Challenges in teaching logic programming

V.Sekovanić*, S. Lovrenčić*
* Faculty of Organization and Informatics, Varaždin, Croatia
vlatka.sekovanic@foi.unizg.hr
sandra.lovrencic@foi.unizg.hr

***Abstract* - Learning Prolog is a challenge on many levels. In relation to other programming languages, it carries additional weight in understanding due to its declarative nature, which is significantly different in relation to procedural languages. Prolog is an extremely important programming language because it is the foundation of artificial intelligence, knowledge-based systems, and other modern systems. The question arises of how to motivate students in the best possible way and make it easier for them to understand Prolog. The literature points to problems in learning the language, but also to possible approaches to quality teaching. This paper describes five-year research on the quality of teaching based on the feedback from students at the end of the semester. The research is an analysis of contact teaching and online teaching in the last part. The results of the research show observed problems in understanding the SWI-Prolog, but also the efforts of teachers to try to alleviate the perceived problems of students in the best possible way. In this way, efforts were made to raise the quality of teaching from year to year, thus making it easier for students to understand and adopt the material related to the basics of the SWI-Prolog.**

Keywords - logic programming; SWI-Prolog; teaching programming

I. INTRODUCTION

Alain Colmerauer and Philippe Roussel developed logic programming language Prolog in 1972 from a project focused on processing natural languages [1]. As a declarative programming language based on first-order predicate logic [2], it is particularly suitable for solving problems that can be described by objects and relationships between them [3]. Prolog is the basis of artificial intelligence [1,3,4], computational linguistics, knowledge-based systems, and other modern systems [5,6,7]. For example, NASA (as one of the leading institutions in the application of formal methods) uses Prolog [8,9,10], parts of IBM's Watson supercomputer are programmed in Prolog [11], as well as a database of the highly successful Human Genome Project [12,13]. Therefore, no system from the above-mentioned fields works today without Prolog, which means that knowledge of this programming language is very important. The question is why Prolog is not popular and more used programming language compared to other programming paradigms? The purpose of this paper is to examine common problems in understanding the basic concepts of Prolog, as well as teaching approaches, and to analyse results of student survey conducted during five year period of teaching Prolog, which will offer guidelines for improving materials and delivery of revised course in the future.

This paper is organized as follows. Chapter 2 reviews previous research on the identified problems in adopting the basic concepts of the Prolog and approaches in teaching. The results of the student survey during five year period are then presented, focusing on the problems observed by students in mastering the material (Chapter 3) and understanding the practical application of the Prolog (Chapter 4). Chapter 5 concludes the paper.

II. BACKGROUND

It is often emphasized that programming is a fundamental digital skill necessary for today's and future careers. Looking at the curricula of European universities, object-oriented paradigms are most prevalent [14], further placing the declarative paradigm in an unequal position. Common problems in teaching programming include [15] a variety of prior knowledge of students, fear of programming, problems with programming language syntax, motivation, learning style, etc.

Another problem with the Prolog is the declarative paradigm, which implies a completely different way of thinking than the imperative paradigm. Imperative pardigm is focused on giving instructions how the program should perform, whereas declarative paradigm is oriented on what should be done within the program, without specifying the steps of the procedure. In other words, the programmer develops the program based on a set of facts and rules in the knowledge base describing the formal specification of the problem, while declarative programming lanugage, such as Prolog determines the algorithm during execution by deriving existing facts and rules, thus providing an answer to the query [12,16,17].

A very simple example in Prolog is the program that states facts about students, courses and which student passed which course. The only rule in the program defines that if a student passed the course, then this student knows the material of this course. How the program will operate to determne what material student knows doesn't have to be specified. According to facts and the rule in the knowledge base, it can be asked what an individual student knows:

```
student (ana).
course (logic_programming).
passed (ana, logic_programming).
knows (Student, Course) :- passed (Student, Course).

? – knows (ana, Course).
```

Curriculums often start with imperative paradigm and students later encounter different approach to solve problems with declarative paradigm [18]. Those that have good knowledge in former woud likely have more problems in expressing facts and rules and could try to write instructions to the program (in above example, how to determine what a student knows). There are also different ways of expressing other features of Prolog, such as cut, negation and recursion [3,5] that also make the adjustment to a different paradigm difficult.

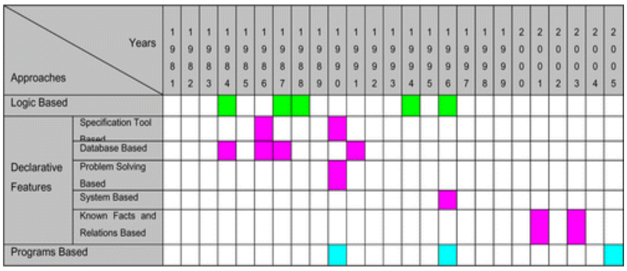
Teachers (researchers/authors) agree on one thing - adopting the basic concepts of the Prolog and the declarative paradigm itself is a problem for students [12,16,17,19,20,21,22]. Therefore, their approach to teaching is to identify the problems and develop new and more interesting teaching approaches to ultimately facilitate student adoption of the Prolog. Three basic pedagogical goals in teaching programming languages are the acquisition of language syntax, the development of program design skills, and creative thinking [23].

To better understand the problem, Yang, S., & Joy, M. [22] conducted an interesting study. In their work, they included and examined the available textbooks used for teaching Prolog from 1980 to 2005. The results of their study include the identification of the most common approaches to teaching Prolog (their characteristics, advantages and disadvantages) and the attitudes from the students' point of view. Based on the textbooks studied (see Fig. 1), they identified three approaches to teaching Prolog over the 25 years [22]:

- logic-based - deal with abstract theories of mathematical logic and/or logic programming; students find this approach most difficult.
- declarative features based - require hands-on experience, and the Prolog mindset is developed concretely. Implies specification tool-based (disliked by students), database-based (students find this approach the most appropriate), problem-solving-based (applies to describing and solving problems; appropriate), system-based (students find it provides a deeper understanding of Prolog), known facts and relationships-based (the most appropriate approach; concrete and applicable to solving real-world problems).
- program-based - requires both theoretical and practical knowledge. This approach introduces students to the basic ideas of Prolog by showing them examples of programs where students can quickly see the components and structure of the program, etc.

The authors emphasize that the most appropriate approach includes both concrete and abstract components, especially approaches based on the initial emphasis on the declarative properties of the Prolog. At the same time, it is necessary to take into account the different preferences of the learning approach. Therefore, it is suggested to use a blended learning strategy to accommodate different learning styles.

Callear, D. [12] in his paper also analyzes the literature on Prolog and points out the problem of poor structure of the material. Namely, the author highlights the most important conceptual steps in learning Prolog, which, when presented in the wrong order (as in the cited literature), interfere with the overall understanding of the programming language. Some of the topics that are difficult for students to understand relate to the use of variables, rules, backtracking, recursion, and lists. He proposed teaching at appropriate pace with easier topics first, teaching one topic at a time, returning to topics and frequent exercises and conducted a research that showed a positive feedback about proposed structured method from students.



Approaches	Years	1981	1982	1983	1984	1985	1986	1987	1988	1989	1990	1991	1992	1993	1994	1995	1996	1997	1998	1999	2000	2001	2002	2003	2004	2005
Logic Based					🟩			🟩	🟩						🟩		🟩									
Declarative Features	Specification Tool Based						🟪				🟪															
	Database Based				🟪		🟪	🟪				🟪														
	Problem Solving Based										🟪															
	System Based																🟪									
	Known Facts and Relations Based																					🟪		🟪		
Programs Based											🟦						🟦									🟦

Figure 1. Prolog Teaching Trends by Yang, S., & Joy, M. (2007)

Motivation is the main factor that can influence the positive results in learning the Prolog, which is often emphasized in the papers. First and foremost, the teacher is the one who must create a motivating environment for knowledge transfer. Motivation includes applying the Prolog to interesting and advanced applications in real-world problems [22], interactive environments [24, 25], presenting lessons within intelligent tutoring systems [16,26,27], using expert system shells and application [28], but also hints about the value of logical programming skills in the profession [23]. Therefore, it is important not only to invest in high-quality (innovative) teaching processes but also to monitor (measure) student feedback. New high-frequency, automated algorithms for data collection and analysis could offer new insights into complex learning processes [29].

III. WHAT IS THE HARDEST TO LEARN?

Prolog was taught at second year of undergraduate university study during programming exercises at the course Introduction to Formal Methods at the Faculty of Organization and Informatics. The programming environment used was SWI-Prolog, which is free and widely used. The last year it was taught in full scope was 2020/2021. From academic year 2022/2023 aforementioned couse will be taught as a revised course Introduction to Knowledge Modeling. Therefore, it was important to analyse various aspects of the course delivery with the goal to adjust both theoretical and practical elements.

To improve the quality of teaching, a student surveys were already regularly conducted at the end of each semester in the academic years 2016/2017-2020/2021. In addition to basic student demographics, the survey collected information related to - understanding the material, optimal proportionality of material and class time, learning preferences, the optimal number of examples, problems in mastering the material, recognizing the practical application of the Prolog, and comments and ideas. As shown in Table 1, a total of 864 students enrolled in the course during the specified period, while 675 students completed the survey. A much weaker response to the survey was obtained in the academic year 2020/2021 when classes were held online due to the COVID-19 pandemic.

In this paper, the focus is on the results examining the problems of mastering the material (Question 14) and identifying the areas of the practical application of the Prolog (Question 15). The perception of problems students face in learning Prolog is very important, because programming paradigm differ from those they are familiar with. Students had to rank problems they face when learning Prolog on a five point Likert scale, where one was designating the biggest problem, and five the smallest. Problems were: syntax, logic/semantics, reasoning procedure, declarative paradigm, and examples used in class.

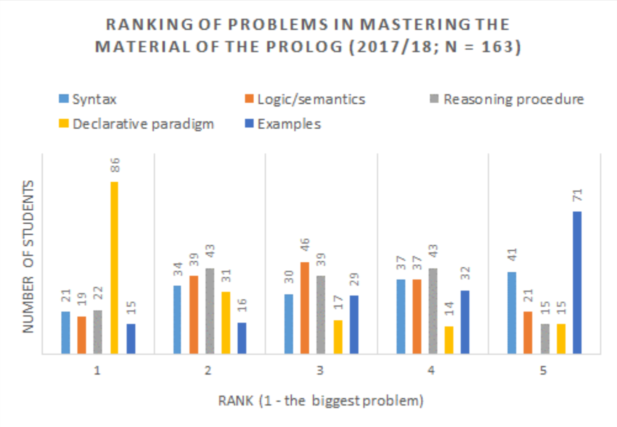


Figure 2. Ranking of problems in mastering the material of the Prolog (the academic year 2017/2018; N = 163)

Fig. 2 shows the results of the ranking of problems for academic year 2017/2018. This year was chosen because of the largest number of students surveyed within the years observed (N = 163), although the results for other years are similarly spaced. This means that the biggest problem for students is the concept of the declarative paradigm, while the smallest problems are examples. On the other hand, Fig. 3 shows a more detailed structure in terms of the biggest problem in mastering the Prolog material (looking at the whole five-year period; N = 616). The biggest problem during all observed years for students is the declarative paradigm (as confirmed by the observed research through a review of the literature). It is followed by logic/semantics, reasoning procedures, language syntax, while examples[1] are mostly ranked as the biggest problem with a relatively small number of students. In the online year (2020/2021), the structure is slightly different. Students were also allowed to express their opinions, describe their experience of learning Prolog through the disadvantages and advantages of this teaching approach, and suggest improvements (point out problems) that can improve the teaching in next academic year.

TABLE I. ANNUAL NUMBER OF ENROLLED AND SURVEYED STUDENTS

	Number of enrolled students	Number of completed surveys	Question 14	Question 15	Percentage of completed surveys
2016	182	154	128	138	84,6
2017	205	177	163	169	86,3
2018	159	126	123	126	79,2
2019	165	144	134	144	87,3
2020	153	74	68	49	48,4
Total	**864**	**675**	**616**	**626**	**78,1**

To improve the quality of teaching and better understanding of Prolog, we tried to influence the causes of the identified problems. The main example of the treasure hunt was designed in the Escape Room to make it as interesting as possible for the students while learning the basics of the Prolog. In other examples, students had the opportunity to work independently or in teams to design and write a stand-alone Prolog program for extra credits. Since the declarative paradigm differs from the others in the way it solves problems, students were encouraged to use the same kind of logical thinking when creating programs as they do when solving problems in everyday life. During classes, we slowed down the parts of the material that were critical to understanding Prolog and emphasized the importance of each segment (syntax, reasoning procedures, etc.). Student work and comprehension were systematically monitored. Interactive e-books with knowledge tests within the H5P module in Learning Management System (LMS) Moodle were developed specifically for online instruction (2020/2021). Because they are used for problems within the SWI-Prolog editor, work was also possible in a more stable online environment SWISH. Additional teacher consultations and demonstrations were available to students (when possible). Unfortunately, despite all the motivation and efforts of teachers, Prolog was still largely declared as an unpopular programming language, as

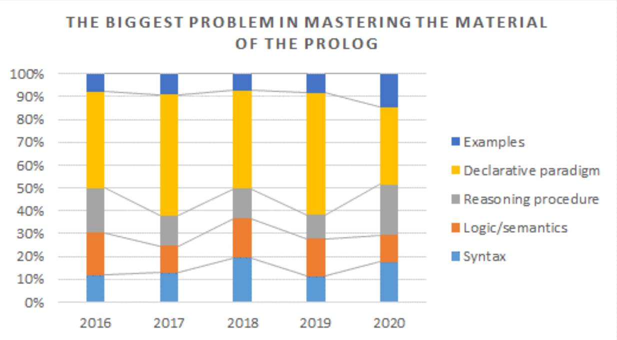


Figure 3. Overview of the structure of the biggest problem in mastering the material of the Prolog in the total amount (2016-2020; N = 616)

[1] The lessons were based on the main example of the Escape Room, through which the basic concepts of the Prolog are introduced during the programming exercises. This was followed by examples of the family tree for the purpose of gradual independent application of what has been learned, illustrative examples of problem tasks as part of teamwork, and an optional example of creating an independent program in the Prolog for additional credits and whose topic was determined by the student.

shown by the results in the next chapter - recognition of the fields of the practical application of Prolog.

IV. STUDENT PERCEPTION OF THE PRACTICAL APPLICATION OF THE PROLOG

Students were also asked the following question: „Based on your previous education and experience, in what areas do you see the practical application of the Prolog / declarative paradigm alone or in combination with other program paradigms?“ As can be seen in Fig. 4, in the pooled results for the entire five-year period observed, 78% of students responded that they do not see any practical application of Prolog. The remaining 22% of students see the practical application in the field of artificial intelligence (7%), computer games (5%), databases (4%), expert systems (1%), and other applications (5%), such as Prolog connection with other programming languages, military or business systems, virtual reality (VR), etc. The results are similar when considered individually within the observed years.

Why is Prolog not a popular programming language? Some of the students' answers are that they do not have the mindset necessary for the declarative paradigm, that they do not see the point of applying complicated and demanding logic/syntax when these examples can be solved much easier and simpler in other programming languages (C ++, Python). One of the answers is that the job market does not require programming skills in Prolog, so there is no additional motivation to learn Prolog. Of course, the teoretical part of the course also have influence on understanding practical application of Prolog and logic programming languages in general. Teachers usually have motivational examples at the introductory lectures but successfull examples of how Prolog is used in practice should obviously be more emphasized.

Students experience the practical application of Prolog mainly when they create an independent task and especially when they write a thesis, in which they have a task to combine Prolog with some other programming language (e.g. Prolog and Python via the programming module Pyswip) and thus realize the advantages of such symbiosis (shorter code, greater security, etc.). Based on the students' responses, the conclusion is that perhaps the focus in the examples should be on combining Prolog with other programming paradigms, some of which could certainly impact motivation.

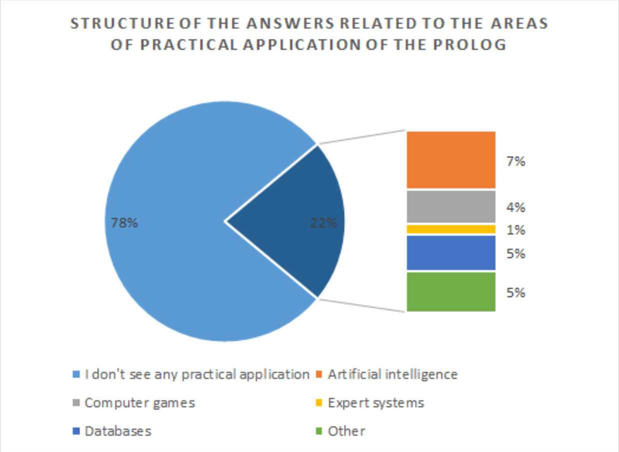


Figure 4. Overview of the structure of the answers related to the areas of the practical application of the Prolog (2016-2020; N = 626)

V. CONCLUSION

On one hand, the Prolog is necessary (as the basis of all systems using artificial intelligence); on the other hand, it is unpopular (compared to other program paradigms). The results of the conducted survey not only reveal problems in mastering the basic concepts of the Prolog and its general unpopularity but thus also confirm the observed problems of other studies described in the literature. It is important to understand the aspects that cause problems in mastering the basics of the Prolog. Understanding the challenges allows for a more creative and innovative approach to teaching (e.g., Platform for Teaching Logic Programming Using Virtual Worlds [30]). We also need to be aware that teacher motivation and effort will not be enough if students are not positively engaged. Thus, it is complex thinking that involves understanding problems and eliminating them by applying possible teaching approaches, appropriately structuring teaching content, considering different learning styles, systematically monitoring the results of learning analysis, and constantly innovating improvements to promote motivation.

To fully understand problems students face when learning logic programming, further analysis of other survey answers will be made. Also, the data about student activity and grades were collected for all years at LMS Moodle. This enables analytics of selected data that can give more answers of students' learning process and connection to learning outcomes. Information obtained from further analysis will be used to adjust materials and delivery of revised course from academic year 2022/2023 onward and to implement apropriate learning analytics for continuous monitoring and improving of students' learning process and results.

ACKNOWLEDGMENT

This work has been fully supported by the Croatian Science Foundation under the project IP-2020-02-5071.

REFERENCES

[1] A. Colmerauer and P. Roussel, „The birth of Prolog“, in History of programming languages, II pp. 331-367, 1996.

[2] P. Brna, A. Bundy, T. Dodd, et al., „Prolog programming techniques“, Instr Sci 20, pp. 111–133, 1991.

[3] I. Bratko, „Prolog Programming for Artificial Intelligence“, 4th Edition, Pearson Education, 2012.

[4] W. Ertel, „Introduction to artificial intelligence“, Springer, 2018.

[5] M. M. Gooley, and B. W. Wah, „System architectures for Prolog execution“, Structuring Expert Systems Domain Design and Development, Edited by J. Liebowitz and DA de Salvo, Prentice-Hall Inc., New Jersey, 171-213, 1989.

[6] A. Lyall, P. Hammond, D. Brough and D. Glover, „BIOLOG-a DNA sequence analysis system in Prolog“, Nucleic Acids Research, Volume 12, Issue 1Part2, pp 633–642, 11 January 1984.

[7] C. Mungall, „Experiences using logic programming in bioinformatics“, in International Conference on Logic Programming, pp. 1-2, Springer Berlin Heidelberg, July 2009.

[8] National Aeronautics and Space Administration NASA – Clarissa, https://ti.arc.nasa.gov/tech/cas/user-centered-technologies/clarissa/

[9] National Aeronautics and Space Administration NASA - Formal Methods Symposium, https://shemesh.larc.nasa.gov/nfm2021/

[10] National Aeronautics and Space Administration NASA Langley Formal Methods, https://shemesh.larc.nasa.gov/fm/fm-what.html

[11] A. Lally and P. Fodor, „Natural Language Processing with Prolog in the IBM Watson System“, The Association for Logic Programming (ALP) Newsletter, 9, 2011.

[12] D. Callear, „Teaching Programming: Some Lessons from Prolog“, in Annual Conference on Teaching and Learning, 2000, https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.176.7535&rep=rep1&type=pdf

[13] National Human Genome Research Institute, https://www.genome.gov/human-genome-project

[14] V. Aleksić and M. Ivanović, „Introductory programming subject in European higher education“, Informatics in Education, 15(2), pp. 163-182, 2016.

[15] D. Radošević, T. Orehovački and A. Lovrenčić, „New approaches and tools in teaching programming“, in Central European Conference on Information and Intelligent Systems, pp. 49-57, September 2009.

[16] C. K., Looi, "Automatic debugging of Prolog programs in a Prolog Intelligent Tutoring System", Instr Sci 20, pp. 215–263, 1991.

[17] A. Stathaki, H. Kondylakis, E. Marakakis and M. Kalogerakis, „i-Prolog: a web-based intelligent tutoring system for learning Prolog“, in Interactivity, Game Creation, Design, Learning, and Innovation, pp. 337-346, Springer, Cham, 2017.

[18] S. A. Rebelsky, P. B. Henderson, A. N. Kumar and F. N. Springsteel, "Why I do declare! declarative programming in the undergraduate curriculum.", in SIGCSE technical symposium on Computer Science Education, pp. 398-399, February 2001.

[19] H. Coelho and J. C. Cotta, „Prolog by example: how to learn, teach and use it“, Springer-Verlag Berlin Heidelberg, 1988.

[20] A. E. Fluck, „Lessons from discarded computer architectures“, in IFIP International Conference on the History of Computing, pp. 198-205, Springer Berlin Heidelberg, September 2010.

[21] M.W. Van Someren, „What's wrong? Understanding beginners' problems with Prolog“, Instr Sci 19, pp. 257–282, 1990.

[22] S. Yang and M. Joy, „Approaches for Learning Prolog Programming“, Innovation in Teaching and Learning in Information and Computer Sciences, 6:4, pp. 88-107, 2007.

[23] S. Mohorovicic and V. Strcic, „An overview of computer programming teaching methods“, in Central European Conference on Information and Intelligent Systems, pp. 47-52, September 2011.

[24] M. Jebe, S. Münker, J. Gnyp and W. Petersen, „eLearning-Plattform zur Unterstützung des Erlernens des logischen Programmierens“, in DELFI 2021, pp. 351-352, September 2021.

[25] J. Wielemaker, T. Schrijvers, M. Triska and T. Lager, „Swi-Prolog“, Theory and Practice of Logic Programming, 12(1-2), pp. 67-96, 2012.

[26] G. J. Nalepa and I. Wojnicki, "Concept of an Interactive Web Portal for Teaching Prolog“, in FLAIRS Conference, pp. 240-244, 2008.

[27] D. I. Popov and O. Y. Lazareva, „A Knowledge Testing Production Model Based on a Cognitive Map for SWI-Prolog Applications“, International Journal of Emerging Technologies in Learning (iJET), 10(4), 2015.

[28] D. Merritt, "Building expert systems in Prolog", Springer, 1989.

[29] P. Blikstein, M. Worsley, C. Piech, M. Sahami, S. Cooper and D. Koller, „Programming pluralism: Using learning analytics to detect patterns in the learning of computer programming“, Journal of the Learning Sciences, 23(4), pp. 561-599, 2014.

[30] S. Vosinakis, P. Koutsabasis and G. Anastassakis, „A platform for teaching logic programming using virtual worlds“, in 2014 IEEE 14th International Conference on Advanced Learning Technologies, pp. 657-661, IEEE, July 2014.