An Event-B Sequent Prover in Prolog and ProB

Katharina Engels, Jan Gruteser, and Michael Leuschel

Faculty of Mathematics and Natural Science, Institute of Computer Science,
Heinrich Heine University Düsseldorf, Universitätsstr. 1, D-40225 Düsseldorf
{katharina.engels,jan.gruteser,michael.leuschel}@hhu.de

We present the latest advances of ProB's sequent prover for Event-B proof obligations [2] (POs) generated by Rodin. The sequent prover has been integrated into the Prolog core of ProB [6], implementing almost all of Rodin's inference and rewrite rules[1]. ProB PO files, exported from the ProB Rodin plugin, can now be directly loaded into the animator. The proof rules then induce a labelled transition system with sequents as states and the rule applications as transitions. The user can interactively perform proofs with the ProB animator, either through the state visualisation or by selecting transitions. A first automated prover using iterative deepening is also available. The ultimate goal is to automatically generate a comprehensible proof with details about the applied proof rules, in contrast to other external provers in Rodin, which only indicate whether a proof was successful or not.

In order to inspect single proof steps in more detail, we added a proof tree export as an interactive HTML document (cf. Figure 1). It is possible to generate a Rodin proof file (BPR) from a ProB proof trace, which enables replay in Rodin. For this, we map our proof rules to the corresponding Rodin reasoners and include necessary rewrite actions. Currently, the generated BPR file still needs to be entered manually into the Rodin project, but most proof rules are already replayed correctly (e.g., Figure 2). Particularly challenging are Rodin's simplification rules, as they are applied in one large step, making it difficult to apply our finely graded steps. However, replay is still possible, if these steps are marked as reviewed. If the correct reasoner can be used, we obtain additional validation of our proof, and the ProB prover could be integrated as a second chain.

Another interesting aspect is the comparison of the existing Java code with our Prolog implementation. The Prolog code is considerably more compact, see Table 1 for a comparison. For example, the rule `DERIV_DOM_TOTALREL` (which replaces the domain of a total relation with its value) comprises only one line in Prolog, whereas in Rodin a Java class with more than 200 lines is required. The effort to implement the similar rule `DERIV_RAN_SURJREL`, which is not yet available in Rodin, is just adding one more line:

```
simp_rule_with_hyps(domain(R),Dom,'DERIV_DOM_TOTALREL',Hyps) :-
   member(member(R,RType),Hyps), is_rel(RType,total,Dom,_).
simp_rule_with_hyps(range(R),Ran,'DERIV_RAN_SURJREL',Hyps) :-
   member(member(R,RType),Hyps), is_surj(RType,_,Ran).
```

[1] https://wiki.event-b.org/index.php/Inference_Rules/All_Rewrite_Rules

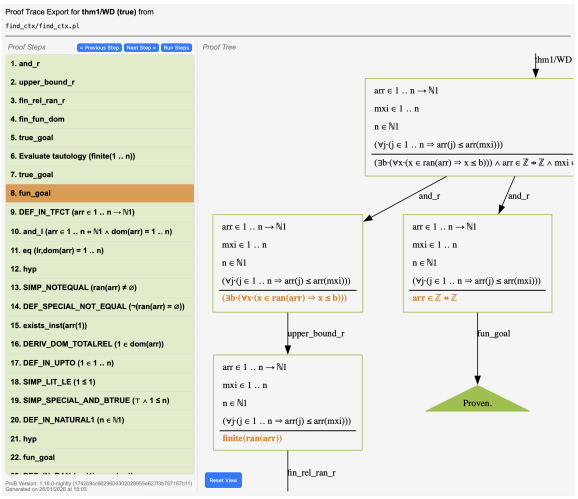

Fig. 1: Interactive HTML Proof Tree

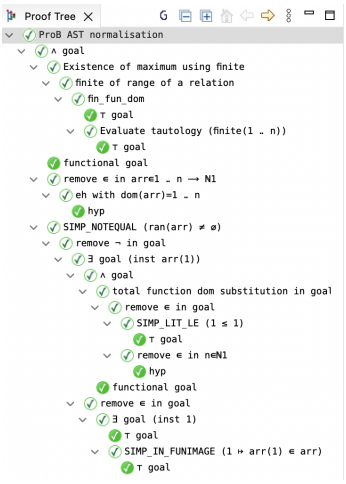

Fig. 2: Proof replayed by the Sequent Prover in Rodin

Table 1: Statistics of Java and Prolog Implementation

Tool	Language	Code for Sequent Prover				Missing Rules	
		Files	LOC	SLOC	Person Years	Rewrite	Inference
Rodin	Java	320	50182	27580	6.51	55/536	16/123
ProB	Prolog	4	4208	3705	0.79	12/536	8/123

There is related work, e.g. a translation of Rodin proofs to TLAPS [4]. The Why3 Plug-In for Rodin [5] translates proof obligations into Why3, but proofs are not back-translated to Event-B proofs. Some first works [3,7] try to translate B/Event-B proofs to the $\lambda\Pi$-Calculus.

In the future, we plan to improve the auto prover and integrate proving features into the ProB2-UI [1]. By using the AI interface of SimB [8], we could already perform AI-controlled proofs, given a suitably trained AI. We aim to extend and adapt the rules for the classical B method and also generate POs in ProB. For ProB in general, the rewrite rules could be used for AST simplifications.

References

1. J. Bendisposto, D. Geleßus, Y. Jansing, M. Leuschel, A. Pütz, F. Vu, and M. Werth. ProB2-UI: A Java-based User Interface for ProB. In *Proceedings FMICS*, LNCS 12863, pages 193–201, 2021.
2. K. Engels, J. Gruteser, and M. Leuschel. Interactive Proving with ProB. In *12th Rodin User and Developer Workshop*, 2025.
3. A. Grieu. From Event-B to Lambdapi. In *Proceedings ABZ*, pages 387–391, 2024.
4. A. Grieu, J.-P. Bodeveix, and M. Filali. Translating Event-B Models and Development Proofs to TLA+. In *Proceedings ABZ*, LNCS 15728, pages 124–142. Springer, 2025.
5. A. Iliasov, P. Stankaitis, D. Adjepon-Yamoah, and A. B. Romanovsky. Rodin platform why3 plug-in. In *Proceedings ABZ 2016*, LNCS 9675, pages 275–281, 2016.

6. M. Leuschel and M. J. Butler. ProB: an automated analysis toolset for the B method. *STTT*, 10(2):185–203, 2008.
7. C. Stolze, O. Hermant, and R. Guillaumé. Towards Formalization and Sharing of Atelier B Proofs with Dedukti. working paper or preprint, Jan. 2024.
8. F. Vu, J. Dunkelau, and M. Leuschel. Validation of Reinforcement Learning Agents and Safety Shields with ProB. In *Proceedings NFM*, volume 14627 of *LNCS*, pages 279–297. Springer, 2024.