EJTAS
ISSN 2786-7447
EUROPEAN JOURNAL OF THEORETICAL AND APPLIED SCIENCES
OPEN ACCESS
PEER REVIEWED

Language Outlay for the Sokoto Cement Production System Using Deterministic Finite Automata Scheme

Zaid Ibrahim
Department of Mathematics, Sokoto State University, Sokoto, Nigeria

Suggested Citation
Ibrahim, Z. (2023). Language Outlay for the Sokoto Cement Production System Using Deterministic Finite Automata Scheme. *European Journal of Theoretical and Applied Sciences, 1*(6), 105-114.
DOI: 10.59324/ejtas.2023.1(6).10

Abstract:

This paper constructs the compact, detailed and extended models, and focuses on the algebraic theoretic strings and language of each transition of the finite automata scheme. It was discovered that from the initial stage to the final stage of cement production processes, each transition or production process can have a particular language. In addition, a language scheme is developed for each of the sub-states (sub-model) that leads to a theoretic study of language scheme and semantics of the model. It can be deduced that when represented as binary codes, the established schemes in the sub-states can be studied as a Boolean algebraic scheme.

Keywords: *Finite automata, Deterministic, String, Word, Alphabet, Language, Compact model, Detailed model, Extended model.*

Introduction

Model theory deals with the relations between the properties of sentences or sets of sentences specified in a formal language on one hand, and of the mathematical structures or sets of structures, which satisfy these sentences upon referred to as algebraic theories of fields, rings, groups, etc. (Robinson, 1965).

On the other hand, an algebraic theory is a small category T with finite products while an algebra for the theory T is a function A : T →Set, preserving finite products (Adámek, Rosický, Vitale, & Lawvere, 2010). Thus, it can be said that there is an interesting relationship between algebra and algebraic theory for which algebra is a system of constructs based on a defined function on sets while an algebraic theory represents study of the collection of attributes associated with that collection.

Finite Automata

The term 'finite automata' describes a class of models of computation that are characterized by having finite states. The use of the word 'automata' harks back to the early days of the subject in the 1950's when they were viewed as abstract models of real circuits (Lawson, 2005). A Finite-State Automaton (plural: automata), or simply a State Machine, is conceived as an abstract machine that can be in one of a finite number of states, the machine is in only one state at a time; the state it is in at any given time is called the current state. It can change from one state to another when initiated by a triggering event or condition and this is called a transition. A particular Finite State Machine is defined by a list of its states, and the triggering condition for each transition.

A typical production system is composed of multiple machines and workstations that perform various operations on a part of

production, and a material handling system that interconnect these workstations. Parts are processed to completion by transiting them through various machines and workstations according to their individual process plan. After processing is complete the parts leave the system and proceeds to the next state until the final state of production is reached (Yalcin, Tai, & Boucher, 2004).

Typical Production System of Sokoto Cement

The Production Process of Sokoto Cement begins from the quarry where the major raw material (limestone) is excavated. After geological test and survey has confirmed a particular piece of land containing limestone, the excavation process begins on the soil. The topmost part of the quarry area contains sand, laterite and marl that are removed to properly get to the limestone. Limestone can be categorized into two categories; the high grade and the low-grade limestone, the identification of these different grades of limestone can be noticed physically. The high grade is harder and brighter while the low grade is darker and softer. The required amount of calcium carbonate to be present in a limestone to make it a high grade is 80% upward while in low grade is 67%- 75% (Quality Control Department, CCNN, n.d.).

The limestone is further crushed in to smaller particles that will ease the process of production and packed into piles (Fitsum Consultancy, 2008). It has been shown by Zaid, et al., (2014A) that there are four mixing beds in which the high-grade limestone is stockpiled in beds 1 and 2 with different concentration of calcium carbonate while the low-grade limestone is piled in beds 3 and 4 also with different concentration of calcium carbonate. Here, the limestones in the mixing beds are moved to the next stage of production by means of a reclaimer and conveyer belts to Silos in the Raw Mill where the targeted quality concentration mixing of calcium carbonate of 74.5% – 75% is carried out. It is at this point the raw material is ready for the next stage of production, which is the clinker process. At this stage the raw material is passed into the cyclone where it is preheated (at 680^0C), the production further proceeds to the rotary kiln where the clinker (a grain like substance) is produced. After the production of clinker, it is passed into the cooler where it is cooled and ready for the next state of production process.

After the clinker is cooled, it is further transferred to the cement mill where it takes another phase of production. The clinker is crushed and mixed with additives into the grinder to produce the powder, which is the required cement. Although we have the required cement but that does not mean that it is the final state of production. The required cement is moved to the final state of packaging and bulk dispatching which complete the cement production process (Quality Control Department, CCNN, n.d.).

Using Finite Automata in Modeling Cement Production System

The idea of using finite automata (FA) in modeling the production system of cement is strictly based on the fact that the system has finite states of production process with a finite link from a state to the other expressed in terms of machine sequence. A detailed approach to modeling a cement production process should include all stages of production and transitions from the raw material to the finished cement (Garcia & Vidal, 1990). In particular, finite automata models have been used to model and develop a control for manufacturing systems (Kim, Shin, et al., 2010).

In cement production process, the state of production changes after each instruction is executed, and each state is completely determined by the prior state and the input. The production simply starts at an initial state, changes from state to state until it reaches the final state (O'Castillo & Tapia, 1999).

Preliminaries

The paper presents the algebraic theoretic applications of the designed models in relation to the states/stages to be followed from the initial stage to the final stage of the production processes of the Sokoto cement. A compact

model with its concatenation was developed in terms of algebraic language. Similarly, the paper constructs some algebraic languages from the detailed constructed deterministic finite automata model. In the same manner, the paper developed languages of the particular activity/stage of the main processes (quarry, Crushing, Raw Milling, Preheating and Finished cement) as extended. In this paper, strings over an alphabet of any finite length, concatenation of string and the strings that bring the automata to an accepting state was considered as words and subwords of the language.

Finite Automata as an Algebraic Model

The two major elements in a Finite Automata are states and inputs, starting with the initial (start) state the process can be changed from one state to another up to the final state. A model of this tye of rule-based recursive action or machine where the number of states and inputs are finite and can only be in one state at a time is called the deterministic finite automata (Gribkoff, 2013), This model can be summarized as follows:

a. A finite set of states.
b. A finite set of inputs.
c. A finite set of transitions

According to (Lawson, 2005), a deterministic finite automaton (DFA) consists of;

1. a finite set of states (often denoted as Q)
2. a finite set Σ of symbols (alphabet)
3. a transition function that takes as argument a state and a symbol and returns a state (often denoted as δ)
4. a start state (often denoted q_0)
5. a set of final or accepting states (often denoted as F)

We have $q_0 \in Q$ and $F \subseteq Q$

So, a DFA is Mathematically represented as a 5-tuple $(Q, \Sigma, \delta, q_0, F)$.

The transition function δ is a function in $Q \times \Sigma \rightarrow Q$ and $Q \times \Sigma$ is the set of 2-tuples (q, a) with $q \in Q$ and $a \in \Sigma$

Gribkoff, (2013), focused on examining real-world applications of DFAs to gain an appreciation of the usefulness of this theoretical concept and these applications include protocol analysis, text parsing, video game, security analysis, CPU control units, natural language processing and speech recognition. He also ascertained that, mechanical devices are frequently designed and implemented using DFAs, such as elevators, vending machines, mealy machines and traffic lights. According to Hopcroft, Motwani, & Ullman, (1979), a deterministic finite automaton (DFA) known as deterministic finite state machine is a finite state machine that accepts/rejects finite strings of symbols and only produces a unique computation of the automaton for each input string.

Deterministic Finite Automata (DFA)

If $Q = \{q_0, q_1, q_2\}$ where q0 is the initial state and $F = \{q_1\}$ is the final state.

The transition function δ is a function in $Q \times \Sigma \rightarrow Q$, so if the alphabet $\Sigma = \{0,1\}$, then

$\delta:(q_0,1) = q_0$

$\delta:(q_0,0) = q_2$

This defines the following state diagram and its transition table as in table 1.

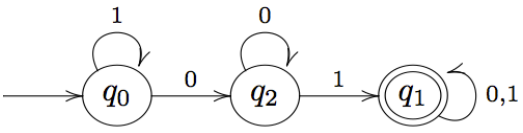

Figure 1. Transition Diagram

Table 1. Transition Table

		0	1
→	q_0	q_2	q_0
	*q_1	q_1	q_1
	q_2	q_2	q_1

The * indicates the final state(s) and in this case there is only one final state q_1.

Strings and Language

A string over Σ is a finite-length sequence of elements of Σ, also the set of all strings over Σ is denoted by Σ^* and for a string x, |x| is the length of x. The unique string of length 0 will be denoted by ε and will be called the empty or null string and a language over Σ is a set of strings over Σ and is denoted by L.

Let $w = a_1, a_2, ..., a_n$ be a string over the alphabet Σ. The automaton M accepts the string w if a sequence of states, $r_0, r_1, ..., r_n$, exists in Q with the following conditions:

1. $r_0 = q_0$
2. $r_{i+1} = \delta(r_i, a_{i+1}), for\, i = 0,...,n-1$
3. $r_n \in F$

The first condition says that the machine starts in the start state q_0. The second condition says that given each character of string w, the machine will transit from state to state according to the transition function δ. The last condition says that the machine accepts w if the last input of w causes the machine to halt in one of the accepting states. Otherwise, it is said that the automaton rejects the string. The set of strings M accepted is the language recognized by M and this language is denoted by L(M).

Strings over an Alphabet

A string of length (n ≥ 0) over an alphabet Σ is just an ordered n-tuple of elements of Σ, written without punctuation.

e.g., if Σ = {a, b, c}, then a, ab, aac, and bbac are strings over Σ of lengths one, two, three and four respectively.

Σ^* = set of all strings over Σ of any finite length.

Also, there is a unique string of length zero over Σ, called the null string (or empty string) and denoted (no matter which Σ we are talking about).

Concatenation of Strings

The concatenation of two strings u, v∈Σ∗ is the string uv obtained by joining the strings end-to-end.

e.g., if u = ab, v = ra and w = cad, then vu = raab, uu = abab and wv = cadra.

This generalizes to the concatenation of three or more strings as uvwuv = abracadabra. The length of a string u will be denoted by length (u). We make no notational distinction between a symbol a ∈Σ and the corresponding string of length one over Σ, also since Σ is regarded as null string ε a subset of Σ^*, then Σ^* is never empty, as it always contains the, the unique string of length zero.

Note also that for any u,v,w ∈ Σ^* , uε = u = εu and (uv)w = uvw = u(vw) and length(uv) = length(u) + length(v).

Motivation

(i) If Σ = {a}, then Σ^* contains ε, a, aa, aaa, aaaa, ...

(ii) If Σ = {a, b}, then Σ^* contains ε, a, b, aa, ab, ba, bb, aaa, aab, aba, abb, baa, bab, bba, bbb,…

(iii) If Σ = ∅ (the empty set — the unique set with no elements), then Σ^* = {ε}, the set just containing the null string.

Languages

A (formal) language L over an alphabet Σ is just a set of strings in Σ^*. Thus, any subset $L \subseteq \Sigma^*$ determines a language over Σ. The language determined by a regular expression r over Σ is L(r) = {u∈Σ^* | u matches r}.

Two regular expressions r and s (over the same alphabet) are equivalent if and only if L(r) and L(s) are equal sets (i.e., have exactly the same members).

There are only finitely many different states that a finite automaton can be in. In figure 1.1 there are three states, labelled q_0, q_1, and q_2. If we think of the elements of Σ as input symbols. Thus, all the possible transitions of the finite automaton can be specified by giving a finite graph whose vertices are the states and whose edges have both a direction and a label (drawn from Σ). In the example Σ = {0, 1} and the only possible transitions from state q_0 are $q_0 \rightarrow q_2$

and $q_0 \rightarrow q_0$. In other words, in state q_0 the machine can either input the symbol 0 and enter state q_2, or it can input the symbol 1 and loop to same state q_0.

There is a distinguished start state q_0, which in the graphical representation of a finite automaton, the start state is usually indicated by means of an unlabelled arrow. The states are partitioned into two kinds: accepting states and non-accepting states and in the graphical representation of a finite automaton, the accepting states are indicated by double circles round the name of each such state, and the non-accepting states are indicated using single circles. In figure 1 there is only one accepting state q_1, the other two states are non-accepting. It is also allowed for the start state to be accepting. The reason for this partitioning of the states into 'accepting' and 'non-accepting' has to do with the use to which one puts finite automata to recognize, that is whether or not a string $u \in \Sigma^*$ is in a particular language.

Given u, we begin in the start state of the automaton and traverse its graph of transitions, using up the symbols in u in the correct order reading the string from left to right. If we can use up all the symbols in u in this way and reach an accepting state, then u is in the language 'accepted' (or 'recognized') by this particular automaton; otherwise, u is not in that language.

Results and Discussion

This section presents the results of the paper as follows:

Languages Accepted by the Constructed Models

Definition: The language L_M of a finite automata M contains all input strings accepted by M.

L_M = {strings that bring M to an accepting state} and the strings that bring the automata to an accepting state are the words of these languages.

Notations: The following notations are used in this paper and defined to be:

W_C: Words and subwords of compact model

L_C: Language of compact model

W_D: Words and subwords of detailed model

L_D: Language of detailed model

W_Q: Words and subwords of extended model of quarry stage

L_Q: Language of extended model of quarry stage

$W_{C\&S}$: Words and subwords of extended model of crushing and stockpiling stage

$L_{C\&S}$: Language of extended model of crushing and stockpiling stage

W_R: Words and subwords of extended model of raw milling stage

L_R: Language of extended model of raw milling stage

W_P: Words and subwords of extended model of preheating stage

L_P: Language of extended model of preheating stage

W_F: Words and subwords of extended model of finished cement stage

L_F: Language of extended model of finished cement stage

Compact Model

Therefore, modifying the compact model constructed in (Zaid, I. et al 2014a) to be as in in figure 2 and for clarity, if the impossible transitions of the compact model are removed and we assign $t_1 = a,\ t_2 = b$ *and* $t_3 = c$.

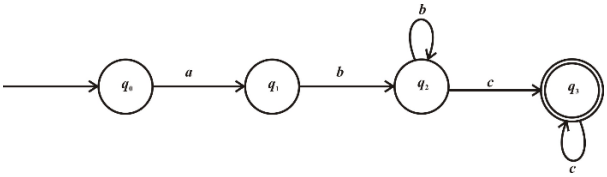

Figure 2. Modified Compact Model

Then, the words and subwords of the model can be generated as:

$$W_C = \varepsilon, a, ab, abb, abbb, ..., ab^n, abc, abcc, abccc, ..., abc^n, abbc^n, abbbc^n ..., ab^nc^n \quad (1)$$

and in that case, the language of the compact model is:

$$L_C = \left\{ \varepsilon abc, abc, abbc, ..., abb^{n-1}, abcc^{n-1}c, abb^{n-1}cc, ..., abb^{n-1}c^{n-1} \mid n \in Z^+ \right\} \quad (2)$$

which is an infinite language.

Detailed Model

Also, for the detailed model of the Sokoto cement production system of figure 3, as constructed in (Zaid, I. et al 2014b), where there are repetitions of transition during raw milling and cement milling and by assigning as follows $t_1 = a, t_2 = b$ *and* $t_3 = c$ then, the figure can be represented as follows:

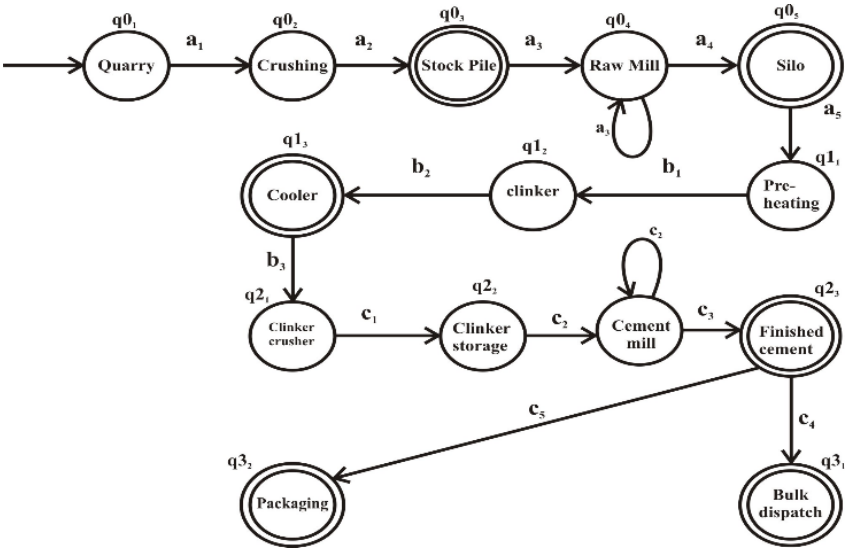

Figure 3. Modified Detailed Model

$$W_D = a_1, a_1a_2, a_1a_2a_3, a_1a_2a_3a_3^*, a_1a_2a_3a_3^*a_4, a_3^*a_4, a_1a_2a_3a_3^*a_4a_5, a_5b_1,$$
$$a_5b_1b_2, a_1a_2a_3a_3^*a_4a_5b_1b_2, b_3c_1, b_3c_1c_2, b_3c_1c_2c_2^*, b_3c_1c_2c_2^*c_3, c_4, c_5,$$
$$b_3c_1c_2c_2^*c_3c_4, b_3c_1c_2c_2^*c_3c_5, a_1a_2a_3a_3^*a_4a_5b_1b_2b_3c_1c_2c_2^*c_3c_4,$$
$$a_1a_2a_3a_3^*a_4a_5b_1b_2b_3c_1c_2c_2^*c_3c_5 \quad (3)$$

So, the language of the detailed automata is given by:

$$L_D = \left\{ a_1a_2, a_1a_2a_3a_3^*a_4, a_3a_3^*a_4, a_1a_2a_3a_3^*a_4a_5b_1b_2, a_5b_1b_2, b_3c_1c_2c_2^*c_3, c_4, c_5 \right\}$$
$$= \left\{ a_1a_2, a_1a_2a_3a_3a_3^{n-1}a_4, a_3a_3a_3^{n-1}a_4, a_1a_2a_3a_3a_3^{n-1}a_4a_5b_1b_2, a_5b_1b_2, b_3c_1c_2c_2c_2^{n-1}c_3, c_4, c_5, ... \right\} \quad (4)$$

Which also produce an infinite word and subwords.

Extended Model of the Quarry Stage

If $q_{0_1} and\ q_{0_2}$ in figure 3 are considered to be quarry state and q_{0_i} to be re-presented as $q_{0_{ij}}$ where , the state can be expanded as in figure 4 to be termed as the extended model of the quarry stage of the production process denoting $t_{1_i} by\ a_i$ to give the finite automata of the quarry stage as follows:

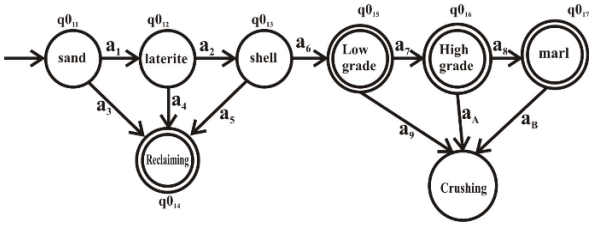

Figure 4. Modified Quarry Stage of the Extended Model

The words and subwords of the automata are:

$$W_Q = a_1, a_1a_2, a_1a_4, a_1a_2a_5, a_1a_2a_6, a_1a_6, a_1a_2a_6a_7, a_6, a_7, a_8, a_6a_9, a_1a_2a_6a_9, a_1a_2a_6a_A, a_1a_2a_6a_7a_8, a_1a_2a_6a_7a_8a_B, \quad (5)$$

from which the language of the quarry automata is given by:

$$L_Q = \{a_1a_2a_5, a_1a_4, a_3, a_1a_2a_6, a_1a_2a_6a_7, a_1a_2a_6a_7a_8\} \quad (6)$$

Extended Model of the Crushing and Stockpiling Stage

Extending the crushing and stockpiling of figure 3 as done in the same manner as that of quarry stage and taking into consideration some of the loops in the activities of stockpiling while assigning $t1_2 = a$, the figure can be represented as figure 5 below:

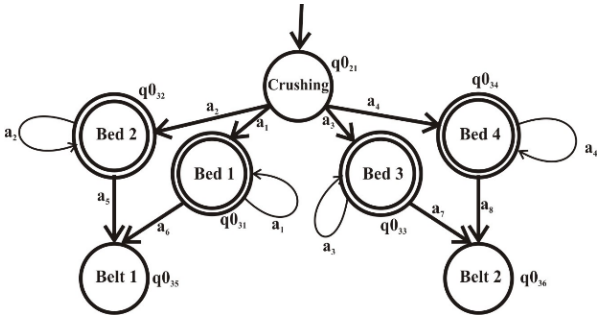

Figure 5. Modified Crushing and Stockpiling Stage of the Extended Model

The words and subwords of the crushing and stockpiling automata are as follows:

$$W_{C\&S} = a_1, a_2, a_1a_1^*, a_1a_1a_1^{n-1}, a_1a_1^*a_6, a_1a_1a_1^{n-1}a_6, a_2a_2^*, a_2a_2^*a_5, a_2a_2a_2^{n-1}, a_3, a_4, a_3a_3^*, a_3a_3a_3^{n-1}, a_3a_3^*a_7, a_3a_3a_3^{n-1}a_7, a_4a_4^*, a_4a_4a_4^{n-1}, a_4a_4^*a_8, a_4a_4a_4^{n-1}a_8 \quad (7)$$

and the language of the automata is as follows:

$$L_{C\&S} = \{a_1a_1a_1^{n-1}, a_2a_2a_2^{n-1}, a_3a_3a_3^{n-1}, a_4a_4a_4^{n-1}, \ldots\} \quad (8)$$

Extended Model of the Raw Milling Stage

Extending the Raw milling stage of figure 3 as done in the same manner as that of quarry or stockpiling stage and taking into consideration some of the loops in the activities of Raw-milling and assigning $t1_3 = a,\ t1_4 = b,$ the figure can be represented as figure 6 below:

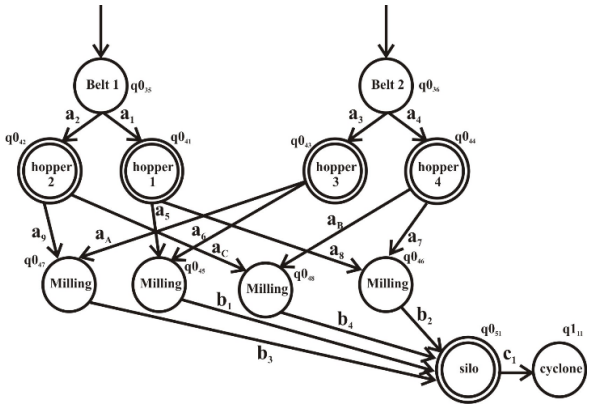

Figure 6. Modified Raw milling Stage of the Extended Model

Therefore, the words and subwords of the automata are:

$$W_R = a_1, a_1a_5, a_1a_5b_1, a_1a_8, a_1a_8b_2, a_2, a_2a_9, a_2a_9b_3, a_2a_C, a_2a_Cb_4, a_3, a_3a_6, a_3a_6b_1, a_3a_A, a_3a_Ab_3, a_4, a_4a_7, a_4a_7b_2, a_4a_B, a_4a_Bb_4, a_9b_3, a_5b_1, a_Cb_4, a_8b_2, a_6b_1, a_Ab_3, a_Bb_4, a_7b_2, a_9b_3c_1, a_5b_1c_1, a_Cb_4c_1, a_8b_2c_1, a_6b_1c_1, a_Ab_3c_1, a_Bb_4c_1, a_7b_2c_1, a_1a_5b_1c_1, a_1a_8b_2c_1, a_2a_9b_3c_1, a_2a_Cb_4c_1, a_3a_6b_1c_1, a_3a_Ab_3c_1, a_4a_7b_2c_1, a_4a_Bb_4c_1 \quad (9)$$

from which the language of raw milling stage is written as:

$$L_R = \{a_1, a_2, a_3, a_4, a_1a_5b_1, a_2a_9b_3, a_2a_Cb_4, a_1a_8b_2, , a_3a_6b_1, a_3a_Ab_3, a_4a_Bb_4, a_4a_7b_2\} \quad (10)$$

Preheating Stage of the Extended Model

Extending the Preheating stage of figure 3 as done in the same manner as that of quarry or stockpiling or raw milling stage and taking into consideration that, $t2_1 = a_1, t2_2 = b_1$ *and* $t2_3 = c_1$, the figure can be depicted as figure 7 below:

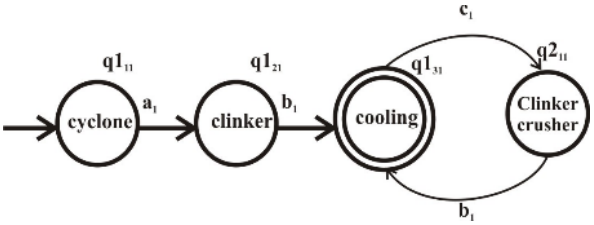

Figure 7. Modified Preheating Stage of the Extended Model

The words and subwords of these automata are:

$$W_P = a_1, b_1, a_1b_1, a_1b_1(c_1 + b_1)^*, a_1b_1(c_1 + b_1)(c_1 + b_1)^{n-1}, \ldots \quad (11)$$

Therefore, the language of the preheating automata is:

$$L_P = \{a_1b_1, a_1b_1(c_1 + b_1)^*, \ldots\} \quad (12)$$

Finished Cement Stage of the Extended Model

Extending the Finished stage of figure 3 as done in the same manner as that of quarry or stockpiling or raw milling stage and taking into consideration the transitions between bulk dispatch and packaging of the powdered cement as well as, denoting, $t3_1 = a_1, t3_2 = b_1, t3_3 = c_1$ *and* $t3_4 = c_1$, the figure can be depicted as figure 8 below:

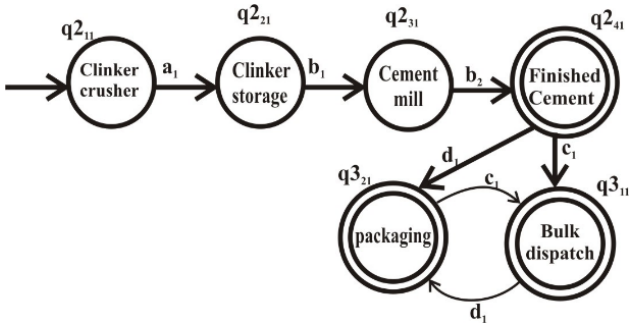

Figure 8. Modified Finished Cement Stage of the Extended Model

Therefore, the words and subwords of the automata are:

$$W_F = a_1, b_1, b_2, a_1b_1b_2, a_1b_1b_2c_1, a_1b_1b_2d_1, c_1, d_1, a_1b_1b_2c_1d_1, a_1b_1b_2c_1d_1, a_1b_1b_2c_1d_1(c_1 + d_1), \ldots \quad (13)$$

So, the language of the finished cement automata is;

$$L_F = \left\{a_1b_1b_2, a_1b_1b_2c_1, a_1b_1b_2d_1, a_1b_1b_2c_1d_1, a_1b_1b_2c_1d_1(c_1+d_1)^*, \ldots\right\} \tag{14}$$

Union of the Languages

The unions of these regular languages are given by:

$$\begin{aligned} L_Q \cup L_{C\&S} \cup L_R \cup L_P \cup L_F = {} & \left\{a_1a_2a_5,\ a_1a_4,\ a_3,\ a_1a_2a_6,\ a_1a_2a_6a_7,\ a_1a_2a_6a_7a_8\right\} \\ & \cup \left\{a_1a_1a_1^{n-1}, a_2a_2a_2^{n-1}, a_3a_3a_3^{n-1}, a_4a_4a_4^{n-1}, \ldots\right\} \cup \\ & \left\{a_1,\ a_2,\ a_3,\ a_4,\ a_1a_5b_1,\ a_2a_9b_3,\ a_2a_Cb_4,\ a_1a_8b_2,\ a_3a_6b_1,\ a_3a_Ab_3,\ a_4a_Bb_4,\ a_4a_7b_2\right\} \cup \\ & \left\{a_1b_1,\ a_1b_1(c_1+b_1)^*, \ldots\right\} \\ & \cup \left\{a_1b_1b_2, a_1b_1b_2c_1, a_1b_1b_2d_1, a_1b_1b_2c_1d_1, a_1b_1b_2c_1d_1(c_1+d_1)^*, \ldots\right\} \end{aligned} \tag{15}$$

$$L_Q \cup L_{C\&S} \cup L_R \cup L_P \cup L_F = \left\{\begin{array}{l} a_1a_4, a_1a_2a_5, a_1a_2a_6, a_1a_2a_6a_7, a_1a_2a_6a_7a_8, a_1a_1a_1^{n-1}, a_2a_2a_2^{n-1}, \\ a_3a_3a_3^{n-1}, a_4a_4a_4^{n-1}, \ldots, a_1,\ a_2,\ a_3,\ a_4,\ a_1a_5b_1,\ a_2a_9b_3,\ a_2a_Cb_4, \\ a_1a_8b_2,\ a_3a_6b_1,\ a_3a_Ab_3,\ a_4a_Bb_4,\ a_4a_7b_2, a_1b_1,\ a_1b_1(c_1+b_1)^*, \ldots, \\ a_1b_1b_2, a_1b_1b_2c_1, a_1b_1b_2d_1, a_1b_1b_2c_1d_1, a_1b_1b_2c_1d_1(c_1+d_1)^*, \ldots \end{array}\right\} \tag{16}$$

Remarks

1. If a_i, b_i, c_1 and d_1 represent the times taken to transit between these processes then, $L_Q \cup L_{C\&L} \cup L_R \cup L_P \cup L_F$ gives the time period of the production cycle.
2. If a_i, b_i, c_1 and d_1 represent the quantities transited between production units then, $L_Q \cup L_{C\&L} \cup L_R \cup L_P \cup L_F$ gives the quantity produced per production cycle.
3. Also, $L_Q \cup L_{C\&L} \cup L_R \cup L_P \cup L_F$ represent the regular language of the general extended model to be considered in further research.

Conclusions

It has been established that the Sokoto cement production system can be modeled into an automata scheme with interesting algebraic theoretic properties that present relevant information on both the production mechanism viewed as sub-states and the overall performance analysis when viewed as compact, detailed and extended scheme; In addition, a language scheme has been developed for each of the sub-states (sub-model) leading to a theoretic study of regular language scheme (machine learning). This has an important application in theoretical computer science, data science, cryptography and networking as it involves graphs among others.

In further research, these particular extended models would be combined into a general extended model so as to establish the deterministic finite automata for the general extended model and its language as well as

regular expression for the general model and possible represented as binary codes.

References

Adamek, J., Rosicky, J., Vitale, E. M., & Lawvere, F. W. (2010). *Algebraic Theories: A Categorical Introduction to General Algebra.* Cambridge Tracts in Mathematics, Cambridge University Press.

Fitsum Consultancy S. (2008). Environmental impact assessment report. *Schulze Global Investments*, *1*, 13-18.

Garcia, P. & Vidal, E. (1990). Inference of k-testable languages in the strict sense and application to syntactic pattern recognition. *Journal of IEEE: Transactions on Pattern Analysis and Machine Intelligence, 12,* 920 – 925. https://doi.org/10.1109/34.57687

Gribkoff, E. (2013). *Applications of deterministic finite automata.* ECS Lecture Notes, University of California, Davis, (ECS 120),1-9.

Hopcroft, J. E., Motwani, R., & Ullman, J. D. (2013). *Introduction to automata theory, languages, and computation. 3rd edition.* Pearson, Addison Wesley, New York.

Kim N., Shin D, Wysk R. A. & Rothrock L. (2010). Using Finite State Automata for Formal Modeling of affordances in Human-Machine Cooperative Manufacturing System: *International Journal of Production Research, 48*(5), 1303-1320. http://doi.org/10.1080/00207540802582235

Lawson M. V. (2005). *Lecture Note.* Department of Mathematics, School of Mathematic and Computer Science Hderiott Watt University.

O'Castillo O. L. & Tapia C. G. (2009). An Environmental and Production Policy Application of Multi-objective Mathematical Programming for Cement Manufacturing in the Philippines. *International Conference on OR and Management Science for Development - Manila, Philippines.*

Robinson, A. (1965). *Introduction to Model Theory and Metamathematics of Algebra (Studies in Logic and the Foundations of Mathematics).* 2nd edition. North-Holland, Netherland.

Yalcin A., Tai T. & Boucher T. O. (2004). *Deadlock Avoidance in Automated Manufacturing Systems Using Finite Automata and State Space Search.*

Zaid, I., Ibrahim, A. A., & Garba, I. A. (2014a). Modeling of sokoto cement production process using a finite automata scheme: An analysis of the detailed model. *International Journal of Computational Engineering Research (IJCER), 4*(5), 2250 – 3005.

Zaid, I., Ibrahim, A. A., Garba, I. A., and Sahabi, D. M. (2014b). Modeling of sokoto cement production process using finite automata: Compact model. *International Journal of Scientific Research (IOSR)–Journal of Applied Physics,* 6(3):21 – 22. http://doi.org/10.9790/4861-06323437