VizAR: Visualization of Automated Reasoning Proofs (System Description) [⋆]

Jan Jakubův[1][0000−0002−8848−5537] and Cezary Kaliszyk[2][0000−0002−8273−6059]

[1] Czech Technical University in Prague, Prague, Czech Republic
jakubuv@gmail.com
[2] University of Innsbruck, Innsbruck, Austria and INDRC, Prague, Czech Republic
cezary.kaliszyk@uibk.ac.at

Abstract. We present a system for the visualization of proofs originating from Automated Theorem Provers for first-order logic. The system can hide uninteresting proof parts of proofs, such as type annotations, translate first-order terms to standard math syntax, and compactly display complex formulas. We demonstrate the system on several non-trivial automated proofs of statements from Mizar Mathematical Library translated to first-order logic, and we provide a web interface where curious users can browse and investigate the proofs.

Keywords: Proof Visualization · First-Order Logic · Automated Theorem Provers

1 Introduction

With the increasing power of *Automated Theorem Proving* systems (ATPs), the size and complexity of the proofs they output are also increasing. This additionally implies that analyzing such automatically generated proofs is becoming more daunting for users. This is of particular importance for proofs that originate from machine-learning-guided provers. The guided version of E, ENIGMA [6] can automatically find proofs of many theorems that have previously been provable only with long manual proofs. A large number of such proofs have been discussed in our recent work on machine learning for Mizar [5]. To allow users to inspect and analyze such proofs conveniently, we developed and present the VizAR system:

http://ai.ciirc.cvut.cz/vizar/

The system can hide uninteresting parts of proofs (such as Mizar soft type system annotations and reasoning about them), translate first-order terms to standard math syntax (such as presenting $\mathsf{Element}(x, y)$ as $x \in y$), and compactly display complex formulas. The system provides several ways to visualize

⋆ Supported by ERC-CZ grant no. LL1902 POSTMAN and EU Regional Development Fund under the Czech project AI&Reasoning no. CZ.02.1.01/0.0/0.0/15_003/00004, and Cost Action CA20111 EuroProofNet.

complex proofs. In the full proof view, the proof is displayed as an interactive SVG image. In order to simplify orientation in large proofs, the system features a conjecture-centered view which helps to identify essential proof steps. Finally, the proof step view allows the user to interactively browse individual proof steps and reveal the proof essence hidden in their symbols.

Related Work. There exist several tools for viewing general automatically found proofs. One of the first generally usable visual viewers for automatically found proofs was the LΩUI [7] viewer offered as part of the Omega system. TPTP tools include an interactive derivation viewer IDV [8] which allows users to focus on particular clauses in TPTP proofs and see their relation (distance) from the axioms and the conjecture. One of the most advanced viewers for proofs is PROOFTOOL [4] which allows viewing GAPT transformed proofs.

Urban et al [10] have developed an online tool for Mizar that checks if particular subgoals are ATP-provable and if so views the premises (rather than proof details as our tool does). Visualizing proof search differs quite a lot from the presentation of complete proofs and has also been investigated [3]. Hammer systems use automated theorem provers to find proofs of conjectures in more complex logics. The reconstruction of such ATP proofs often requires presenting them in a more complex logic including mechanisms able to transform the conjecture to its positive form [2]. Finally, the most advanced tools for presenting non-ATP Mizar proofs are used to render Mizar Library articles in LaTeX for the Journal of Formalized Mathematics [1]. To our best knowledge, we are not aware of any proof visualization tool as advanced as VizAR.

2 VizAR: The Proof Navigator

VizAR can display an arbitrary proof in the TPTP language. In addition, it integrates extended support for proofs of Mizar statements coming from the MPTP [9] translation of Mizar to first-order logic. A large amount of MPTP proofs has been recently generated by ATPs (E and Vampire) with machine learning guidance [5]. Selected proofs can be investigated on the VizAR web page. VizAR shows the original Mizar statements for every conjecture and assumption, and it provides links to Mizar proofs and symbol definitions.

Symbol Translation. MPTP uses its own names for Mizar symbols. VizAR uses Unicode symbols to display terms and predicates in standard mathematical notation when possible. For example, the MPTP symbol `m1_subset_1(X,Y)` corresponds to Mizar symbol $\mathsf{Element}(X,Y)$ and in VizAR it is presented as $X \in Y$.[3] Another example is the MPTP symbol `r2_wellord2(X,Y)` corresponding to the Mizar symbol $\mathsf{are\_equipotent}(X,Y)$ which is written as $|X| = |Y|$ in VizAR. The translation is implemented using simple templates to position arguments. For symbols without special VizAR translations, original Mizar symbol names are used. Mizar names are composed of various ASCII characters resembling the standard math notation, for example, `c=` stands for $\subseteq$.

[3] We use LaTeX to typeset VizAR syntax in this paper. Unicode used in HTML/SVG looks fairly close to it, with the exception of better spacing and fonts in LaTeX.

Clause Visualization. ATP proofs consist of clauses with positive and negative literals. In the VizAR syntax, clauses are displayed as sequents in order to avoid the negation sign ($\sim$ in TPTP). For example, the clause $A \mid B \mid \sim C \mid \sim D$ is considered as the logically equivalent sequent $C, D \Rightarrow A, B$. The *antecedents* (left-hand side) are implicitly connected by logical *and*, while the *consequents* (right-hand side) are implicitly connected by logical *or*. The sequents are visualized as boxes with the content displayed vertically (top-down) as demonstrated in Figure 1. Clauses without negative literals (for example, $A \mid B$) are displayed simply as A, B instead of $\top \Rightarrow A, B$. Clauses without positive literals (for example, $\sim C \mid \sim D$) are displayed as $C, D \Rightarrow \bot$. As an exception, a unit clause with one negative literal is displayed as $\neg A$ instead of $A \Rightarrow \bot$ to save space. This is the only case where the negation sign can be encountered in VizAR.

Clause Simplifications. MPTP first-order translations of Mizar statements typically use soft type guard predicates to specify types of variables. A typical clause (written as a sequent) looks as $\mathsf{natural}(X1), \mathsf{natural}(X2) \Rightarrow \mathsf{natural}(\mathsf{plus}(X1, X2))$. This states that the sum of two naturals is a natural number. To simplify the proof presentation, VizAR hides the type guards applied to variables, and introduces a different variable symbol for each type predicate, for example, N for natural numbers and R for real numbers. Hence the above sequent becomes just $\mathsf{natural}(\mathsf{plus}(N1, N2))$. In the VizAR syntax, this becomes simply $(N_1 + N_2) \in \mathbb{N}$ as VizAR uses Unicode subscript letters to typeset variable indices. This means that, for example, the VizAR statement $N_1 \in \mathbb{R}$ should be interpreted as "every natural number is a real number". As a second step, all negative occurrences of type guard predicates (even with a non-variable argument) are completely hidden. This is because they typically provide no interesting information from the human point of view. While the first simplification preserves all information in the clause, the second removes intuitively trivial literals but the original clause cannot be fully reconstructed.

Proof Transformations. Proofs considered by VizAR are proofs by contradiction because of the underlying ATP provers. The prover first negates the conjecture and then searches for the contradiction with other assumptions. An ATP proof in the TPTP language is a directed acyclic graph where the leaves correspond to assumptions and all the edges can be followed to the sink node representing the contradiction. Every inner node represents an inferred clause and the edges connect premises with the inference outcome. After symbol translations and clause simplifications, two consequent graph nodes might represent syntactically equal clauses. For example, the Mizar statements $\mathsf{Element}(X, \mathsf{NAT})$ and $\mathsf{natural}(X)$ are both represented as $X \in \mathbb{N}$ in VizAR. In these cases, to further simplify the proof graph, we unify consequent nodes labeled with the same VizAR expression and merge their respective source and destination edges.

Proof Visualizations. VizAR uses Graphviz to render proof graphs while the web interface is implemented by the static site generator Jekyll. VizAR web interface provides several ways to investigate ATP proofs. In the *full proof view*, the whole proof graph is displayed as an SVG image with hyperlinks. Graph

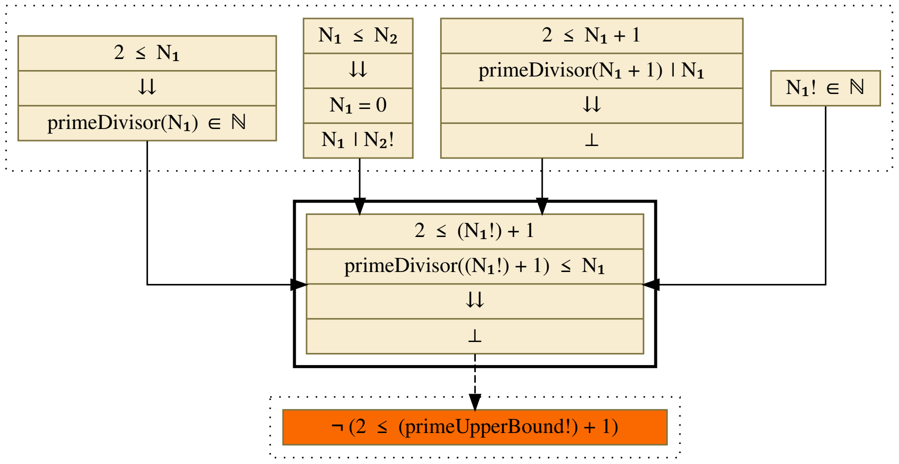
2 ≤ N₁
⇊
primeDivisor(N₁) ∈ ℕ
N₁ ≤ N₂
⇊
N₁ = 0
N₁ ∣ N₂!
2 ≤ N₁ + 1
primeDivisor(N₁ + 1) ∣ N₁
⇊
⊥
N₁! ∈ ℕ
2 ≤ (N₁!) + 1
primeDivisor((N₁!) + 1) ≤ N₁
⇊
⊥
¬ (2 ≤ (primeUpperBound!) + 1)

Fig. 1. Visualization of a proof step from the proof of MPTP theorem `t72_newton`.

leaves corresponding to assumptions are displayed in blue and all the nodes inferred from the negated conjecture are displayed in orange. Hence all non-orange nodes represent statements generally valid in Mizar. Clicking on any node takes the user to a detailed description of the corresponding proof step.

Since the full proof view might be very complex, VizAR features a *conjecture-centered view* where only the statements derived from the conjecture are displayed. This is a subgraph of the full proof view. Additionally, for every conjecture-related node, its non-conjecture premises used to derive this step are displayed. This view can help the user inspect how the negated conjecture is transformed into the contradiction. Thus, it is useful to identify the key steps of the proof.

In the *proof step view*, only a single proof graph node is displayed with its immediate parents and children. Additional information is provided about the symbols appearing in this proof step. Again, the user can click on any of the nodes to see the corresponding proof step view. Fig. 1 shows an example proof step in VizAR. The ATP proved the Mizar theorem `t72_newton`, which states that there is no upper bound on the prime numbers. Proving this in one ATP run is rather impressive, so we inspect the key steps leading to the contradiction. The ATP inferred that when $n!+1 \geq 2$ then $n!+1$ has no prime divisors less or equal to n. We already see the instantiation found $(n! + 1)$ and can inspect the key reasoning steps: In the later step, this is applied to the upper bound on primes assumed by the negated conjecture, which quickly leads to a contradiction since the upper bound must be greater than 2. For the sake of presentation, we display the ATP Skolem symbols (sk_i in VizAR) as $\mathsf{primeDivisor}$ and $\mathsf{primeUpperBound}$ and hide trivially false statements ($\mathsf{primeDivisor}(..) = 0, 1$).

Skolem symbols are introduced by ATPs to eliminate existential quantifiers and they typically constitute an important part of the proof. Hence it is important to understand their meaning and, to help the user with that, VizAR displays their origin in the proof overview. Clicking on the axiom will take the user to the *axiom view* where they will also see the original formula that gave rise to them. The Skolem symbols are also displayed in the proof step view when some of them are included in the step claim.

3 Conclusions and Future Work

We have developed the VizAR ATP proof visualization system and we publish its web interface on GitHub pages with a custom domain redirect. The web interface currently features selected ATP proofs of MPTP statements. In the proof gallery, we present *featured proofs* with improved VizAR syntax for all relevant Mizar symbols. Moreover, the *other proofs* section of the page contains a large number of proofs where Mizar names are used for selected symbols.

The VizAR system can be enhanced in many ways. First, VizAR syntax can be provided for more Mizar symbols to display statements in standard math notations. Second, additional proof simplification rules can be applied, for example, to hide clauses like $A \Rightarrow A$ or $(s = t) \Rightarrow (t = s)$. Such simplification rules could also be detected automatically or provided interactively.

References

1. Bancerek, G., Naumowicz, A., Urban, J.: System description: XSL-based translator of Mizar to LaTeX. In: CICM. LNCS, vol. 11006, pp. 1–6. Springer (2018)
2. Blanchette, J.C., Böhme, S., Fleury, M., Smolka, S.J., Steckermeier, A.: Semi-intelligible Isar proofs from machine-generated proofs. J. Autom. Reason. **56**(2), 155–200 (2016). https://doi.org/10.1007/s10817-015-9335-3
3. Byrnes, J., Buchanan, M., Ernst, M., Miller, P., Roberts, C., Keller, R.: Visualizing proof search for theorem prover development. In: UITP. ENTCS, vol. 226, pp. 23–38. Elsevier (2008). https://doi.org/10.1016/j.entcs.2008.12.095
4. Dunchev, C., Leitsch, A., Libal, T., Riener, M., Rukhaia, M., Weller, D., Paleo, B.W.: PROOFTOOL: a GUI for the GAPT framework. In: UITP. EPTCS, vol. 118, pp. 1–14 (2012). https://doi.org/10.4204/EPTCS.118.1
5. Jakubův, J., Chvalovský, K., Goertzel, Z.A., Kaliszyk, C., Olšák, M., Piotrowski, B., Schulz, S., Suda, M., Urban, J.: MizAR 60 for Mizar 50. CoRR (2023), `https://arxiv.org/abs/2303.06686`
6. Jakubův, J., Chvalovský, K., Olšák, M., Piotrowski, B., Suda, M., Urban, J.: ENIGMA Anonymous: Symbol-independent inference guiding machine (system description). In: IJCAR (2). LNCS, vol. 12167, pp. 448–463. Springer (2020)
7. Siekmann, J.H., Hess, S.M., Benzmüller, C., Cheikhrouhou, L., Fiedler, A., Horacek, H., Kohlhase, M., Konrad, K., Meier, A., Melis, E., Pollet, M., Sorge, V.: $L\Omega UI$: *L*ovely *Ω*mega *U*ser *I*nterface. Formal Aspects Comput. **11**(3), 326–342 (1999), `https://doi.org/10.1007/s001650050053`
8. Trac, S., Puzis, Y., Sutcliffe, G.: An Interactive Derivation Viewer. In: Autexier, S., Benzmüller, C. (eds.) Proceedings of the 7th Workshop on User Interfaces for Theorem Provers. Electronic Notes in Theoretical Computer Science, vol. 174, pp. 109–123 (2007). https://doi.org/10.1016/j.entcs.2006.09.025
9. Urban, J.: MPTP 0.2: Design, implementation, and initial experiments. J. Autom. Reason. **37**(1-2), 21–43 (2006). https://doi.org/10.1007/s10817-006-9032-3
10. Urban, J., Rudnicki, P., Sutcliffe, G.: ATP and presentation service for Mizar formalizations. J. Autom. Reason. **50**(2), 229–241 (2013)