Open Geometry Prover Community Project [*]

Nuno Baeta[0000-0002-1629-7924]
CISUC
University of Coimbra, Portugal
nmsbaeta@gmail.com

Pedro Quaresma[0000-0001-7728-4935]
CISUC, Department of Mathematics
University of Coimbra, Portugal
pedro@mat.uc.pt

Mathematical proof is undoubtedly the cornerstone of mathematics. The emergence, in the last years, of computing and reasoning tools, in particular automated geometry theorem provers, has enriched our experience with mathematics immensely. To avoid disparate efforts, the *Open Geometry Prover Community Project* aims at the integration of the different efforts for the development of geometry automated theorem provers, under a common “umbrella”. In this article the necessary steps to such integration are specified and the current implementation of some of those steps is described.

1 Introduction

Mathematical proof is undoubtedly the cornerstone of mathematics. All mathematics practitioner know its centrality and the difficulty in mastering it [8]. The emergence, in the last years, of computing and reasoning tools, in particular automated geometry theorem provers, has enriched our experience with mathematics immensely. Building such tools and exploring their applicability require a coherent, well-organized community of researchers working in a collaborative way, to avoid disparate efforts, as recalled by T. Han et al. [7]. Reuse of previous knowledge is vital for human beings in all kinds of learning activities, and so much more in mathematics. The reuse of practical implementations of an abstract idea is usually much harder than the reuse of the abstract idea itself. The same algorithm may be implemented several times using different programming languages and data formats due to engineering mismatches.

The *Open Geometry Prover Community Project* (*OGPCP*) aims at the integration of the different efforts for the development of geometry automated theorem provers, under a common “umbrella”. As such, a contribution to the larger goal of establishing a network of researchers working in the area of formal reasoning, knowledge-based intelligent software and geometric knowledge management, to explore efficient methodologies for the creation and reuse of electronic tools in geometry.

To bring up such a framework a series of tools and protocols must be implemented/established. The *Open Geometry Prover Community Project* framework, goals are:

- to provide a common open access repository for the development of Geometry Automated Theorem Provers (GATP);
- to provide an API to the different GATP in such a way that they can be easily used by users, stand-alone or integrated in other tools;
- to develop portfolio strategies to allow choosing the best GATP for any given geometric conjecture;
- to interface with repositories of geometric knowledge [27] (e.g. *TGTP*[1] [24], *TPTP*[2] [30]);
- to develop a GATP System Competition to be able to rate GATPs [2, 25].

[*]This work is funded by national funds through the FCT - Foundation for Science and Technology, I.P., within the scope of the project CISUC - UID/CEC/00326/2020 and by European Social Fund, through the Regional Operational Program Centro 2020.

[1]Thousand of Geometric problems for geometric Theorem Provers, http://hilbert.mat.uc.pt/TGTP/

[2]Thousands of Problems for Theorem Provers, http://www.tptp.org/

Overview of the paper. The paper is organised as follows: first, in §2, the current status of the framework implementation is described. In §3 a short description of the GATP currently incorporated in the *OGPCP* is given. Finally, in §4, conclusions are drawn and future work is discussed.

2 *OGPCP* Implementation Status

The *OGPCP* framework is a never-ending project in the sense that new GATP can be proposed and incorporated in the project at any given moment. Nevertheless many of the steps necessary for its current use and for an easy integration of new future projects are already done.

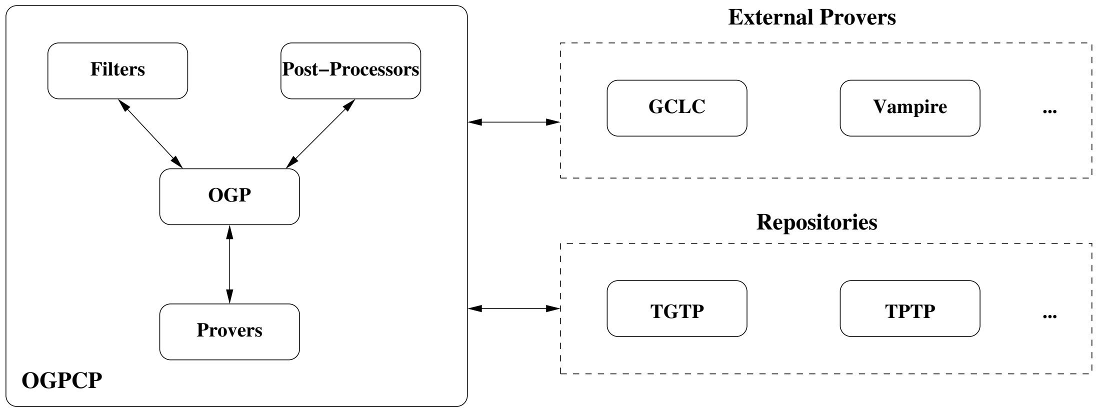

Figure 1: OGPCP Framework

2.1 *OGPCP* Source Repository

The *OGPCP* is hosted at GitHub.[3] The code is made available under the GNU General Public License,[4] version 3 or later, and the documentation under the GNU Free Documentation License,[5] version 1 or later.

OGPCP is available only as source-code and its installation in Unix systems is a straightforward process, provided that GNU Make, Apache Ant and OpenJDK are installed. After downloading the code from the GitHub repository, in the command line just type

```
$ make
$ make install
```

2.2 *OGPCP* Application Programming Interface

OGPCP API is a combination of several command-line tools, e.g., native (i.e. done by the *OGPCP* team and sharing a common base code) and external provers, filters, post-processors, prepared to seamlessly work together or with independent tools.

Native *OGPCP* provers must:

[3]https://github.com/opengeometryprover/OpenGeometryProver
[4]https://www.gnu.org/licenses/gpl.html
[5]https://www.gnu.org/licenses/fdl.html

- use *TPTP*'s first-order format (FOF) syntax as their default conjecture format;
- accept the same command-line arguments;
- provide the same output;

All this is explained in Open Geometry Prover Community Project *Programmer Manual*, available at the *OGPCP*'s GitHub repository.

External provers will be developed by other teams, with different base codes, even, eventually, using different programming languages (no enforcement is done on those matters). External provers conjecture's format may not adhere to TPTP's FOF syntax (see Section 4). In such cases, for *OGPCP* to take advantage of those provers, and vice-versa, filters to/from the FOF syntax must be written, as well as post-processors to interpret the output of those provers.

Example of usage and already implemented features:

Using conjectures *in situ*, contained in local files

`$ ogp ceva.gcl`	use of GCL prover, native language, area method
`$ ogp ceva.gcl -w`	use of GCL prover, native language, Wu's method
`$ ogp ceva.coqam`	use of CoqAM, native language, area method
`$ ogp ceva.fof`	use of inbuilt portfolio mechanism
`$ ogp -t 30 ceva.fof`	use of inbuilt portfolio mechanism with a time limit (30 seconds)

Using conjectures in a remote repository

`$ ogp --tgtp=GEO0001 gclc`	connection to the *TGTP* repository (see § 2.5).

The command line *OGPCP* meta-syntax is the following:

```
ogp [<option>] [<conjecture> [<prover> [<prover-options>]]]
```

The available options are:

`-h`, `--help`
prints a help message and exits — to be used alone;

`-p`, `--provers`
lists the available (to *OGPCP*) GATP and exits — to be used alone;

`-V`, `--version`
prints *OGPCP* version and exits — to be used alone;

`-t <time>`, `--timeout=<time>`
redefines the default time limit, in seconds, when proving a conjecture.

The conjecture is provided to the prover in a local file or, when using the remote repository *TGTP*, by its unique id in the repository, using the syntax

`--tgtp=<conjecture_id>`.

When attempting to prove a conjecture, the choice of the prover, if none is indicated, proceeds according to the following rules:

1. if the conjecture is provided in a local file, then
   (a) if file name extension is `fof`, then use *OGPCP* portfolio prover;
   (b) otherwise, use the default prover associated with that extension
2. otherwise, use *OGPCP* portfolio prover.

When the prover is given, a check is made to certify if the conjecture's format is one accepted by the prover. If that is not the case, whenever possible filters are used to convert to a format accepted by the prover. If all this fails, an error occurs and the process ended.

2.3 *OGPCP* Filters & Post-processors

A set of filters are already ready to be used.

`filterGCLtoFOF`	GCL language to FOF
`filterGEOGEBRAtoFOF`	GeoGebra[6] to FOF
`filterJGEXtoFOF`	JGEX to FOF

for the moment all these filters, `filter*toFOF`, assume the inclusion of the axioms of the deductive database full-angle method [6], given that these are already converted to FOF syntax.[7] That is, a plain conversion is made and an `include` instruction is added at the begin with the above mentioned axiom set.

Post-processors are to be used in conjunction with independent provers. They are used to obtain information about the proof's result, e.g., if the proof was successful or not, time, file with the proof steps, if any, etc., as the output of an independent prover must not adhere to that of a native *OGPCP* prover.

As of this writing, there is only one post-processor — for the Vampire ATP, to get the time of a proof.

2.4 *OGPCP* Portfolio

Portfolio problem solving is an approach in which for an individual instance of a specific problem, one particular, hopefully the most appropriate, solving technique is automatically selected among several available ones and used. Weindenbach [32] makes the distinction between syntactic and semantic approaches. With a *Simple-Syntactic portfolio solver* the selection of the core solvers is done by purely syntactic problem properties and there is no exchange of results between different core solvers. In a *Sophisticated-Semantic portfolio solver* the selection of the core solvers is done by semantic or structural problem properties and the solvers exchange results [32].

Already some work in the area of geometric automated theorem proving has been done, namely in the prover mechanism implemented in GeoGebra [16, 17, 22]. It is expected that this research can be incorporated into the *OGPCP*.

2.5 *OGPCP* Interface with Repositories

A server/client architecture to connect *OGPCP* and *TGTP* is already available. On the side of the *TGTP* repository a query-server is already implemented, always listening to client requests.

The code for the clients is open-source and available as part of the *OGPCP* project. The clients are build in such a way that a SQL query can be send to the *TGTP* database, receiving in return the code of the desired conjecture. The exchange of information between the server and the client is done using the *JSON* format.[8] The implementation of new clients to other GATP it is easy and opens the use of the information contained in *TGTP* from any GATP. This server/client architecture is currently being used to establish a connection between the e-learning environment *Web Geometry Laboratory* and the *TGTP* repository [27, 28].

For example, using the *OGPCP* API we could write: `ogp --tgtp=GEO0001 gclc`. This call will trigger the `tgtpToOgpcp` client, sending a query about the *gclc* code for problem *GEO0001* in *TGTP*. If

[6]`https://www.geogebra.org/`

[7]GEO012+0.ax, `http://www.tptp.org/cgi-bin/SeeTPTP?Category=Axioms`

[8]`https://www.json.org/`

the problem do not exist an error code will be returned, if the *gclc* for such a problem do not exist, the *FOF* code for that problem will be returned. After receiving an error free answer, the `ogp` command will pursue as usual.

2.6 Geometry Automated Theorem Provers Systems Competition

To be able to compare the different methods and implementations, a competition will have the virtue of pushing towards the standardization of the input language, the standardization of test sets, the direct comparability and the easier exchange of ideas and algorithmic techniques. The results of such a competition will also constitute a showcase, where potential users will look for the best GATP for their goals [2, 25].

A first trial-run of the *Geometry Automated Theorem Provers Systems Competition*, GASC 0.2, was already run, at ThEdu'19, the *8th International Workshop on Theorem proving components for Educational software*, August 2019, Natal, Brazil [25, 26] and a second trial-run is being prepared.

Not being directly related to the *OGPCP* the GASC will be used to test the different GATP in the project, pushing towards the development of new and better implementations.

3 External Geometry Automated Theorem Provers

A set of external GATP are already part of the OGPCP. These are autonomous open source projects that recognise the *OGPCP* and from which filters to/from the native syntax and FOF are already implemented, or will be implemented in a near future.

Those GATP must be downloaded and installed in a separate way, simple instructions on how to do it will be part of the *OGPCP* documentation.

GeoGebra Automated Reasoning Tools. The standard version of *GeoGebra*[9] includes several Automated Reasoning Tools (ART):

- for conjecturing a geometric property (e.g. such three points visually "seem" to be aligned), the `Relation` command;
- for rigorously denying or confirming a given conjecture (e.g. providing an affirmative answer to the conjecture after internally verifying, using Computer Algebra tools, that some determinant involving the coordinates of the three selected points is zero), the `Prove` and `ProveDetails` commands;
- for presenting some complementary hypotheses for the truth of a given (actually false) statement (e.g. remarking that the truth of the proposed statement needs some further steps in the geometric construction describing the statement), the `LocusEquation` command.

See [3] for a detailed explanation about the project and [15] for a tutorial-like paper about the different commands, as well as [19] for a quite updated version.

Moreover there are two other reasoning toolsets, already implemented but in (yet) non-standard versions of *GeoGebra* [4, 20]. The first one contains the `Discover` tool and command, and the `Compare` command, can be used in the *GeoGebra Discovery* fork,[10] available in two different options: one, operating over *GeoGebra Classic 5*, for *MS-Windows*, *Mac* and *Linux* systems; and, the other, working over *GeoGebra Classic 6*, that requires starting it in a browser, for tablets and smartphones.

[9] `https://www.geogebra.org/download`

[10] `https://github.com/kovzol/geogebra-discovery`, `http://autgeo.online/geogebra-discovery/`

The `Discover` command automatically finds all theorems (of a certain kind: parallelism, congruence, perpendicularity, etc.) holding over a given element of a construction (e.g. involving a point), by considering some combinatorial heuristics to formulate different *Relation* tests involving always the selected element, plus some other one, and presenting as output the collection of obtained properties. The `Compare` command is used to find a general relationship between two quantities (for example, by comparing the sum of the lengths of the catheti a and b and the length of the hypotenuse c in a right triangle—clearly, here the relationship is an inequality, namely, $c < a+b < \sqrt{2}c$). This low-level command is usually called from an improved version of the `Relation` command [31].

The second currently on-going improvement deals with the development of an *AG=Automated Geometer*,[11] a "geometer" that does not require human intervention, except that of launching the computation process over a figure. It is a web-based module that allows GeoGebra to automatically produce different conjectures over the given geometric construction, and to internally confirm or deny them using tools similar to those in GeoGebra Discovery, but here not limited to exploring relations involving a single, specific element.

The algorithms behind all these tools deal with the algebraic translation of the geometric statements and the symbolic manipulation—via the embedded computer algebra system GIAC[12] [13]—of the corresponding complex algebraic geometry varieties. See [14, 18, 21, 29], for a detailed description of the involved theoretical approach.

It must be remarked that the chosen method is quite effective and is able to deal, in milliseconds and over a variety of popular electronic devices (laptops, smartphones, etc.), with very complicated statements but, on the other hand, it does not provide any human-understandable arguments for the declared truth/falsity of the involved statements.

Finally, we summarize how GeoGebra's features can be directly exploited by OGPCP. GeoGebra offers two application programming interfaces for external programs:

- A JavaScript Application Programming Interface (API), available for web applications. The suggested method is to set up the construction via JavaScript calls and then execute the command `ProveDetails` to obtain the result.

- The desktop application can be directly called with an input GeoGebra file. The file structure is given in XML. By creating the XML data as input, and calling GeoGebra via command line, the debug information can be directly processed to get the result.

GCLC Automated Reasoning Tools. Within the mathematical software GCLC, there is an implementation of the area method [12] by Janičić and Quaresma, and implementations of (simple) Wu's method and Gröbner based method, by Predović and Janičić [11]. Apart a graphical user interface all the GATP can be used in stand-alone mode, begin usable in the overall *Open Geometry Prover Community Project* interface.

CoqAM. The formalization within the *Coq* proof assistant of the area method, a decision procedure for affine plane geometry [12, 23].[13] It can be used in stand-alone mode (*Coq* must be installed), being possible its use within overall *Open Geometry Prover Community Project* interface.

[11] `http://autgeo.online/ag/`, `https://github.com/kovzol/ag`

[12] Giac/Xcas, `https://www-fourier.ujf-grenoble.fr/~parisse/giac.html`

[13] `https://dpt-info.u-strasbg.fr/~narboux/area_method.html`

JGEX. Java Geometry Expert, *JGEX*, is a software which combines dynamic geometry software (DGS), automated geometry theorem prover (GATP) and an approach for visually dynamic presentation of proofs. As a dynamic geometry software, *JGEX* can be used to build dynamic visual models to assist teaching and learning of various mathematical concepts.[14]

Apart the use of the GATP systems inside the overall graphical DGS interface, they can be used standalone, being possible its use within the overall *Open Geometry Prover Community Project* interface.

Generic ATP Apart from these ATP, specific to geometry (GATP), the generic ATP can also be used. It is a question of including an axiomatic theory specific to geometry, e.g. those in the *Geo* domain in *TPTP* (Hilbert geometry; Tarski geometry, Rules of construction (von Plato), deductive database method, among others). Not being in the core of the project its use is, nevertheless, being taken in consideration and the *Open Geometry Prover Community Project* command line tool will process an input related to such tools.

4 Conclusions and Future Work

The problems related to the integration between different geometry provers can be much more harder than the presented above. Different algorithms/provers do not assume all the same mathematical setting. Different axiomatizations exist, e.g. Tarski's, Hilbert's, von Plato's; Area method. Different kinds of geometry, e.g. euclidean 2D or 3D, non-euclidean. Different types of approaches, geometric, e.g. area method, algebraic, e.g. Wu's method. More than a, maybe unrealistic, full integration, the *OGPCP* should aim to: give a simple, documented, open source, API to allow the use of GATP by experts and non-experts and to constitute itself as a forum, a space of discussion, about the deductive tools for geometry.

Apart many improvements in the existing framework, e.g. improve the API, linking with external provers, filters and post-processing, new native provers are planned: a new implementation of the full-angle method [5], the deductive database method [6] (using the axioms of the full-angle method), and a novel approach, the deductive graphs method, based on the deductive database method but using deductive graphs. Some initial work has already been done in those methods [1, 9, 10].

As said at the beginning the *OGPCP* is meant to be a never ending project in the sense that new improvements in the area of automated deduction will be made and incorporated in it. New methods, new implementations, improvements in the existing approaches, etc. To enlarge the usefulness and conquer new "audiences" (e.g. teachers in primary and secondary tools) the GATP need to be more modular, being able to be incorporated into "friendly" tools, that can cover the "difficult nature" of many GATP. The *OGPCP* should help on the goal of "bring the automated deduction to all geometers".

Acknowledgements

Zoltán Kovács, Tomás Recio, Francisco Botana (*GeoGebra*), Predrag Janičić (*GCLC*) and Julien Narboux (*COQAm*), contributed to this work by helping writing section 3.

[14]`https://github.com/yezheng1981/Java-Geometry-Expert`

References

[1] Nuno Baeta & Pedro Quaresma (2013): *The full angle method on the OpenGeoProver*. In C. Lange, D. Aspinall, J. Carette, J. Davenport, A. Kohlhase, M. Kohlhase, P. Libbrecht, P. Quaresma, F. Rabe, P. Sojka, I. Whiteside & W. Windsteiger, editors: *MathUI, OpenMath, PLMMS and ThEdu Workshops and Work in Progress at the Conference on Intelligent Computer Mathematics*, *CEUR Workshop Proceedings* 1010, Aachen. Available at `http://ceur-ws.org/Vol-1010/paper-08.pdf`.

[2] Nuno Baeta, Pedro Quaresma & Zoltán Kovács (2020): *Towards a Geometry Automated Provers Competition*. In: *Proceedings 8th International Workshop on Theorem proving components for Educational software*, *Electronic Proceedings in Theoretical Computer Science* 313, pp. 93–100, doi:10.4204/EPTCS.313.6. (ThEdu'19), Natal, Brazil, 25th August 2019,.

[3] Francisco Botana, Markus Hohenwarter, Predrag Janičić, Zoltán Kovács, Ivan Petrović, Tomás Recio & Simon Weitzhofer (2015): *Automated Theorem Proving in GeoGebra: Current Achievements*. *Journal of Automated Reasoning* 55(1), pp. 39–59, doi:10.1007/s10817-015-9326-4.

[4] Francisco Botana, Zoltán Kovács & Tomás Recio (2020): *A Mechanical Geometer*. *Mathematics in Computer Science*, doi:10.1007/s11786-020-00497-7.

[5] Shang-Ching Chou, Xiao-Shan Gao & Jing-Zhong Zhang (1996): *Automated Generation of Readable Proofs with Geometric Invariants, II. Theorem Proving With Full-Angles*. *Journal of Automated Reasoning* 17(13), pp. 349–370, doi:10.1007/BF00283134.

[6] Shang-Ching Chou, Xiao-Shan Gao & Jing-Zhong Zhang (2000): *A Deductive Database Approach to Automated Geometry Theorem Proving and Discovering*. *Journal of Automated Reasoning* 25, p. 219–246, doi:10.1023/A:1006171315513.

[7] The Anh Han, Lúis Moniz Pereira & Tom Lenaerts (2019): *Modelling and Influencing the AI Bidding War: A Research Agenda*. In: *AAAI/ACM conference on AI, Ethics and Society 2019*, doi:10.1145/3306618.3314265. Available at `http://www.aies-conference.com/wp-content/papers/main/AIES-19_paper_28.pdf`.

[8] Gila Hanna, David Reid & Michael de Villiers, editors (2019): *Proof Technology in Mathematics Research and Teaching*. Springer, doi:10.1007/978-3-030-28483-1.

[9] Yannis Haralambous & Pedro Quaresma (2014): *Querying Geometric Figures Using a Controlled Language, Ontological Graphs and Dependency Lattices*. In S. Watt et al., editor: *CICM 2014*, *LNAI* 8543, Springer, pp. 298–311, doi:10.1007/978-3-319-08434-3_22.

[10] Yannis Haralambous & Pedro Quaresma (2018): *Geometric Search in TGTP*. In Hongbo Li, editor: *Proceedings of the 12th International Conference on Automated Deduction in Geometry*, SMS International. Available at `http://adg2018.cc4cm.org/ADG2018Proceedings`.

[11] Predrag Janičić (2006): *GCLC — A Tool for Constructive Euclidean Geometry and More Than That*. In Andrés Iglesias & Nobuki Takayama, editors: *Mathematical Software - ICMS 2006*, *Lecture Notes in Computer Science* 4151, Springer, pp. 58–73, doi:10.1007/11832225_6.

[12] Predrag Janičić, Julien Narboux & Pedro Quaresma (2012): *The Area Method: a Recapitulation*. *Journal of Automated Reasoning* 48(4), pp. 489–532, doi:10.1007/s10817-010-9209-7.

[13] Z. Kovács & B. Parisse (2015): *Computer algebra and polynomials*, chapter Giac and GeoGebra – improved Gröbner basis computations, pp. 126–138. *LNCS* 8942, Springer Cham, doi:10.1007/978-3-319-15081-9_7.

[14] Z. Kovács, T. Recio & C. Sólyom-Gecse (2019): *Rewriting input expressions in complex algebraic geometry provers*. *Annals of Mathematics and Artificial Intelligence* 85(2-4), pp. 73–87, doi:10.1007/s10472-018-9590-1.

[15] Zltan Kovács, Tomas Recio & Maria Pilar. Vélez (2018): *Using Automated Reasoning Tools in GeoGebra in the Teaching and Learning of Proving in Geometry*. *International Journal for Technology in Mathematics Education* 25(2), pp. 33–50, doi:10.1564/tme_v25.2.03.

[16] Zoltán Kovács (2014): *The portfolio prover in GeoGebra 5.* In: *Proceedings of the 10th International Workshop on Automated Deduction in Geometry (ADG 2014).*

[17] Zoltán Kovács (2015): *The Relation Tool in GeoGebra 5.* In Francisco Botana & Pedro Quaresma, editors: *Automated Deduction in Geometry*, *Lecture Notes in Computer Science* 9201, Springer International Publishing, pp. 53–71, doi:10.1007/978-3-319-21362-0_4.

[18] Zoltán Kovács, Tomás Recio & M. Pilar Vélez (2019): *Detecting truth, just on parts.* *Revista Matemática Complutense* 32(2), pp. 451–474, doi:10.1007/s13163-018-0286-1.

[19] Zoltan Kovács, Tomas Recio & Maria Pilar Vélez (2022): *Mathematics Education in the Age of Artificial Intelligence*, chapter Automated Reasoning Tools with GeoGebra: What are they? What are they good for? Springer Nature. (to appear).

[20] Zoltán Kovács & Tomas Recio (2020): *GeoGebra Reasoning Tools for Humans and for Automatons.* In: *Electronic Proceedings of the 25th Asian Technology Conference in Mathematics*, Radford University, Radford, Virginia, USA, and Suan Sunandha Rajabhat University, Thailand, Mathematics and Technology, LLC, pp. 16–30, doi:10.13140/RG.2.2.26851.58407. Available at `http://atcm.mathandtech.org/EP2020/invited/21786.pdf`.

[21] Manuel Ladra, Pilar Páez-Guillán & Tomás Recio (2020): *Dealing with negative conditions in automated proving: tools and challenges. The unexpected consequences of Rabinowitsch's trick.* *Revista de la Real Academia de Ciencias Exactas, Físicas y Naturales. Serie A. Matemáticas* 114(4), doi:10.1007/s13398-020-00874-8.

[22] Zoltán Kovács & Predrag Janičić Mladen Nikolić, Vesna Marinković (2019): *Portfolio theorem proving and prover runtime prediction for geometry.* *Annals of Mathematics and Artificial Intelligence* 85(2-4), pp. 119–146, doi:10.1007/s10472-018-9598-6.

[23] Julien Narboux (2004): *A Decision Procedure for Geometry in Coq.* *Lecture Notes in Computer Science* 3223, pp. 225–240, doi:10.1007/b100400. Available at `http://portal.acm.org/citation.cfm?id=1784950.1784959`.

[24] Pedro Quaresma (2011): *Thousands of Geometric Problems for Geometric Theorem Provers (TGTP).* In Pascal Schreck, Julien Narboux & Jürgen Richter-Gebert, editors: *Automated Deduction in Geometry*, *Lecture Notes in Computer Science* 6877, Springer, pp. 169–181, doi:10.1007/978-3-642-25070-5_10.

[25] Pedro Quaresma & Nuno Baeta (2019): *Geometry Automated Theorem Provers Systems Competition 0.2 Report.* techreport 1, CISUC. Available at `https://www.cisuc.uc.pt/ckfinder/userfiles/files/TR2019-01.pdf`.

[26] Pedro Quaresma, Walther Neuper & João Marcos, editors (2020): *Proceedings 8th International Workshop on Theorem Proving Components for Educational Software.* 313, Open Publishing Association, doi:10.4204/EPTCS.313.

[27] Pedro Quaresma, Vanda Santos & Nuno Baeta (2018): *Exchange of Geometric Information Between Applications.* *Electronic Proceedings in Theoretical Computer Science* 267, pp. 108–119, doi:10.4204/eptcs.267.7.

[28] Pedro Quaresma, Vanda Santos & Milena Marić (2018): *WGL, a web laboratory for geometry.* *Education and Information Technologies* 23(1), pp. 237–252, doi:10.1007/s10639-017-9597-y.

[29] T. Recio & M. P. Vélez (1999): *Automatic Discovery of Theorems in Elementary Geometry.* *J. Autom. Reason.* 23, pp. 63–82, doi:10.1023/A:1006135322108. Available at `http://dl.acm.org/citation.cfm?id=594128.594243`.

[30] G. Sutcliffe (2017): *The TPTP Problem Library and Associated Infrastructure. From CNF to TH0, TPTP v6.4.0.* *Journal of Automated Reasoning* 59(4), pp. 483–502, doi:10.1007/s10817-017-9407-7.

[31] Róbert Vajda & Zoltán Kovács (2020): *GeoGebra and the realgeom Reasoning Tool.* In P. Fontaine, K. Korovin, I. S. Kotsireas, P. Rümmer & S. Tourret, editors: *PAAR+SC-Square 2020. Workshop on Practical Aspects of Automated Reasoning and Satisfiability Checking and Symbolic Computation Work shop 2020*, pp. 204–219. arXiv:http://ceur-ws.org/Vol-2752/paper15.pdf.

[32] Christoph Weidenbach (2017): *Do Portfolio Solvers Harm?* In Giles Reger & Dmitriy Traytel, editors: *ARCADE 2017. 1st International Workshop on Automated Reasoning: Challenges, Applications, Directions, Exemplary Achievements*, *EPiC Series in Computing* 51, EasyChair, pp. 76–81, doi:10.29007/vpxm.