Bi-directional transformations (bx) – Theory and Applications Across Disciplines

Jeremy Gibbons (University of Oxford, UK),
Richard F. Paige (University of York, UK),
Andy Schürr (Technische Universität Darmstadt, Germany),
James F. Terwilliger (Microsoft Research, USA),
Jens Weber (University of Victoria, Canada)

December 1, 2013–December 6, 2013

1 Overview of the Field

Bidirectional transformations *(bx)* are mathematics-based formalisms used in applied computer science for maintaining the consistency of two (or more) related data structures [4]. Researchers from many different areas including software engineering (SE), programming languages (PL), databases (DB), and graph transformation (GT) are actively investigating the use of bx to solve a diverse set of problems. Moreover, new fields in applied computer science are discovering bx as a potential solution to current and ongoing challenges. The mathematics of bx are of significant interest to many researchers and developers in computer science. Applications of this mathematics are of great value to practitioners who need to manage large datasets. Problems of interest to researchers in bx include:

- calculation and synchronization of different views of software models in software development
- consistency maintenance of a graphical user interface and its underlying application model in user interface design
- visualization and animation of data and algorithms that operate with that data
- construction of updatable views of databases to integrate data from different sources

While interest in bx pervades different disciplines of mathematics and computer science, there is no commonly agreed *vocabulary* for talking about bx, no commonly held conceptual models for explaining bx principles and practices, no standardised tools for experimenting with bx scenarios, and no benchmarks that will allow researchers to compare theories and solutions scientifically.

The bx community – drawn loosely and inclusively from researchers in the SE, PL, DB and GT areas mentioned above – has been making progress on these issues, firstly by organising a series of technical workshops at relevant conferences, but also by organising week-long intensive research seminars. Past seminars were held at the GRACE International Meeting in Shonan, Japan in 2008 [10] and at Schloss Dagstuhl, Germany in 2011 [5]. Each seminar contributed new understanding and better structure, as well as advances to the technical and theoretical underpinnings, of the research field.

2 Recent Developments and Open Problems

The state of the art in bx can be summarised as: "a great deal of activity, with an emerging conceptual model", and with several coordinating efforts being made – in the form of regular scientific workshops and seminars – to connect and identify synergies between these activities. Some of the most recent developments can be categorised as "bottom-up" activities (i.e., focusing on concrete problems, both theoretical and practical, to which bx can be applied) and "top-down" (i.e., conceptual activities, classifying bx problems, or identifying a broadly applicable theory for bx). Substantial contributions have also been made in terms of building and exploiting usable tools for bx (of note are eMOFLON [1], HenshinTGG [11], GROOVE [8], SDMLib [15], Echo [14], USE [9], GroundTRam [12], Eclipse QVT [6] and Epsilon [13]). Mathematical theories for bx have been developed that address certain bx scenarios (of note are Stevens's game theoretical semantics [17], triple graph grammars [16], updateable views [2] and lenses [7]). Examples and case studies for bx have also been identified (e.g., health informatics case studies involving database/application co-evolution, document synchronisation [3]). The open problems that still face the community include:

- the absence of a common vocabulary for bx: each research discipline still has its own vocabulary for the key concepts of bx (e.g., what does a bx operate on, how is a bx defined, what are the constituent components of a bx?)
- limited benchmarks for comparison of partial or complete bx solutions;
- no common repository of bx scenarios or challenge problems that can be used to test and evaluate bx solutions;
- no clear conceptual and mathematical basis for bx that crosses all disciplines, providing a unified perspective;
- tools to support particular bx scenarios but with very limited interoperability and integration, both in terms of being able to share problem definitions, but also in terms of the mathematical formalisms supported, the operators provided, and the mechanisms used to execute bx scenarios.

3 Presentation Highlights

A number of highlights of the workshop presented themselves across the week, both conceptual and technical. We briefly summarise some of the key observations in this section.

3.1 Common conceptual framework

The workshop attempted to provide introductions to each research community's current views on the state-of-the-art in bx; this was presented concisely and rapidly via a number of mini-tutorials on the first day. One of the most interesting observations from these mini-tutorials (on PL, GT, SE and DB approaches to bx) is the amount of overlap in the conceptual terminology of bx: most communities did use a notion of *lens* to explain the key ideas behind bx. A lens, in PL terms, is a program that, when read from left to right, describes a function that maps primary to secondary data. When the same program is read from right to left, it describes a "backwards" function that maps a modified output, together with the original input, back to a modified input. In this sense, changes can be propagated in different directions. This concept appeared in different guises in each mini-tutorial, and provided the start of a common basis for discussions of collaboration and synergy over the rest of the week.

3.2 Challenge Problems

What makes bx challenging in a particular domain or research field? This was a question asked of all attendees before their arrival and was a theme for discussion throughout the week. Participants were asked to try to identify conceptual or theoretical challenges, practical challenges associated with using bx, and tool/technology challenges involved with building software systems that supported or exploited bx. Several challenge problems were noted and detailed during the week:

- One problem involved using bx to support a scenario where semantics-preservation between data structures was not required: the target of the bx included information that was propagated to the source, though the propagation itself was not a direct mapping: concepts in the target language had no direct equivalent in the source language. A concrete example of this, which was presented at the workshop, is the situation where a structural or behavioural model of a system is forward-transformed into a mathematical representation. Analysis is carried out on the mathematical representation – for example, a property is checked against the representation – and a counter-example identified. This counter-example is then backwards-transformed into the structural or behavioural source model. There was some discussion amongst the participants as to whether this was strictly a bx problem, or whether bx was best suited to semantics-preserving transformations.
- A related problem that was discussed involved *synchronisation* between two different documents: consistency relationships were defined on “fragments” of the documents (e.g., capturing numerical properties as well as structural properties), and bx could be defined to propagate changes. An engineering variant of the problem was also discussed wherein documents were in different formats (e.g., XML and PDF). A challenge with this scenario was in identifying the fragments of the artefacts to be transformed, and to apply a bx to those fragments without disrupting the consistency of other, related fragments.
- Several participants presented non-bx transformation tools – particularly, SDMlib, GROOVE and USE – which could nevertheless be used to support aspects of bx. The challenge with such applications was reconciling the theory of bx with that supported by the tools. An interesting observation with some of the proposed ‘unidirectional’ approaches to bx was that intermediate structures (e.g., intermediate graphs, intermediate models) could provide a clear mathematical and technical mechanism for using unidirectional approaches to support bx.
- A challenging bx problem involving structural and application co-evolution was presented; in such scenarios, both the structures (schemas) that constrain data, and the application (software) that uses the structure and data must be evolved simultaneously in order to maintain consistency. The use of *channels* to support decoupling of structures/data/applications was advocated as a possible mechanism to support this. Channels appear to have some relationship to the intermediate models/structures discussed in the previous problem [18].

3.3 Mathematical representations

There was substantial interest and discussions during the week about mathematical representations of both bx and additional structures (i.e., not just the artefacts for which consistency is to be established) associated with bx problems. There was much discussion on different kinds of *lenses*, including symmetric lenses (which can be associated with problems of model synchronisation) and asymmetric lenses (where one of the two artefacts involved in the bx is taken as primary). Discussions on mathematical formulations of lens structures (including providing mechanisms that capture lens equivalence) and multiplicity of lens structures was a running discussion.

There was also substantial interest in mathematical formulations of “least change” in the context of bx, and how type theory could be used to unify extensional approaches to bx.

3.4 Benchmarks

Benchmarks for bx were a recurring topic in both formal and informal discussions and presentations. Consideration of questions like – What categories of bx benchmarks exist? How do we distinguish a bx example from a benchmark? What are bx benchmarks expected to measure? How should bx benchmarks be expressed? – were considered.

An initial categorisation of benchmarks was suggested: separating benchmarks into functional ones (e.g., focusing on properties of consistency relations, types of bx transformation scenarios, laws for bx approaches) and non-functional ones (e.g., scalability with respect to size of primary artefact, space/time complexity, size of propagated deltas) were also proposed.

A precise definition of a bx benchmark was also proposed: a *bx benchmark* is a *bx example* that has a precise and executable definition of a binary consistency relation on source and target artefacts; an explicit definition of, or a generator for, input artefact elements; a set of precisely defined update scenarios for certain input artefact elements; and a set of executable metric definitions. It was also suggested that a bx benchmark should capture the essence of certain aspects of a class of real-world bx application scenarios.

3.5 Scenarios

A number of scenarios for bx were developed through brainstorming activities. Six important scenarios were identified (described below). Some effort was also made towards expressing these scenarios using lenses and suitable operators. The scenarios were:

- *Database view update:* where a view is generated from a source database, the source database is then updated, and the view must also be updated to remain consistent with the updated database. The challenges associated with different kinds of database update (specifically, that some updates are more difficult/expensive to manage than others) was considered.
- *Synchronisation of artefacts:* where two artefacts with consistency relationships defined between them would be synchronised (via a bx) at specific points in time. An interesting point of variation between different concrete instances of this scenario is *when* synchronisation via a bx takes place: must consistency always hold? Must it hold when one or both artefacts are available to be used by external actors? Two variants of this scenario were discussed: a model-text scenario where one artefact is text-based (e.g., a document, program), and a second where both artefacts are models (with graph-based structures).
- *Artefact co-evolution:* where a definition-instance consistency relationship exists between two artefacts (which differs from the previous scenario where both artefacts are instances). While some similarities between this scenario and the previous one were discussed, differences were identified too, particularly with variants. Two variants were identified: program-data coevolution, where a program makes use of data (via a database schema), and either program or schema are updated; and model-metamodel coevolution. There was some debate as to whether this is really a bx problem, and it was suggested that it is when you are concurrently using different versions of the database schemas/metamodels.
- *Concurrent development using artefacts:* this very practical scenario for bx arises where artefacts are being used by several distributed teams working concurrently. Each team acquires use of ("checks out") a part of an artefact (e.g., a submodel, a subgraph), makes changes, then commits the changes back to the "master" artefact. This is a variant on the first scenario (with the added dimension of concurrency), but also with further variation points, e.g., where the checked-out parts of an artefact are disjoint, and where they aren't.
- *Model-view architecture:* another practical scenario for bx is the *model view controller* software architecture, where a view represents a graphical display, and a model represents the data that is being represented. A controller mechanism is used to resolve inconsistencies in models and to orchestrate when updates to the view take place. A bx can be defined between model and view/controller to ensure that consistency between artefacts is established.
- *Consistency relation repair:* the final scenario that was discussed was related to repairing artefacts where consistency relationships between them had been violated: bx could be used to operationalise the repairs that re-establish consistency.

3.6 Repository

A repository of bx examples was also discussed and presented; the repository is meant to be a shared resource for all researchers in bx, to which anyone can contribute. bx examples are presented in a common format using a consistent template that captures the metadata that will help users categorise, identify, formulate and search for bx examples. The repository can be found at `http://bx-community.wikidot.com/examples:home`.

3.7 Tool Demos

The state of software tool support for bx was highlighted over the week. Previously we mentioned a number of demonstrations of non-bx tools and how they could be used to support specific bx scenarios (e.g., GROOVE, SDMlib and USE). Other novel tools that were discussed include GRoundTram [12], which focuses on support for compositional development of bx. eMOFLON, based on triple graph grammars, was also demonstrated, particularly focusing on its support for specifying *deltas*, which can be propagated incrementally. Finally, the Echo tool was presented, which aims to support inconsistency detection and repair by using a solver 'under-the-hood'. The tools demonstrated both the state-of-the-art and the very recent and significant progress made in providing realistic engineering support for use of bx.

4 Synthesis of Scientific Progress Made

We briefly highlight some of the most significant progress made during the week.

- Significant coalescence on the use of and appreciation for lenses as a common mathematical framework for describing, reasoning about, and implementing bx across programming languages, software engineering, graph transformation and databases. Both novel theoretical contributions and novel technical/engineering contributions towards the use of lenses were presented throughout the programme.
- The requirement for significant and realistic examples and benchmarks for bx in order to make sustained progress in both the theory and the practice. The contribution of a bx repository was a significant advance in achieving this requirement.
- The establishment of a precise definition of a 'useful' bx benchmark, the properties expected of a bx benchmark, and several examples of potential benchmarks.
- The enrichment of the theory of bx, both in terms of lenses (for example, precisely specifying the relationship between symmetric and asymmetric lenses), and in terms of type theory.

5 Outcome of the Meeting

Several concrete outcomes arose as a result of the meeting. An immediate outcome was a plan to propose a multi-disciplinary tutorial on bx at a leading database conference, bringing together experts from each of the PL, SE, DB and GT domains. A proposal to organise a tutorial at SIGMOD'14 was submitted in early January 2014. The motivation for proposing such a tutorial is best expressed by quoting from the proposal itself:

> The database research discipline, in many ways, pioneered research on [bx] transformations through the advent of updateable views. In the intervening time, other disciplines in computer science have taken up research in similar areas. There has been very little communication or citation between these disciplines ...
>
> With this tutorial, we hope to continue to open the lines of communication and introduce members of the database community to lines of research from other disciplines and to the state of our collaboration so far. As an outcome, we expect tutorial attendees will learn of ideas from other disciplines that will lead to identification of new research problems (or solutions) applicable to their own discipline. As such, this tutorial focuses on the development of synergies across disciplines.

Planning for a follow-up 1-week workshop (a successor to this BIRS workshop) was also started, and proposals for location were solicited and discussed. These included a return to Japan, a follow-up Dagstuhl seminar, and hosting an event Bertinoro, Italy, which regularly holds summer schools and related activities.

A further outcome was the plan to develop a *survey-tutorial* paper, based on the tutorial proposed at SIGMOD'14. This would capture both the state of the art, and the open challenges, for bx theory and practice. As well, plans to edit a book on bx were also put in place.

Finally, we are pleased that a number of new collaborations have developed as a result of this workshop. This is evidenced by the program for the upcoming *BX 2014* workshop, at which seven papers written by attendees of this BIRS workshop will be presented, with four of the papers being developed while at BIRS.

References

[1] A. Anjorin, K. Saller, S. Rose and A. Schürr, A Framework for Bidirectional Model-to-Platform Transformations. In *Proc. SLE 2012*, LNCS 7745, Springer-verlag, 2012.

[2] F. Bancilhon and N. Spyratos, Update Semantics of Relational Views. *ACM Trans. Database Syst.* 6(4): 557-575, 1981.

[3] M. Beine, N. Hames, J.H. Weber and A. Cleve, Bidirectional Transformations in Database Evolution: A Case Study "At Scale". In *Proc. Workshops at 17th International Conference on Extending Database Technology (EDBT)*, 8 pages, OpenProceedings.org, 2014.

[4] K. Czarnecki, J.N. Foster, Z. Hu, R. Lämmel, A. Schürr and J.F. Terwilliger, Bidirectional Transformations: A Cross-Discipline Perspective. In *Proc. ICMT 2009*, LNCS 5563, Springer-Verlag, 2009.

[5] Dagstuhl Seminar #11031 on Bidirectional Transformations, `http://www.dagstuhl.de/en/program/calendar/semhp/?semnr=11031`, last accessed January 2014.

[6] Eclipse QVT, `http://www.eclipse.org/mmt/?project=qvtd`, last accessed, January 2014.

[7] J.N. Foster, M.B. Greenwald, J.T. Moore, B.C. Pierce and A. Schmitt, Combinators for bidirectional tree transformations: A linguistic approach to the view-update problem. In *ACM Trans. Program. Lang. Syst.* 29(3), 2007.

[8] A. Ghamarian, M. de Mol, A. Rensink, E. Zambon and M. Zimakova, Modelling and analysis using GROOVE. In *STTT* 14(1): 15-40, 2012.

[9] M. Gogolla, F. Büttner and M. Richters, USE: A UML-based specification environment for validating UML and OCL. In *Sci. Comput. Program.* 69(1-3), 2007.

[10] GRACE International Meeting on Bidirectional Transformations, `http://grace.gsdlab.org/index.php?title=Main_Page`, last accessed January 2014.

[11] F. Hermann, H. Ehrig, C. Ermel, F. Orejas, Concurrent Model Synchronization with Conflict Resolution Based on Triple Graph Grammars. In *Proc. FASE 2012*, LNCS 7212, Springer-Verlag, 2012.

[12] S. Hidaka, Z. Hu, K. Inaba, H. Kato and K. Nakano, GRoundTram: An Integrated Framework for Developing Well-Behaved Bidirectional Model Transformations. *Progress in Informatics* 10:131-148, March 2013.

[13] D. Kolovos, R.F. Paige, L.M. Rose and A. Garcia Dominguez. *The Epsilon Book*, 2013.

[14] N. Macedo, T. Guimarães and A. Cunha, Model repair and transformation with Echo. In *Proc. ASE 2013*, ACM Press, 2013.

[15] U. Norbisrath, R. Jubeh and A. Zündorf, *Story Driven Modeling*, CreateSpace Independent Publishing Platform, 2013.

[16] A. Schürr and F. Klar, 15 Years of Triple Graph Grammars. In *Proc. ICGT 2008*, LNCS 5214, Springer-Verlag, 2008.

[17] P. Stevens, Bidirectional model transformations in QVT: semantic issues and open questions. In *Software and System Modeling* 9(1), 2010.

[18] J.F. Terwilliger, Bidirectional by necessity: Data persistence and adaptability for evolving application development. In *Proc. Generative and Transformational Techniques in Software Engineering IV*, LNCS 7680, Springer-Verlag, 2011.