Journal of Computer Science

Review

Evolution of Artificial Intelligence Programming Languages - a Systematic Literature Review

[1,2,3]Emmanuel Adetiba, [1]Temitope M. Adeyemi-Kayode, [1]Adekunle A. Akinrinmade, [1]Funmilayo S. Moninuola, [1]Oladipupo O. Akintade, [1,2]Joke A. Badejo, [4,5]Obiseye O. Obiyemi, [4]Surendra Thakur and [6]Abdultaofeek Abayomi

[1]Department of Electrical and Information Engineering, College of Engineering, Covenant University, Ota, Nigeria
[2]Covenant Applied Informatics and Communications African Center of Excellence, Covenant University, Ota, Nigeria
[3]HRA, Institute for Systems Science, Durban University of Technology, P.O. Box 1334, Durban, South Africa
[4]KZN e-Skills CoLab, Durban University of Technology, Durban, South Africa
[5]Department of Electrical and Electronic Engineering, Osun State University, Osogbo, Osun State, Nigeria
[6]Department of Information and Communication Technology, Mangosuthu University of Technology, P.O. Box 12363 Jacobs, 4026 Durban, South Africa

Article history
Received: 12-02-2021
Revised: 07-07-2021
Accepted: 19-07-2021

Corresponding Author:
Emmanuel Adetiba
Department of Electrical and Information Engineering, College of Engineering, Covenant University, Ota, Nigeria
Email: emmanuel.adetiba@covenantuniversity.edu.ng

Abstract: Artificial Intelligence (AI) has received significant attention in recent years. It is being adopted to provide solutions to medicine, engineering, education, government and several other domains. To analyze the state-of-the-art of research in AI, we present a systematic literature review focusing on the Evolution of AI programming languages. Our search returned 7,604 documents; after reviewing these documents, 78 which were relevant for this study were retained. Our research revealed that the prevalence of AI programming language by volume of publications had experienced peaks and valleys between 1963 and 2018; however, between 2015 to 2020, related publications have been experiencing peaks. During the review period, the PROLOG programming language received the most attention in about 49% of publications; this was followed by LISP, which received almost 22%. The remaining attention was shared between Logic and Object-Oriented Programming (LOOP), ARCHLOG, Epistemic Ontology Language with Constraints (EOLC), Python, C++, ADA and JAVA. However, the predominant AI programming language in recent AI software is C/C++, which takes 70% of the modern AI libraries analyzed in this study. Python is used in 60% of the modern AI libraries analyzed. Their prevalence is as a result of their speed, portability and ease of coding, making them effective in developing trending AI libraries such as TensorFlow and Keras.

Keywords: Artificial Intelligence, Programming Language, Python, AI, LISP, PROLOG, JAVA, C++, EOLC, ADA

Introduction

Artificial Intelligence (AI) is concerned with intelligent behaviors in artifacts such as perception, reasoning, learning, communicating and acting in a complex environment. AI involves the science and engineering of machines that possess the listed characteristics, which humans can do, better and faster. The physical symbol system hypothesis states that it has the necessary and sufficient means for general intelligent action. A physical symbol system is a machine like a digital computer capable of manipulating symbolic data-adding numbers, rearranging lists of symbols and replacing some symbols with others (Nilsson, 2010).

Traditionally, computing is used for performing mechanical computations using fixed procedures. Unfortunately, this approach implies complex problems would be more difficult to solve. Another shortcoming is that computers so programmed would have difficulties understanding and adapting to new situations as humans do. AI is different from this traditional approach in that it requires machines to think and tackle such complex assignments. AI was formally coined by John McCarthy in a workshop conducted by IBM at Dartmouth College in 1956 (Nilsson and Nilsson,1998).

When digital computers were first developed in the 1940 s and 1950 s, researchers wrote several programs; these programs could play chess, checkers and prove theorems. In the 1960 s and 1970 s, AI explored various ways to represent problems by developing different search techniques and general heuristics. These enabled the development of

programs used to solve algebraic word problems and symbolic integration. In the 1970 s and 1980 s, as a result of more powerful systems, AI programs were used to build expert systems and by 1997, an IBM program named DEEP BLUE defeated the world chess champion, Garry Kasparov. Interest in AI sagged in the late 1950s. It however, resumed with vigor in the 1980s. Networks of non-linear elements with adjustable-strength interconnections are now recognized as an important class of non-linear modeling tools. AI programming languages are languages capable of implementation of logic (Clark and Gregory, 1986), vastly useful in building expert systems (Hawkinson, 1986) and with features for natural language processing (Weeks and Berghel, 1986). There are many AI programming languages. Some are List Processor (LISP), French for Programming in Logic (PROLOG), Python, JAVA, C++ and ADA.

LISP is a computer programming language with a long history and a distinctive, fully parenthesized prefix notation (Reilly, 2003). It was first conceived in 1958 and it became the programming language of choice for AI research. It was the basis for many ideas in computer science, including tree data structures, automatic storage management, dynamic typing, conditionals, higher-order functions, recursion, the self-hosting compiler (Graham) and the read-eval-print loop (Widemann *et al*., 2013). Linked lists are LISP's major data structures and LISP source code is made of lists. Thus, LISP programs can manipulate source code as a data structure, giving rise to the macro systems that allow programmers to create new syntax or new domain-specific languages embedded in LISP.

The idea of PROLOG was first conceived in 1970 and implemented in 1972 by AI scientists, namely, Alain Colmerauer, Robert Kowalski and Philippe Roussel (Kowalski, 1988). Since its inception, it was one of the most popular programming languages and has remained the most popular with many variants (Szuba, 1984). Since it is a general-purpose logic programming language, PROLOG is widely used for programming in AI (Clocksin and Mellish, 2012). It differs from other programming languages in that it is intended mainly as a declarative programming language. It has its roots in first-order logic expressed in terms of relations, represented as facts and rules while a computation is initiated by running queries over these relations (Lloyd, 2012). The language is well suited for different branches of AI. For example, logical problems that are randomly selected (Szuba, 1984) (De Raedt *et al*., 2007); expert systems (Merritt, 2012) and natural language processing (Lally and Fodor, 2011; Pereira and Shieber, 2002).

Python is a general-purpose, high-level programming language whose design philosophy emphasizes code readability. Guido Van Rossum developed it in the early 1990 s (Van Rossum and Drake Jr, 1995). Python's syntax allows programmers to express concepts in fewer lines of code than would be possible in languages such as C (Mark, 2007). Python is well adapted for AI tasks (Blank *et al*., 2003; Laugier and Pertin-Troccaz, 1986), especially in machine learning and natural language processing. It was conceived towards the end of 1980 (Venners, 2003). Still, its implementation began in December 1989 as a successor to the ABC language capable of exception handling and interfacing with the Amoeba operating system (Van Rossum and Drake Jr, 2014). By October 2000, Python 2.0 was released with new features, which included a full garbage collector and support for Unicode (Van Rossum, 2007), versions 2.6 and 2.7 followed while Python 3.0 was released on December 3, 2008 (Blank *et al*., 2003). Several AI libraries run primarily on the Python infrastructure. For example, Keras, noted for its user-friendliness, modularity and easy extensibility, is a high-level neural network API compatible with Python 2.7-3.6 (Keras, 2018); Theano allows for the evaluation of mathematical expressions using multi-dimensional arrays efficiently (Theano, 2018).

Java was developed in the early 1990 s by James Gosling from Sun Microsystems (McKinley, 2016b). Some of the features of Java are - it can be easily coded and it is highly scalable, making it desirable for AI projects (Rodrigues *et al*., 2011). It is also portable and can easily be implemented on different platforms since it uses virtual machine technology (Arnold *et al*., 2005).

C++ was developed in 1979 by Bjarne Stroustrup at Bells Labs and standardized in 1998 as a C language extension. Of all the AI programming languages, it is the fastest and most used by developers in AI projects because of its time sensitivity and also when speed is of higher priority to improve their project execution time (Stroustrup, 2013).

In this study, we carry out a systematic review of published articles on AI programming languages from databases such as SCOPUS, IEEE Xplore and Google scholar. The published articles were reviewed to retrieve the year of execution, development team, capabilities, features and the limitations and applications of AI programming languages.

Methods

To provide a systematic review of AI programming languages, we followed the guidelines put forward by (Kitchenham *et al*., 2004). This systematic review aims to access original articles and full-length review articles that relate to AI programming languages. The process is detailed below:

Research Questions

The research questions address forthwith are:

RQ 1: What is the prevalence of AI programming language publications since 1963?
RQ 2: Which of the AI programming languages have received more attention regarding the volume of research publication produced?

RQ 3: What are the predominant AI programming languages on which recent AI software rely?
RQ 4: What characteristics of AI programming languages make them suitable or unsuitable for use across platforms?

In addressing RQ 1, we considered the volume of the publications published for each AI programming language since 1963. Each of these programming languages has been described in Table 2 - 10. For RQ 2, we discussed the major AI programming languages; LISP, Prolog, LOOP, ARCHLOG, EOLC, Phyton, C++, ADA and JAVA. We considered the popularity of the languages based on the number of publications that have been published from 1963 to 2020. To answer RQ 3, we analyzed a few recently developed AI software platforms/frameworks and determined the foundational AI programming languages that underlie the frameworks.

Table 1: Search terms on Scopus and IEEE Explore (1963 to 2020)

#	Searches
1	Artificial Intelligence
2	AI
3	Programming Language
4	1 OR 2
5	3 AND 4

Table 2: Extracted studies on LISP artificial intelligence programming languages

Programming language	Authors	Year	Development team	Capabilities/features	Limitations	Applications
LISP	John (1962)	1962	Research Laboratory of Electronics, Massachusetts Institute of Technology, U.S. Army, U.S. Navy and U.S. Air Force	LISP is a mathematical language that uses Symbolic data processing(S-expressions). These S-expressions are stored in a structured list. It performs computational analysis on sequential programs, it has simple internal structure and compatible with other systems.	Ultra slow numerical computation and lack of better representation of block of registers. It has a higher overhead when compared with other conventional programming languages. This has narrowed its use in AI. It is limited with respect to processing and memory requirements. Its size, unwieldiness, "kitchen sink" design strategy and general ADA fication is disliked by most of its critics. Slower compare to C++	Used in differential and integral calculus, electrical circuit theory, mathematical logic, game playing and other fields of artificial intelligence. A vastly useful tool for the programming of many Expert Systems. It is programming language of choice of most AI researchers. It has been applied in Musical composition and processing foundational language for all programming languages.
	Dixon (1986)	1986	EG&G Idaho Inc, Idaho Falls, ID,, USA.	Able to rapidly analyze large trees by efficiently utilizing a list-based tree structure, search space and rule-based		
	Hawkinson (1986)	1986	LISP Machine Inc, Cambridge, MA, USA	More powerful for providing symbolic computing compared with conventional programming methods.		
	Kaisler (1986)	1986		Very flexible programming language compared with most others because it is built around a kernel of mathematical principles.		
	Hahne *et al*. (1988)	1988	Siemens AG, Munich, West Ger, Siemens AG, Munich, West Ger	LISP, a Programming Language and Its Computational Models		
	Rahn (1990)	1990	The University of Washington	Its features includes, a data structure for music representation, Lisp front (a language for composing music), it is object-oriented and the ability to output Music4P score data		
	Takeuchi (2002)	2002	Computer Science Department, University of Electro-Communications, Chofu, 182-8585, Japan	Lisp has potentials for systems programming and is suitable for writing operating systems. It also possess capabilities for heterogeneous parallel computing		
	Ellis (2012)	2012	Department of Electrical Engineering and Computer Science, Information and Telecommunication Technology Center, University of Kansas, Lawrence, KS 66045, United States	Coding is fast and efficient due compilers Automatic garbage collections was invented for LISP language		
	Jaakkola *et al*. (2019)	2019	Tampere University	Self-modifying codes based on the state of the computer. Ability to provide solutions in terms of algebraic formulas	Its implementation in computer architecture	Implementation of self-modifying applications. Developing intelligent software
	Khomtchouk *et al*. (2018)	2018	Center for Therapeutic Innovation and Department	LISP allows programmers to more quickly write programs	Not stated in the field	Build of AI systems that may ultimately

Table 2: Continue

	of Psychiatry and Behavioral Sciences, University of Miami Miller School of Medicine	that run faster than in other languages.	transform machine learning and AI research in bioinformatics and computational biology.

Table 3: Extracted Studies on PROLOG Artificial Intelligence programming language

Programming language	Authors	Year	Development team	Capabilities/features	Limitations	Applications
PROLOG	Clark and Gregory	1986	Dept. of Computing, Imperial College of Science and Technology, University of London	Performs efficient implementation by incorporating parallel mode of computation	The coding of the knowledge and the organization and modularization requires a lot of creativity. Some examples include fact/consequent representation problem and existential quantification problems. One of its drawbacks is the lack of natural mechanism to tackle the issue of uncertainties since by default it is designed to be a two-valued logic programming language. Prolog by default provides limited support for real-life knowledge engineering. In order to improve performance in Prolog applications, the performance of the microarchitecture of the uniprocessor engine needs to be developed. Prolog side-effects are typical necessary evil. Convenient logic program properties no longer hold and upsetting practical problems sprout, namely the debugging of programs with side-effects is harder. By default Prolog does not take into account the dynamic nature of agents such as knowledge acquisition and action execution. This poses a problem because the agent might work in a dynamic environment where unexpected things can happen. Slower compare to C++	Suitably adapts to applications requiring implementation in logic programming environments. Applied in Octree solid modeling applications. Relational Database applications, natural language processing, theorem proving, automated reasoning. Used for establishing expert systems in specific research activities. Used in logic-circuit model building, expert systems, AI and natural-language interfacing. Used in clinics for detection and classification of QRSs in Electrocardiography (ECG). Forth-based Prolog is used as basis of the expert system component of an astronaut interface for a series of Spacelab experiments It is a well adapted technology for expert systems design Used in real-time expert system to develop astronaut interface for a series of Spacelab vestibular experiments. Used in the probing of a facility's defenses and to find potential attack paths that meet designated search criteria. Used in electric utility applications, for volt/VAR dispatch and to increase the capability of electric energy management centers to successfully monitor power system operation and promptly respond to emergencies. Used in representation of a nursing knowledge base. Finds application in HICOM communication system language of choice for knowledge-processing systems suitable for implementing an inference engine for agents. Finds applications in insulin pump systems in hospitals for treatment of diabetic patients. Widely used in artificial intelligence Research Construction of expert systems. Applied in the interpretation of IR spectra
	Wilkerson	1986	Univ of Missouri, Rolla, MO, USA, Univ of Missouri, Rolla, MO, USA			
	Wilson	1986		PROLOG for applications programming		
	Guerrieri and Grover	1986	Softech Inc, Waltham, MA, USA,	Suitable support for programming to modeling with Octrees		
	Weeks and Berghel	1986	Department of Computer Science, University of Nebraska, Lincoln, NE 68588, United States	Support for relational databases, natural language processing and automated reasoning		
	Herther	1986	Univ of Minnesota Humanities/Social, Science Libraries, Minneapolis, MN, USA	PROLOG to the future: A glimpse of things to come in artificial intelligence It supports symbolic programming, therefore, by manipulating trigonometric relations and identities, it is able to derive useful kinematic equations of open kinematic chains It is easy to learn and easily adaptable		
	Giakoumakis and Papakonstantinou	1987	Natl Technical Univ of Athens, Athens, Greece	It's facts, rules, questions and inference features are used in recognition tasks in clinical analysis		
	Martin *et al*.	1987	Information Technology Research Centre, Department of Engineering Mathematics, University of Bristol, Bristol, BS8 1TR, United Kingdom			
	Odette and Paloski	1987	Applied Expert Systems Inc, Cambridge, MA, USA,	Forth-based Prolog possess the features for symbolic reasoning Knowledge Engineering Library (KEL) written in Prolog, expands Prolog's knowledge processing functionalities by supporting rule-based knowledge representation and approximate reasoning.		
	Paloski *et al*.	1987	KRUG Int, Houston, TX, USA,	Forth-based Prolog includes a predicate that can be used to execute Forth definitions. It also has the facility for rule based clauses and a procedure base containing Prolog goals that provides support for coding in Forth. PROLOG and automatic program generation from specifications		
	Zimmerman	1987	Quintus Computer Systems, Mountain, View, CA, USA	It operates on the principles of logic using the same inference methods that define symbolic logic (predicate calculus) that human's use in reasoning. It has the ability to manipulate complex data structures and flexibly represent real -world knowledge greatly speeds development time		
	Anon (1988)	1988				
	Ozbolt and Swain	1988	Univ of Virginia, Sch of Nursing,, Charlottesville, VA, USA	Provides support for graphical illustrations, natural language text, description of major knowledge domains and depictions of the relationships within and among them.		
	Beer and Manz	1989	Siemens AG	It is user-friendly and many		

Table 3: Continue

	Beerand Manz	(1989)		Osterreich, Austria faster and more economically than with other customary languages such as COBOL or PASCAL.	user tasks can be carried out	
	Buzzi	1989	Comparative Physiology and Behavioral Biology Laboratory, Federal Institute of Technology, Zurich, Switzerland	Allows definition and characterization of groups of subjects and single objects. Classifications performed using PROLOG compares very well with the methods of logistic regression and with discriminant analysis.		
	Colmerauer	1989	Faculté des Sciences de Luminy, Unité de recherche associée au Cnrs 816, Case 901, 70, route Léon Lachamp, F-13288 Marseille Cedex 9	Prolog III programming language expands Prolog functionalities by redefining the fundamental process at its heart, integrates into this mechanism, refined processing of trees and lists, greater number processing and processing of complete propositional calculus.		
	Patt (1989)	1989	Computer Science Division, University of California, Berkeley, CA, United States			
	Pereira and Calejo	1989	Logic Programming and Artificial Intelligence Group, Universidade Nova de Lisboa (UNL), Monte da Caparica, Portugal			
	Hayashi *et al*. (2002)	2002	Computer and Network Systems Laboratory, Corporate Research and Development Center, TOSHIBA Corporation	Prolog-like procedures can be developed to resolve issues of implementing inference engine for dynamic agents		
	Ellis and Agah	(2012)	Prolog has a built-in list handling essential in representing tree-based data structures and pattern matching .it can also backtrack automatically	Probabilistic measuring, robotic domain and biological domain The performance Rate is highly improved when connected to Java		
	Ostermayer *et al*. (2014)	2014				
	De Raedt (2015)	2015				
	Jaśkiewicz (2016)	2016				
	Nickles (2016)	2016				
	Morozov *et al*. (2017)	2017				
	Zhang *et al*. (2017)	2017				
PROLOG	Futo and Papp (1986)	1986	Computer Research Inst, Budapest, Hung,	Ability to strategically search for the right dosage of drug by leveraging on a combination of pharmacokinetic/pharmacodynamics models and logical decision rules		
PROLOG	Martin *et al*. (1987)	1987	Information Technology Research Centre, Department of Engineering Mathematics, University of Bristol, Bristol, BS8 1TR, United Kingdom	FPROLOG Builds generality and flexibility over conventional Prolog by empowering the language with breadth-first and depth-first search capabilities.		
PROLOG, LISP	Postma *et al*. (1987a)	1987	Department of Analytical Chemistry, Faculty of Science, University of Nijmegen, Toernooiveld, Nijmegen, The Netherlands	They were designed to primarily manipulate symbols rather than numbers. It, therefore, possess toolkits that consist of various knowledge representation methods and inference engines		
PROLOG, LISP	Postma *et al*. (1987b)	1987	Department of Analytical Chemistry, Faculty of Science, University of Nijmegen, Toernooiveld, Nijmegen, The Netherlands	Implementation of a teaching program for IR spectrometry in Lisp and Prolog (Useful to get full paper)		
PROLOG, LISP	Falk (1988)	1988	Computer Design, United States		Compared with coding in conventional languages such as C or Ada, code executions in PROLOG or LISP is slower. In addition to this, their memory demands are extensive.	
PROLOG	Jaakkola *et al*. (2019)	2019	Tampere University	Logic-based language Ability to execute rules, create new rules and modify old rules Ability to provide solutions in terms of algebraic formulas	Its implementation in computer architecture	Developing intelligent software And Expert System
PROLOG	Duque-Méndez *et al*. (2018)	2018	Universidad Nacional de Colombia, Manizales, Colombia	Can interface with Java		

Table 4: Extracted Studies on LOOP Artificial Intelligence programming language

Programming language	Authors	Year	Development team	Capabilities/features	Limitations	Applications
LOOP (Logic and. Object-Oriented Programming language)	Suciu *et al*. (2001)	2001	Dept. of Comput. Sci., Tech. Univ. of Cluj-Napoca, Romania	LOOP extends PROLOG logic programming paradigm with object-oriented features	Lacks mechanisms for structuring knowledge (program clauses)	Used in LP (Linear Programming)-based AI applications

Table 5: Extracted Studies on ARCHLOG Artificial Intelligence programming language

Programming Language	Authors	Year	Development team	Capabilities/features	Applications
ARCHLOG	Fidjeland and Luk (2006)	2006	Imperial College London, 180 Queen's Gate, London SW7 2AZ, United Kingdom	Can produce high-performance designs without detailed knowledge of hardware development and a framework for designing multiprocessor architectures	Finds application in machine learning and cognitive robotics

Table 6: Extracted Studies on EOLC Artificial Intelligence programming language

Programming Language	Authors	Year	Development team	Capabilities/features	Applications
EOLC (Epistemic Ontology Language with Constraints)	Kumar and Krogh, 2007)	2007	Department of Electrical and Computer Engineering, Carnegie Mellon University,	Used for specifying the epistemic ontology for heterogeneous verification Pittsburgh, PA 15213, USA.	Finds application in redundant flight guidance system and in the heterogeneous Verification of Embedded Control Systems

Table 7: Extracted studies on PYTHON Artificial Intelligence programming languages

Programming Language	Authors	Year	Development team	Capabilities/features	Limitations	Applications
PYTHON	McKinley (2016a)	1991	Department of Analytical Chemistry, Faculty of Science, University of Nijmegen, Toernooiveld, Nijmegen, The Netherlands	Python has improved over a short period of time compare to Java and C++ The efficiency of the programmer is highly improved because of its support for object-oriented design, functional and procedural styles of programming. It has high level syntax. Algorithm can be tested without implementation.	Is not good for mobile computing because of its weak Language for mobile computing The execution is slow in AI development due to the fact that it works with the help of an interpreter unlike C++ and Java	
	Bahana *et al*. (2018)	2018	Computer Science Department, Faculty of Computing and Media, Bina Nusantara University, Indonesia	Contain the utilities; Scrapy, Beautiful Soup and Link Grabber for web crawling, web content extractor and Uniform Resource Locator (URL) extractors, respectively	Beautiful Soup cannot be used as a web crawler by itself	Web content crawling. Developing advanced probabilistic models in AI research
	Bingham *et al*. (2019)	2019	Uber AI Labs, Uber Technologies, Inc., 555 Market St., San Francisco, CA 94103, United States	Expressivity, scalability, flexibility and minimality		

Table 8: Extracted studies on C++ Artificial Intelligence programming languages

Programming language	Authors	Year	Capabilities/features	Limitations	Applications
C++	Park *et al*. (2017)	2017	C++ can organize data because it is a multi-paradigm programming that supports object-oriented principles. It has high level of abstraction which makes it good for solving complex problem in AI	The standard library base in C++ is small and even though it has better efficiency of control, big C++ projects are tough to maintain and slow to develop.	Used in virtual robot simulation and synthesis, grasp synthesis, 3D drawing. Any kind of data can be modelled and simulated easily in AI
	McKinley (2016a)	2016			
	Kurniawan *et al*. (2015)	2015			
	Ellis and Agah (2012)	2012			

Table 9: Extracted Studies on JAVA, HASKEL and ADA Artificial Intelligence programming languages

Programming language	Authors	Year	Capabilities/features	Limitations	Applications
JAVA	Ostermayer *et al*. (2014)	2012	Java is portable. Implementation on different platforms is easy because of virtual machine technology. Algorithms coding is very easy.	Java's Response time is more and less execution speed this makes it slower than C++	It enables the Automatic Speech Recognition (ASR) systems to have Improved performance when linked with PROLOG
	Babu *et al*. (2015)	2015			
	Kurniawan *et al*. (2015)	2015			
	Mittal and Mandalika (2015)	2015			
	McKinley (2016a)	2016			
	Garg and Kumar (2017)	2017			
	Raff (2017)	2017			
HASKELL	Magalhães and Löh (2015)	1990	The algorithms can be gotten in cabal. Can be simulated on CPU cloud, bytecode complier and CUDA binding.		Uses algebraic datatype to reduce code duplication in generic programming.
ADA	Hattori *et al*. (1985)	1985	High performance and maintainability, list processing facility		Development of large scale AI software

Table 10: Extracted studies of other programming languages

Programming language	Authors	Year	Development team	Capabilities/features	Limitations	Applications
LARS (Logic -based framework for Analytic Reasoning over Streams)	Beck *et al*. (2018)	2018	Department of Mathematics and Computer Science, University of Perugia, Italy.	Extend propositional logic with generic window operators and additional controls to handle temporal information. Extend Answer Set Programming (ASP) with rich stream reasoning. Analytic reasoning over streams.	Relies on ASP.	Due to its linkage to ASP, it provides a uniform basis for developing AI applications in a streaming context, such as diagnosis, configuration, planning and many others.
Arden Syntax	Kraus (2018; Kraus *et al*., 2018a-b)	2018	Center for Communication and Information Technology, University Hospital Erlangen, Glückstraße 11, Erlangen, 91054, Germany Department of Medical Informatics, Biometrics and Epidemiology, Chair of Medical Informatics, Friedrich-Alexander-University Erlangen-Nuremberg, Wetterkreuz 13, Erlangen-Tennenlohe, 91058, Germany Department of Anaesthesiology, University Hospital Erlangen, Krankenhausstraße 12, Erlangen, 91054, Germany	Creation of multi-patient clinical decision support(CDS) routines in Arden Syntax	Installation is limited to focus on a single patient and can only preprocess microbiology findings in MLMs separately	Easy implementation of prototypical dashboard and concurrent reuse of basic components of the dashboards. Dashboards can also be shared concurrently.
Perkedel	Noprianto (2018)	2018	Computer Science Department, Bina Nusantara University Jakarta, Indonesia 11480	Domain specific language for simple data entry application	A lot of functionalities have not been designed and implemented.	It is a real multi-platform application that is not dependent on language run time. It can be used to develop a data entry application in few minutes and it's an open source software.
Parrot	Tu *et al*. (2019)		School of Computer Science, Central China Normal University Wuhan, China. School of Information Technology, York University Toronto, Canada. School of Computer Science, Central China Normal University Wuhan, China.	Python-based toolkit to perform Information Retrieval (IR) experimentation within Jupyter notebook.	Classical learning-to-rank models is not implemented.	Provides an interactive experimental framework for IR. Facilitates the development of neural ranking models Improves research proficiency in information retrieval.
Battlebot	Noval *et al*. (2019)	2019	Game Technology Study Program, Multimedia Creative Department, Politeknik Elektronika Negeri Surabaya Surabaya, Indonesia	a visual programming game		Offer a simplified approach in understanding the basic logic of Programming by using some icons in it.

Concerning the characteristics of AI programming languages (RQ 4), we considered the following: Capabilities or features, limitations and application of the AI programming language.

Search Strategy

The search for relevant literature was conducted in May 2021 using SCOPUS and IEEE Xplore databases; the dates was set to 1963 to 2020. Furthermore, we extracted more related documents from Google Scholar and reference list. Table 1 shows the search terms used for the databases.

Selection Criteria

We searched for studies that are related to AI Programming Language. Specifically, we included studies that revealed the features or capabilities, limitations and

applications of AI programming languages. We excluded reviews, viewpoints, or editorials. We also reviewed studies that were written in English only.

Case Definitions

For this research, we signposted our definition of AI based on the definition put forward by Nils Nilsson (Nilsson, 2010) in his book. "Artificial intelligence is that activity devoted to making machines intelligent and intelligence is that quality that enables an entity to function appropriately and with foresight in its environment." This definition would guide our final selection of retained studies in subsequent sections.

Quality Criteria

We ensured that the AI programming language was explicitly defined and met the case definition for each of the full text accessed as stated in 2.3. Studies with ambiguous details of the features or capabilities, limitations and AI applications were excluded from this review.

Data Extraction

Data extraction was conducted by the following reviewers; TJ, AA, FA and OA. The selection of relevant articles was performed by all reviewers. Any disagreement between reviewers on our selection of studies was resolved by the fifth and sixth reviewer EA and JB. From each study, we extracted data on programming language, version, year of execution, development team, capabilities or features, limitations and applications. This study seeks to present the evolution of AI programming languages, as such, the retained studies are presented in a chronological order and grouped based on the type of the AI programming language.

Results

Systematic Search

Our search returned 7,604 documents related to AI programming languages; Scopus (5,818) and IEEE Xplore (435). Furthermore, a thorough manual search was performed on Google scholar and returned 1,039 documents. Hence, a total of 7,604 documents were prepared for analysis. Afterward, 434 duplicated documents were removed. Of the 7,170 documents remaining, 6,665 documents were excluded based on the following criteria; studies that do not relate to Artificial Intelligence programming languages and reviews, books, book chapters, reports, notes, short survey, letter, viewpoints and editorials. A total of 195 full texts and abstracts were assessed for eligibility; after that, 78 studies were retained for qualitative and quantitative synthesis. The highlighted search procedure is graphically represented in Fig. 1.

Study Characteristics

The bulk of the retained studies on Artificial Intelligence programming languages can be grouped under the main categories LISP, PROLOG. Two studies in 1986 and 1987 (Futo and Papp, 1986; Martin *et al*., 1987) respectively dealt with different variants of PROLOG namely TC- PROLOG and FPROLOG. In recent years, we see documentations on new Artificial Intelligence programing language like Logic and Objected-Oriented Programming Language (LOOP) documented by (Suciu *et al*., 2001) in 2001 which extended PROLOG logic programming language with object oriented features while (Ostermayer *et al*., 2014) discuss a connection architecture between PROLOG and JAVA. ARCHLOG documented in 2006 (Fidjeland and Luk, 2006) can produce high-performance designs without detailed knowledge of hardware development and a framework for designing multiprocessor architectures; Epistemic Ontology Language with Constraints (EOLC), which is used for specifying the epistemic ontology for heterogeneous verification was documented in 2007 by (Kumar and Krogh, 2007); McKinley described Python in his 2016 paper (McKinley, 2016a), (Ellis and Agah, 2012; Kurniawan *et al*., 2015; McKinley, 2016a; Park *et al*., 2017) in 2012,2015,2016 and 2017 respectively discussed the use of C++ in Artificial intelligence programming languages while (Babu *et al*., 2015; Garg and Kumar, 2017; Kurniawan *et al*., 2015; McKinley, 2016b; Mittal and Mandalika, 2015; Raff, 2017) discussed the use of JAVA in Artificial Intelligence programming languages.

Successes, Challenges and Limitations

Successes of Artificial Intelligence

Optimization of AI language has enabled a revolutionary change across many sectors globally. For instance, AI has been deployed across different fields like finance, health, engineering and education. The proliferation of AI languages across these sectors has helped to birth new tech businesses (Rauf and Alanazi 2014). In Education and Research, AI has brought about significant improvement in the quality of delivery of educational resources globally; expert systems have been created to provide seamless learning content to students and researchers across borders (McJones, 2017). In the health sector, AI has been used to assist in automated data management (Giakoumakis and Papakonstantinou, 1987), developing an artificial neural network to assist in rapid patient care (King, 2018). The advancements in AI has provided further dynamics of analysis of face or object recognition techniques for audiovisuals. Also, in music evolution in composing human like notes (Baumann *et al*., 2002). The invasion of cognitive problem-solving skill has brought innovative ideas that is answering engineering questions (Rauf and Alanazi 2014).

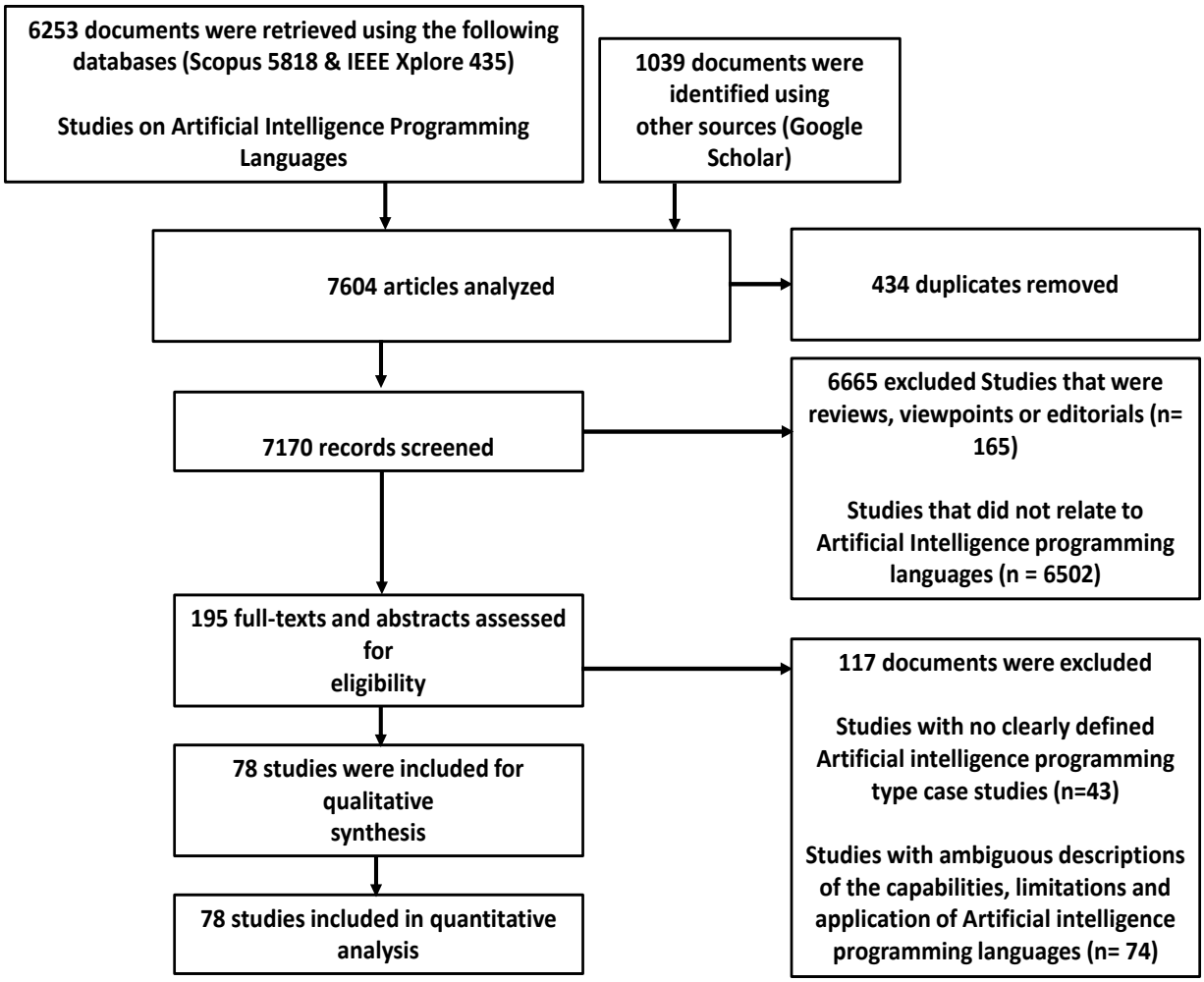

Fig. 1: PRIMA flowchart for selected studies

Challenges and Limitations of Artificial Intelligence

The main aim of AI is to build an intelligent machine that will make life easier for humans. The machine should be able to think like humans with some intelligent traits added to it. The programmers want to build some emotional quotient into the machines. Expert development is one of the major problems of AI programming language. LISP; a functional language created as a mathematical notation for computer programmers was developed for lambda calculus, which is not part of undergraduate curriculum in higher institutions.

This makes it more tedious for beginners to master LISP than other object-oriented languages like JAVA. The expert community and library capacity are limited due to this difficulty (Falch and Elster, 2018; McJones, 2017; Priestley, 2017).

Since AI involves building an intelligent machine like humans, it must also face some challenges like humans. Identifying some of these challenges will minimize the associated risks and, at the same time, make sure that we take full advantage of this technology.

Most researchers believe and agree that a super intelligent AI is unlikely to showcase human emotions like love or hate and therefore, it cannot become intentionally benevolent or malevolent. However, the most likely scenario to pose a threat to society is via autonomous weapons. These are weapons that AI systems are programmed to use to kill. If in the hands of the wrong person, these weapons could easily cause mass casualties. This could even lead to an AI war that would also result in mass causalities (Mason, 2010).

Legal challenges related to AI's application in the financial industry could be related to the consequences of erroneous algorithms and data governance. Due to the lack of appropriate data, erroneous algorithms can leave a big dent in the profits of an organization by making incorrect and perhaps detrimental predictions.

Poor data governance can result in data breaches where customers' Personal Identifiable Information (PII) that acts as a feedstock to an algorithm may get into the hands of hackers and can cause legal challenges for the organization (Heinl, 2014).

Discussion

This study was conducted to clearly understand the evolution of AI programming language from conception to date. The study confirms that John McCarthy pioneered the concept of AI, which has given birth to AI based high-level programming languages such as LISP, PROLOG, etc. Our findings suggest that AI; a multi-disciplinary field, can be highly enriched if further research is geared towards developing more syntax and semantic interaction that could produce robust language understanding systems.

We discuss the answers to our research questions in the following sub-sections.

What is the Prevalence of AI Programming Language Publications Since 1963?

AI programming languages have seen significant research interest since 1963. The research area has seen different peaks and valleys in research outputs. Between 1980 and 1986; a steep increase of 8 to 242 documents in SCOPUS was observed. There is however a significant reduction between 1987 and 2003 before a mild increase to 278 in 2004. Between 2005 and 2014, a dip occurred in research outputs in AI programming language based publications.

Which of the AI Programming Languages Have Received More Attention with Respect to the Volume of Research Publications Being Produced?

The volume of research outputs focusing on the PROLOG programming language has been rather significant. From Table 3, 34 research publications were reviewed for PROLOG in this systematic review, 15 documents were reviewed for the LISP programming language while the remaining 20 documents were reviewed for Logic and OBJECT Oriented Programming (LOOP), ARCHLOG, Epistemic Ontology Language with Constraints (EOLC), Phyton, C++, ADA and JAVA programming languages making it a total of 78 reviewed articles.

What are the Predominant AI Programming Languages on which Recent AI Software's Rely?

A number of AI libraries have been written based on some of the reviewed foundational AI programming languages, some of these programs are listed in Table 11. As illustrated in the Table, C/C++ has enjoyed the highest level of patronage (N = 7) by developers of modern AI libraries. This is closely followed by Python (N = 6). The reasons for the high adoption of these languages may be attributed to their unique strengths compared with other reviewed languages as earlier illustrated in Table 7 and 8.

What are the Characteristics of AI Programming Languages Which Make Them Suitable or Unsuitable for Use Across Platforms?

One of the most desirable qualities of AI programming languages is speed. PROLOG and LISP are slower compared to C/C++ and Python. An additional ability to support object-oriented principles is desirable. Sometimes it may be necessary to trade between quick response time and execution speed. Other desirable qualities are portability and ease of coding. The combination of C/C++ with Python in developing most of the trending AI libraries (such as Tensor Flow and Keras), as illustrated in Table 11 is born out of the need to complement the weakness of one language with the strength of the other and vice-versa.

Table 11: Details of modern AI libraries and their foundational AI programming languages

S/N	Modern Ai libraries	Initial release date	Original author	AI programming language platform
1	ALICE	1995	Joseph Weizenbaum	Java
2	OpenNN	2003	International Center for Numerical Methods in Engineering (CIMNE)	C++
3	OpenCog			C++, Python
4	TensorFlow	2015	Google Brain Team	Phyton, C++, CUDA
5	Siri	2011	Apple	Objective C
6	Neural Designer		Arteinics	C++
7	Keras	2015		Python
8	Scikit-learn	2007	David Cournapeau	Python, Cython, C, C++
9	Pandas	2017	Wes McKinney	Python
10	SciPy	2017	Travis Oliphant, Pearu Peterson, Eric Jones	Python Fortran, C, C++

Conclusion

This review covers the period between 1963 and 2020, the search criteria on relevant databases was based on the keywords "Artificial Intelligence", "AI" and "programming language". The search returned 7,604 documents, which were narrowed to 78 based on the relevance of information in the publication for our systematic review task. Based on quantitative and qualitative analysis of the resulting documents, we present the successes, challenges and limitations of AI. Our findings revealed that AI programming languages have experienced periods of peaks and valleys from 1963 but have been experiencing peaks since 2015, indicating their prevalence in recent times. Also, in the period of review, PROLOG and LISP have received the most attention. However, in recent AI software, C/C++ and Python are the most deployed owing to their speed, portability and ease of coding. Significant progress has been made in the field that has grown over the last 55 years and practically helped to solve many potential human problems concerning machine vision, game-playing and automatic natural language processing. This has even stretched beyond the scope of the originators of AI like Turing and McCarthy ever envisaged. More research was geared towards fine tuning the programming languages being functional and having a trace of embedded logic capabilities. Conclusively, as a science of intelligence, AI can be legitimately looked upon in the field of psychology because of its relevance to cognitive psychology.

Acknowledgement

This study was carried out at the Advanced Signal Processing and Machine Intelligence Research (ASPMIR) group laboratory, Covenant University, Ota, Nigeria.

KZN e-Skills CoLab, Durban University of Technology, Durban, South Africa provided support for the publication of this work.

Author's Contributions

Emmanuel Adetiba: Concept development, writing, critical review, final approval.

Temitope M. Adeyemi-Kayode: Writing, critical review, editing.

Adekunle A. Akinrinmade, Fumnilayo S. Moninuola and Oladipupo O. Akintade: Writing, editing.

Joke A. Badejo: Editing, critical review of final draft.

Obiseye O. Obiyemi: Critical review of final draft and galley proof.

Surendra Thakur: Critical review of final draft and source for funding of manuscript publication.

Abdultaofeek Abayomi: Editing of the final manuscript and critical review of the final draft.

Ethics

This article is original and the corresponding author confirms that all of the other authors have read and approved the manuscript and no ethical issues involved.

References

Anon. (1988). Knowledge-based systems for electric utility operation using the PROLOG language. https://www.scopus.com/inward/record.uri?eid=2-s2.0-0023960071&partnerID=40&md5=26113d56b21ca2692289d4e7a7a8c757

Arnold, K., Gosling, J., & Holmes, D. (2005). The Java programming language. Addison Wesley Professional. https://www.acs.ase.ro/Media/Default/documents/java/ClaudiuVinte/books/ArnoldGoslingHolmes06.pdf

Babu, A. A., Rao, A. A., & Yellasiri, R. (2015, November). Unsupervised adaptation of ASR systems: An application of dynamic programming in machine learning. In 2015 SAI Intelligent Systems Conference (IntelliSys) (pp. 245-253). IEEE. doi.org/10.1109/IntelliSys.2015.7361150

Bahana, R., Adinugroho, R., Gaol, F. L., Trisetyarso, A., Abbas, B. S., & Suparta, W. (2017, November). Web crawler and back-end for news aggregator system (Noox project). In 2017 IEEE International Conference on Cybernetics and Computational Intelligence (CyberneticsCom) (pp. 56-61). IEEE. doi.org/10.1109/CYBERNETICSCOM.2017.8311684

Baumann, S., Kluter, A., & Norlien, M. (2002, December). Using natural language input and audio analysis for a human-oriented MIR system. In Second International Conference on Web Delivering of Music, 2002. WEDELMUSIC 2002. Proceedings. (pp. 74-81). IEEE. https://ieeexplore.ieee.org/abstract/document/1176196

Beck, H., Dao-Tran, M., & Eiter, T. (2018). LARS: A logic-based framework for analytic reasoning over streams. Artificial Intelligence, 261, 16-70. doi.org/10.1016/j.artint.2018.04.003

Beer, A., & Manz, J. (1989). PROLOG. A user-friendly Hicom tool. Telcom Report (English Edition), 12(4), 130-131. doi.org/10.1007/978-3-322-89479-3_1

Bingham, E., Chen, J. P., Jankowiak, M., Obermeyer, F., Pradhan, N., Karaletsos, T., ... & Goodman, N. D. (2019). Pyro: Deep universal probabilistic programming. The Journal of Machine Learning Research, 20(1), 973-978. https://www.jmlr.org/papers/volume20/18-403/18-403.pdf

Blank, D., Kumar, D., Meeden, L., & Yanco, H. (2003). Pyro: A python-based versatile programming environment for teaching robotics. Journal on Educational Resources in Computing (JERIC), 3(4), 1-es. doi.org/10.1145/1083310.1047569

Buzzi, R. (1989). Classifications of subjects with the language PROLOG. Computer methods and programs in biomedicine, 28(3), 171-182. doi.org/10.1016/0169-2607(89)90146-6

Clark, K., & Gregory, S. (1986). Parlog: Parallel programming in logic. ACM Transactions on Programming Languages and Systems (TOPLAS), 8(1), 1-49. doi.org/10.1145/5001.5390

Clocksin, W. F., & Mellish, C. S. (2012). Programming in Prolog: Using the ISO standard. Springer Science & Business Media. doi.org/10.1007/978-3-642-55481-0

Colmerauer, A. (1989, May). Une introduction à Prolog III. In Annales des télécommunications (Vol. 44, No. 5, pp. 229-241). Springer-Verlag. doi.org/10.1007/BF02995669

De Raedt, L. (2015). Probabilistic programming and its applications (Keynote Abstract). In 9th International Workshop on Multi-disciplinary Trends in Artificial Intelligence (MIWAI 2015), Fuzhou, China, November 13-15, 2015 (Vol. 9426, pp. xiii-xiv). Springer International Publishing. https://www.diva-portal.org/smash/record.jsf?pid=diva2%3A1552681&dswid=235

De Raedt, L., Kimmig, A., & Toivonen, H. (2007, January). ProbLog: A Probabilistic Prolog and Its Application in Link Discovery. In IJCAI (Vol. 7, pp. 2462-2467). https://www.aaai.org/Papers/IJCAI/2007/IJCAI07-397.pdf

Dixon, B. W. (1986). Integrated fault tree development environment (No. EGG-M--05086). Idaho National Engineering Lab. https://inis.iaea.org/search/search.aspx?orig_q=RN:18051258

Duque-Méndez, N. D., Tabares-Morales, V., & González, H. (2018, November). ESIA Expert System for Systems Audit Risk-Based. In Ibero-American Conference on Artificial Intelligence (pp. 483-494). Springer, Cham. doi.org/10.1007/978-3-030-03928-8_39

Ellis, I. C., & Agah, A. (2002). Software development paradigms for artificial intelligence applications. In Proceedings of the 6th IASTED International Conference on Software Engineering and Applications, SEA (Vol. 2012, pp. 7-12). https://www.actapress.com/PDFViewer.aspx?paperId=24402

Falch, T. L., & Elster, A. C. (2018). ImageCL: Language and source-to-source compiler for performance portability, load balancing and scalability prediction on heterogeneous systems. Concurrency and Computation: Practice and Experience, 30(9), e4384. doi.org/10.1002/cpe.4384

Falk, H. (1988). AI techniques enter the realm of conventional languages. Computer Design, 27(19), 45-49.

Fidjeland, A., & Luk, W. (2006, August). Archlog: High-level synthesis of reconfigurable multiprocessors for logic programming. In 2006 International Conference on Field Programmable Logic and Applications (pp. 1-6). IEEE. doi.org/10.1109/FPL.2006.311234

Futo, I., & Papp, I. (1986). Use of TC-PROLOG for medical simulation. In SCS Multiconference (pp. 29-34).

Garg, S., & Kumar, S. (2016, August). JOSN: JAVA oriented question-answering system combining semantic web and natural language processing techniques. In 2016 1st India International Conference on Information Processing (IICIP) (pp. 1-6). IEEE. doi.org/10.1109/IICIP.2016.7975361

Giakoumakis, E., & Papakonstantinou, G. (1987). PROLOG program for the QRS detection and classification. https://dspace.lib.ntua.gr/xmlui/handle/123456789/33363

Graham, P. (2002). Revenge of the nerds, May 2002: URL.

Guerrieri, E., & Grover, V. (1986). Octree Solid Modeling with PROLOG. doi.org/10.1007/978-3-662-21626-2_80

Hahne, K., Moeckel, P., & Thuerlings, K. J. (1988). LISP, a Programming Language and Its Computational Models. Siemens Forschungs- und Entwicklungsberichte/Siemens Research and Development Reports, 17(2), 52-58.

Hattori, F., Kushima, K., & Wasano, T. (1985). Comparison of Lisp, Prolog and Ada Programming Productivity in Ai Area. Paper presented at the Proceedings - International Conference on Software Engineering. https://dl.acm.org/doi/abs/10.5555/319568.319655

Hawkinson, L. (1986). Lisp and Lisp Machines: Tools For Al Programming. Chilton's instruments & control systems, 59(1), 37-40.

Hayashi, H., Cho, K., & Ohsuga, A. (2002). Logic programming for agents Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) (Vol. 2417, pp. 591). doi.org/10.1007/3-540-45683-X_63

Heinl, C. H. (2014). Artificial (intelligent) agents and active cyber defence: Policy implications doi.org/10.1109/CYCON.2014.6916395

Herther, N. K. (1986). Prolog To The Future: A Glimpse of things to come in Artificial Intelligence. Microcomputers for information management, 3(1), 31-45. https://eric.ed.gov/?id=EJ337855

Jaakkola, H., Henno, J., Mäkelä, J., & Thalheim, B. (2019). Artificial intelligence yesterday, today and tomorrow. Paper presented at the 2019 42nd International Convention on Information and Communication Technology, Electronics and Microelectronics, MIPRO 2019 - Proceedings. doi.org/10.23919/MIPRO.2019.8756913

Jaśkiewicz, G. (2016). Prolog-Scripted Tactics Negotiation and Coordinated Team Actions for Counter-Strike Game Bots. IEEE Transactions on Computational Intelligence and AI in Games, 8(1), 82-88. doi.org/10.1109/TCIAIG.2014.2331972

John, M. (1962). Functions and Constants in the LISP System LISP 1.5 Programmer's Manual (pp, 56-69): MIT Press. https://ieeexplore.ieee.org/abstract/document/6277736

Kaisler, S. (1986). Lisp: An AI programming language. IEEE Journal of Oceanic Engineering, 11(4), 468-473. doi.org/10.1109/JOE.1986.1145204

KERAS. (2018). Keras: The Python Deep learning library. https://keras.io/

Khomtchouk, B. B., Weitz, E., Karp, P. D., & Wahlestedt, C. (2018). How the strengths of Lisp-family languages facilitate building complex and flexible bioinformatics applications. Briefings in Bioinformatics, 19(3), 537-543. doi.org/10.1093/bib/bbw130

King, B. F. (2018). Artificial Intelligence and Radiology: What Will the Future Hold? Journal of the American College of Radiology. doi.org/10.1016/j.jacr.2017.11.017

Kitchenham, B. A., Dyba, T., & Jorgensen, M. (2004). Evidence-based software engineering. Paper presented at the Proceedings of the 26th international conference on software engineering. doi.org/10.1109/ICSE.2004.1317449

Kowalski, R. A. (1988). The early years of logic programming. Communications of the ACM, 31(1), 38-43. doi.org/10.1145/35043.35046

Kraus, S. (2018) Generalizing the Arden Syntax to a common clinical application language. Vol. 247. Studies in Health Technology and Informatics (pp. 675-679). https://ebooks.iospress.nl/volumearticle/48877

Kraus, S., Drescher, C., Sedlmayr, M., Castellanos, I., Prokosch, H. U., & Toddenroth, D. (2018). Using Arden Syntax for the creation of a multi-patient surveillance dashboard. Artificial Intelligence in Medicine, 92, 88-94. doi.org/10.1016/j.artmed.2015.09.009

Kumar, R., & Krogh, B. H. (2007, 9-13 July 2007). A New Logic Programming Framework for Heterogeneous Verification of Embedded Control Systems. Paper presented at the 2007 American Control Conference. doi.org/10.1109/ACC.2007.4282481

Kurniawan, A., Soesanto, C., & Wijaya, J. E. C. (2015). CodeR: Real-time Code Editor Application for Collaborative Programming. Paper presented at the Procedia Computer Science. doi.org/10.1016/j.procs.2015.07.531

Lally, A., & Fodor, P. (2011). Natural language processing with prolog in the ibm watson system. The Association for Logic Programming (ALP) Newsletter. https://www.cs.miami.edu/home/odelia/teaching/csc419_spring19/syllabus/IBM_Watson_Prolog.pdf

Laugier, C., & Pertin-Troccaz, J. (1986). SHARP: A System for Automatic Programming of Manipulation Robots.

Lloyd, J. W. (2012). Foundations of logic programming: Springer Science & Business Media. ISBN-10: 9783642831898.

Magalhães, J. P., & Löh, A. (2015). Hierarchy in generic programming libraries Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) (Vol. 9129, pp. 93-112). doi.org/10.1007/978-3-319-19797-5_5

Martin, T. P., Baldwin, J. F., & Pilsworth, B. W. (1987). The implementation of Fprolog-A fuzzy Prolog interpreter. Fuzzy Sets and Systems, 23(1), 119-129. doi.org/10.1016/0165-0114(87)90104-7

Mark, S. (2008). Rapid GUI programming with Python and Qt: The definitive guide to PyQt programming: Prentice Hall.

Mason, C. (2010). The Logical Road to Human Level AI Leads to a Dead End. IEEE Self-Adaptive and Self-Organizing Systems Workshop (SASOW), 2010. doi.org/10.1109/SASOW.2010.63

McJones, P. (2017). The LISP 2 Project. IEEE Annals of the History of Computing, 39(4), 85-92. doi.org/10.1109/MAHC.2018.1221040

McKinley, K. S. (2016a). Programming the world of uncertain things (keynote). ACM SIGPLAN Notices, 51(1), 1-2. doi.org/10.1145/2837614.2843895

McKinley, K. S. (2016b). Programming the world of Uncertain things (Keynote). Paper presented at the Conference Record of the Annual ACM Symposium on Principles of Programming Languages. doi.org/10.1145/2837614.2843895

Merritt, D. (2012). Building expert systems in Prolog: Springer Science & Business Media. ISBN-10: 9781461389118.

Mittal, H., & Mandalika, S. D. (2015). Framework for evaluation of programming language examinations SpringerBriefs in Applied Sciences and Technology (Vol. Part F2, pp. 121-128). doi.org/10.1007/978-981-287-338-5_10

Morozov, A. A., Sushkova, O. S., & Polupanov, A. F. (2017). Towards the distributed logic programming of intelligent visual surveillance applications Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) (Vol. 10062 LNAI, pp. 42-53). doi.org/10.1007/978-3-319-62428-0_4

Nickles, M. (2016). A tool for probabilistic reasoning based on logic programming and first-order theories under stable model semantics Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) (Vol. 10021 LNAI, pp. 369-384). doi.org/10.1007/978-3-319-48758-8_24

Nilsson, N. J. (2010). The Quest for Artificial Intelligence: Stanford University. doi.org/10.1017/CBO9780511819346

Nilsson N. J., & Nilsson N. J. (1998). Artificial intelligence: a new synthesis. Morgan Kaufmann. ISBN-10: 9781558604674.

Noval, B. A., Safrodin, M., & Hakkun, R. Y. (2019). Battlebot: Logic Learning Based on Visual Programming Implementation in Multiplayer Game Online. Paper presented at the IES 2019 - International Electronics Symposium: The Role of Techno-Intelligence in Creating an Open Energy System Towards Energy Democracy, Proceedings. doi.org/10.1109/ELECSYM.2019.8901628

Noprianto, E. (2018). Pemanfaatan media sosial dan penerapan social media analytics (SMA) untuk perpustakaan di Indonesia. Jurnal Pustaka Budaya, 5(2), 1-10. http://journal.unilak.ac.id/index.php/pb/article/view/1583

Odette, L. L., & Paloski, W. H. (1987). Use of a Forth-based Prolog for real-time expert systems. II. A full Prolog interpreter embedded in Forth. Journal of Forth Application and Research, 4(4), 477-486. https://dl.acm.org/doi/abs/10.5555/37380.37383

Ostermayer, L., Flederer, F., & Seipel, D. (2014, August). CAPJA-A Connector Architecture for Prolog and Java. In KESE@ ECAI.

Ozbolt, J. G., & Swain, M. A. P. (1988, November). Representing a nursing knowledge base for a decision support system in Prolog. In Proceedings of the Annual Symposium on Computer Application in Medical Care (p. 119). American Medical Informatics Association. https://www.ncbi.nlm.nih.gov/pmc/articles/pmc2245187/

Park, J., Park, C. H., Park, D. I., & Kim, H. S. (2017, June). The library for grasp synthesis & robot simulation. In 2017 14th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI) (pp. 418-423). IEEE. doi.org/10.1109/URAI.2017.7992766

Patt, Y. N. (1989). Alternative Implementations of Prolog: The Microarchitecture Perspective. IEEE Transactions on Systems, Man and Cybernetics, 19(4), 693-698. doi.org/10.1109/21.35333

Pereira, F. C., & Shieber, S. M. (2002). Prolog and natural-language analysis: Microtome Publishing. ISBN-10: 0971977704

Pereira, L. M., & Calejo, M. (1989). Algorithmic debugging of prolog side-effects Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) (Vol. 390 LNAI, pp. 151-162). doi.org/10.1007/3-540-51665-4_82

Postma, G. J., Vandeginste, B. G. M., van Halen, C. J. G., & Kateman, G. (1987a). Comparison Of Lisp and Prolog. TrAC - Trends in Analytical Chemistry, 6(1), 2-5. doi.org/10.1016/0165-9936(87)85011-2

Postma, G. J., Vandeginste, B. G. M., van Halen, C. J. G., & Kateman, G. (1987b). Implementation of a teaching program for IR spectrometry in Lisp and Prolog. Trends in Analytical Chemistry, 6(2), 27-30. doi.org/10.1016/0165-9936(87)87007-3

Priestley, M. (2017). AI and the Origins of the Functional Programming Language Style. Minds and Machines, 27(3), 449-472. doi.org/10.1007/s11023-017-9432-7

Paloski, W. H., Slosberg, R. B., & Kamm, R. D. (1987). Effects of gas properties and waveform asymmetry on gas transport in a branching tube network. Journal of Applied Physiology, 62(3), 892-901. https://journals.physiology.org/doi/abs/10.1152/jappl.1987.62.3.892

Raff, E. (2017). JSAT: Java statistical analysis tool, a library for machine learning. Journal of Machine Learning Research, 18, 1-5.

Rahn, J. (1990). Lisp kernel. A portable software environment for composition. Computer Music Journal, 14(4), 42-58. doi.org/10.2307/3680790

Rauf, A., & Alanazi, M. N. (2014, August). Using artificial intelligence to automatically test GUI. In 2014 9th International Conference on Computer Science & Education (pp. 3-5). IEEE. doi.org/10.1109/ICCSE.2014.6926420

Reilly, E. D. (2003). Milestones in computer science and information technology: Greenwood Publishing Group. ISBN-10: 9781573565219.

Rodrigues, A. O., Viana, J. J., & Ramirez, J. A. (2011). A Java platform for the implementation of the finite-difference time-domain method. In 2011 SBMO/IEEE MTT-S International Microwave and Optoelectronics Conference (IMOC 2011) (pp. 927-931). IEEE. doi.org/10.1109/IMOC.2011.6169373

Stroustrup, B. (2013). The C++ Programming Language US: Edwards Brothers Malloy in Ann Arbor, Michigan. ISBN-10: 0-201-88954-4

Suciu, A., Pusztai, K., Muresan, T., & Simon, Z. (2001, 7-9 Nov 2001). LOOP-a language for LP-based AI applications. Paper presented at the Proceedings 13th IEEE International Conference on Tools with Artificial Intelligence. ICTAI 2001. doi.org/10.1109/ICTAI.2001.974477

Szuba, T. (1984). PC-PROLOG for Process Control Applications. Angewandte Informatik, Applied Informatics, 26(4), 164-171.

Takeuchi, I. (2002). The future of Lisp. Systems and Computers in Japan, 33(6), 10-18. doi.org/10.1002/scj.10000

THEANO. (2018). Welcome. http://deeplearning.net/software/theano/

Tu, X., Huang, J., Luo, J., Zhu, R., & He, T. (2019). Parrot: A python-based interactive platform for information retrieval research. Paper presented at the SIGIR 2019 - Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval. doi.org/10.1145/3331184.3331393

Van Rossum, G. (2007). Python Programming Language. Paper presented at the USENIX Annual Technical Conference. https://thereaderwiki.com/en/Python_(programming_language)

Van Rossum, G., & Drake Jr, F. L. (1995). Python tutorial: Centrum voor Wiskunde en Informatica Amsterdam, The Netherlands.

Van Rossum, G., & Drake Jr, F. L. (2014). Python frequently asked questions. https://scicomp.ethz.ch/public/manual/Python/2.7.6/faq.pdf

Venners, B. (2003). The making of Python. Artima. com. http://www. artima. com/intv/python. html

Weeks, J., & Berghel, H. (1986). A comparative feature-analysis of microcomputer prolog implementations. ACM SIGPLAN Notices, 21(2), 46-61. doi.org/10.1145/15022.15028

Widemann, B. T. Y., Bolz, C. F., & Grelck, C. (2013). The functional programming language R and the paradigm of dynamic scientific programming (Position paper) Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) (Vol. 7829 LNCS, pp. 182-197). doi.org/10.1007/978-3-642-40447-4_12

Wilkerson, R. W. (1986). PROLOG. IEEE Potentials, 5(3), 22-25. doi.org/10.1109/MP.1986.6500804

Wilson, W. G. (1986). PROLOG for applications programming. IBM Systems Journal, 25(2), 190-206. doi.org/10.1147/sj.252.0190

Zhang, S., Khandelwal, P., & Stone, P. (2017). Dynamically constructed (PO)MDPs for adaptive robot planning. Paper presented at the 31st AAAI Conference on Artificial Intelligence, AAAI 2017.

Zimmerman, J. (1987). Prolog uses powerful logic to deftly control data. Computer Design, 26(11), 91-94. https://dl.acm.org/doi/abs/10.5555/26504.26512