Formal Representation and Solution of Plane Geometric Problems

Xiaokai Zhang, Na Zhu, Cheng Qin, Yang Li, Tuo Leng[*]
School of Computer Engineering and Science
Shanghai University
tleng@shu.edu.cn

Zhenbing Zeng
College of Sciences
Shanghai University

Project: https://github.com/FormalGeo/FormalGeo

Abstract

Geometric problem solving has always been a long-standing challenge in the fields of mathematical reasoning and artificial intelligence. In this paper, we present formalgeo7k, a geometric problem dataset based on rigorous geometry formalization theory and consistent geometry formal system, serving as a benchmark for various tasks such as geometric diagram parsing and geometric problem solving. All problems are annotated with problem text, problem diagram, formal descriptions, and solution. Combining symbolic solver and deep learning techniques, we can achieve human-like, traceable and explainable solutions, which are stored in a hypergraph for graph-related tasks. We experiment with various methods and the best method achieves only 86.40% on formalgeo7k. This shows that formalgeo7k presents a challenge for future research.

1 Introduction

Geometric problem solving (GPS) has always been a long-standing challenge [20, 28, 12] in the fields of mathematical reasoning and artificial intelligence, owing to the cross-modal forms of knowledge and the absence of automated solving methods. A typical geometric problem consists of a textual problem description and a geometric diagram. GPS requires solvers to possess multimodal fusion and reasoning capabilities, which have attracted much attention recently. Existing works mostly construct a neuro-symbolic system for GPS. Deductive Database (DD) methods [13, 38, 22] parse the problem text and diagram into formal language and then solve the problem by logical reasoning. Program sequence generation (PSG) methods [5, 34, 27] encode the problem text and diagram, input the encoding into a decoder, and generate a program sequence, which is then executed by a program executor.

However, existing methods focus on the research of the neural part while neglecting the symbolic part. First, existing methods fail to achieve a human-like problem-solving process. DD methods cannot eliminate redundant theorems, and PSG methods lack mathematical rigor. This not only undermines the readability of the solutions but also limits their application in mathematics education. Second, existing methods lack research on geometry formalization theory. This not only fails to ensure the consistency of the solver's reasoning process but also hinders the expansion of symbolic systems. Defining new theorems requires modifying the solver's code, making it difficult to represent more complex problems, such as problems at International Mathematical Olympiad (IMO) level. Furthermore, existing datasets are small in scale and cannot serve as benchmarks for training and evaluating large language model (LLMs). Some datasets are poorly annotated or contain errors, making them unsuitable as a unified benchmark.

[*] Corresponding Author

38th Conference on Neural Information Processing Systems (NeurIPS 2024) Workshop on MATH-AI.

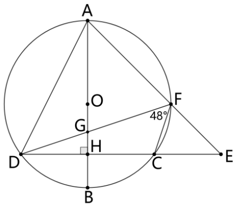


Image CDL
Equal(MeasureOfAngle(CFD),48)
PerpendicularBetweenLine(DH,GH)

Text CDL
Equal(MeasureOfAngle(CFD),48)
IsCentreOfCircle(O,O)
IsDiameterOfCircle(AB,O)
PerpendicularBetweenLine(DH,GH)

Construction CDL
Shape(OAD,DA)
Shape(AD,DG,GO,OA)
Shape(AO,OG,GF,FA)
Shape(OFA,AF)
Shape(GD,DH,HG)
Shape(FG,GH,HC,CF)
Shape(OCF,FC)
Shape(OCF,CE,EF)
Shape(BH,HD,ODB)
Shape(CH,HB,OBC)
Collinear(AOGHB)
Collinear(DGF)
Collinear(AFE)
Collinear(DHC)
Cocircular(O,ADBCF)

Problem Text
English: As shown in the diagram, ∠CFD=48°, O is the center of ⊙O, the diameter of ⊙O is AB, DH is perpendicular to GH. Find the measure of ∠EFC.
Chinese: 如图所示，∠CFD=48°，O是⊙O的圆心，⊙O的直径为AB，DH垂直于GH。求∠EFC的大小。

Image Annotation (parsed from .gbb)
Points: {A:[919,980], B:[1378,827], C:[919,1103], ..., O:[919,827]}

Goal CDL	**Answer**
Value(MeasureOfAngle(EFC))	66

Theorem Sequences
circle_property_chord_perpendicular_bisect_arc(1,ODC,OHB)
arc_addition_length(1,ODB,OBC)
similar_arc_judgment_cocircular(1,ODB,ODC)
similar_arc_property_length_ratio(1,ODB,ODC)
similar_arc_property_measure_ratio(1,ODB,ODC)
arc_property_circumference_angle_external(1,ODB,A)
arc_property_circumference_angle_external(1,OAC,F)
arc_property_circumference_angle_external(1,ODC,F)
triangle_property_angle_sum(1,ADH)
arc_property_circumference_angle_internal(1,OAC,D)
adjacent_complementary_angle(1,EFC,CFA)

Figure 1: An example of annotated geometric problem in formalgeo7k.

To address these issues, we develop FormalGeo, a consistent geometry formal system based on rigorous geometry formalization theory [35]. We further construct the formalgeo7k dataset, which contains 7,000 geometric problems. All problems are annotated with problem text, problem diagrams, formal descriptions, and solutions, as shown in Fig. 1. This dataset can serve as a benchmark for various tasks, such as GPS, geometric diagram parsing (GDP), geometric relation extraction (GRE), geometric problem formalization and the evaluation of LLMs reasoning abilities.

To assess the dataset difficulty, we experiment with several methods and the best method only achieves 86.40% problem solving success rate (PSSR). This shows that formalgeo7k presents a challenge for future research.

2 formalgeo7k dataset

We present formalgeo7k, a geometric problem dataset based on rigorous geometry formalization theory and consistent geometry formal system, serving as a benchmark for various tasks such as geometric diagram parsing and geometric problem solving. All code and datasets are open-source and available at `https://github.com/FormalGeo/FormalGeo`. Refer to Appx. D for instructions on how to use *pip* to build FormalGeo formal systems and download formalgeo7k.

2.1 Geometry formal system

We build FormalGeo, a consistent geometry formal system based on rigorous geometry formalization theory [35]. FormalGeo incorporates diagrammatic reasoning, algebraic computation, relational reasoning, and logical operations into a unified framework of geometry predicate logic, whose syntax is similar to first-order predicate logic. Within this formal framework, we can transform geometric diagrams and text into a consistent formal language, which not only maintains good readability but can also be mechanically processed by computer. The final solution of geometric problems can be represented as hypergraph, where conditions serve as hypernodes and theorems as hyperedges, thus transforming into a human-like solution. In addition, FormalGeo uses geometry definition language (GDL) to define new predicates or theorems and uses condition declaration language (CDL) to describe geometry problems, making it extremely convenient to extend the formal system and represent complex problems. Details of GDL are provided in Appx. A. A formal representation and solution of a 2022 IMO geometry problem can be found in Appx. B.

2.2 Dataset annotation

Our formalgeo7k Dataset contains 7,000 plane geometric problems, sourced from the Geometry3K [13] (40.69%), GeoQA [5] (53.38%), and GeoQA+ [3] (2.93%). Our annotation task can be divided into four parts: 1. Standardizing the style of problem texts (English and Chinese). For problems lacking English or Chinese descriptions, we add the missing descriptions. 2. Redrawing geometric diagrams using GeoGebra and saving the *.gbb* files, which contains detailed positional information of the geometric elements. 3. Annotating geometric diagrams and texts using FormalGeo formal language. 4. Adding the theorem sequences required to solve the geometric problems.

Once the problems are annotated, we assign a different annotator to review and verify the correctness of the annotations. Finally, all problems are input into a symbolic solver FGPS [36] to check the correctness of the syntax, ensure that the annotated theorems can solve the problem and eliminate redundant theorems. By treating the intermediate results of a problem as a new goal, we can decompose a geometric problem into multiple sub-problem, automatically expanding the number of problems to tens of thousands. This results in a dataset with a smoother difficulty curve, and the number of problems becomes sufficient for training and evaluating LLMs.

Sixteen trained graduate master's students participated in the dataset annotation tasks. The annotation and reviewing process took approximately 1,500 person-hours.

2.3 Dataset Statistics

Our dataset has 7,000 geometric problems. Each problem text (English) has a maximum of 444 characters and a minimum of 58 characters, with an average problem text length of 114.22. The construction CDL describes the topological structure of a geometric problem, and the number of its statements roughly reflects the complexity of the geometric diagram. The average length of construction CDL statements is 5.97. Both text CDL and image CDL describe the conditions of the geometric problem, with the average number of statements being 4.02 and 2.87, respectively. In the text CDL, the most frequently occurring predicates are *Equal* (61.04%), *IsCentreOfCircle* (9.67%), *PerpendicularBetweenLine* (8.69%), *ParallelBetweenLine* (4.92%), and *IsTangentOfCircle* (3.22%). In the diagram CDL, The proportions of *Equal*, *PerpendicularBetweenLine*, and *ParallelBetweenLine* are 80.95%, 12.16%, and 6.89%, respectively. The number of theorems required for GPS serves as an measure of the difficulty of the problem. Each problem involves a maximum of 28 theorems and a minimum of 1 theorems, with an average theorem number of 4.34. The detailed statistical information can be found in Appx. C.

3 Experiments

3.1 Benchmark methods

We tested several methods on formalgeo7k. It is important to note that most SOTA models for GPS are trained in specific symbolic environments (such as those provided by Geometry3K [13] or GeoQA [5]), and adapting them to the FormalGeo would require significant manual effort. Therefore, we only compare methods using the FormalGeo environment. All methods use annotated text CDL and image CDL as input to the model. Parsing CDL from the original problem diagram and text is still a challenge, and we leave it for future work.

Forward Search [36]. This is a purely symbolic approach. It starts from the initial conditions of the problem and continuously apply theorems to derive new conditions until the goal is achieved. We run the forward search method using four different strategies (breadth-first, depth-first, random and beam) and presented the results of the best strategy (random). The maximum search depth is set to 15, the beam size to 20, and the timeout for each problem is set to 600 seconds.

Backward Search [36]. This is a purely symbolic approach. It begins with the problem-solving goal, expands it into multiple sub-goals, and repeats this process until all sub-goals are resolved. The best strategy for the backward search is breadth-first, with the other parameters set the same as in the forward search.

FGeo-TP [9]. This method utilizes the language model to predict the theorem sequences for GPS. The implementation details and training methods are consistent with the original paper.

Table 1: Details of PSSR.

Method	Total	L_1	L_2	L_3	L_4	L_5	L_6
Forward Search	39.71	59.24	40.04	33.68	16.38	5.43	4.79
Backward Search	35.44	67.22	33.72	11.15	6.67	6.07	1.03
FGeo-TP	80.86	96.43	85.44	76.12	62.26	48.88	29.55
FGeo-DRL	86.40	97.65	94.21	85.87	70.45	46.81	32.18
HyperGNet	85.53	95.44	89.46	84.25	77.84	50.00	45.76

FGeo-DRL [38]. This method leverages a pre-trained natural language model to establish a policy network for theorem selection and employ monte carlo tree search for heuristic exploration. The implementation details and training methods are consistent with the original paper.

HyperGNet [37]. This method builds a neural-symbolic system to effectively embed geometry knowledge and automatically perform human-like geometric deductive reasoning. The implementation details and training methods are consistent with the original paper.

3.2 Experimental results

To provide a more detailed comparison of different models' capabilities, we divided the dataset into 6 levels based on the length of the theorem sequence l required to solve the problem: $L_1(l \leq 2)$, $L_2(3 \leq l \leq 4)$, $L_3(5 \leq l \leq 6)$, $L_4(7 \leq l \leq 8)$, $L_5(9 \leq l \leq 10)$, $L_6(l \geq 11)$. The experimental results are shown in Tab. 1.

We run the search method on two Intel i9-10900X processors, one AMD Ryzen 9 5900X, and one AMD Ryzen 9 7950X, and train the neural network on a single GeForce RTX 4090. We run the search method using multiple processes while maintaining a CPU utilization rate of 80%. The total duration of the search is approximately 3 days, and executing a single training session of the neural network require approximately 30 minutes.

It is evident that the longer the theorem sequence required to solve the problem, the higher the difficulty and the lower PSSR. We can see that, compared to traditional brute-force search methods, heuristic search combined with deep learning techniques has significantly improved the problem-solving success rate.

4 Related Work

The study of GPS has a long history, which can be broadly divided into algebraic methods and synthetic methods. Algebraic methods transform geometric problems into a system of algebraic equations consisting of polynomials and inequalities, such as Wu's method [25], Gröbner basis methods [2] and elimination methods [24]. Synthetic methods encompass a wider range of approaches, including search-based methods [8, 15], knowledge-based methods [7], geometric invariants-based methods [32, 6], and machine Learning and optimization methods [21, 1, 19]. GPS has seen further advancements in recent years. Existing methods predominantly integrate deep learning and symbolic reasoning to construct a neuro-symbolic system for solving geometric problems. DD methods parse the problem text and diagram into formal language and then solve the problem by logical reasoning. Representative DD methods include Inter-GPS [13], GeoDRL [17], AlphaGeometry [22], FGeo [38, 9] and E-GPS [26]. PSG methods encode the problem text and diagrams, input the encoding into a decoder, and generate a program sequence, which is then executed by a program executor. Representative PSG methods include NGS [5], Geoformer [4], DPE [3], PGPSNet [34], SAC-GPS [16], UniMath [11], LANS [10], DualGeoSolver [27] and adaptive learning model [30]. Several geometry formal systems and datasets have been developed. We compared existing datasets with formalgeo7k, as shown in Tab. 2. In addition to geometric problem solving, tasks such as geometric diagram parsing [33] and geometric formalization [14] have also begun to attract increasing attention.

5 Conclusion

Based on rigorous geometry formalization theory and a consistent geometry formal system, we developed formalgeo7k, a dataset containing 7,000 annotated geometry problems, including problem

Table 2: Comparative analysis with existing geometric problem datasets.

Method	Task	Size	Comparative Metrics for GPS task							
			FM	FW	BW	NT	PT	FS	HS	CR
GEOS [21]	GPS	186	DD	✓		✓				
GEOS++ [18]	GPS	1,406	DD	✓		✓				
GEOS-OS [19]	GPS	2,235	DD	✓		✓				
Geometry3K [13]	GPS	3,002	DD	✓		✓				
GeoQA [5]	GPS	5,010	PSG	✓		✓				
GeometryQA [23]	GPS	1,398	PSG	✓		✓				
GeoRE [29]	GRE	1,398	-	-	-	-	-	-	-	-
GeoQA+ [3]	GPS	7,528	PSG	✓		✓				
UniGeo [4]	GPS	14,541	PSG	✓		✓	✓			
PGDP5K [33]	GDP	5,000	-	-	-	-	-	-	-	-
PGPS9K [34]	GPS	9,022	PSG	✓		✓				
GeoEval [31]	GPS	5050	-	✓		✓				✓
formalgeo7k	All	7000	DD	✓	✓	✓	✓	✓	✓	✓

* All denotes GPS+GDP+GRE. The 8 comparative metrics are: Formalization Methods, ForWard solving, BackWard solving, Numerical Targets, Proving Targets, Formal System, Human-like Solutions and Complexity Ratings.

text, problem diagrams, formal descriptions, and solutions. formalgeo7k serves as a benchmark for various tasks such as geometric diagram parsing and geometric problem solving, directly benefiting both AI4MATH and AI4EDU research. Experimental results indicate that formalgeo7k presents a challenge for future research.

References

[1] Alvin, C., Gulwani, S., Majumdar, R., Mukhopadhyay, S.: Synthesis of solutions for shaded area geometry problems. In: The Thirtieth International Flairs Conference (2017)

[2] Buchberger, B.: Applications of gröbner bases in non-linear computational geometry. Mathematical aspects of scientific software pp. 59–87 (1988)

[3] Cao, J., Xiao, J.: An augmented benchmark dataset for geometric question answering through dual parallel text encoding. In: Calzolari, N., Huang, C.R., Kim, H., Pustejovsky, J., Wanner, L., Choi, K.S., Ryu, P.M., Chen, H.H., Donatelli, L., Ji, H., Kurohashi, S., Paggio, P., Xue, N., Kim, S., Hahm, Y., He, Z., Lee, T.K., Santus, E., Bond, F., Na, S.H. (eds.) Proceedings of the 29th International Conference on Computational Linguistics. pp. 1511–1520. International Committee on Computational Linguistics, Gyeongju, Republic of Korea (Oct 2022), https://aclanthology.org/2022.coling-1.130

[4] Chen, J., Li, T., Qin, J., Lu, P., Lin, L., Chen, C., Liang, X.: UniGeo: Unifying geometry logical reasoning via reformulating mathematical expression. In: Goldberg, Y., Kozareva, Z., Zhang, Y. (eds.) Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing. pp. 3313–3323. Association for Computational Linguistics, Abu Dhabi, United Arab Emirates (Dec 2022). https://doi.org/10.18653/v1/2022.emnlp-main.218, https://aclanthology.org/2022.emnlp-main.218

[5] Chen, J., Tang, J., Qin, J., Liang, X., Liu, L., Xing, E., Lin, L.: GeoQA: A geometric question answering benchmark towards multimodal numerical reasoning. In: Zong, C., Xia, F., Li, W., Navigli, R. (eds.) Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021. pp. 513–523. Association for Computational Linguistics, Online (Aug 2021). https://doi.org/10.18653/v1/2021.findings-acl.46, https://aclanthology.org/2021.findings-acl.46

[6] Chou, S.C., Gao, X.S., Zhang, J.Z.: Automated production of traditional proofs in solid geometry. Journal of Automated Reasoning **14**(2), 257–291 (1995)

[7] Chou, S.C., Gao, X.S., Zhang, J.Z.: A deductive database approach to automated geometry theorem proving and discovering. Journal of Automated Reasoning **25**(3), 219–246 (2000)

[8] Gelernter, H.L.: Realization of a geometry theorem proving machine. In: IFIP congress. pp. 273–281 (1959)

[9] He, Y., Zou, J., Zhang, X., Zhu, N., Leng, T.: Fgeo-tp: A language model-enhanced solver for euclidean geometry problems. Symmetry **16**(4) (2024). https://doi.org/10.3390/sym16040421, https://www.mdpi.com/2073-8994/16/4/421

[10] Li, Z.Z., Zhang, M.L., Yin, F., Liu, C.L.: LANS: A layout-aware neural solver for plane geometry problem. In: Ku, L.W., Martins, A., Srikumar, V. (eds.) Findings of the Association for Computational Linguistics ACL 2024. pp. 2596–2608. Association for Computational Linguistics, Bangkok, Thailand and virtual meeting (Aug 2024), https://aclanthology.org/2024.findings-acl.153

[11] Liang, Z., Yang, T., Zhang, J., Zhang, X.: UniMath: A foundational and multimodal mathematical reasoner. In: Bouamor, H., Pino, J., Bali, K. (eds.) Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing. pp. 7126–7133. Association for Computational Linguistics, Singapore (Dec 2023). https://doi.org/10.18653/v1/2023.emnlp-main.440, https://aclanthology.org/2023.emnlp-main.440

[12] Littman, M.L., Ajunwa, I., Berger, G., Boutilier, C., Currie, M., Doshi-Velez, F., Hadfield, G., Horowitz, M.C., Isbell, C., Kitano, H., et al.: Gathering strength, gathering storms: The one hundred year study on artificial intelligence (ai100) 2021 study panel report. arXiv preprint arXiv:2210.15767 (2022)

[13] Lu, P., Gong, R., Jiang, S., Qiu, L., Huang, S., Liang, X., Zhu, S.C.: Inter-GPS: Interpretable geometry problem solving with formal language and symbolic reasoning. In: Zong, C., Xia, F., Li, W., Navigli, R. (eds.) Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers). pp. 6774–6786. Association for Computational Linguistics, Online (Aug 2021). https://doi.org/10.18653/v1/2021.acl-long.528, https://aclanthology.org/2021.acl-long.528

[14] Murphy, L., Yang, K., Sun, J., Li, Z., Anandkumar, A., Si, X.: Autoformalizing Euclidean geometry. In: Salakhutdinov, R., Kolter, Z., Heller, K., Weller, A., Oliver, N., Scarlett, J., Berkenkamp, F. (eds.) Proceedings of the 41st International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 235, pp. 36847–36893. PMLR (21–27 Jul 2024), https://proceedings.mlr.press/v235/murphy24a.html

[15] Nevins, A.J.: Plane geometry theorem proving using forward chaining. Artificial Intelligence **6**(1), 1–23 (1975)

[16] Ning, M., Wang, Q.F., Huang, K., Huang, X.: A symbolic characters aware model for solving geometry problems. In: Proceedings of the 31st ACM International Conference on Multimedia. p. 7767–7775. MM '23, Association for Computing Machinery, New York, NY, USA (2023). https://doi.org/10.1145/3581783.3612570, https://doi.org/10.1145/3581783.3612570

[17] Peng, S., Fu, D., Liang, Y., Gao, L., Tang, Z.: GeoDRL: A self-learning framework for geometry problem solving using reinforcement learning in deductive reasoning. In: Rogers, A., Boyd-Graber, J., Okazaki, N. (eds.) Findings of the Association for Computational Linguistics: ACL 2023. pp. 13468–13480. Association for Computational Linguistics, Toronto, Canada (Jul 2023). https://doi.org/10.18653/v1/2023.findings-acl.850, https://aclanthology.org/2023.findings-acl.850

[18] Sachan, M., Dubey, K., Xing, E.: From textbooks to knowledge: A case study in harvesting axiomatic knowledge from textbooks to solve geometry problems. In: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing. pp. 773–784 (2017)

[19] Sachan, M., Xing, E.: Learning to solve geometry problems from natural language demonstrations in textbooks. In: Proceedings of the 6th Joint Conference on Lexical and Computational Semantics (SEM 2017). pp. 251–261 (2017)

[20] Selsam, D., de Moura, L., Buzzard, K., Barton, R., Liang, P., Loos, S., Wiedijk, F.: Imo grand challenge (2019), https://imo-grand-challenge.github.io/, accessed on September 14, 2024

[21] Seo, M., Hajishirzi, H., Farhadi, A., Etzioni, O., Malcolm, C.: Solving geometry problems: Combining text and diagram interpretation. In: Proceedings of the 2015 conference on empirical methods in natural language processing. pp. 1466–1476 (2015)

[22] Trinh, T.H., Wu, Y., Le, Q.V., He, H., Luong, T.: Solving olympiad geometry without human demonstrations. Nature **625**(7995), 476–482 (2024). https://doi.org/10.1038/s41586-023-06747-5, https://doi.org/10.1038/s41586-023-06747-5

[23] Tsai, S.h., Liang, C.C., Wang, H.M., Su, K.Y.: Sequence to general tree: Knowledge-guided geometry word problem solving. In: Zong, C., Xia, F., Li, W., Navigli, R. (eds.) Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers). pp. 964–972. Association for Computational Linguistics, Online (Aug 2021). https://doi.org/10.18653/v1/2021.acl-short.121, https://aclanthology.org/2021.acl-short.121

[24] Wang, D.: An elimination method for polynomial systems. Journal of Symbolic Computation **16**(2), 83–114 (1993)

[25] Wu, W.T.: On the decision problem and the mechanization of theorem proving in elementary geometry. Scientia Sinica **21**, 157–179 (1978)

[26] Wu, W., Zhang, L., Liu, J., Tang, X., Wang, Y., Wang, S., Wang, Q.: E-gps: Explainable geometry problem solving via top-down solver and bottom-up generator. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 13828–13837 (June 2024)

[27] Xiao, T., Liu, J., Huang, Z., Wu, J., Sha, J., Wang, S., Chen, E.: Learning to solve geometry problems via simulating human dual-reasoning process. In: Larson, K. (ed.) Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence, IJCAI-24. pp. 6559–6568. International Joint Conferences on Artificial Intelligence Organization (8 2024). https://doi.org/10.24963/ijcai.2024/725, https://doi.org/10.24963/ijcai.2024/725, main Track

[28] XTXMarkets: Artificial intelligence mathematical olympiad prize (aimo prize) (2023), https://aimoprize.com/, accessed on September 14, 2024

[29] Yu, W., Wang, M., Wang, X., Zhou, X., Zha, Y., Zhang, Y., Miao, S., Liu, J.: Geore: A relation extraction dataset for chinese geometry problems. In: 35th Conference on Neural Information Processing Systems (NeurIPS 2021) Workshop on Math AI for Education (MATHAI4ED) (2021)

[30] Zeng, X., Liu, S.: Research on the application of knowledge mapping and knowledge structure construction based on adaptive learning model. Expert Systems with Applications **249**, 123400 (2024). https://doi.org/https://doi.org/10.1016/j.eswa.2024.123400, https://www.sciencedirect.com/science/article/pii/S0957417424002653

[31] Zhang, J., Li, Z.Z., Zhang, M.L., Yin, F., Liu, C.L., Moshfeghi, Y.: GeoEval: Benchmark for evaluating LLMs and multi-modal models on geometry problem-solving. In: Ku, L.W., Martins, A., Srikumar, V. (eds.) Findings of the Association for Computational Linguistics ACL 2024. pp. 1258–1276. Association for Computational Linguistics, Bangkok, Thailand and virtual meeting (Aug 2024), https://aclanthology.org/2024.findings-acl.73

[32] Zhang, J.Z., Chou, S.C., Gao, X.S.: Automated production of traditional proofs for theorems in euclidean geometry i. the hilbert intersection point theorems. Annals of Mathematics and Artificial Intelligence **13**(1-2), 109–137 (1995)

[33] Zhang, M.L., Yin, F., Hao, Y.H., Liu, C.L.: Plane geometry diagram parsing. In: Raedt, L.D. (ed.) Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22. pp. 1636–1643. International Joint Conferences on Artificial Intelligence Organization (7 2022). https://doi.org/10.24963/ijcai.2022/228, https://doi.org/10.24963/ijcai.2022/228, main Track

[34] Zhang, M.L., yin, F., Liu, C.L.: A multi-modal neural geometric solver with textual clauses parsed from diagram. In: Elkind, E. (ed.) Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI-23. pp. 3374–3382. International Joint Conferences on Artificial Intelligence Organization (8 2023). https://doi.org/10.24963/ijcai.2023/376, https://doi.org/10.24963/ijcai.2023/376, main Track

[35] Zhang, X., Zhu, N., He, Y., Zou, J., Huang, Q., Jin, X., Guo, Y., Mao, C., Li, Y., Zhu, Z., Yue, D., Zhu, F., Wang, Y., Huang, Y., Wang, R., Qin, C., Zeng, Z., Xie, S., Luo, X., Leng, T.: Formalgeo: An extensible formalized framework for olympiad geometric problem solving (2024)

[36] Zhang, X., Zhu, N., He, Y., Zou, J., Qin, C., Li, Y., Leng, T.: Fgeo-sss: A search-based symbolic solver for human-like automated geometric reasoning. Symmetry **16**(4) (2024). https://doi.org/10.3390/sym16040404, https://www.mdpi.com/2073-8994/16/4/404

[37] Zhang, X., Zhu, N., He, Y., Zou, J., Qin, C., Li, Y., Zeng, Z., Leng, T.: Fgeo-hypergnet: Geometry problem solving integrating formal symbolic system and hypergraph neural network (2024)

[38] Zou, J., Zhang, X., He, Y., Zhu, N., Leng, T.: Fgeo-drl: Deductive reasoning for geometric problems through deep reinforcement learning. Symmetry **16**(4) (2024). https://doi.org/10.3390/sym16040437, https://www.mdpi.com/2073-8994/16/4/437

Table 3: Detailed statement examples for defining a predicate.

name	item	content

IsMidpointOfLine(M,AB)	ee_check	Point(M)
		Line(AB)
		Collinear(AMB)

	fv_check	M,AB
	multi	M,BA
	extend	Equal(LengthOfLine(AM),LengthOfLine(MB))
LengthOfLine(AB)	ee_check	Line(AB)
	sym	ll
	multi	BA

A Details of GDL

FormalGeo formal languages are categorized into geometry definition language (GDL) and condition declaration language (CDL). The former is used to define relations, attributes, theorems, and other elements of a geometry formal system, while the latter is employed for declaring conditions and goals in geometric problems. This chapter introduces GDL. An example of CDL can be found in Appx. B.

GDL can be divided into predicate definition language and theorem definition language.

A.1 Predicate definition language

Predicate definition language is used to define geometric relations and geometric attributions. FormalGeo comprises 89 predicates, including 25 fundamental predicates (Tab. 4) built into the solver and 12 entities (Tab. 5), 31 entity relationships (Tab. 6), and 21 attributions (Tab. 7) defined using the predicate definition language.

The detailed statements for defining a predicate is as shown in the Tab. 3, including the predicate name and point variable declaration, validity check declaration, multiple representations, and automatic expansion. Additionally, when defining attributes, it also includes symbolic form declaration.

A.2 Theorem definition language

Theorems are defined using the geometry predicate logic, comprising two parts: premises and conclusions, as shown in the Tab. 8. FormalGeo defines 200 theorems.

B An example of annotated IMO geometry problem

Based on rigorous geometry formalization theory and a carefully developed symbolic solver, FormalGeo can represent, verify, and solve IMO-level geometry problems. Fig. 2 shows the original problem from the 2022 IMO Problem 4 and its formalized description.

The final solution of geometric problems can be represented as a hypergraph, where conditions serve as hypernodes and theorems as hyperedges, as shown in Fig. 3. For this hypergraph, we can easily apply rule-based methods to transform it into a human-like solution, as illustrated in Fig. 4. We can also use LLMs to convert it into a more readable solution, which we leave for future research.

Table 4: Predicates built into the solver.

id	type	name	examples
1	Construction	Shape	Shape(AB,BC,CA)
2	Construction	Collinear	Collinear(ABCD)
3	Construction	Cocircular	Cocircular(O,ABC)
4	BasicEntity	Point	Point(A)
5	BasicEntity	Line	Line(AB)
6	BasicEntity	Arc	Arc(OAB)
7	BasicEntity	Angle	Angle(ABC)
8	BasicEntity	Polygon	Polygon(ABCD)
9	BasicEntity	Circle	Circle(O)
10	Algebra	Equal	Equal(a,b)
11	Algebra	Equation	Equation(a-b)
12	Attribution	Free	Free(x)
13	Operation	Add	Equal(Add(a,b,c),1)
14	Operation	Sub	Equal(Sub(a,b),1)
15	Operation	Mul	Equal(Mul(a,b,c),1)
16	Operation	Div	Equal(Div(a,b),1)
17	Operation	Pow	Equal(Pow(a,b),1)
18	Operation	Mod	Equal(Mod(a,b),1)
19	Operation	Sqrt	Equal(Sqrt(a),1)
20	Operation	Sin	Equal(Sin(a),1/2)
21	Operation	Cos	Equal(Cos(a),1/2)
22	Operation	Tan	Equal(Tan(a),1)
23	Target	Value	Value(a)
24	Target	Equal	Equal(a,b)
25	Target	Relation	Relation(RightTriangle(ABC))

Table 5: Entities defined using the predicate definition language.

id	type	examples
26	Entity	RightTriangle(ABC)
27	Entity	IsoscelesTriangle(ABC)
28	Entity	IsoscelesRightTriangle(ABC)
29	Entity	EquilateralTriangle(ABC)
30	Entity	Kite(ABCD)
31	Entity	Parallelogram(ABCD)
32	Entity	Rhombus(ABCD)
33	Entity	Rectangle(ABCD)
34	Entity	Square(ABCD)
35	Entity	Trapezoid(ABCD)
36	Entity	IsoscelesTrapezoid(ABCD)
37	Entity	RightTrapezoid(ABCD)

Table 6: Relations defined using the predicate definition language.

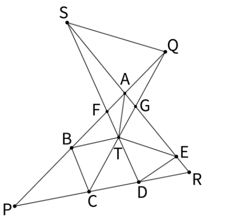
id	type	examples
38	Relation	IsMidpointOfLine(M,AB)
39	Relation	IsMidpointOfArc(M,OAB)
40	Relation	ParallelBetweenLine(AB,CD)
41	Relation	PerpendicularBetweenLine(AC,BC)
42	Relation	IsPerpendicularBisectorOfLine(AB,CD)
43	Relation	IsBisectorOfAngle(BD,ABC)
44	Relation	IsMedianOfTriangle(AD,ABC)
45	Relation	IsAltitudeOfTriangle(AD,ABC)
46	Relation	IsMidsegmentOfTriangle(DE,ABC)
47	Relation	IsCircumcenterOfTriangle(O,ABC)
48	Relation	IsIncenterOfTriangle(O,ABC)
49	Relation	IsCentroidOfTriangle(O,ABC)
50	Relation	IsOrthocenterOfTriangle(O,ABC)
51	Relation	CongruentBetweenTriangle(ABC,DEF)
52	Relation	MirrorCongruentBetweenTriangle(ABC,DEF)
53	Relation	SimilarBetweenTriangle(ABC,DEF)
54	Relation	MirrorSimilarBetweenTriangle(ABC,DEF)
55	Relation	IsAltitudeOfQuadrilateral(EF,ABCD)
56	Relation	IsMidsegmentOfQuadrilateral(EF,ABCD)
57	Relation	IsCircumcenterOfQuadrilateral(O,ABCD)
58	Relation	IsIncenterOfQuadrilateral(O,ABCD)
59	Relation	CongruentBetweenQuadrilateral(ABCD,EFGH)
60	Relation	MirrorCongruentBetweenQuadrilateral(ABCD,EFGH)
61	Relation	SimilarBetweenQuadrilateral(ABCD,EFGH)
62	Relation	MirrorSimilarBetweenQuadrilateral(ABCD,EFGH)
63	Relation	CongruentBetweenArc(OAB,OCD)
64	Relation	SimilarBetweenArc(OAB,OCD)
65	Relation	IsDiameterOfCircle(AB,O)
66	Relation	IsTangentOfCircle(PA,O)
67	Relation	IsCentreOfCircle(P,O)
68	Relation	ConcyclicBetweenPoints(A,B,C,D)

Table 7: Attributions defined using the predicate definition language.

id	type	examples
69	Attribution	LengthOfLine(AB)
70	Attribution	MeasureOfAngle(ABC)
71	Attribution	PerimeterOfTriangle(ABC)
72	Attribution	AreaOfTriangle(ABC)
73	Attribution	HeightOfTriangle(ABC)
74	Attribution	RatioOfSimilarTriangle(ABC)
75	Attribution	RatioOfMirrorSimilarTriangle(ABC)
76	Attribution	PerimeterOfQuadrilateral(ABCD)
77	Attribution	AreaOfQuadrilateral(ABCD)
78	Attribution	HeightOfQuadrilateral(ABCD)
79	Attribution	RatioOfSimilarQuadrilateral(ABCD)
80	Attribution	RatioOfMirrorSimilarQuadrilateral(ABCD)
81	Attribution	LengthOfArc(OAB)
82	Attribution	MeasureOfArc(OAB)
83	Attribution	RatioOfSimilarArc(OAB)
84	Attribution	RadiusOfCircle(O)
85	Attribution	DiameterOfCircle(O)
86	Attribution	PerimeterOfCircle(O)
87	Attribution	AreaOfCircle(O)
88	Attribution	PerimeterOfSector(OAB)
89	Attribution	AreaOfSector(OAB)

Table 8: Detailed statement examples for defining a theorem.

name	item	content
midpoint_of_line_judgment(M,AB)	premise	Collinear(AMB)& Equal(LengthOfLine(AM),LengthOfLine(MB))
	conclusion	IsMidpointOfLine(M,AB)
vertical_angle (AOC,BOD)	premise	Collinear(AOB)&Collinear(COD)& Angle(AOC)&Angle(BOD)
	conclusion	Equal(MeasureOfAngle(AOC),MeasureOfAngle(BOD))



Problem Text

English: As shown in the figure, BC=DE, TB=TD, TC=TE, and ∠ABT=∠TEA. Prove that points P, R, Q, and S are concyclic.

Chinese: 如图所示，BC=DE，TB=TD，TC=TE，∠ABT=∠TEA。求证P、R、Q、S四点共圆。

Text CDL

Equal(LengthOfLine(BC),LengthOfLine(DE))
Equal(LengthOfLine(TB),LengthOfLine(TD))
Equal(LengthOfLine(TC),LengthOfLine(TE))
Equal(MeasureOfAngle(ABT),MeasureOfAngle(TEA))

Goal CDL

Relation(ConcyclicBetweenPoints(P,R,Q,S))

Construction CDL

Shape(SA,AQ,QS)
Shape(SF,FA,AS)
Shape(QA,AG,GQ)
Shape(AF,FT,TA)
Shape(AT,TG,GA)
Shape(FB,BT,TF)
Shape(GT,TE,EG)
Shape(BP,PC,CB)
Shape(BC,CT,TB)
Shape(TC,CD,DT)
Shape(TD,DE,ET)
Shape(ED,DR,RE)
Collinear(SAGER)
Collinear(SFTD)
Collinear(PBFAQ)
Collinear(PCDR)
Collinear(CTGQ)

Theorem Sequences

congruent_triangle_judgment_sss(1,TBC,TDE)
congruent_triangle_property_angle_equal(1,TBC,TDE)
adjacent_complementary_angle(1,CTB,BTQ)
adjacent_complementary_angle(1,STE,ETD)
mirror_similar_triangle_judgment_aa(1,QBT,STE)
mirror_similar_triangle_property_line_ratio(1,QBT,STE)
mirror_similar_triangle_property_line_ratio(1,BTQ,EST)
concyclic_between_points_judgment_circular_power(1,STD,CTQ)
concyclic_between_points_property_angle_equal(1,C,D,Q,S)
mirror_similar_triangle_property_angle_equal(1,QBT,STE)
triangle_property_angle_sum(1,QPC)
adjacent_complementary_angle(1,PCQ,QCR)
angle_addition(1,QSA,ASF)
concyclic_between_points_judgment_sum_of_angles(2,P,R,Q,S)

Figure 2: An example of annotated geometry problem (2022 IMO Problem 4).

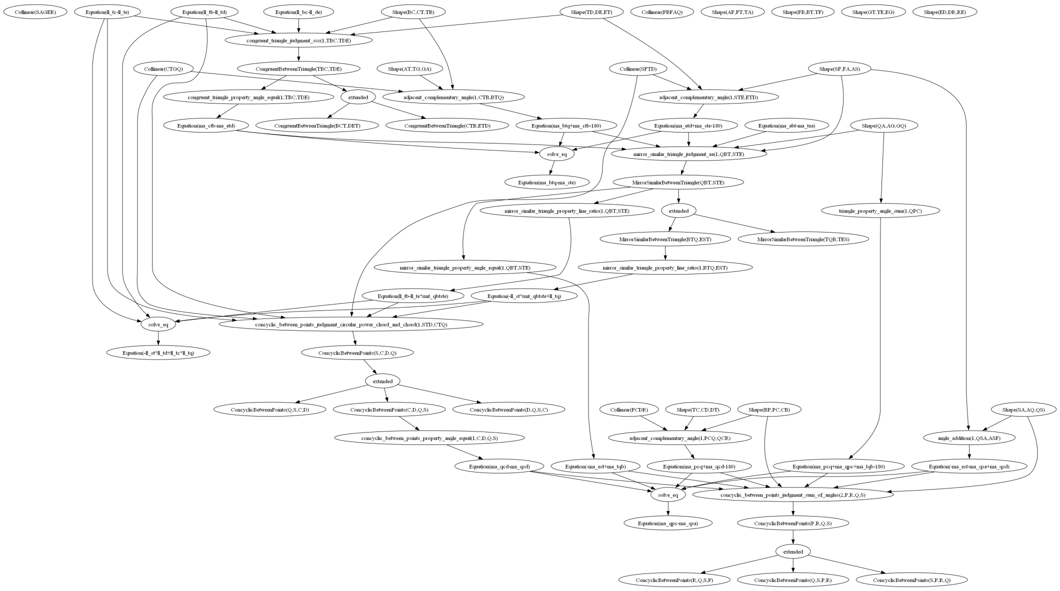
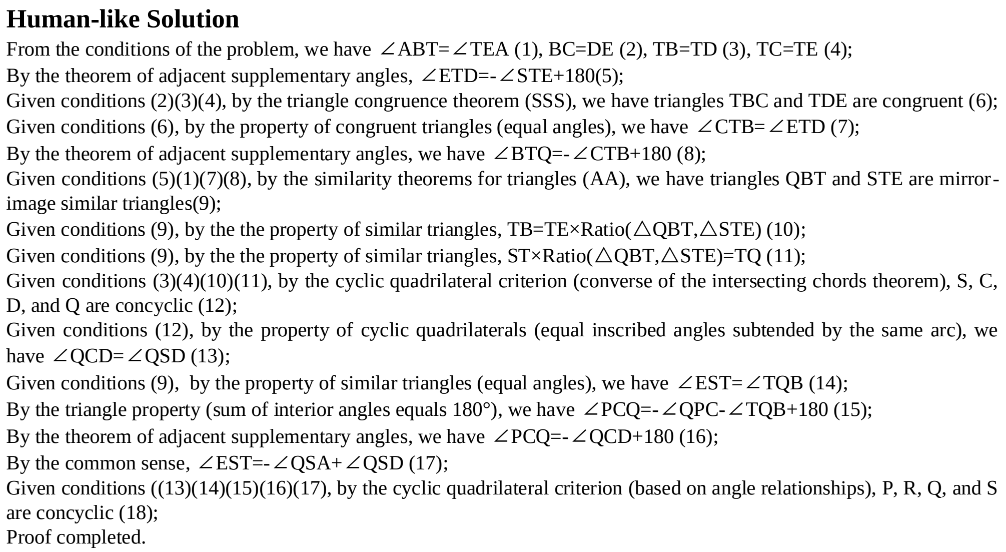

Collinear(SAGER)
Equation(ll_tc-ll_te)
Equation(ll_tb-ll_td)
Equation(ll_bc-ll_de)
Shape(BC,CT,TB)
Shape(TD,DE,ET)
Collinear(PBFAQ)
Shape(AF,FT,TA)
Shape(FB,BT,TF)
Shape(GT,TE,EG)
Shape(ED,DR,RE)
congruent_triangle_judgment_sss(1,TBC,TDE)
Collinear(CTGQ)
CongruentBetweenTriangle(TBC,TDE)
Shape(AT,TG,GA)
Collinear(SFTD)
Shape(SF,FA,AS)
congruent_triangle_property_angle_equal(1,TBC,TDE)
extended
adjacent_complementary_angle(1,CTB,BTQ)
adjacent_complementary_angle(1,STE,ETD)
Equation(ma_ctb-ma_etd)
CongruentBetweenTriangle(BCT,DET)
CongruentBetweenTriangle(CTB,ETD)
Equation(ma_btq+ma_ctb-180)
Equation(ma_etd+ma_ste-180)
Equation(ma_abt-ma_tea)
Shape(QA,AG,GQ)
solve_eq
mirror_similar_triangle_judgment_aa(1,QBT,STE)
Equation(ma_btq-ma_ste)
MirrorSimilarBetweenTriangle(QBT,STE)
mirror_similar_triangle_property_line_ratio(1,QBT,STE)
extended
triangle_property_angle_sum(1,QPC)
MirrorSimilarBetweenTriangle(BTQ,EST)
MirrorSimilarBetweenTriangle(TQB,TES)
mirror_similar_triangle_property_angle_equal(1,QBT,STE)
mirror_similar_triangle_property_line_ratio(1,BTQ,EST)
Equation(ll_tb-ll_te*rmt_qbtste)
Equation(-ll_st*rmt_qbtste+ll_tq)
solve_eq
concyclic_between_points_judgment_circular_power_chord_and_chord(1,STD,CTQ)
Equation(-ll_st*ll_td+ll_tc*ll_tq)
ConcyclicBetweenPoints(S,C,D,Q)
extended
ConcyclicBetweenPoints(Q,S,C,D)
ConcyclicBetweenPoints(C,D,Q,S)
ConcyclicBetweenPoints(D,Q,S,C)
Collinear(PCDR)
Shape(TC,CD,DT)
Shape(BP,PC,CB)
Shape(SA,AQ,QS)
concyclic_between_points_property_angle_equal(1,C,D,Q,S)
adjacent_complementary_angle(1,PCQ,QCR)
angle_addition(1,QSA,ASF)
Equation(ma_qcd-ma_qsd)
Equation(-ma_est+ma_tqb)
Equation(ma_pcq+ma_qcd-180)
Equation(ma_pcq+ma_qpc+ma_tqb-180)
Equation(-ma_est-ma_qsa+ma_qsd)
solve_eq
concyclic_between_points_judgment_sum_of_angles(2,P,R,Q,S)
Equation(ma_qpc-ma_qsa)
ConcyclicBetweenPoints(P,R,Q,S)
extended
ConcyclicBetweenPoints(R,Q,S,P)
ConcyclicBetweenPoints(Q,S,P,R)
ConcyclicBetweenPoints(S,P,R,Q)

Figure 3: Solution hypergraph generated by symbolic solver FGPS [36] (2022 IMO Problem 4).



Human-like Solution

From the conditions of the problem, we have ∠ABT=∠TEA (1), BC=DE (2), TB=TD (3), TC=TE (4);
By the theorem of adjacent supplementary angles, ∠ETD=-∠STE+180(5);
Given conditions (2)(3)(4), by the triangle congruence theorem (SSS), we have triangles TBC and TDE are congruent (6);
Given conditions (6), by the property of congruent triangles (equal angles), we have ∠CTB=∠ETD (7);
By the theorem of adjacent supplementary angles, we have ∠BTQ=-∠CTB+180 (8);
Given conditions (5)(1)(7)(8), by the similarity theorems for triangles (AA), we have triangles QBT and STE are mirror-image similar triangles(9);
Given conditions (9), by the the property of similar triangles, TB=TE×Ratio(△QBT,△STE) (10);
Given conditions (9), by the the property of similar triangles, ST×Ratio(△QBT,△STE)=TQ (11);
Given conditions (3)(4)(10)(11), by the cyclic quadrilateral criterion (converse of the intersecting chords theorem), S, C, D, and Q are concyclic (12);
Given conditions (12), by the property of cyclic quadrilaterals (equal inscribed angles subtended by the same arc), we have ∠QCD=∠QSD (13);
Given conditions (9), by the property of similar triangles (equal angles), we have ∠EST=∠TQB (14);
By the triangle property (sum of interior angles equals 180°), we have ∠PCQ=-∠QPC-∠TQB+180 (15);
By the theorem of adjacent supplementary angles, we have ∠PCQ=-∠QCD+180 (16);
By the common sense, ∠EST=-∠QSA+∠QSD (17);
Given conditions ((13)(14)(15)(16)(17), by the cyclic quadrilateral criterion (based on angle relationships), P, R, Q, and S are concyclic (18);
Proof completed.

Figure 4: Human-like Solution (2022 IMO Problem 4).

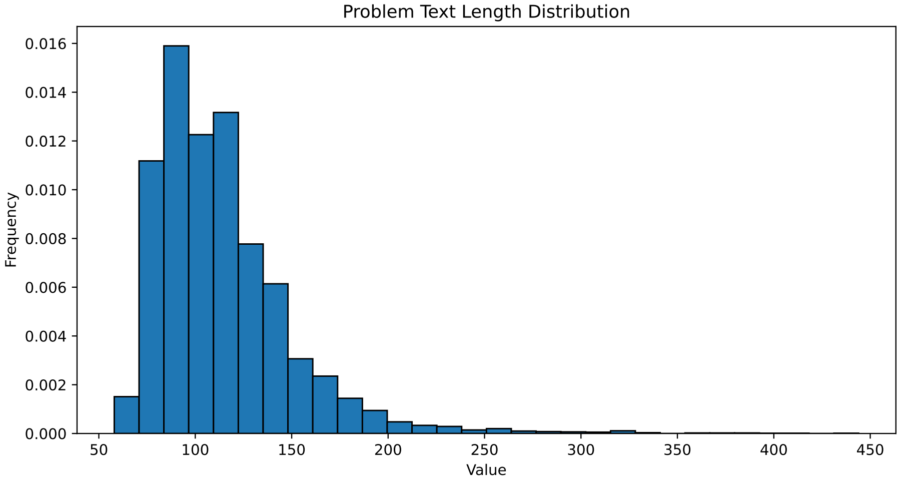


Figure 5: Problem Text Length Distribution.

C Statistics details

We collect information on the text length distribution, theorem length distribution, and predicate frequency in the formalgeo7k dataset.

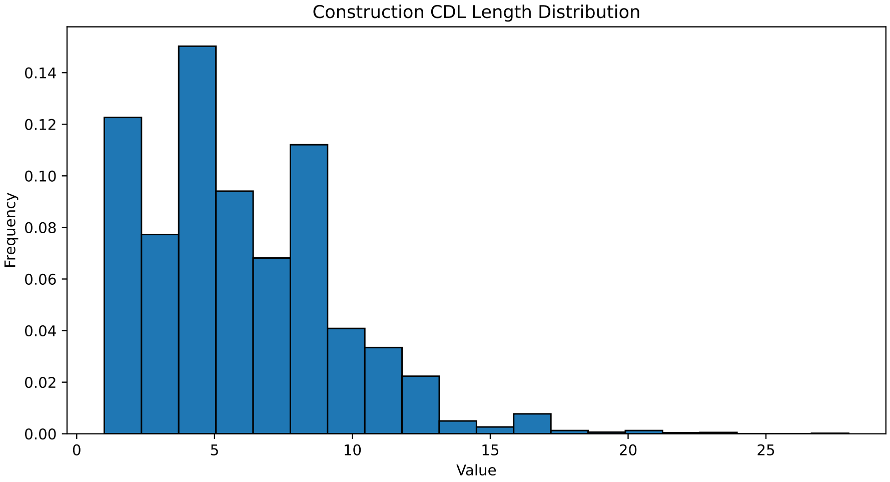


Figure 6: Construction CDL Length Distribution.

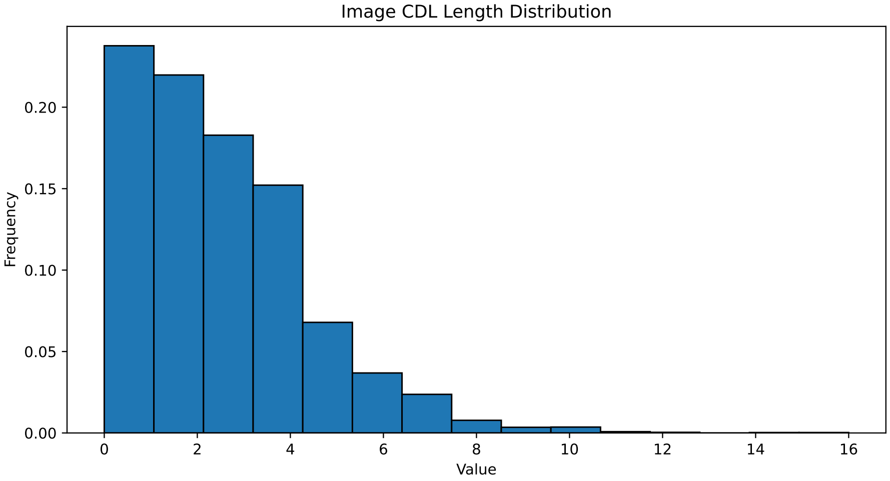


Figure 7: Image CDL Length Distribution.

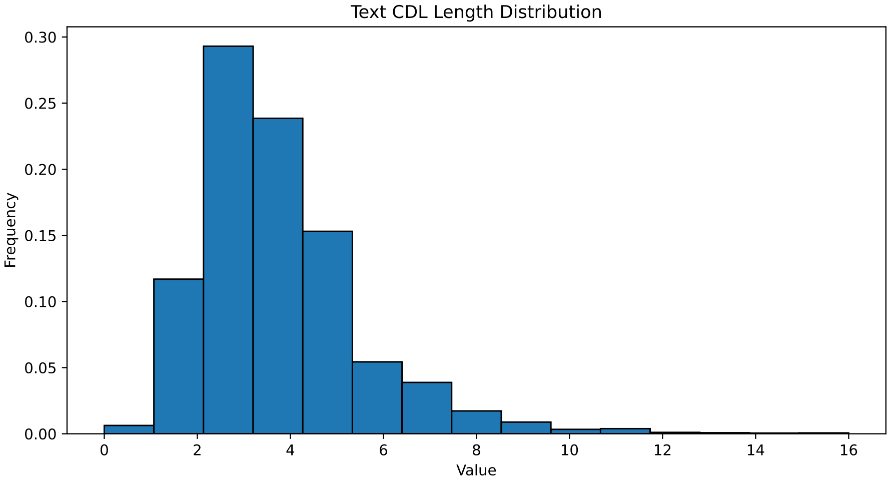


Figure 8: Text CDL Length Distribution.

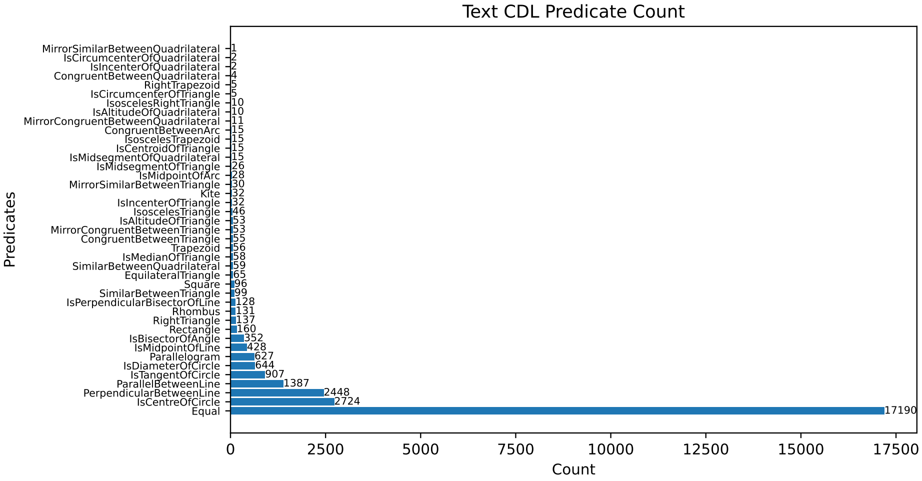


Figure 9: Text CDL Predicate Count.

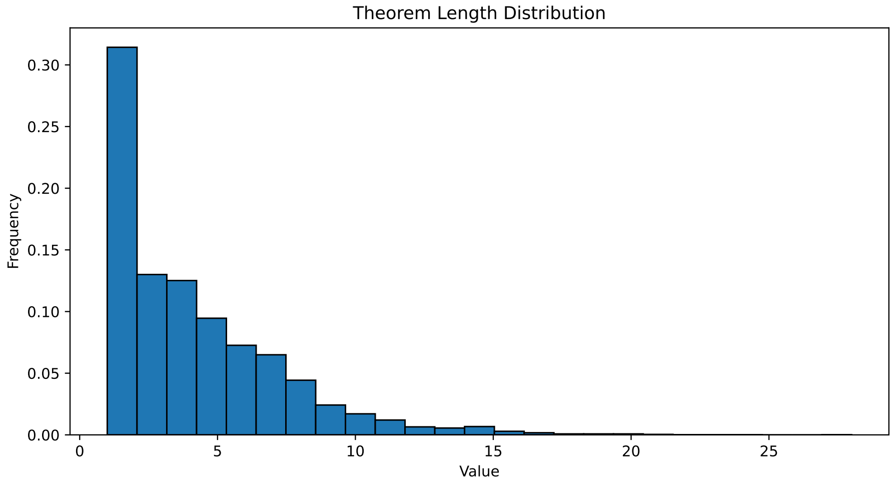


Figure 10: Theorem Length Distribution.

D Dataset download and Usage

You can use pip to download and use our formalgeo7k dataset.

```
$ conda create -n <your_env_name> python=3.10
$ conda activate <your_env_name>
$ pip install formalgeo

>>> from formalgeo.data import download_dataset, DatasetLoader
>>> from formalgeo.solver import Interactor
>>> from formalgeo.parse import parse_theorem_seqs

>>> download_dataset(dataset_name="formalgeo7k_v2",
                     datasets_path="your_datasets_path")
>>> dl = DatasetLoader(dataset_name="formalgeo7k_v2",
                       datasets_path="your_datasets_path")

>>> solver = Interactor(dl.predicate_GDL, dl.theorem_GDL)
>>> problem_CDL = dl.get_problem(pid=1)
>>> solver.load_problem(problem_CDL)
>>> for t_name, t_branch, t_para in parse_theorem_seqs(problem_CDL["theorem_seqs"]):
        solver.apply_theorem(t_name, t_branch, t_para)
>>> solver.problem.check_goal()

>>> from formalgeo.tools import show_solution
>>> show_solution(solver.problem)
```