Open Journal of Discrete Mathematics, 2012, 2, 105-108
http://dx.doi.org/10.4236/ojdm.2012.23020 Published Online July 2012 (http://www.SciRP.org/journal/ojdm)

An Entertaining Example of Using the Concepts of Context-Free Grammar and Pushdown Automation

Krasimir Yordzhev
Faculty of Mathematics and Natural Sciences, South-West University, Blagoevgrad, Bulgaria
Email: yordzhev@swu.bg

Received April 13, 2012; revised May 10, 2012; accepted June 4, 2012

ABSTRACT

A formal-linguistic approach for solving an entertaining task is offered in this paper. The well-known task of the Hanoi towers is discussed in relation to some concepts of formal languages and grammars. A context-free grammar which generates an algorithm for solving this task is described. A deterministic pushdown automation which in its work imitates the work of monks in solving the task of the Hanoi towers is built.

Keywords: Context-Free Grammar; Context-Free Language; Pushdown Automation; Hanoi Towers; Discrete Mathematics Learning

1. Introduction

Task 1 (The Task of the Hanoi Towers [1]). *The Hanoi Towers are made up of three vertical pins. A series of N discs is hung on the first pin. The discs are all different, but ordered by size with the largest being on the bottom and the smallest on top. The task is to move the discs from the first to the third pin, using the second pin as an assistant. There are several conditions to completing this exercise: only one disc may be moved at one time and while one disc is being moved, all other discs must be on one of the pins and also, during this time, it is prohibited for a larger disc to be placed on a smaller one.*

The task of the Hanoi Towers is a classic example used to teach recursion in programming [1-4]. In this paper, we will look at this Task from the standpoint of mathematical linguistics, *i.e.* as a part of the discipline of "discrete mathematics" and "mathematical linguistics" studies by students in Informatics and Computer courses at university [2,5-9].

There are three interesting approaches associated with the task of Hanoi towers in terms of mathematical linguistics:

1) To generate the Hanoi moves using a finite automaton;

2) To generate the Hanoi moves using a context-free grammar;

3) To generate the Hanoi moves using a pushdown automation.

The first approach is described in [10] and an equivalent version (with morphisms) in [11]. In this paper we will consider the second and third tasks. These tasks have been formulated in a Bulgarian in the textbook [12].

The algebraic properties of context-free grammars and languages are discussed in [5,8,13,14]. Several applications of formal grammars and languages and pushdown automata are considered in [8,15].

2. Context-Free Grammars and Languages

Let V be a finite and non-empty set. The elements of this set are called *letters*, and the whole set V—*alphabet*.

We will call a *word over the alphabet* V each finite string of letters from V. A word that does not contain any letter is called an *empty word*, which we will mark with ε. V^* denotes the set of all words over V, including empty set. The term *length of a word* refers to the number of letters in it. The length of the word α will be expressed by $|\alpha|$.

Let α and β be two words over the Alphabet V. By *concatenation* (*multiplication*) $\alpha\beta$ of both words we will mean the word obtained by successive completion of the letters of β after the last letter of α.

Let V be an alphabet. Each subset L of V^* is called *formal language* (or only *language*) over alphabet V.

By *generative grammar* (or only *grammar*) Γ, we will understand the four ordered tuples $\Gamma = \langle V, W, S, P \rangle$, where V is a finite set (Alphabet) of *terminal* symbols, W—a set of *nonterminal* symbols, S—a *start symbol* of the grammar, which is an element of W, and P is a set of ordered pairs (α, β), where $\alpha, \beta \in (V \cup W)^*$, with at least there one non-terminal symbol in α. In a number of sources (see references at the end), an additional condition is placed for sets W and P to be finite. For our

needs this condition is not necessary. It is enough that these sets are countable. The elements of P are called *productions*. If $(\alpha,\beta)\in P$, then it means $\alpha\to\beta$, as the symbol "$\to$" does not belong to $V\cup W$.

Let μ and ν be two words from $(V\cup W)^*$. We will say that μ is *derived directly* from ν in the grammar $\Gamma=\langle V,W,S,P\rangle$ and will write $\nu\overset{\Gamma}{\vdash}\mu$ (or only $\nu\vdash\mu$, if Γ is understandable), if there exists words $\alpha_1,\alpha_2\in(V\cup W)^*$ and a production $\alpha\to\beta$ in P so that $\nu=\alpha_1\alpha\alpha_2$ and $\mu=\alpha_1\beta\alpha_2$.

If $\omega_0,\omega_1,\cdots,\omega_n$ is a word over $V\cup W$, for which $\omega_0\overset{\Gamma}{\vdash}\omega_1\overset{\Gamma}{\vdash}\cdots\overset{\Gamma}{\vdash}\omega_n$, we will say that the number of words is *the derivation* of ω_n from ω_1 in Γ, which we denote by $\omega_0\overset{\Gamma}{\models}\omega_n$ or only $\omega_0\models\omega_n$, if Γ is default. The count n of the immediate derivations $\omega_i\overset{\Gamma}{\vdash}\omega_{i+1}$ will be called *the length of derivation*.

The Set $L(\Gamma)=\left\{\omega\in V^*\middle|S\overset{\Gamma}{\models}\omega\right\}$ is called *formal language over V, generated by the grammar* Γ. The Grammars Γ_1 and Γ_2 are *equivalent* if $L(\Gamma_1)=L(\Gamma_2)$.

A grammar $\Gamma=\langle V,W,S,P\rangle$ is *context-free*, if all the productions are of the type

$$A\to\omega,\quad A\in V,\quad \omega\in(V\cup W)^*,$$

where V and W are alphabets, respectively with terminal and nonterminal symbols.

Task 2. *For a given positive integer N a context-free grammar* Γ_N *should be built with a terminal alphabet encoding possible displacements and if* $\omega\in L(\Gamma_N)$, *then* ω *describes the algorithm that solves the Task* 1. *Prove that for each positive integer N language* $L(\Gamma_N)$ *is not empty, i.e. for each positive integer N there is an algorithm that solves the task of the Hanoi towers*[1].

Solution. Let's consider context-free grammar $\Gamma_N=\langle V,W,S,P\rangle$ where $V=\left\{p_{ij}\middle|i,j\in\{1,2,3\},i\neq j\right\}$. The meaning of p_{ij} is "Move top disk from the *i*-th pin of the *j*-th pin". In this way, if $\omega=\pi_1\pi_2\cdots\pi_k$, where $\pi_i\in V,\ \ i=1,2,\cdots,k$, then ω describes the algorithm for the consecutive moving of k discs in the three pins.

$W=\left\{h_{ij}(n)\middle|i,j\in\{1,2,3\},\ i\neq j,\ n=1,2,\cdots,N\right\}$, the start symbol $S=h_{13}(N)$, P consists of productions $h_{ij}(1)\to p_{ij}$ and $h_{ij}(n)\to h_{ik}(n-1)\,p_{ij}\,h_{kj}(n-1)$ for $n=2,3,\cdots,N$, where $i,j,k\in\{1,2,3\},\quad i\neq j,\quad i\neq k,\ k\neq j$. Apparently, the grammar Γ_N constructed in this manner is context-free.

Let $\omega\in L(\Gamma_N)$, *i.e.* we assume that the derivation $h_{13}(N)\overset{\Gamma_N}{\models}\omega$ exists and let the length of the derivation is equal to s. If $s=1$, then obviously this is possible if and only if the number of discs $N=1$ and there is a direct derivation $h_{13}(1)\overset{\Gamma_N}{\vdash}p_{13}$ and in the presence of a single disc, p_{13} describes an algorithm for solving Task 1. Similarly $h_{ij}(1)\overset{\Gamma_N}{\vdash}p_{ij}$ is a derivation with length 1 with start symbol $h_{ij}(1)$ and p_{ij} describes the algorithm for moving a single disc from pin i to pin j, where $i,j\in\{1,2,3\}$ and $i\neq j$. Let $s>1$. We assume that if a derivation exists with length less than s of type $h_{ij}(n)\overset{\Gamma_N}{\models}\alpha$, where $\alpha\in V^*,\ \ n\le N$ then α describes the moving of *n* discs from pin i to pin j, using pin *k* as assistant according to the constraints in Task 1, where $i,j,k\in\{1,2,3\}$, $i\neq j,\ j\neq k,\ k\neq i$. When $s<1$ obviously derivation $h_{13}(N)\overset{\Gamma_N}{\models}\omega$ with length *s* (if existing) will be of the type $h_{13}(N)\overset{\Gamma_N}{\vdash}h_{12}(N-1)\,p_{13}h_{23}(N-1)\overset{\Gamma_N}{\models}\omega$ then the next derivations $h_{12}(N-1)\overset{\Gamma_N}{\models}\omega_1$ and $h_{23}(N-1)\overset{\Gamma_N}{\models}\omega_2$ exist with lengths less then *s*, where $\omega_1,\omega_2\in V^*$ and $\omega=\omega_1p_{13}\omega_2$. According to the induction assumption, ω_1 describes the algorithm for moving of $N-1$ discs from the first to the second pin, using the third one as assistant, and ω_2 describes the algorithm for moving $N-1$ discs from the second to the third pin, using thr first one as assistant. Then $\omega=\omega_1p_{13}\omega_2$ describes the following algorithm: first under the constraints of Task 1 we move the top number t of discs from the first pin to the second, then we move the largest bottom disk from the first pin of the empty third and finally, we move t number of discs from the second pin to the third one. Therefore $\omega=\omega_1p_{13}\omega_2$ (if existing) describes the solution of the task of the Hanoi towers.

Let's prove for each positive integer N that language $L(\Gamma_N)$ is not empty. When $N=1$, the only production of P which can be applied is $h_{13}\to p_{13}$ and therefore $L(\Gamma_1)=\{p_{13}\}$, *i.e.* $L(\Gamma_1)$ is not an empty language. Let's assume that for each positive integer $t\le N$ the languages $L(\Gamma_t)$ are not empty and put $N=t+1$. Let's consider the context-free grammars $\Gamma_{t'}=\langle V,W,h_{12}(t),P\rangle$ and $\Gamma_{t''}=\langle V,W,h_{23}(t),P\rangle$. Apparently $\Gamma_{t'}$ and $\Gamma_{t''}$ work by analogy of Γ_t and according to the above proven if $\omega'\in L(\Gamma_{t'})$, then ω' describes the algorithm for moving t discs from the first tos the second pin, using third one as assistant under the constraints described in Task 1, and if $\omega''\in L(\Gamma_{t''})$, then ω'' describes the algorithm for moving t discs from the second pin to the third one using the first one as assistant. According to the induction assumption ω' and ω'' there exists. Then in Γ_{t+1} derivation exist

[1]It is not necessary $L(\Gamma_N)$ to describe all solutions of Task 1. Some of them may be ineffective, for instance if they involve the useless relocation of disk as soon as moving the same disk to another pin.

$h_{13}(t+1) \overset{\Gamma_{t+1}}{\vdash} h_{12}(t)p_{13}h_{23}(t) \overset{\Gamma_{t+1}}{\models} \omega' p_{13} \omega''$, where $\omega' \in L(\Gamma_{t'})$ and $\omega'' \in L(\Gamma_{t''})$. Therefore $\omega \in L(\Gamma_{t+1})$, *i.e.* $L(\Gamma_{t+1})$ is a non empty language. □

When $N=5$ the following word is produced p_{13} p_{12} p_{32} p_{13} p_{21} p_{23} p_{13} p_{12} p_{32} p_{31} p_{21} p_{32} p_{13} p_{12} p_{32} p_{13} p_{21} p_{23} p_{13} p_{21} p_{32} p_{31} p_{21} p_{23} p_{13} p_{12} p_{32} p_{13} p_{21} p_{23} p_{13}. We can verify the correctness of the algorithm using the example of the five consecutive playing cards.

The following is easy to prove (e.g. using induction):

Proposition 1. *Let N be a positive integer, and* Γ_N *be defined as the solution of Task* 2 *context-free grammar, then*

$$\left|L(\Gamma_N)\right| = 1$$

and if $\omega \in L(\Gamma_N)$, then

$$|\omega| = 2^N - 1$$

In other words, for each positive integer *N*, grammar Γ_N generates exactly one word that describes an algorithm for solving the task of the Hanoi towers with exactly $2^N - 1$ displacements of the disks from one pin to another.

3. Pushdown Automata

By *nondeterministic pushdown automation* one will understand each ordered septuple

$$M = \langle K, V, W, \delta, q_0, z_0, F \rangle,$$

where:

- *K* is a finite set of *states* of automaton;
- *V* is a finite set of entry letters (*entry alphabet*);
- *W* is a finite, non empty set of *stack symbols* (*stack alphabet*);
- $\delta : K \times (V \cup \{\varepsilon\}) \times W \to P(K \times W^*)$ is *a transition function*;[2]
- $q_0 \in K$ is *a start state* of automaton;
- $z_0 \in W$ is *a start* stack symbol;
- $F \subseteq K$ is a set of *accepting* states.

The ordered triple $\langle q, \alpha, \gamma \rangle \in K \times V^* \times W^*$ will be called *configuration* of nondeterministic pushdown automaton *M*.

Let $\alpha = a_1 a_2 \cdots a_s \in V^*$, $\gamma = z_1 z_2 \cdots z_t \in W^*$. Then the transition function δ defines a transition configuration $\langle q, \alpha, \gamma \rangle$ to the next configuration in the following way:

1) For each pair $\langle p, \gamma' \rangle \in \delta(q, a_1, z_1)$ the configuration $\langle q, \alpha, \gamma \rangle$ passes in the configuration $\langle p, \alpha_1, \gamma_1 \rangle$, where $\alpha_1 = a_2 a_3 \cdots a_s$, $\gamma_1 = \gamma' z_2 z_3 \cdots z_t$, which we denote by $\langle q, \alpha, \gamma \rangle \vdash \langle p, \alpha_1, \gamma_1 \rangle$.

2) For each pair $\langle p, \gamma' \rangle \in \delta(q, \varepsilon, z_1)$ the configuration $\langle q, \alpha, \gamma \rangle$ passes in the configuration $\langle p, \alpha, \gamma' z_2 z_3 \cdots z_t \rangle$, which we denote by $\langle q, \alpha, \gamma \rangle \vdash \langle p, \alpha, \gamma' z_2 z_3 \cdots z_t \rangle$.

If the nondeterministic pushdown automation is initially given the word $\alpha_1 = a_2 a_3 \cdots a_s$, then, according to the start configuration $\langle q_0, \alpha, z_0 \rangle$, the following possible configurations are obtained by using a function of transitions δ. For each new configuration using δ all possible next configurations are obtained and so on.

The nondeterministic pushdown automation *M recognizes the word* $\alpha_1 = a_2 a_3 \cdots a_s$, *by accepting state*, if its work at the beginning of given word α, it reaches a configuration of type $\langle q, \varepsilon, \gamma \rangle$, for each $\gamma \in W^*$, when $q \in F$.

The nondeterministic pushdown automation *M recognizes the word* $\alpha_1 = a_2 a_3 \cdots a_s$, *by empty stack,* if its work at the beginning of a given word α reaches a configuration of type $\langle q, \varepsilon, \varepsilon \rangle$.

The pushdown automation $M = \langle K, V, W, \delta, q_0, z_0, F \rangle$ is called *deterministic*, if for each $q \in K$ and $z \in W$ exactly one of the following two conditions is valid:

1) $\delta(q, a, z)$ contains no more than one element for each $a \in V$ and $\delta(q, \varepsilon, z) = \varnothing$.

2) $\delta(q, a, z) = \varnothing$ for each $a \in V$ and $\delta(q, \varepsilon, z)$ contains no more than one element.

A language which is recognized by some deterministic pushdown automation is called a *deterministic* language. As it is known the relationship between context-free languages and pushdown automation is given by the following statements:

For each context-free language *L* a nondeterministic pushdown automation *M* exists, such that *L* is recognized by *M* through an accepting state.

Language *L* is recognized of a nondeterministic pushdown automation through an empty stack if and only if *L* is recognized of a nondeterministic pushdown automation through an accepting state.

If *L* is a language which is recognized by a nondeterministic pushdown automation, then *L* is a context-free language.

Task 3. *For each positive integer* $N \geq 2$ *a deterministic pushdown automation* M_N *should be built, which in its work should imitate the work of monks in solving the task of the Hanoi towers* (*see Task* 1).

Solution. The requested pushdown automation is the following: $M = \langle K, \varnothing, W, \delta, q_0, z_0, \varnothing \rangle$, where $K = \{q_0\}$, $W = \{p_{ij} \mid i \neq j, i, j = 1,2,3\}$ $\cup \{h_{ij}(n) \mid i \neq j, i, j = 1,2,3, n = 1,2,\cdots, N-1\} \cup \{z_0\}$, and $\varnothing$ is the empty set. Let $i, j, k \in \{1,2,3\}$, $i \neq j$, $j \neq k$, $k \neq i$. Then the transition function δ is defined in following way:

$$\delta(q_0, \varepsilon, z_0) = \langle q_0, h_{12}(N-1)\, p_{13} h_{23}(N-1) \rangle \qquad (1)$$

[2]As usual with *P*(*A*) is denoted the set of all subsets of the set *A*, including the empty.

$$\delta\left(q_0,\varepsilon,h_{ij}(1)\right)=\left\langle q_0,p_{ij}\right\rangle \quad (2)$$

$$\delta\left(q_0,\varepsilon,h_{ij}(n)\right)=\left\langle q_0,h_{ik}(n-1)\,p_{ij}h_{kj}(n-1)\right\rangle,\quad n=2,3,\cdots,N \quad (3)$$

$$\delta\left(q_0,\varepsilon,p_{ij}\right)=\left\langle q_0,\varepsilon\right\rangle. \quad (4)$$

Immediately after the inclusion of M_N, before being submitted as any input signal, the automation replaces the start stack symbol z_0 with word $h_{12}(N-1)\,p_{13}h_{23}(N-1)$ according to (1) and after a number of actions depending on the current stack symbol (2), (3), or (4). Moreover, we assume that automation M_N is designed so that after the reading of the stack symbol of the type p_{ij} $i,j\in\{1,2,3\}$, $i\neq j$, simultaneously with the action according to (4) another action is carried out, namely the removal of top disk i-th pin on j-th. Transient function is defined so that after a finite number of beats the stack is empty and stops. This occurs because if the current stack symbol of the kind p_{ij}, then M_N deletes it, and if the current stack symbol is of the type $h_{ij}(n)$, then at the next beat of the parameter n decreases by one unit, if $n>1$, or $h_{ij}(n)$ passes into the symbol p_{ij} at $n=1$, then that symbol is deleted.

We will prove that $1\le s<N$ the stack M_N of configuration $\left\langle q_0,\varepsilon,h_{ij}(s)\gamma\right\rangle$ where $i,j\in\{1,2,3\}$, $i\neq j$, $\gamma\in W^*$ as a result of their work reaches a configuration $\left\langle q_0,\varepsilon,\gamma\right\rangle$ and in the process it transfers, according to the restrictions of Task 1, s discs from pin i to pin j. When $s=1$ we have $\left\langle q_0,\varepsilon,h_{ij}(1)\gamma\right\rangle\vdash\left\langle q_0,\varepsilon,p_{ij}\gamma\right\rangle\vdash\left\langle q_0,\varepsilon,\gamma\right\rangle$, *i.e.* the assertion is met. Let's assume that the assertion is fulfilled for any t, such that $1\le t<s<N$ and let $t=s$. Then in $i,j,k\in\{1,2,3\}$, $i\neq j$, $j\neq k$, $k\neq i$ we have $\langle q_0,\varepsilon,h_{ij}(t)\gamma\rangle\vdash\left\langle q_0,\varepsilon,h_{ik}(t-1)\,p_{ij}h_{kj}(t-1)\gamma\right\rangle$. According to the induction assumption, M_N reaches the configuration $\left\langle q_0,\varepsilon,p_{ij}h_{kj}\gamma\right\rangle$ moving $t-1$ discs from i-th pin of k-th pin after which it passes in a configuration $\left\langle q_0,\varepsilon,h_{kj}(t-1)\gamma\right\rangle$ moving the next disc from pin i to pin j and again according to the induction, it moves $t-1$ discs (as all are obviously smaller size) on this disk on pin j which are taken from pin k. Therefore the assertion is true for any $s=1,2,\cdots,N-1$.

According to the assertion that has just been just proved, we have:

$$\begin{aligned}\left\langle q_0,\varepsilon,z_0\right\rangle&\vdash\left\langle q_0,\varepsilon,h_{12}(N-1)\,p_{13}h_{23}(N-1)\right\rangle\vdash\cdots\\&\vdash\left\langle q_0,\varepsilon,p_{13}h_{23}(N-1)\right\rangle\\&\vdash\left\langle q_0,\varepsilon,h_{23}(N-1)\right\rangle\vdash\ldots\vdash\left\langle q_0,\varepsilon,\varepsilon\right\rangle\end{aligned},$$

while the stack moves to the upper $N-1$ discs from the first to the second pin, then it moves the biggest at the bottom from the first to the third pin and finally it moves discs ($N-1$ numbers from the second to the third pin observing the restrictions described in Task 1). Therefore the pushdown automation M_N solves the task of the Hanoi towers.

REFERENCES

[1] J. Arsac, "Jeux et Casse—Tête a Programmer," BORDAS, Paris, 1985.

[2] A. V. Anisimov, "Recursive Information Transducers," Vishcha Shkola, Kiev, 1987.

[3] A. V. Anisimov, "Informatics, Creativity, Recursion," Naukova Dumka, Kiev, 1988.

[4] N. Wirth, "Algorithms + Data Structures = Programs," Prentice Hall, Boston, 1976.

[5] I. Chiswell, "A Course in Formal Languages, Automata and Groups," Springer-Verlag, London, 2009. doi:10.1007/978-1-84800-940-0

[6] J. Denev, R. Pavlov and I. Demetrovich, "Discrete Mathematics," Science and Art, Soa, 1984.

[7] J. Denev and S. Shtrakov, "Discrete Mathematics," South-West University "N. Rilski", Blagoevgrad, 1995.

[8] J. E. Hopcroft, R. Motwani and J. D. Ullman, "Introduction to Automata Theory, Languages, and Computation," Addison-Wesley, Boston, 2001.

[9] K. Manev, "Introduction in Discrete Mathematics," KLMN, Soa, 2003.

[10] J.-P. Allouche and F. Dress, "Tours de Hanoï et Automates," *Informatique Théorique et Applications*, Vol. 24, No. 1, 1990, p. 1.

[11] J.-P. Allouche and J. Shallit, "Automatic Sequences: Theory, Applications, Generalizations," University Press, Cambridge, 2003. doi:10.1017/CBO9780511546563

[12] S. Shtrakov, K. Yordzhev and M. Todorova, "Guide for Solving of Tasks in Discrete Mathematics," South-West University "N. Rilski", Blagoevgrad, 2004.

[13] S. Ginsburg, "The Mathematical Theory of Context-Free Languages," McGraw-Hill, Boston, 1966.

[14] G. Lallemant, "Semigroups and Combinatorial Applications," John Wiley & Sons, Hoboken, 1979.

[15] A. V. Aho and J. D. Ullman, "The Theory of Parsing, Translation and Computing," Prentice-Hall, Upper Saddle River, 1972.

