A Review Paper on CFG to PDA Converter

Sanika Kadam, Omkar Kale, Anish Kamble, Aditya Kanawade
and Sejal Kharche

March 29, 2022

A Review Paper on CFG To PDA Converter

Sanika Kadam
Artificial Intelligence and Data Science
Vishwakarma Institute of Technology,
Pune,411037, Maharashtra.
sanika.kadam20@vit.edu

Omkar Kale
Artificial Intelligence and Data Science
Vishwakarma Institute of Technology,
Pune,411037, Maharashtra.
omkar.kale20@vit.edu

Anish Kamble
Artificial Intelligence and Data Science
Vishwakarma Institute of Technology,
Pune,411037, Maharashtra.
anish.kamble20@vit.edu

Aditya Kanawade
Artificial Intelligence and Data Science
Vishwakarma Institute of Technology,
Pune,411037, Maharashtra.
aditya.kanawade20@vit.edu

Sejal Kharche
Artificial Intelligence and Data Science
Vishwakarma Institute of Technology,
Pune,411037, Maharashtra.
sejal.kharche201@vit.edu

***Abstract*— Grammar Induction (Language studying) is the technique of getting to know grammar by training data of the good and poor strings of the language. The ability that of a CFG and a Pushdown Automata are equal. A CFG generates context-free language, and a Pushdown Automata finds out context-free language. A context-free grammar is formed using four parts: variables, terminals, a start variable, and rules. Each rule should have one single variable on the "left aspect," and on the "right aspect,". A PDA is a machine infinite-state within which things can be pushed or popped off of an infinitely tall stack on every transition. This paper describes the way to convert a Context Free Grammar into a Pushdown Automata that finds out the CFG's language. With Greibach Normal Form inputs, a context-free grammar to pushdown automaton converter is available. PDA are shown to provide CF language class. A finite automaton recognizes each and every regular language.**

Keywords— Finite Automata, Context-Free Grammar, Pushdown Automata, Conversion.

I. INTRODUCTION

Wide variety of languages and grammars are studied and worked upon by computer scientists that fall under the hierarchy proposed by linguist Noam Chomsky in 1956 also known as Chomsky Hierarchy. A quadruple (V, T, P, S) is produced by a CFG and a set of finite grammatical policies, where V acts as a variable (non-terminals). T stands for terminals. There is no appropriate or left context for P. The first image is S. Pushdown automata are a means to implement a CFG and a structure DFA for a standard grammar in the same way. A DFA can only retain a certain number of records, whereas a PDA can retrieve an infinite quantity of data.

This language categorization scheme is linked to the categorization of automata that may be used to comprehend such languages, as well as the grammar that is used to create them. The task of automaton M designing which accepts as sequences of input is the challenge of detecting between Language L and alphabet its challenge to select Language L over it. If sequences are acceptable factors of the L, M accepts in other case, M would reject them. Every grade of language in the hierarchy of Chomsky is created using an type of grammar specially designed and can be identified using a specific kind of automaton. As one goes up the language hierarchy, the kind of automata which is essential to understanding the language grows more hard, and the grammar required to create the language becomes more common. Fig. 1.

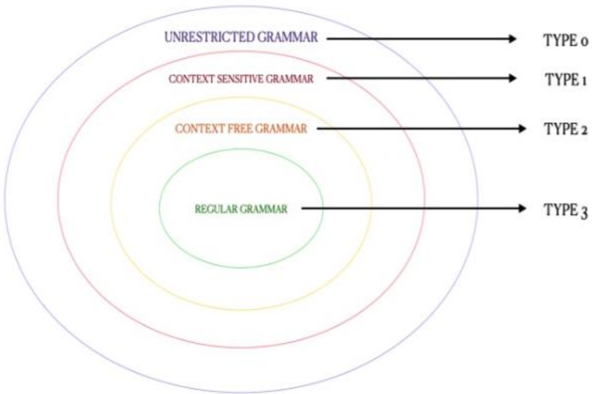

Fig. 1. Hierarchy of Chomsky

Illustration CFG languages are types of different languages discovered using PDA, which are type of automata that is finite with an stack which is indefinite. The Pushdown Automata (PDA) is depicted in Figure 2 is a block diagram.

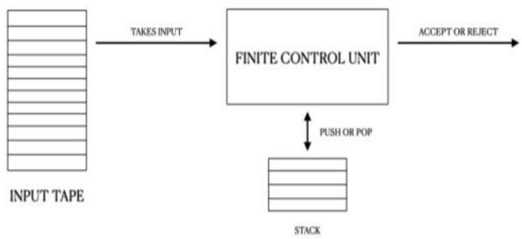

Fig. 2. Schematic diagram of the PDA

An evolutionary strategy based on commonly used context-free grammars (Backus-Naur form, BNF) surpassed their representations when the impact of different grammar representations was investigated. The introduction of evolutionary algorithms made it easier to analyse CFG by offering grammar-specific heuristic operators and better populating the preliminary population.

II. LITERATURE REVIEW

Context-Free Grammar is a notation that can be used to specify a language's grammar (CFG). Terminals, non-terminals, a start symbol, and production rules make up Context-Free Grammar. The terminal symbols are a group of tokens[5]. These are the fundamental symbols that makeup strings. The collection of strings marked by the start sign is the grammar's stated language[5]. A context-free grammar, which is type 2 in Chomsky's hierarchy, has a wide range of applications in formal language theory, pattern recognition, speech recognition, machine learning, compiler design, and genetic engineering, to name a few [4].

Several papers have explained the benefits of using Context-Free Grammars (CFGs) using proofs and Parse Trees to reduce uncertainty in Natural Language Sentences[5]. These advantages, however, are contingent on the Derivation processes and CFG Rules.

The paper[6] presents a context-free grammar to limit the identification of a system's use cases to adhere to the requirements of the engineering discipline. The suggested CFG will be used to represent and record the rules that control the identification of events (designated as syntactic categories), use cases, and other information system requirement components (designated as the set of valid terminals). the

Grammatical inference/grammar induction encompasses a wide range of subjects, including identifying unfamiliar Context Free grammar in the chosen language from positive examples[4].

In this study[4], a new method is presented for determining the Chomsky Normal form's equivalent. It is been proved that GFGness is Exp Time complete for properties other than PDA. VPA can be exponentially more concise than GFG-VPA, and GFG-VPA can be exponentially more succinct than deterministic VPA.

The topic of learning context-free grammar is investigated from a corpus in the paper [7]. A technique based on the concept of the corpus's minimum description length is been looked into.

Path queries are used to specify paths inside a data network that corresponds to a specific pattern[1]. Regular path patterns created by regular expressions are frequently supported by query languages like SPARQL. A context-free path query defines a path whose language could be determined by context-free grammar[1].

This type of query is useful in a variety of fields, including genetics, data science, and source code analysis. The approach for handling context-free path queries is presented in this paper. In contrast to existing systems that must process the entire graph, the algorithm mentioned searches for localised pathways, allowing it to process subgraphs. It also accepts any context-free grammar as input, avoiding the use of troublesome normal forms[1]. It looks at the traditional correspondence theorem between context-free grammars and pushdown automata in this study[2]. Changing the sequential composition process operator to a sequence alignment operator with conditional acceptance improves the correlation in the scenario[2]. The missing factor in reconstructing the entire correspondence is revealed to be the addition of a concept of situation awareness [2].

In input-driven pushdown automata (IDPDA), the next activity on the pushdown store (push, pop, nothing) is solely controlled by the possible input. Nowadays, many similar devices are designed so that pushing from an unoccupied pushdown does not terminate the calculation, but rather maintains it [3].

The impact of DIDPDAs' novel behaviour on their power is investigated, and their capabilities are compared to those of input-driven pushdown automata, which are effectively IDPDAs with input preprocessed by duration finite-state transducers [3], are two types of IDPDAs. It is investigated that the determinization of DIDPDAs and their sheer intricacy, closure features, and decidability [3].

By investigating, it can be said that these transformations from the perspective of discretional complexity in this paper [9]. The ideal optimum values are presented for the majority of states of nondeterministic and deterministic finite automata that are comparable to Chomsky’s normal form unary context-free grammars [9]. The amount of elements within grammar in question determines the boundaries. There are additional upper limitations for the majority of states of finite automata that emulate unary pushdown automata [9].

As a result, it shows that one-way auxiliary pushdown automata's workspace for accepting non-regular unary languages has a log log n lower bound.

SCFGs (synchronous context-free grammars) are increasingly widely used in statistical machine translation, with Hiero (Chiang 2007)[10] being the most well-known example. The task at hand was to use an SCFG and an n-gram language model to decode source text and generate a target translation [10].

In practice, decoding is challenging, yet the fundamental linguistics and relationships involved may be articulated simply and clearly. Because of the formal description, PDAs will be available for comparison to current decoders based on other forms of automata [10].

III. COMMON INFORMATION ON THE SYSTEM

In this paper, it had been created as context-free descriptive linguistics to pushdown automaton convertor that operates with Greibach ancient sort inputs. The paper used the following methodology to convert the CFG to pushdown automata. On R.H.S. producing, the initial image ought to be a terminal sign.

The steps for obtaining personal organiser from the CFG area unit as follows:

Step 1: Amend the CFG outputs into GNF outputs.

Step 2: there will be only 1 state on the pushdown automata

Step 3: The CFG's initial image area unit getting to be the PDA's image

Step 4: Add the appropriate rule to non-terminal symbols:

$$\delta\ (q, \varepsilon, A) = (q, \alpha)$$

The production rule could be a could be a.

Step 5: for each terminal image, add the following rule:

$$\delta\ (q, a, a) = (q, \varepsilon) \text{ for every image.}$$

.

$$\delta\ (q, 1, Z_0) = \{ (q, X Z_0) \}$$

$$\delta\ (q, 1, X) = \{ (q, X X) \}$$

$$\delta\ (q, 0, X) = \{ (p, X) \}$$

$$\delta\ (q, \in, X) = \{ (q, \in) \}$$

$$\delta\ (p, 1, X) = \{ (p, \in) \}$$

$$\delta\ (p, 0, Z_0) = \{ (q, Z_0) \}$$

Fig 3: -CFG to PDA converter [25]

IV. ADVANTAGES AND DISADVANTAGES OF EXISTING SYSTEM

A PDA recognizes a string, as per the first, when it achieves its final state after reading the entire string. Per the second, a PDA validates a string when it has read the entire string and has emptied its stack [13].

Defining the CFG to PDA transfer process may create an incremental parsing method for any CFG [13]. When the NPDA is approached with a non-deterministic decision, a retracing search can be prepared; if the NPDA is confronted with a non-deterministic decision, a piece of code is organised that iteratively hunts for one of the paths to approve [13]. The stacked alphabet of the PDA produced in the CFG to PDA conversion procedure is formed by the non-terminals and terminals of the supplied grammar [13].

It could be said that alike LSTM and Transformer networks visualise context-free languages with constrained iteration and similar representation power. However, the disadvantage of the LSTM model is that it fails to factorise its innate area to encrypt the state and numerous aspects of the stack without clear and specific guidance, which is the main pillar to its vulnerable results in real-time projects and practice [12]. However, the lack of explicit breakdown normalisation has a slight effect on the Transformer [12]. Since the results are sensitive to the configurations of PDAs, the working practice of the models, and the standard operating procedures, language identification is not a real job to evaluate the factual ability of LSTM and Transformer [12].

V. PROPOSED SYSTEM

PDA cannot recognize the context-sensitive and unrestricted alphabet, it can only recognize the context-free alphabet so we need to find a way in which PDA will recognize the context-sensitive and unrestricted alphabet. Every-deterministic PDA cannot be converted to a deterministic PDA. Indeed, if we've got a nondeterministic PDA that is guaranteed to have a deterministic original there is no mechanical procedure to find it.

CONCLUSION

An adequate fit of method graphs under language similarity is called a language. A pushdown automaton's set of languages matches the set of languages provided by either context-free grammar. Integration of automata theory and process theory is a project we're working on right now. As a result, we may utilise an interactive computational model to demonstrate the fundamentals of computer science. The use of formal languages theories in software engineering will aid in improving the overall accuracy of the measurements indicated. Based on several versions of minimization of conventional nondeterministic automata, minimal automata can be created. One can create a separate expense function that considers symbol frequencies and reduces the number of bits even.

REFERENCES

[1] Querying graph databases using context-free grammarsCiro M.Medeiros, Martin A.Musicante, Umberto S.Costa, 2022.*)*

[2] PUSHDOWN AUTOMATA AND CONTEXT-FREE GRAMMARS IN BISIMULATION SEMANTICS , JOS C. M. BAETEN, CESARE CARISSIMO , AND BAS LUTTIK, 2022

[3] Digging input-driven pushdown automata, Martin Kutrib and Andreas Malcher, 2021

[4] Context Free Grammar Identification from Positive Samples, 1K.Senthil Kumar, 2D.Malathi, 2018

[5] NLP : Context Free Grammars and Parse Trees for Disambiguating Telugu Language Sentences Jinka Sreedhar, Dr Sk Althaf Hussain Basha , D. Praveen Kumar, A.Jagan, Baijnath Kaushik, 2017

[6] A Context Free Grammar for Requirements Modeling Michelle P. Banawan, 2012

[7] Automatic Learning of Context-Free Grammar, Tai-Hung Chen, Chun-Han Tseng, Chia-Ping Chen, 2006

[8] A Bit of Nondeterminism Makes Pushdown Automata Expressive and Succinct, Shibashis Guha, Isma¨el Jecker, Karoliina Lehtinen and Martin Zimmermann, 2021

[9] Unary Context-Free Grammars and Pushdown Automata, Descriptional Complexity and Auxiliary Space Lower Bounds, Giovanni Pighizzini , Jeffrey Shallit, Ming-wei Wang, 2002

[10] Pushdown Automata in Statistical Machine Translation, Cyril Allauzen, Bill Byrne, Adri`a de Gispert, Gonzalo Iglesias, Michael Riley, 2013.

[11] PDA Simulator for CFG Induction Using Genetic Algorithm

[12] Learning Bounded Context-Free-Grammar via LSTM and the Transformer:Difference and Explanations, Hui Shi, Sicun Gao, Yuandong Tian, Xinyun Chen, Jishen Zhao, 2021

[13] Push-down Automata and Context-free Grammars.

[14] Vayadande, Kuldeep, Ritesh Pokarne, Mahalaxmi Phaldesai, Tanushri Bhuruk, Tanmai Patil, and Prachi Kumar. "SIMULATION OF CONWAY'S GAME OF LIFE USING CELLULAR AUTOMATA." International Research Journal of Engineering and Technology (IRJET) 9, no. 01 (2022): 2395-0056.

[15] Vayadande, Kuldeep, Ram Mandhana, Kaustubh Paralkar, Dhananjay Pawal, Siddhant Deshpande, and Vishal Sonkusale. "Pattern Matching in File System." International Journal of Computer Applications 975: 8887.

[16] Vayadande, Kuldeep, Neha Bhavar, Sayee Chauhan, Sushrut Kulkarni, Abhijit Thorat, and Yash Annapure. Spell Checker Model for String Comparison in Automata. No. 7375. EasyChair, 2022.

[17] VAYADANDE, KULDEEP. "Simulating Derivations of Context-Free Grammar." (2022).

[18] Vayadande, Kuldeep, Neha Bhavar, Sayee Chauhan, Sushrut Kulkarni, Abhijit Thorat, and Yash Annapure. Spell Checker Model for String Comparison in Automata. No. 7375. EasaafyChair, 2022.

[19] Varad Ingale, Kuldeep Vayadande, Vivek Verma, Abhishek Yeole, Sahil Zawar, Zoya Jamadar. Lexical analyzer using DFA, International Journal of Advance Research, Ideas and Innovations in Technology, www.IJARIIT.com.

[20] Kuldeep Vayadande, Harshwardhan More,Omkar More, Shubham Mulay,Atahrv Pathak, Vishwam Talanikar, “Pac Man: Game Development using PDA and OOP”, International Research Journal of Engineering and Technology (IRJET), e-ISSN: 2395-0056, p-ISSN: 2395-0072, Volume: 09 Issue: 01 | Jan 2022, www.irjet.net

[21] Kuldeep B. Vayadande, Parth Sheth, Arvind Shelke, Vaishnavi Patil, Srushti Shevate, Chinmayee Sawakare, “Simulation and Testing of Deterministic Finite Automata Machine,” International Journal of Computer Sciences and Engineering, Vol.10, Issue.1, pp.13-17, 2022.

[22] Rohit Gurav, Sakshi Suryawanshi,Parth Narkhede,Sankalp Patil,Sejal Hukare,Kuldeep Vayadande,” Universal Turing machine simulator”, International Journal of Advance Research, Ideas and Innovations in Technology, ISSN: 2454-132X, (Volume 8, Issue 1 - V8I1-1268, https://www.ijariit.com/

[23] Kuldeep Vayadande, Krisha Patel, Nikita Punde, Shreyash Patil, Srushti Nikam, Sudhanshu Pathrabe, “Non-Deterministic Finite Automata to Deterministic Finite Automata Conversion by Subset Construction Method using Python,” International Journal of Computer Sciences and Engineering, Vol.10, Issue.1, pp.1-5, 2022.

[24] Kuldeep Vayadande and Samruddhi Pate and Naman Agarwal and Dnyaneshwari Navale and Akhilesh Nawale and Piyush Parakh,” Modulo Calculator Using Tkinter Library”, EasyChair Preprint no. 7578, EasyChair, 2022

[25] CFG to PDA Converter https://www.chegg.com/homework-help/questions-and-answers/draw-pda-grammar-convert-cfg-addressed-pda-p-q-p-q-0-1-g-x-z0-q-f-given-q16290065