AGTIVE 2007 Graph Transformation Tool Contest

Arend Rensink[1] and Gabriele Taentzer[2]

[1] University of Twente, The Netherlands
rensink@cs.utwente.nl
[2] Philipps-Universität Marburg, Germany
taentzer@mathematik.uni-marburg.de

Abstract. In this short paper we describe the setup and results of a new initiative to compare graph transformation tools, carried out as part of the AGTIVE 2007 symposium on "Applications of Graph Transformation with Industrial Relevance". The initiative took the form of a contest, consisting of two rounds: the first round was a call for cases, the second round a call for solutions. The response to both rounds was very good, leading to the conclusion that this is an initiative worth repeating. There are, however, a number of lessons to be learned; these are summarised here, in order to improve the organisation and the eventual benefits of this type of contest.

1 Introduction

Tools are crucial for the promotion of graph transformation in industry. It is only with the ready availability of reliable, easy-to-use tools that the attractions and benefits of graph transformation can ever become clear to anyone not having a prior education in this field. Furthermore, given the inherent complexities of the method, tool performance is an important issue. As a community we should be constantly working to improve tool support in all these aspects.

A variety of tool environments exists, supporting different graph transformation approaches and to some degree serving different purposes. There are some examples of tool comparisons, e.g., [5, 3, 2]; furthermore, Varro et al. [9] propose some benchmarks to be used for such purposes. Nevertheless, having a certain application in mind, it is difficult for newcomers to decide the right graph transformation tool to use. Moreover, even for most of the tool experts it is true that they know much about one or two tools but little about the others.

To stimulate both the continued improvement of tools and the wider dissemination of knowledge about existing tools, we have organised a *tool contest* as part of the AGTIVE 2007 symposium. The aim of this event was to compare the expressiveness, the usability and the performance of graph transformation tools, along a number of selected case studies. The desired outcome was twofold:

- To learn about the pros and cons of each tool considering different applications. A deeper understanding of the relative merits of different tool features will help to further improve graph transformation tools and to indicate open problems.
- To instill a sense of challenge and competition that will motivate tool developers to continue their efforts. There is nothing like seeing, and being inspired by, the features supported by other tools to stimulate progress in one's own development.

The case studies were selected from the entries received after an open call for cases, which was distributed among the graph transformation tool providers. The call and selection procedure are outlined in Sect. 2. This was followed by a call for solutions, distributed more widely through the usual channels for calls for papers. Since the response exceeded our expectations, it was not possible to demonstrate all solutions at the contest session, as we had originally planned. The setup eventually chosen for this session is described and evaluated in Sect. 3. In Sect. 4 we draw conclusions from the experience gained in this way, and we give recommendations for next instances of the contest.

As a final word, let us repeat the motto that we stated on the call for solutions. Tool improvement is what we seek, and this contest was a means of achieving it. To paraphrase a famous saying (see [7]):

't Is better to have competed and lost, then never to have competed at all.

2 Call for Cases

Although the idea for the tool contest has arisen at ICGT 2006, in September 2006, it took some time to put it into practice. The call for cases (constituting, in fact, the first announcement of the tool contest) was issued only three months in advance of the event, and contained a deadline of a mere two weeks for case descriptions. The call was very broad, merely asking for case studies of any kind, from which a "small but representative" subset was to be selected.

The response far exceeded our expectations: we received 13 case descriptions, of varying size and amount of detail. Since we wanted to select at most 3, we had to set up a ranking system. The following criteria were used:

Nature. What is being modelled; in other words, what is the application area? Among the cases received, the application areas were: real-world systems (a game and a biological system, respectively), semantics, model transformations of various kinds, and algorithms.

Size. What is the expected size of the solution? This can be expressed in terms of the expected number of rules (order of magnitude), the expected complexity of the meta-model and the rules. In general, the best measure is the effort to create a correct solution.

Challenge. What is (or are) the core problem(s) in solving the case? Some of the more specific challenges identified were: showing confluence and termination, verifying correctness, offering sufficient (space and time) performance, allowing human interaction. For most of the cases, however, the main challenge was to come up with a "good" model – where, of course, it is not at all easy to define the "quality" of a model in the first place.

Detail. What is the detail of the case description? Some of the submissions were quite detailed, clearly constituting cases that had already been carried out by the submitters — which is in fact something we very much encouraged in our communications, believing it to be the only way to ensure fast response. Clearly, a fair amount of detail is an advantage in a case description, in particular if this includes a good indication of the challenges (see above). On the other hand, a too precise description of the case runs the danger of leaving too little room for different solutions and creativity. In particular, a case description should describe *what* is to be done, and not *how*.

On all these dimensions, the submissions received were quite diverse. In the end we identified a partitioning from which we selected three representative cases:

Category. On a certain level of abstraction, the following three categories could be distinguished (as a mixture of the nature and the challenge of the case):

1. *General graph transformation cases.* These are real world applications, as well as algorithms, for which the main challenge lies in actually providing a model.
2. *Model transformation cases.* This is a very important application area on its own, to which much tool development has been devoted.
3. *Performance cases.* These are cases inspired by algorithms and decision problems in which the speed of transformation and/or memory consumption are the main challenges.

In each of these categories there were case submissions with a sufficient amount of detail to be usable. In the end we made the following selection:

1. *Ludo game*. This case was actually submitted by two teams, in slightly different form. It involved modelling a (fairly simple) board game, i.e., a real-world application. The challenges of this game are in modelling, visualisation and (human) interaction, and to a minor degree in analysis. This case is reported in [4].
2. *UML-to-CSP model transformation.* This is a non-trivial model transformation case, which had in fact already been studied before (see [1]). The challenges are the ease of definition and understandability of the rules, as well as the ability to read and write models in common formats. This case is reported in [8].
3. *Sierpinski triangles.* This case involves the fast and efficient generation of very large graphs, based on a simple transformation. Challenges are time and memory performance. This case is reported in [6].

It should be noted that none of these categories addresses analysis or verification issues, and indeed this was perceived as an omission in the contest. Similarly, the chosen case studies have little or no need for backtracking — which is an important element when modelling NP-complete problems using graph grammars. We will come back to this in Sect. 4.

3 Solutions

The call. The call for solutions was issued just before the summer holidays, two-and-a-half months before the workshop. The deadline for submissions was one month before the event, with notification promised ten days later. The danger with such tight deadlines obviously is that attendance may depend upon acceptance, and so late notifications can give rise to late registrations.

In the call for solutions, we merely asked that submissions should

- Contain a description of the chosen case study variant (if any);
- Present the chosen solution, including a discussion of design decisions.

As a guideline this is rather weak. In response to queries we added that submissions should

- Not exceed 5 pages in length;
- Include enough information so that readers should be able to reconstruct the solution.

As with the call for cases, the response was very good: we received 30 solutions altogether, reasonably well divided over the cases:

Case	**Solutions**
Ludo	8
UML-to-CSP	11
Sierpinski triangles	13

Given the absence of strict guidelines, submissions were quite diverse and, as a consequence, hard to judge and compare. For that reason we accepted all of them to the contest session. In turn, this meant we could not allow all solutions to be demonstrated during the workshop, as we had originally planned: not only would the available time be too short, but also an over-long demonstration session would not be attractive.

The event. Because of the relatively large number of submissions, the actual contest session was held in two stages, the first of which was split in three parallel meetings, one for each case. During these case meetings, a small number of submissions were demonstrated; these were selected by the organisers on the basis of the diversity of the approaches involved. The demonstrations were followed by a discussion on the aspects identified in the case, and the various solutions offered for those aspects by the different tools.

In the second (plenary) stage, after a brief report on the case meetings, a more global discussion took place on the setup and principles of the contest, the manner in which the outcome was to be published, suggestions for case studies and recommendations for future editions. For inspiration to future organisers, we include the outcome of this discussion in the form of a list of suggestions.

- Distinguish between the transformation language and the transformation tool. Criteria for the language are (among others): naturality for the domain, ease of modelling. Criteria for the tool are (among others): efficiency, usability.
- Include test suites in the case description.
- Ideas for types of case studies:
  - Large matches in irregular graphs (e.g., compiler construction problems). The main challenge is performance.
  - Refactoring. An important challenge in this context if *copying* graphs (in particular trees).
  - Comparison of different matching strategies, along the lines of the benchmarks provided in [9].

  - Algorithmic problems with well-known solutions, involving backtracking
  - Complex text-to-model and model-to-text transformations. A challenge is the flexibility and adaptability of the tools for this (important, and often ignored) type of transformation.
  - Cases involving formal analysis/correctness proofs. Challenges are the power to address certain types of problems, and the performance in producing results.
- Ideas for organisation:
  - Throw a bunch of students at a problem, using different tools
  - Present all solutions and let the audience judge them, using a questionnaire
  - Set up a central server for a fair comparison of execution performance
  - Produce solutions under time constraints
- Set up a repository of case studies and solutions. (This has in the meanwhile been put into practice: see `gtcases.cs.utwente.nl`.)

4 Evaluation and Recommendations

In the following, we draw some conclusions from the past tool contest and give recommendations for the next contest round.

4.1 Evaluation

Strong points. The response to this tool contest far exceeded our expectations. It shows us that the time is ripe to initiate competition for graph transformation tools. Each part of this contest was borne by a remarkable enthusiasm of the participants. In the run-up of the tool contest, there was already remarkable stimulus for further tool development. The positive experience with this tool contest leads to enough excitement for a next edition of such a contest.

Weak points. However, the first round of this tool contest had less contest character, since we did it without any ranking. Although general challenges have been identified, they were not precisely given and could not be used to judge and to rank. Furthermore, we did not have enough time for everyone to demonstrate their solution. The tool contest could have been a workshop by its own. The time restriction led to a shift of discussions among tool builders into the preparation phase of papers [4, 6, 8] which report on the tool contest in detail.

4.2 Recommendations

For the next round of tool contests we like to give some recommendations. For truly creating a contest, a ranking should be possible. This starts with the identification of case categories and the submission of cases within these categories. Besides categories mentioned in Section 2, further categories are needed to cover all kinds of challenges for graph transformations tools. For example, a category "NP-complete problem" could be an interesting new category to test the efficiency of rule matching. Furthermore, verification issues should be covered by case studies.

All challenges should be included in case descriptions. They need to be formulated precisely enough to allow comparison. Performed experiments should be repeatable by outsiders. This requirement includes a detailed experiment description mentioning all tool specialities used. In general, the comparability of solutions has to be increased such that a ranking system can be set up.

Assuming the enthusiasm for graph transformation tool contests will hold on, the next contests should be organised as some kind of workshop which offer enough time for demonstrating all solutions, including live-demos of experiments. Since tool contests are an important incitement for tool improvements, deadlines should be less tight. Continuous comparisons and improvements of tools should be possible and supported, to keep the lively contest going on. A wiki for this purpose has been set up at `gtcases.cs.utwente.nl`

References

[1] Engels, G., Heckel, R., Küster, J.M.: The Consistency Workbench: A tool for consistency management in UML-based development. In: Stevens, P., Whittle, J., Booch, G. (eds.) UML 2003. LNCS, vol. 2863, pp. 356–359. Springer, Heidelberg (2003)

[2] Fuß, C., Mosler, C., Ranger, U., Schultchen, E.: The jury is still out: A comparison of AGG, Fujaba, and PROGRES. In: Graph Transformation and Visual Modeling Techniques (GT-VMT). Electronic Communications of the EASST, vol. 6 (2007)

[3] Geiß, R., Batz, G.V., Grund, D., Hack, S., Szalkowski, A.: GrGen: A fast SPO-based graph rewriting tool. In: Corradini, A., Ehrig, H., Montanari, U., Ribeiro, L., Rozenberg, G. (eds.) ICGT 2006. LNCS, vol. 4178, pp. 383–397. Springer, Heidelberg (2006)

[4] Rensink, A., Dotor, A., Ermel, C., Jurack, S., Kniemeyer, O., de Lara, J., Maier, S., Staijen, T., Zündorf, A.: Ludo: A case study for graph transformation tools. In: Schürr, A., Nagl, M., Zündorf, A. (eds.) AGTIVE 2007. LNCS, vol. 5088. Springer, Heidelberg (2008)

[5] Rensink, A., Schmidt, V.D.: Model checking graph transformations: A comparison of two approaches. In: Ehrig, H., Engels, G., Parisi-Presicce, F., Rozenberg, G. (eds.) ICGT 2004. LNCS, vol. 3256, pp. 226–241. Springer, Heidelberg (2004)

[6] Taentzer, G., Biermann, E., Bisztray, D., Bohnet, B., Boneva, I., Boronat, A., Geiß, R., Horvath, Á., Kniemeyer, O., Mens, T., Ness, B., Plump, D., Vajk, T.: Generation of Sierpinski triangles: A case study for graph transformation tools. In: Schürr, A., Nagl, M., Zündorf, A. (eds.) AGTIVE 2007. LNCS, vol. 5088. Springer, Heidelberg (2008)

[7] Tennyson, A.: In memoriam (1850)

[8] Varró, D., Asztalos, M., Bisztray, D., Boronat, A., Dang, D.H., Geiß, R., Greenyer, J., Van Gorp, P., Kniemeyer, O., Narayanan, A., Rencis, E., Weinell, E.: Transforming UML models to CSP: A case study for graph transformation tools. In: Schürr, A., Nagl, M., Zündorf, A. (eds.) AGTIVE 2007. LNCS, vol. 5088. Springer, Heidelberg (2008)

[9] Varró, G., Schürr, A., Varró, D.: Benchmarking for graph transformation. In: IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC), pp. 79–88. IEEE Computer Society, Los Alamitos (2005)