text up to date and of providing some assurance of the completeness, and correctness, of the results presented.

We hope to have convinced the reader that our notion of living monograph, as outlined in this paper, does address these issues: the use of wiki-like systems for providing broad coverage and fast reactivity to required updates is well known. Coupled with the benefits of a proof assistant, once enabled with ergonomic means to create and manipulate commutative diagrams and perform associated reasoning, the resulting system would indeed constitute a highly novel kind of on-line mathematical editor. We anticipate that this would not only aid the writing process but also assist the writer in the very development of the mathematics.

Although not part of our original motivation, we might also envisage certain advantages of a living monograph for the reader. Beyond the obvious convenience of hyperlinking, to navigate easily within the text, let us note, in particular, the possible application of the Coq knowledge graph. Given a Coq document, this details the dependencies between its different definitions and proofs. This could be a powerful tool for understanding as it would allow the reader, at any moment, to inspect the provenance of a particular result in order to understand better the overall structure of the mathematical theory described in the text. In effect, it would provide a finer-grained and on-demand version of the chapter and section dependencies often given at the start of research monographs and textbooks.

Indeed, let us emphasise that this use of the knowledge graph is representative of how we imagine the living monograph would make use of its underlying proof assistant. The aim would be to exploit and display the insights that the assistant can bring, and which would be practically inaccessible without it, in order to augment the typical content of a traditional monograph. In particular, we would not—at least by default—wish to display the formal proofs of results as part of the standard reading experience; instead, the reader should be able to click in order to access the formalised proof should they wish to do so—but we imagine that the simple existence of the underlying formalisation would suffice for the majority of readers.

Let us conclude by noting some more general potential consequences of the notion of living monograph. If a research community succeeds in organising itself around a comprehensive living monograph, this would obviously provide clear entry points, for the newcomer, into the primary literature. However, it could also give rise to a new mode of attribution, finer-grained than the traditional scientific paper and previously typically restricted to lecture notes or theses, whereby small, but significant, improvements to definitions or results could be acknowledged by their incorporation into the living monograph—as a kind of *micro-publication*. In a similar vein, the existence of the living monograph would hopefully encourage the formalisation of existing results, major or minor, as a legitimate and attributed activity in and of itself.

At a time when the traditional publication model of academia is increasingly under question, the ability to acknowledge such work—generally ignored by the traditional model—might help to advance further the debate on how a scientist's contribution to their field might best be identified. It is to be hoped that it would at least provide a more nuanced picture than does an H-index. Finally, let us note that the notion of living monograph is clearly not restricted to the field of graph transformation or to categorical reasoning per se. The basic concept could apply to many fields of mathematics and perhaps foster better communication between neighbouring fields by enabling hyperlinking *between* living monographs.

Related work

We are aware of two existing lines of work on the formalisation of certain aspects of graph transformation. The first, initiated by Strecker and summarised in [25], uses the Isabelle proof assistant to formalise certain concrete notions of graphs, homomorphisms and rewriting steps in order to provide formal proofs of the preservation of certain properties under transformation. More recently, Söldner and Plump have formalised, again in Isabelle, the equivalence of the operational definition of direct derivations (by deletion and glueing operations) with that of a double-pushout diagram [24] for a certain class of concrete graphs and their homomorphisms.

The notion of micro-publication outlined above bears some similarity to the idea of the Archive of Formal Proofs although the content in the AFP is typically more substantial in size and scope in that it typically presents entire libraries or non-trivial results rather than individual lemmata.

The idea of a *living review* [26], i.e. a survey article that is *required* to be maintained up-to-date as a condition of publication, was pioneered in the late 1990s by the Max Planck Institute for Gravitational Physics in the form of its *Living Reviews in Relativity*. In 2015, this initiative became a Springer journal—retaining the requirement of maintenance—and has been joined by similar efforts in solar physics and computational astrophysics; the idea also reached biology in 2014 [15]. Of course, these notions of living review do not have, or even need, the possibility of an underlying formalisation, but they do strongly illustrate the need for constant integration of new results in dynamic research fields.

The seminal initiative in mathematics to build a comprehensive formalised library of knowledge is clearly the QED project, dating from the 1990s, which set out to "build a computer system that effectively represents all important mathematical knowledge and techniques" [1]. Despite an active mailing list and two international workshops (held in 1994 and 1995), QED never proceeded beyond the stage of initial plans; the reasons for this have been discussed elsewhere—see, for example, [19,27]—but seem mainly to have stemmed from the difficulty either of choosing a single logic upon which to found everything or, alternatively, of providing for the inter-operability of different logics.

While it seems unlikely that a consensus will ever be reached on the first question, considerable progress has subsequently been made on the second—see, for example, [2,19] and the MMT system—and the success of the Twenty Years of the QED Manifesto workshop, held as part of the 2014 Vienna Summer of Logic, proves that the original vision remains intact. Indeed, many other projects now exist such as Kerodon and Stacks or, with a more general intention, Xena or ForMath. Although the notion of living monograph has different, albeit overlapping, motivations to these various approaches, we hope that it will fulfil a useful rôle within the larger ecosystem of formalised mathematics.

References

1. The QED manifesto. In: Bundy, A. (ed.) Automated Deduction – CADE-12. CADE 1994. LNCS, vol. 814, pp. 238–251. Springer, Berlin, Heidelberg (1994)
2. Alama, J., Brink, K., Mamane, L., Urban, J.: Large formal wikis: Issues and solutions. In: Davenport, J., Farmer, W., Urban, J., Rabe, F. (eds.) Intelligent Computer Mathematics. CICM 2011. LNCS, vol. 6824, pp. 133–148. Springer, Berlin, Heidelberg (2011)
3. Arias, E.J.G., Pin, B., Jouvelot, P.: jsCoq: Towards hybrid theorem proving interfaces. EPTCS **239**, 15–27 (2017), Proceedings of UITP 2016
4. Arsac, S.: Coq formalization of graph transformation. Master's thesis, Université Paris-Cité (2022), https://www.samuelarsac.fr/rapport22.pdf
5. Baresi, L., Heckel, R.: Tutorial introduction to graph transformation: a software engineering perspective. In: Corradini, A., Ehrig, H., Kreowski, H., Rozenberg, G. (eds.) Graph Transformation. ICGT 2002. LNCS, vol. 2505, pp. 402–429. Springer, Berlin, Heidelberg (2002)
6. Behr, N., Harmer, R., Krivine, J.: Concurrency theorems for non-linear rewriting theories. In: Gadducci, F., Kehrer, T. (eds.) Graph Transformation. ICGT 2021. LNCS, vol. 12741, pp. 3–21. Springer, Cham (2021)

7. Behr, N., Harmer, R., Krivine, J.: Fundamentals of compositional rewriting theory. Journal of Logical and Algebraic Methods in Programming **135**, 100893 (2023)
8. Bertot, Y., Kahn, G., Théry, L.: Proof by pointing. In: Hagiya, M., Mitchell, J. (eds.) Theoretical Aspects of Computer Software. TACS 1994. LNCS, vol. 789, pp. 141–160. Springer, Berlin, Heidelberg (1994)
9. Bertot, Y., Castéran, P.: Interactive theorem proving and program development. Texts in Theoretical Computer Science. An EATCS Series, Springer, Berlin, Heidelberg (2004)
10. Claus, V., Ehrig, H., Rozenberg, G.: Graph-grammars and their application to computer science and biology. LNCS, vol. 73. Springer, Berlin, Heidelberg (1979)
11. Ehrig, H., Ehrig, K., Prange, U., Taentzer, G.: Fundamentals of Algebraic Graph Transformation. Texts in Theoretical Computer Science. An EATCS Series, Springer, Berlin, Heidelberg (2006)
12. Ehrig, H., Engels, G., Kreowski, H.J., Rozenberg, G. (eds.): Handbook of Graph Grammars and Computing by Graph Transformation, vol. 2: Applications, Languages and Tools. World Scientific, Singapore (1999)
13. Ehrig, H., Kreowski, H.J., Montanari, U., Rozenberg, G. (eds.): Handbook of Graph Grammars and Computing by Graph Transformation, vol. 3: Concurrency, Parallelism, and Distribution. World Scientific, Singapore (1999)
14. Ehrig, H., Nagl, M., Rozenberg, G., Rosenfeld, A.: Graph-grammars and their application to computer science. LNCS, vol. 291. Springer, Berlin, Heidelberg (1987)
15. Elliott, J., Turner, T., Clavisi, O., Thomas, J., Higgins, J., Mavergames, C., Gruen, R.: Living systematic reviews: an emerging opportunity to narrow the evidence-practice gap. PLoS Medicine **11**(2), e1001603 (2014)
16. Garner, R., Lack, S.: On the axioms for adhesive and quasiadhesive categories. Theory and Applications of Categories **27**(3), 27–46 (2012)
17. Harju, T., Petre, I., Rozenberg, G.: Tutorial on DNA computing and graph transformation. In: Corradini, A., Ehrig, H., Kreowski, H., Rozenberg, G. (eds.) Graph Transformation. ICGT 2002. LNCS, vol. 2505, pp. 430–434. Springer, Berlin, Heidelberg (2002)
18. Heckel, R., Taentzer, G.: Graph Transformation for Software Engineers. Springer, Cham (2020)
19. Kohlhase, M., Rabe, F.: QED reloaded: Towards a pluralistic formal library of mathematical knowledge. Journal of Formalized Reasoning **9**(1), 201–234 (2016)
20. Lack, S., Sobociński, P.: Adhesive and quasiadhesive categories. RAIRO – Theoretical Informatics and Applications **39**(3), 511–545 (2005)
21. Pfaltz, J.L., Rosenfeld, A.: Web grammars. In: Proceedings of the 1st International Joint Conference on Artificial intelligence. pp. 609–619 (1969)
22. Rozenberg, G. (ed.): Handbook of Graph Grammars and Computing by Graph Transformation, vol. 1: Foundations. World Scientific, Singapore (1997)
23. Schneider, H.J.: Chomsky-languages for multidimensional input-media. In: Proceedings of the International Computing Symposium. pp. 599–608 (1970)
24. Söldner, R., Plump, D.: Towards mechanised proofs in double-pushout graph transformation. EPTCS **374**, 59–75 (2022), Proceedings of GCM 2022
25. Strecker, M.: Interactive and automated proofs for graph transformations. Mathematical Structures in Computer Science **28**(8), 1333–1362 (2018)
26. Wheary, J., Schutz, B.: Living reviews in relativity: Making an electronic journal live. Journal of Electronic Publishing **3**(1) (1997)
27. Wiedijk, F.: The QED manifesto revisited. Studies in Logic, Grammar and Rhetoric **10**(23), 121–133 (2007)