

Syllabification, Normalization and Lexicographic Ordering of Myanmar Texts using Formal Approaches

(ミャンマー語テキストの形式手法による音節分割，正規化と辞書順排列)

By

TIN HTAY HLAING

A Dissertation

Submitted in partial fulfillment of the requirement for the degree

DOCTOR OF ENGINEERING

Department of Information Science and Control Engineering

Graduate School of Engineering

Nagaoka University of Technology, Japan

August, 2014

Supervisor : Professor Yoshiki MIKAMI

DEDICATED TO

My daughter, Zin Myint Mo Thein Htut @ ももちゃん

My son, Khant Zay Thein Htut @ ケンくん

ACKNOWLEDGEMENT

First of all, I would like to express my sincere gratitude to my academic advisor, Professor Mikami Yoshiki for the guidance and encouragement he gave me throughout my study career. I am deeply indebted to my advisor for his invaluable guidance, continuous support as well as teaching the concept of formal approaches to apply in natural language related research. His unique ideas and logical way of thinking inspire me to do this research.

I would like to take this opportunity to express my earnest thanks to Prof. Yukawa Takashi, Prof. Takei Yoshinori, Associate Prof. Yamamoto Kazuhide and Prof. Machida Kazuhiko (Tokyo University of Foreign Languages) for being my thesis examiners, and providing insightful feedbacks. Without their critical comments and suggestions, this research would not have progressed.

Further, I would like to express my deepest thanks to Prof. Hirao Yuji and Prof. Fukuda Takabumi for spending their precious time on the discussion of my research work from different perspective and providing invaluable suggestions.

I gratefully acknowledge the scholarship foundations: (1) The Kubota Fund (Nippon Koei Co., Ltd) (April, 2010-March, 2012) (2) The KDDI Foundation (April, 2012- March, 2013) (3) Rotary Yoneyama Memorial Foundation (April, 2013-August, 2014). They remove my financial concerns and as a result, I could solely focus on my research which leads to the completion of PhD.

I also would like to deliver my special thanks to MEXT Research Grant (Ministry of Education, Culture, Sports, Science and Technology in JAPAN) for their financial support. And my most sincere thanks go to all teachers at Japanese Language Center for their kind support and language training. Special thanks to all members of GII/STS Laboratory and NUT staffs for their kind hearts and smiles.

Last, but not least, I would like to give my heartfelt thanks to my family especially my husband, Win Thein Htut, for their endless care and encouragement during my study in JAPAN. This study is dedicated to my two children, Zin Myint Mo Thein Htut and Khant Zay Thein Htut.

TIN HTAY HLAING

Contents

1	INTRODUCTION	1
1.1	Typology of Writing Systems	1
1.2	Objectives of the Study	4
1.3	Generic Myanmar Language Applications using Orthographic Syllable Model	5
1.4	Organization of the Study	7
2	THEORETICAL FRAMEWORK	9
2.1	Introduction	9
2.2	Formal Grammar	10
2.3	Regular Expression	12
2.4	Regular Language	14
2.5	Finite State Automata (FSA)	15
2.6	Finite State Transducers (FST)	18
2.6.1	Stuttgart Finite State Transducer Tool (SFST)	20
2.7	Introduction to Order Theory	22
2.7.1	Binary Relation on Sets	22
2.7.2	Lexicographic Order	23
3	FINITE STATE TRANSDUCER (FST) FOR AUTOMATIC MYANMAR SYLLABIFICATION	24
3.1	Introduction	24
3.2	Categories of Myanmar Characters	25
3.3	Myanmar Syllable Structure	27
3.3.1	Myanmar Syllable Structure in Phonetic View	27
3.3.2	Myanmar Syllable Structure in Orthographic View	30

3.4	Syllabification of Irregular Words	31
3.4.1	Myanmar Irregular Words	31
3.4.2	Example of Syllabification	33
3.5	Problem Statement	34
3.6	Literature Review	35
3.7	Finite State Transducer for Myanmar Syllabification	38
3.7.1	Constructing Regular Grammar and Transducer.....	38
3.7.2	Implementation	41
3.7.3	Experimental Result	46
3.8	Contributions of the Research.....	48
4	NORMALIZED CODE SEQUENCE FOR MYANMAR LANGUAGE : FSA BASED CODE SEQUENCE CHECKER FOR SAFE IDNs	50
3.1	Introduction.....	50
4.1.1	Phishing and Internationalized Domain Names (IDNs)	51
4.2	Problem Statement	52
4.3	Implementation of Myanmar Domain Names: Linguistic Issues	57
4.4	Finite State Automata (FSA) for Myanmar Code Sequence Checking	59
4.4.1	Myanmar Combining Marks in a Syllable	59
4.4.2	FSA for checking code sequence in Myanmar Domain Names	61
4.4.3	Experimental Results and Discussion	64
5	FORMAL DEFINITION OF LEXICOGRAPHIC ORDER AND SORTING ALGORITHM	68
5.1	Introduction.....	68
5.2	Problem Statement	69
5.3	Formal Description of Order.....	70

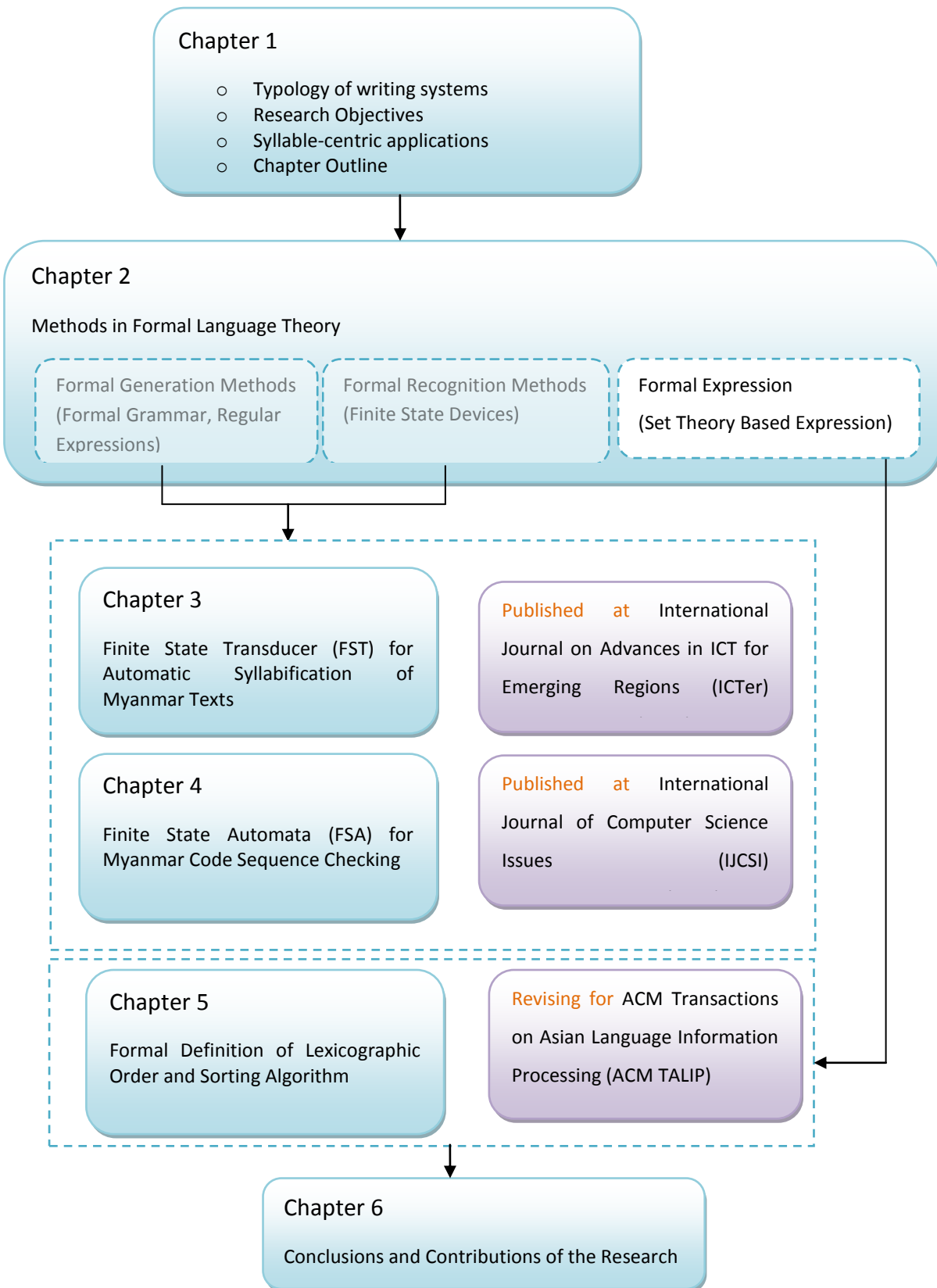
5.4	Myanmar Lexicographic Sorting Algorithm	78
5.4.1	Primary Lexicographic Order Elements	78
5.4.2	Conventional Multi-level Sorting of Syllables	81
5.4.3	Sub-syllable level Sorting.....	82
5.5	Demonstration and Discussion	85
5.6	Conclusions and future work	87
6	CONCLUSIONS	89
	REFERENCES.....	93

LIST OF FIGURES

Figure 1 Chomsky Hierarchy of Languages.....	12
Figure 2. FSA that accepts Myanmar words "House" and "Houses"	15
Figure 3 A Simple FST.....	19
Figure 4 A Path in the Transducer for English.....	20
Figure 5 Finite State Transducer for Myanmar syllabification	41
Figure 6. FSA for generalized Myanmar domain names.....	62
Figure 7. FSA for Myanmar syllable with sub-syllabic element in correct order	63
Figure 8. Registration process with proposed FSA module	64

LIST OF TABLES

Table 1. Summary of Writing Systems	3
Table 2. Categories of Myanmar Charcters in UCS/Unicode	25
Table 3. Proposed Categories of Myanmar Characters	25
Table 4. List of added Vowels.....	26
Table 5. List of Added Medials	26
Table 6. Syllable Segmentation Examples and Results.....	37
Table 7. Updated Keyboard Layout in my-kdb.mim.....	42
Table 8. Details of Experimental Results	46
Table 9. Comparison of Syllabification Results on Irregular Words	47
Table 10. Summary of Myanmar Syllabification Methods	47
Table 11. Myanmar look-alike digits	52
Table 12. Myanmar homoglyphs.....	53
Table 13. Myanmar look-alike consonants.....	53
Table 14. Summary of types of spoofing, existing mitigation methods and threats for Myanmar IDNs	54
Table 15. Combining Mark Order Spoofing in Myanmar.....	56
Table 16. ICANN's Myanmar character in IDNA2008	57
Table 17. Vowel combining marks.....	60
Table 18. Medial combining marks	60
Table 19. Romanized URLs for Myanmar	67
Table 20. Comparison of Multi-level Sort and Simple Sort.....	74
Table 21. Example of Japanese Lexicographic Sorting	76
Table 22. Categories of Myanmar Characters	78
Table 23. List of Added Vowels.....	79
Table 24. List of Added Medials	80
Table 25. Monosyllable order defined by the Myanmar Orthography Dictionary.....	82
Table 26. Analysis of multi-syllable order	83



Academic Achievements

International Referred Journal Papers

1. “Automatic syllabification of Myanmar Texts using Finite State Transducer”. (2013). Tin Htay Hlaing and Yoshiki Mikami, **International Journal on Advances in ICT for Emerging Regions (ICTer)**, Volume 6, Number 2.
2. “FSA based Code Sequence Checking to Prevent Mal Use of Myanmar IDNs”. (2014). Tin Htay Hlaing and Yoshiki Mikami, **International Journal on Computer Science Issues (IJCSI)**, Volume 11, Issue 1.
3. (Under Revising). “Formal Definition of Lexicographic Order and its Application to Sorting Algorithm: A Case for Myanmar”. (2014). Tin Htay Hlaing and Yoshiki Mikami, **ACM Transactions on Asian Language Information Processing (ACM TALIP)**.

International Referred Conference Paper

4. “Manually Constructed Context-free Grammar for Myanmar Syllable Structure”. (2012). Tin Htay Hlaing. Proceedings of European Chapter for the Association of Computational Linguistics (EACL), *Student Research Workshop*, Avignon, France, 26 April, pages 32–37.

Research Internship

5. Internship Program at Faculty of Informatics, Otto-von-Guericke University Magdeburg, Germany, from 14th March, 2013 to 27th March, 2013.

Syllables govern the world.

- John Selden (1584 – 1654)

1 INTRODUCTION

One of the linguistics structures that gained the attention of the scientific community from Natural Language Processing area was the syllable (Dinu 2006). New and exciting researches regarding the formal, quantitative, or cognitive aspects of syllables arise, and new applications of syllables in various fields are proposed: speech recognition, automatic transcription of spoken language into written language, or language acquisition are just few of them. These works define the syllable structure information in phonetic domain.

Syllables can also be defined in orthographic domain of Indic-based scripts especially Myanmar and it would be unusual to the users of Alphabetic scripts. Myanmar language processing includes generic and crucial language processing tasks where orthographic model of syllable structure is compulsory. This research focuses on the formal representation of orthographic syllable model of Myanmar language and development of applications namely automatic syllabification, normalization and lexicographic ordering of Myanmar texts using formal approaches.

1.1 Typology of Writing Systems

Writing systems have been classified in a number of different ways. Scripts can be classified into a number of groups based on their historical genealogy. For example, scripts currently used in Asia can be classified into Indian, Uyghur, and Chinese groups. Scripts can also be classified into two camps: ideograms and phonograms. Furthermore, scripts can be classified based on what each character represents: word, syllable or phoneme. Namely, scripts can be classified into logography or logograms, a syllabary or syllabics, and monophonic characters or simply an alphabet. Phonograms are divided into syllabaries and alphabets. An ideograph whose smallest unit represents the meaning of a word is described in a logography.

If we look at languages and scripts in the world from the viewpoint of diversity, we find that Asia has the greatest diversity. If we list the top 100 languages ranked in terms of

the number of speakers, 62 of them are Asian languages. Again, most of the scripts in Asia are “Indian-based scripts” which refer to a group of scripts that derived from the Brāhmī script of ancient India, for example, Hindi, Bengali, Thai, Lao and Myanmar.

Basically the scripts of Asia can be classified into five types namely consonantal writing systems, alphabets, logography, syllabary and Indic scripts.

A consonantal writing system (or consonantary) is a writing system in which a word is spelled using only consonant letters. Since vowels are not explicit, the reader needs to guess the vowels hidden in a sequence of consonant letters, and to add these to identify a word.

In a pure monophonic system, one letter represents one sound. A character set of monophonic system consists of consonant letters and vowel letters. A word is expressed with a series of these letters. Although an alphabet is a phonogram system with the simplest structure, it can express numerous words with a small character set of around 30 letters.

The third category is logosyllabary. Of course, a typical example treated in this classification is Chinese characters. Chinese characters are so classified because they are both logography and syllabary at the same time.

The fourth category, syllabary, is defined as “letters with no graphical relationship or rules between letters with similar sounds”. In its broad sense, a syllabary is a “script that is written using a syllable as a unit.” Typical syllabaries considered in this document are Hangul, Indian-based scripts and Kana in Japanese.

The fifth category is “Indian-based scripts” which refer to a group of scripts that derived from the Brāhmī script of ancient India, example, Bengali, Sinhala and Myanmar. The phonographic characteristic of the Indian-based scripts is that they are based on “a-vowel accompanying consonant syllabics.” Unlike consonant letter in monophonic characters, a consonant letter, including its inherent vowel, can constitute a syllable (Daniel and Bright 1996).

The presence of such a special vowel allows Indian-based scripts to be regarded as alphabets. The way a variety of signs are added to a consonant letter to express a change in the vowel or a repetition of the consonant looks similar to the way a variety of diacritics are added to an alphabet letter. Due to this similarity in appearance, Indian-based scripts are defined as “diacritically modified consonant syllabic scripts”, and are sometimes called an “alphasyllabary”. Alphasyllabary means that it is an intermediate form between an alphabet and a syllabary. ISO/IEC 10646 gives the prefix of “letters” to the consonant letters of Indian-based script just as it does to alphabet letters, and yet it points to minor differences in the characteristics of letters and signs between Indian-based scripts and alphabets. (Mikami 2009).

The summary of writing systems in Asia is described together with their word structure information in Table 1.

Table 1. Summary of Writing Systems

Typology of Writing Systems				
No.	Typology	Representing sound or meaning	Example	Word Structure Information (W)
1.	Consonantal Writing Systems	Phonogram	Arabic script	$W := \gamma \dots \gamma = \gamma^*$ $\gamma := C \mid V_0$ where C = consonantal letter γ = basic character set V_0 = basic or inherent vowel
2.	Monophonic characters or Alphabet	Phonogram	Latin script	$W := L \dots L = L^*$ (Note: In orthographic view, there are letters whose functions cannot

				identify clearly as consonant or vowel in some contexts and thus we say just L as letters.)
3.	Logography	Ideograph	Chinese script	$W := L$ where L = character
4.	Syllabary	Phonogram	Japanese script	$W := S \dots S = S^*$
5.	Indian-based script (Alphasyllabary)	Phonogram	Myanmar script	$W := S \dots S = S^*$ $S := C \mid V \mid \overline{CV}$ where S = syllable C = consonant with inherent vowel V = standalone vowel syllable \overline{V} = vowel sign (Note : vowel sign cannot be used alone)

1.2 Objectives of the Study

This study targets in developing applications for Myanmar language using formal approaches by taking structural regularity of Myanmar syllable structure in orthographic domain.

For Alphabetic languages, syllable-based development with formal approaches requires supportive information. The syllable structures of these languages have low regularity as the functions of letters are context-dependent. Thus language resources such

as annotated corpus or stochastic information are necessary to get the correct and unambiguous results.

Myanmar script is descended from Brahmic script of Ancient India and includes in the type of alphasyllabaries. A number of diacritics can be added to the base consonant to form a syllable. By analyzing the orthographic characteristics of Myanmar, it is found that Myanmar syllable structure is highly regular. There is no documented work on Myanmar language processing which takes advantages of structuredness of orthographic syllable model and this study fills this gap.

As a first step, orthographic model of Myanmar syllable is represented in formal grammar. Secondly, to prove the importance of syllable structure model, this study develops and demonstrates, for the first time, a computational framework for Myanmar language processing using formal approaches by implementing three generic and critical language processing tasks: (1) Automatic syllabification (2) Normalized code sequence checking (3) Lexicographic Ordering and sorting algorithm. It is expected that our study can serve as a good ground for other Indic-based scripts.

1.3 Generic Myanmar Language Applications using Orthographic Syllable Model

This section explains three generic applications which rely on the syllabic complexity and works on orthographic input strings.

(1) Automatic Syllabification of Myanmar Texts: Languages differ considerably in the syllable structures that they permit. Syllabification may be performed in either the spelling domain (using letter) or the pronunciation domain (using phoneme). Generally, syllabification is done in pronunciation domain because syllable structure definition on written texts is almost impossible for Alphabetic languages. For example, basic understanding on what an English text string is composed of, is consonants and vowels. However, from linguistic view, it cannot be differentiated exactly as consonants and vowels

because their functions are context-dependent. For instance, letters “m”, “y”, “l” can act as vowels in certain situation. That is, if “y” is both preceded and followed by a consonant, we mark that instance as a vowel. Therefore, orthographic syllabification of such languages is impossible and not straight forward.

This is the first significant difference between other alphabetic writing systems and syllabic writing system, Myanmar, in which the role or function of letters never change in any context. Though Myanmar language has more number of character groups rather than consonant and vowel, syllable structure information could be defined systematically. For this reason, syllable-based Myanmar language applications could be implemented in formal approaches. This research develops automatic syllabification of Myanmar texts using finite state transducer (FST).

(2) FSA based Code Sequence Checker for Myanmar Domain Names: The introduction of Unicode support in operating systems and applications has lead to a vastly increased number of available homoglyphs and a new threat arose from the use of characters which are visually indistinguishable from Western characters but belong to a non western script (ICANN, 2005).¹ Considering that a large number of users are not scholars of the language and hence can be easily cheated by homographs, and spoofing, and phishing will occur to a large extent in languages. This calls for great care and caution in supporting local languages and scripts in the domain names.

Phishing attacks can also occur to Myanmar domain names as Myanmar script has similar looking characters and different ways of combination of characters (code sequences) within a syllable. After internationalized domain names (IDNs) had been introduced in 2007, possible threats on IDNs have been discussed and also, guidelines and recommendations for IDN based attacks are developed as mitigating strategies by international organizations. Though general guidelines and mitigating methods are stated to

¹ <http://www.ietf.org/rfc/rfc3743.txt>

cover IDN implementation in all scripts, there are some scripts like Myanmar, which need language specific mitigation methods.

This is also an area of application where syllable model is critical. This research proposes normalized code sequence for Myanmar syllable which is checked using finite state automata (FSA).

(3) Formal Definition of Lexicographic Order and Sorting Algorithm for Myanmar:

Lexicographic order or collation order is an important aspect of localization and it varies accordantly with language and culture. Defining collation order of Asian languages is complicated as it deals with a number of different character groups and complex linguistic conventions are taken into account. Lexicographic order of Myanmar language is also complicated if it is compared with European languages. The formal model of describing collation order of a language is imperative to express multi-level comparison of different character sets (sub-syllabic elements) in simple and unambiguous way.

Therefore, this study presents a generalized formal model of describing lexicographic order by deploying set theory and it is applied to Myanmar language as an example. The order in a given character set and among different character sets encoded in UCS/Unicode can be presented by using formal notations. This work also reveals the lexicographic order of Myanmar language at sub-syllable level. Further, based on the proposed formal description, a sorting algorithm without syllabification process is developed for Myanmar language.

1.4 Organization of the Study

This study has been organized into six chapters. Chapter 1 introduces typology of writing systems and the objectives of this study. Further, it gives a brief introductory for each of three application areas where formal methods are applied.

In Chapter 2, theoretical framework of the study is presented with some examples.

Chapter 3, Chapter 4 and Chapter 5 explains details about three application areas respectively. Each chapter addresses problems of the research and then presents research methodologies, related works, experimental results and contributions.

Finally, we summarize the outputs of the study in the last chapter, Chapter 6, together with limitation and future work.

2 THEORETICAL FRAMEWORK

2.1 Introduction

Language processing based on finite state devices has been an important and active field of research and development for a number of decades. Finite-state machines have been used in many areas of computational linguistics. Their use can be justified by both linguistic and computational arguments.

Linguistically, finite automata are convenient since they allow one to describe easily most of the relevant local phenomena encountered in the empirical study of language. They often lead to a compact representation of lexical rules, or idioms and clichés, that appears as natural to linguists. Graphic tools also allow one to visualize and modify automata. This helps in correcting and completing a grammar. Other more general phenomena such as parsing context-free grammars can also be dealt with using finite-state machines such as recursive transition networks, RTN's. Moreover, the underlying mechanisms in most of the methods used in parsing are related to automata.

From the computational point of view, the use of finite-state machines is mainly motivated by considerations of time and space efficiency. Time efficiency is usually achieved by using deterministic automata. The output of deterministic machines depends, in general linearly, only on the input size and can therefore be considered as optimal from this point of view. Space efficiency is achieved with classical minimization algorithms for deterministic automata. Applications such as compiler construction have shown deterministic finite automata to be very efficient in practice.

Finite automata now also constitute a rich chapter of theoretical computer science. Their recent applications in natural language processing which range from the construction of lexical analyzers and the compilation of morphological and phonological rules to speech processing, show the usefulness of finite-state machines in many areas (Mohri 1997).

This chapter presents theoretical framework employed in finite state based Myanmar language processing. We give brief explanation on formal grammar, regular expression, finite state automaton, finite state transducer and Stuttgart Finite State Tool (SFST). These theoretical concepts are not new in computer science field, however, we would like to prove that these methods are applicable to Myanmar Language.

In Myanmar language processing, application of finite state methods does exist but very limited and there are many areas left where finite state methods should apply. Thus, this study fills this gap by applying finite state devices to critical language processing tasks, automatic syllabification and code sequence checking of Myanmar Unicode strings.

2.2 Formal Grammar

Formal grammars are mathematical systems for describing the structure of languages, both formal and natural. A formal grammar is a notation used to specify a language and is made of four components, non terminals or variables, terminals, production rules and a start symbol. Therefore, a formal grammar is a 4-tuple (V, T, P, S) where

V is a finite set of symbols called as non terminals or variables.

T is a finite set of symbols called as terminals

P is a finite set of rules called as production rules

S is a member of V called as start symbol.

There are four types of grammars, differing in their powers. A grammar being a language specifying notation, its power is defined in terms of the languages it can specify. The differences lie in the format of production rules that are permitted. The four types of grammar are:

Type 0, also called as unrestricted, semi-thue, phrase structure grammar

Type 1, also called as context-sensitive grammar

Type 2, also called as context-free grammar

Type 3, also called as regular grammar

A brief introduction on these four types of grammar is given below.

Type 0 or unrestricted grammar: A type 0 grammar is the one whose every production is of the form $\alpha \rightarrow \beta$, where α and β are in $(V \cup T)^*$, i.e, they can be any strings of terminal and non-terminals, and $\alpha \neq \epsilon$. Since there is no restriction on what can appear on the left hand side of production as well as right hand side of production, it is capable of specifying the largest number of languages. And the languages specifiable using type 0 grammar are called as recursively enumerable languages.

Type 1 or context-sensitive grammar: A type 1 grammar is the one whose every production is of the form $\alpha \rightarrow \beta$, where α and β are in $(V \cup T)^*$, $\alpha \neq \epsilon$ and $|\beta| \geq |\alpha|$. There is a restriction that the right hand side of a production rule must be at least as long as that of the left side. It is so called as context sensitive grammar because every production rule being of the form

$\alpha A \beta \rightarrow \alpha \gamma \beta$, it specifies the replace of A by γ but in the context of $\alpha\beta$, i.e, when A is preceded by α and followed by β , then only A can be replaced by γ .

Type 2 or context-free grammar: A type 2 grammar is the one whose every production is of the form $A \rightarrow \alpha$, where α is in $(V \cup T)^*$, i.e, the left side of the production rule is single non-terminal or variable and the right side can be any string of terminals and non terminals. There is a restriction that the left hand side of the production rule must be a single non terminal. It is so called as context-free grammar because every production rule being of the form $A \rightarrow \alpha$, it specifies the replacement of A by α anywhere, and it is not necessary that A should appear in a particular context for its replacement by α .

Type 3 or regular grammar: A type 3 grammar is the one whose every production is of the form $A \rightarrow a$ or $A \rightarrow aB$, where A and B are in V and a is in T , i.e, the left side of the production rule is a single non-terminal or variable and the right side can be a terminal followed by a non-terminal or only a single terminal. It is so called as regular grammar.

It is found that every regular language is a context-free language, but vice-versa is not true. Similarly, every context-free language not containing ϵ is a context-sensitive language, but vice-versa is not true. And every context-sensitive language is a recursively enumerable language, but vice-versa is not true. Therefore, we conclude that the type(i) languages properly include the type(i+1) languages, except for the case of empty string ϵ for $i = 0, 1, 2$. Therefore, these four classes of languages constitute a hierarchy called as Chomsky Hierarchy (O.G. Kakde 2007) The hierarchy is schematically depicted as follow.

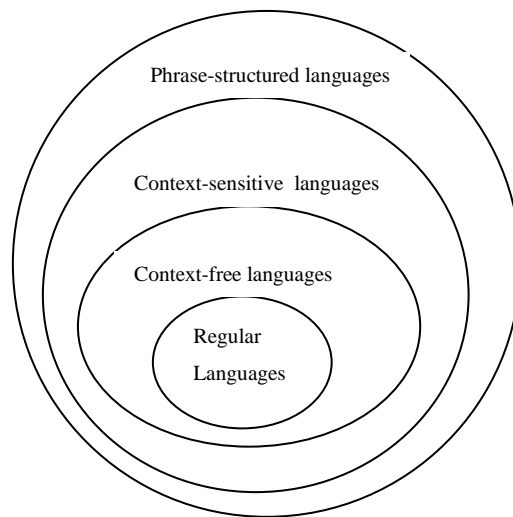


Figure 1 Chomsky Hierarchy of Languages

2.3 Regular Expression

A regular expression is a specific kind of text pattern that you can use with many modern applications and programming languages. Formally, regular expressions are an algebraic notation for characterizing a set of strings. Thus, they can be used to specify search strings as well as to define a language in a formal way (Balaram 2011). Complex

regular expressions can be built up from simpler ones by means of regular expression operators. Square brackets, (), are used for grouping expressions. Because both regular languages and regular relations are closed under concatenation and union, the following basic operators can be combined with any kind of regular expression:

$A \mid B$ Union.

AB Concatenation.

(A) Optionality; equivalent to $(A \mid 0)$.

A^+ Iteration; one or more concatenations of A .

A^* Kleene star; equivalent to (A^+) .

Although regular languages are closed under complementation, subtraction, and intersection, regular relations are not (Kaplan and Kay 1994). Thus the following operators can be combined only with expressions that denote a regular language.

$_A$ Complement

nA Term complement; all single symbol strings not in A .

$A \& B$ Intersection

$A - B$ Subtraction (minus)

Regular relations can be constructed by means of two basic operators:

$A .x. B$ Cross product

$A .o. B$ Composition

The cross product operator, $.x.$, is used only with expressions that denote a regular language; it constructs a relation between them. $(A .x. B)$ designates the relation that maps every string of A to every string of B . The notation $a:b$ is a convenient shorthand for $(a .x.$

b). Composition is an operation on relations that yields a new relation. $(A \circ B)$ maps strings that are in the upper language of A to strings that are in the lower language of B. If A contains the pair $\langle x.y \rangle$ and B contains the pair $\langle y.z \rangle$, the pair $\langle x.z \rangle$ is in the composite relation.

Rather than the above mentioned basic regular expressions, the syntax of regular expression can be extended by defining new operators that allow commonly used constructions to be expressed more precisely. These operators are

- (1) the containment operator $\$$
- (2) the restriction operator \Rightarrow
- (3) the replace operator \rightarrow
- (4) the marking operator \dots

Descriptions consisting of regular expressions can be efficiently compiled into finite-state networks, which in turn can be determinized, minimized, sequentialized, compressed, and optimized in other ways to reduce the size of the network or to increase the application speed. Many years of engineering effort have produced efficient runtime algorithms for applying networks to strings.

Regular expressions have a clean, declarative semantics. At the same time they constitute a kind of high-level programming language for manipulating strings, languages, and relations. Although regular grammars can cover only limited subsets of a natural language, there can be an important practical advantage in describing such sublanguages by means of regular expressions rather than by some more powerful formalism. Because regular languages and relations can be encoded as finite automata, they can be more easily manipulated than context-free and more complex languages (Lauri Karttunen 2000).

2.4 Regular Language

A formal language is a set of strings each of which is composed of symbols from a finite symbol-set called an alphabet. A regular language is a formal language that is possibly an infinite set of finite sequences of symbols from a finite alphabet that satisfies

particular mathematical properties: "A class of languages that are definable by regular expression is exactly the same as the class of languages that are characterized by finite-state automaton is said to be a regular language" (Jurafsky and Martin 1998).

2.5 Finite State Automata (FSA)

Finite-state automata are computational devices that compute regular languages. Formally, the finite state automaton can be defined by the following five parameters:

Q : a finite set of N states q_0, q_1, \dots, q_N

Σ : a finite input alphabet of symbols

q_0 : the start state

F : the set of final states $F \subseteq Q$

$\delta(q,i)$: the transition function or transition matrix between states. Given a state

$q \in Q$. δ is thus a relation from $Q \times \Sigma$ to Q (Jurafsky and Martin 2000).

For illustration, an automaton that accepts a string from the Myanmar language "house" and "houses" and it is visualized in the figure below.

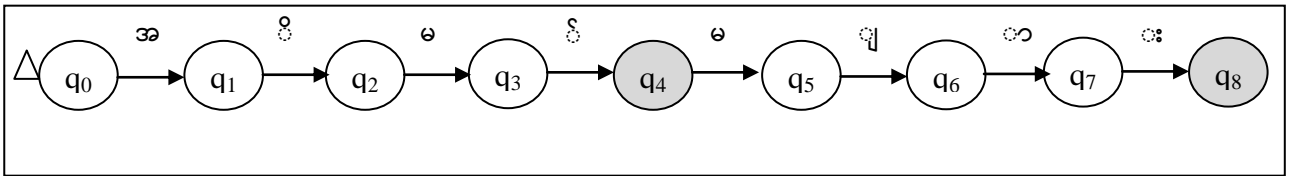


Figure 2. FSA that accepts Myanmar words "House" and "Houses"

The finite state automaton shown in Figure 2 recognizes the words by reading the input string symbol-by-symbol and matching the symbols to the labels on the arcs. This FSA accepts Myanmar words 'house' and 'houses' because the inputs lead to final states. No other strings are accepted by this FSA. For the language automata in Figure 2,

$$Q = \{q_0, q_1, q_2, q_3, q_4, q_5, q_6, q_7, q_8\}$$

$$F = \{q_4, q_8\}$$

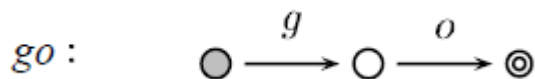
$$\Sigma = \{\text{అ}, \text{ఁ}, \text{అ}, \text{ఁ}, \text{ఁ}, \text{ఁ}, \text{ఁ}\}$$

$\delta(q,i)$ is represented by the following state transition table.

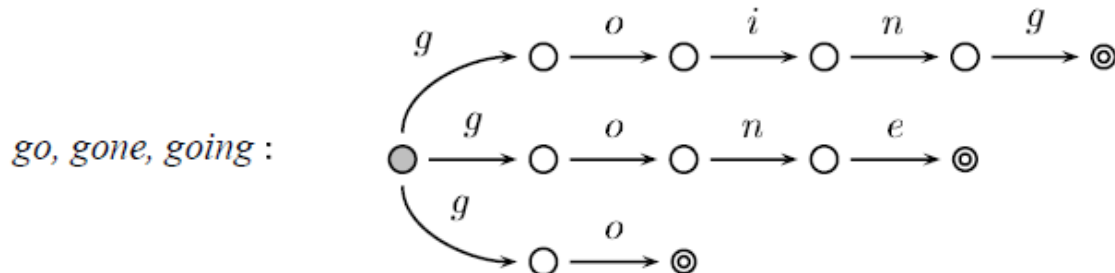
State	Input						
	అ	ఁ	అ	ఁ	ఁ	ఁ	ఁ
0	1	ϕ	ϕ	ϕ	ϕ	ϕ	ϕ
1	ϕ	2	ϕ	ϕ	ϕ	ϕ	ϕ
2	ϕ	ϕ	3	ϕ	ϕ	ϕ	ϕ
3	ϕ	ϕ	ϕ	4	ϕ	ϕ	ϕ
4	ϕ	ϕ	5	ϕ	ϕ	ϕ	ϕ
5	ϕ	ϕ	ϕ	ϕ	6	ϕ	ϕ
6	ϕ	ϕ	ϕ	ϕ	ϕ	7	ϕ
7	ϕ	ϕ	ϕ	ϕ	ϕ	ϕ	8

Further, we introduce two major applications of FSA by showing some examples.

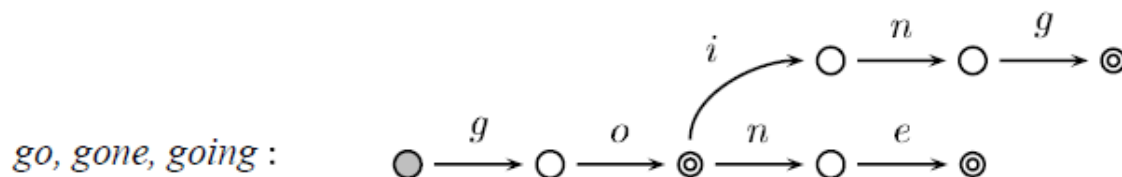
Dictionaries as FSA: Many NLP applications require the use of lexicons or dictionaries, sometimes sorting of hundreds of thousands of entries. Finite state automata provide an efficient means for storing dictionaries, accessing them, and modifying their contents. To understand the basic organization of a dictionary as a finite-state machine, assume that an alphabet is fixed and consider how a single word “go” can be represented. A naïve representation would be to construct an automaton with a single path whose arcs are labeled by the letter of the word “go”.



To represent more than one word, we can simply add paths to our lexicons, one path for each additional path. Thus, after adding the words “gone” and “going”, we might have:



This automaton can then be determinized and minimized as

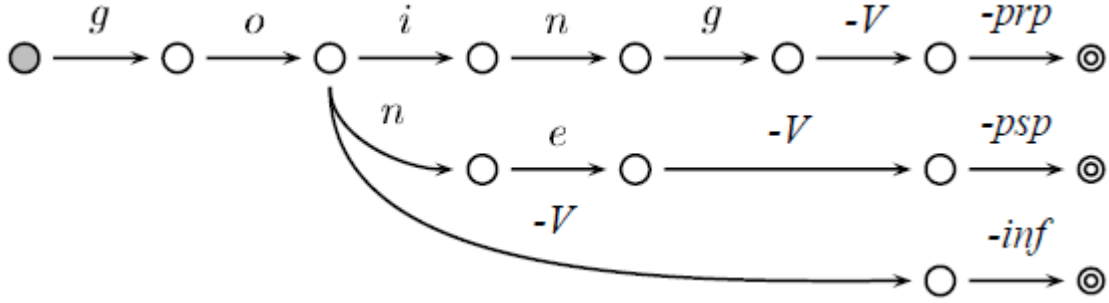


With such a representation, a lexical lookup operation amounts to checking whether a word ω is a member in the language generated by the automaton, which can be done by “walking” the automaton along the path indicated by ω . This is an extremely efficient operation: it takes exactly one “step” for each letter of ω . We say that the time required for this operation is *linear in the length of ω* .

Adding Morphological Information to the Lexicon using FSA: Suppose we want to add to the lexicon information about part-of-speech, and we use two tags: -N for nouns, -V for verbs. Additionally, we encode the number of nouns as -*sg* or -*pl*, and the tense of verbs as -*inf*, -*prp* or -*psp* (for infinitive, present participle and past participle, respectively). It is very important to

note that the additional symbols are multi-character symbols: there is nothing in common to the alphabet symbol -*sg* and the sequence of two alphabet letters $\langle s, g \rangle$. In other words, the extended alphabet is:

$\Sigma = \{a, b, c, \dots, z, -N, -V, -sg, -pl, -inf, -prp, -psp\}$. With the extended alphabet, we can construct the automaton as:



The language generated by the above automaton is no longer a set of words in English. Rather, it is a set of (simplistically) “analyzed” strings, namely $\{go-V-inf, gone-V-psp, going-V-prp\}$ (Shuly Wintner 2001).

2.6 Finite State Transducers (FST)

The finite state automaton accomplishes the task of recognizing strings in a regular language by providing a way to systematically explore all the possible paths through a machine (Jurafsky and Martin 2000). However, this exploration can only address the problem whether the string is present in its language or not. The automaton of this capacity cannot be used to show the relation between two or more languages. However, this problem can be solved by the use of another version of the FSA called ‘Finite State Transducer’ (Balaram 2011).

A transducer determines if the input string is in the domain of the relation, and if it is, computes the corresponding string, or set of strings, that are in the domain of the relation. A regular relation can be thought of as a regular language over n-tuples of symbols, but it is more usefully thought of as expressing relations between sets of strings. The definition of a regular n-relation is as follows:

ϕ is a regular n-relation

For all symbols $a \in (\Sigma \cup \varepsilon) \times \dots \times (\Sigma \cup \varepsilon)$, $\{a\}$ is a regular n -relation.

If R_1 , R_2 and R are regular n -relations, then so are

$R_1 \cdot R_2$, the (n -way) *concatenation* of R_1 and R_2 : for every $r_1 \in R_1$

and $r_2 \in R_2$, $r_1 r_2 \in R_1 \cdot R_2$

$R_1 \cup R_2$

R^* , the n -way *Kleene closure* of R .

For most applications in speech and language processing $n = 2$, so that we are interested in relations between pairs of strings.

With a transducer, a string matches against the input symbols on the arcs, but at the same time the machine is outputting the corresponding output symbol (Roark and Richard 2007). The simple example is given below.

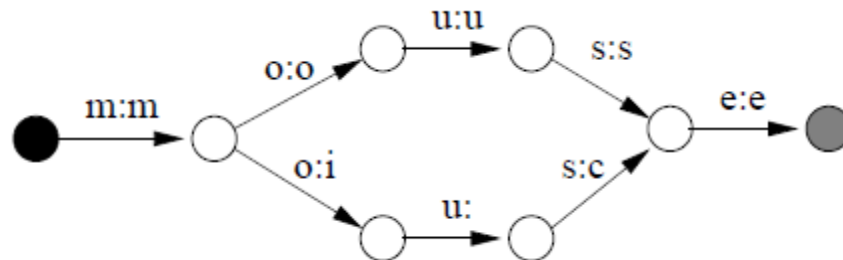


Figure 3 A Simple FST

Analogous to finite automata, FSTs have a single start state and a set of final states. A FST accepts a sequence s of symbol pairs (instead of symbols) if there is a path from the start state to some final state where the sequence of symbol pairs on the transitions is identical to s . FSTs are mainly used to map strings to other strings.

The transducer works in two modes (1) analysis mode and (2) generation mode. In general, a FST maps a string s to a string t (in analysis mode) if there is a path from the start

state to some end state where the right-hand side symbols on the transitions are identical to s and the left-hand side symbols are identical to t . The mapping is reversible: In generation mode, a transducer maps a string s to a string t if the left-hand side symbols on the transitions of some path from the start state to an end state are identical to s and the corresponding right-hand side symbols are identical to t . The transducer shown in the figure generates the strings *mouse* and *mice* for the input string *mouse*.

A typical example is morphological analysis where an inflected word form is analyzed and the base form is returned with additional morphosyntactic information (Helmut Schmid 2005).

Morphology is a domain of linguistics that studies the formation of words. It is traditional to distinguish between surface forms and their analyses, called lemmas. The lemma for a surface form such as the English word *bigger* typically consists of the traditional dictionary citation form of the word together with terms that convey the morphological properties of the particular form. For example, the lemma for “*bigger*” might be represented as “*big+Adj+Comp*” to indicate that *bigger* is the comparative form of the adjective “*big*” (Lauri Karttunen 2001). In the FST below, the zeros represents the epsilon symbol.

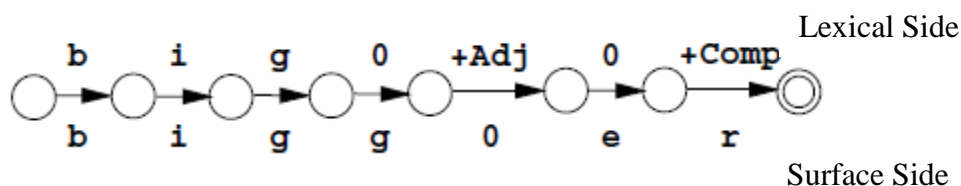


Figure 4 A Path in the Transducer for English

2.6.1 Stuttgart Finite State Transducer Tool (SFST)

The Stuttgart Finite State Transducer (SFST) tools are used for Myanmar syllabification, which is a collection of software tools for the generation, manipulation and processing of finite-state automata and transducers. A finite state transducer (FST) maps strings from one regular language (surface language) onto strings from another regular language (analysis language). One important application of FSTs is morphological analysis,

where a word form such as translations might be mapped to the analysis string `translate<V>ion<N><pl>`. The mapping between surface strings and analysis strings is reversible. The same transducer can be used to generate (i) analyses for a surface form (in analysis mode) and (ii) to generate surface forms for an analysis (in generation mode). The number of generated output strings is not necessarily 1, but can be anywhere between 0 and infinite.

Other important applications of FSTs are lemmatization, tokenization, lexicon representation, spell checking, grapheme to phoneme conversion, low-level parsing, and much more. The SFST tools comprise

- (1) a programming language for the implementation of finite state transducers called "SFST-PL" a compiler for SFST programs called `fst-compiler`
- (2) a set of tools for applying, printing, and comparing finite state transducers and
- (3) a finite state transducer library written in C++

The implementation of the compiler and the other tools is based on the C++ library. The following description of the SFST tools assumes some familiarity with finite state transducers. It also supports a wide range of transducer operations, UTF-8 character coding and weighted transducers (basic functionality only).

The specification of the transducer can be written in SFST-PL, the programming language of the SFST tools. SFST-PL is a general-purpose FST programming language comprising all the descriptive means for the development of finite-state transducers in a single formalism. SFST-PL is flexible and not committed to a single morphological formalism like two-level morphology or ordered rewrite rules. Any regular expression is already a valid SFST program. An example is the expression: `Hello\ world\!` It defines a transducer which maps the string `Hello world!` onto itself and rejects any other input (Helmut Schmid 2005).

2.7 Introduction to Order Theory

Order theory is a branch of mathematics which investigates our intuitive notion of order using binary relations. It provides a formal framework for describing statements such as "this is less than that" or "this precedes that".

2.7.1 Binary Relation on Sets

Given a set X , one of the simplest and yet one of the most frequently occurring additional structures on X consists of what is called a binary relation on X . As the name implies, a binary relation on a set X describes some type of relationship which may exist between certain pairs of elements of X . We can give a rigorous, mathematical definition of a binary relation. Let X be the set of all persons in a town. Now we define $R = \{ (x,y) \in X \times X : x \text{ is a neighbor of } y \}$.

Now we consider a relation which ranks them according to some criteria for comparison. Such relation arises frequently in practice, for example, words are listed in a dictionary in the alphabetical order or the guests at a dinner are seated according to their importance and so on. But such as comparison may not always be possible, with the result that certain pairs of letters will have to be left incomparable. And we formally define it as partial order.

A partially ordered set (P, \leq) , POSET for short, is a set P equipped with a binary relation \leq , called the ordering relation, such that for all x, y, z in P we have;

(Reflexibility) $x \leq x$

(Transitivity) if $x \leq y$ and $y \leq z$ then $x \leq z$

(Antisymmetry) if $x \leq y$ and $y \leq x$ then $x = y$.

Example: $R = \{(x,y) : x \text{ divides } y\}$ is a partial order relation on the set N of natural numbers.

A total or linear or simple order on a set X is a relation \leq on which X is reflexive, transitive and antisymmetric and which has the following property, known as the law of

dichotomy, for every $x, y \in X$ either $x \leq y$ or $y \leq x$. Verbally, every two elements are comparable to each other.

2.7.2 Lexicographic Order

To figure out which of the two words comes first in an English dictionary, you compare their letter one by one from left to right. If all letters have been the same to a certain point and one words run out of letters, that word comes first in the dictionary. For example, *play* comes before *playhouse*. If all letters up to a certain point are the same and the next letters differ, then the word whose next letter is located earlier in the alphabet comes first in the dictionary. For instance, *playhouse* comes before *playmate*.

More generally, if A is any set with partial order relation, then a dictionary or lexicographic order can be defined on a set of strings over A as indicated in the following theorem.

Let A be a set with partial order relation R and let S be a set of strings over A . Define a relation \leq on S as follows.

For any two strings in S , a_1, a_2, \dots, a_m and b_1, b_2, \dots, b_n , where m, n are positive integers,

1. If $m \leq n$ and $a_i = b_i$ for all $i = 1, 2, \dots, m$ then

$$a_1, a_2, \dots, a_m \leq b_1, b_2, \dots, b_n.$$

2. If for some integer k with $k \leq m$, $k \leq n$ and $k \geq 1$, $a_i = b_i$ for all $i = 1, 2, \dots, k-1$, and $a_k \neq b_k$, but $a_k R b_k$ then

$$a_1, a_2, \dots, a_m \leq b_1, b_2, \dots, b_n$$

3. If ϵ is the null string and s is any string in S , then $\epsilon \leq s$.

If no strings are related other than by these three conditions, then \leq is a partial order relation (Susanna S. Epp 2011).

3 FINITE STATE TRANSDUCER (FST) FOR AUTOMATIC MYANMAR SYLLABIFICATION

3.1 Introduction

Automatic syllabification lies at the heart of script processing especially for the South East Asian scripts like Myanmar. Myanmar syllabification algorithms implemented so far are either rule-based or dictionary-based approach. Syllabification algorithms are mainly used in text-to-speech (TTS) systems in producing natural sounding speech, and in speech recognizers for dealing with out-of-vocabulary words. Also for Myanmar script, syllabification algorithm for Myanmar Text-to-Speech (TTS) system has been developed (Kyawt Yin Win 2011). However, such phonetic syllabification cannot be used for some major Myanmar language processing tasks such as lexicographic sorting, word breaking, line breaking and spelling checking. In other words, it is necessary to use orthographic syllabification for these tasks.

This research proposes a new method for Myanmar syllabification which deploys formal grammar and un-weighted finite state transducers (FST). Our proposed method focuses on orthographic way of syllabification for the input texts encoded in Unicode. Although a few researchers have documented attempts at syllabifying Myanmar words orthographically, this is the first known documented method for Myanmar orthographic syllabification using finite state transducers (FST).

The objectives of this study are to represent Myanmar syllable structure in formal description (e.g, regular grammar) by taking advantage of its structuredness and unambiguity. Another objective of our study is to achieve correct syllabification without applying step-by-step rules and the need of corpus. In other words, our proposed method is neither heuristic approach nor statistical approach but based on the regularity of orthographic syllable structure model of Myanmar language.

3.2 Categories of Myanmar Characters

This section describes categories of Myanmar characters encoded in UCS/Unicode. However, it is necessary to re-categorized them based on their functions.

Table 2. Categories of Myanmar Characters in UCS/Unicode

No.	Type	Characters
1.	Consonants	က, ခ, ဂ, ဃ, င, စ, ဆ, ...
2.	Independent Vowels/ Free standing Vowel Syllables	အ, ဣ, ဤ, ဥ, ဦ, ...
3.	Dependent Vowels	ါ, ဝ, ဝိ, ဝီ, ဖ, ဖူ, ...
4.	Various Signs I	း, ိ, ...
5.	Dependent Consonant Signs/ Medials	ချ, ြ, ိ, ိ
6.	Digits	၀, ၁, ၂, ၃, ၄, ...
7.	Various Signs II	ံ, ိ, ိ, ိ

Proposed categories of Myanmar characters are consonants C, vowels V, medials M, finals F, tone marks D, and other characters.

Table 3. Proposed Categories of Myanmar Characters

UCS/Unicode Character Category with number of elements	Proposed Functional Groups with number of elements	Remark
Consonant letters (33)	Consonants C (34)	Myanmar vowel letter A (U+1021) is added
Dependent vowel signs (8) Independent vowel letters (8)	Vowels V (11)	3 combinations of vowel signs are added (Please see Table 4)

Various Signs (5)	Final consonants F (34)	Each member of F is a combination of consonant C and Myanmar sign Asat K and F is isomorphic with C
	Tone marks D (2)	
Dependent consonant signs(4)	Medials M (11)	7 combinations of medial signs are added (Please see Table 5)
Various Signs (4) and Myanmar Consonant Great SA (1)	ငြ်, ဂြ်, ငှ်, ဂှ်, သြ	

Table 4. List of added Vowels

Glyph	Code sequence representation	Description
◌ေ + ◌့	1031+102C	Vowel sign E + AA
◌ေ + ◌့ + ◌်	1031+102C+103A	Vowel sign E + AA + <i>Asat</i>
◌ိ + ◌့	102D + 102F	Vowel sign I + UU

Table 5. List of Added Medials

Glyph	Code sequence representation	Description
◌ျ + ◌့	103B + 103D	Consonant Sign Medial YA + WA
◌ြ + ◌့	103C + 103D	Consonant Sign Medial RA + WA
◌ျ + ◌့	103B + 103E	Consonant Sign Medial YA + HA
◌ြ + ◌့	103C + 103E	Consonant Sign Medial RA + HA
◌့ + ◌့	103D + 103E	Consonant Sign Medial WA + HA
◌ျ + ◌့ + ◌့	103B + 103D + 103E	Consonant Sign Medial YA+WA + HA
◌ြ + ◌့ + ◌့	103C + 103D + 103E	Consonant Sign Medial YA+WA + HA

The process of combination of one or more vowels is known as contraction and it is the same for medials.

3.3 Myanmar Syllable Structure

Myanmar is the country where eight major races and other minority ethnic groups live together. Major ethnic groups namely Kachin, Kayar, Kayin, Chin, Myanmar (Burmese), Mon, Rakkhine and Shan have their own scripts but Myanmar script is used as a major communication medium even by those non-Myanmar language speakers.

Myanmar text consists of string of characters without explicit word boundary markup, written in sequence from left to right without regular inter-word spacing, though inter-phrase spacing may sometimes be used for the ease of reading.

Myanmar script is derived from Brahmi script of ancient India and there are other Indian-based scripts such as Sinhala, Bengali, Dzongka, Thai, Khmer and so on. Myanmar script is characterized as “a-vowel accompanying consonant syllabics” (Peter and William 1996), i.e, this syllabic script consists of consonant letters accompanied by inherent vowels. Since a consonant associated with its inherent vowel can indicate a standalone syllable, it would be appropriate to call it a consonant syllable, but we simply call it a consonant letter for simplicity (Peter and William 1996).

3.3.1 Myanmar Syllable Structure in Phonetic View

Myanmar script is known as diacritically modified consonant syllabic scripts or alphasyllabaries as it is derived from Brahmi and it inherited systemic features of Brahmi. The Brahmi system based on the unit of the graphic “syllable” or aksara, which by definition always ends with a vowel (type V, CV, CVV, etc). Further, the basic consonantal character without any diacritic modification is understood to automatically denote the consonant with the inherent vowel.

Myanmar sentence is composed of words, and words written in Myanmar script are made of a series of distinct single syllables. In phonological representation, each syllable is made up of two parts:

- (1) A consonant or sometimes two consonant together and

(2) A vowel or a vowel and a final consonant together.

The first part of the syllable is known as *head* and the second part the *rhyme*. The rhyme may also contain tone.

As an example, here is the same principle applied to some English words:

(a) head	+	(b) rhyme	=	Syllable
(consonants or two consonants)		(vowel or vowel and final consonant)		
T	+	EE	=	TEE
T	+	ICK	=	TICK
TR	+	ICK	=	TRICK
TR	+	EE	=	TREE

In Burmese script the head of a syllable may be either

(1) an “initial consonant”; for example, the consonants
written: ဝ လ န ဝ

pronounced: p- l- n- thor

or

(2) an initial consonant combined with a second consonant, referred to below as a
“medial consonant”; e.g.

written: ပြ လှ နှ ဝှ

pronounced: py- ly- hn- thw-

There are only four medial consonants in Burmese script.

The rhyme of a syllable may be written with either

(1) an attached vowel symbol; e.g.

written: ဝိ လူ နာ သို

pronounced: pi lu na tho

or

(2) a consonant marked as a final consonant by carrying the “killer” symbol န် ; e.g.

written: ပန် လန် နတ် သတ်

pronounced: pan lan naq theq

or

(3) a combination of an attached vowel symbol and a final consonant; e.g.

written: ပုန် လိန် နောင် သိုက်

pronounced: poun lein naun thaiq

In addition, tones are part of the rhyme and are mostly represented by the two tone marks ◌် and ◌း; e.g.

written: ပုန် လိန်း နား သို့

pronounced: pou'n lei'n na' tho'

Other ways of representing tone are used for certain rhymes.

It is also good to note that vowels in the Myanmar script are always attached to their heads (consonants) — they are not letters in their own right, like the *a*, *e*, *i*, *o*, *u* of the roman alphabet, and they are not normally written independently. However, a Myanmar letter A “အ” (U+1021) is used to write syllables that have no initial consonant, such as

“i” written အိ, “an” written အန်, “oun” written အုန်

The Myanmar letter A “အ” occupies the position of the initial consonant in the written syllable, but is read aloud as “no initial consonant” (John Okell 2008).

From the view of phonology, Myanmar syllable has the phonemic shape of

$$C[M]V[CK]D \mid V[CK]D$$

where C refers an initial consonant, M for medial consonant, V for attached vowels, CK stands for a ending consonant (a combination of consonant C and vowel killer K), and D for the tones respectively. The symbol [] means optional.

Therefore, minimal syllable in phonetic view is CVD or VD.

3.3.2 Myanmar Syllable Structure in Orthographic View

Myanmar syllable structure contains up to 5 sub-syllabic groups namely consonant (C), medial consonant/ dependent consonant sign (M), dependent vowel (V), *Asat* or a vowel killer (K) and tones (D) (which are taken from the group of Various Signs) and these groups can appear in a syllable as one of the following combinations listed below. These combinations can be described as a regular expression $S := C M^+ V^+ F^+ D^+ \mid I F^+$ if the contraction process is done. According to that regular expression, only a consonant can be syllable breaking point in orthographic syllabification which means minimal syllable in orthographic view is C.

Consonant only	Consonant followed by Vowel	Consonant followed by Ending Consonant	Consonant followed by Medial
C	CV	CF	CM
	CVF	CFD	CMV
	CVD		CMVD
	CVFD		CMVF
			CMVFD
			CMF
			CMFD

And S becomes $S := C M^* V^* F^+ D^+ | I F^+$ if the contraction of vowels and medials is not considered. Here, the symbol “+” stands for 0 or 1 occurrence of the character where as “*” represents 0 or more occurrences. The combination of consonant and *Asat* or vowel killer (CK) is called ending consonant (F) and the syllables end with this combination are known as closed syllables.

3.4 Syllabification of Irregular Words

3.4.1 Myanmar Irregular Words

Additional complexity of Myanmar language arises from several irregular writing practices. As Myanmar script is adapted from ancient Mon script, and ultimately based on Indian Brahmi prototype, it has the convention of preserving the original spelling of (mostly) Indian loan words and also follows the Indian practice of stacking geminate and homorganic² consonants (Peter and William 1996). Therefore, we generally name the group of words with such particular forms as *irregular* throughout the book for readability and simplicity. Imported words also spelt with another kind of irregular conventions. The *Irregular* words refer the words with kinzi, consonant stacking, consonant repetition, loan or imported words, great SA and contractions and which are briefly discussed below.

- (1) **Kinzi.** As mentioned in the previous section, final consonant (F or CK) can follow the main consonant and the resulting syllable is called closed syllable (CF or CCK). In a few words (many of them loanwords), the combination of consonant Nga “င” and vowel killer “့” is “င့” which is not written on the line as the usual way, but is placed above the first consonant of the next syllable. The name of the reduced form is called kinzi and meaning “forehead rider” for obvious reasons (John Okell 1994). Some examples are shown below.

² **Homorganic consonants** (from *homo-* "same" and *organ* "(speech) organ") is a phonetics term for consonant sounds that are articulated in the same position or place of articulation in the mouth, such as (m), (p), (b) (pronounced with both lips), or (t), (d), (s), (z), (n), (l) (pronounced by touching the tip of the tongue to the upper gums).

Words with reduced form of “င”	Meaning	Unicode Sequence
အင်္ဂလန်	England	1021 1004 103A 1039 1002 101C 1014 103A
သင်္ဘော	Ship	101E 1004 103A 1039 1018 1031 102C
အင်္ဂါနေ့	Tuesday	1021 1004 103A 1039 1002 102B 1014 1031 1037

- (2) **Consonant Stacking.** There are some final consonants which are not written in the reduced form like kinzi. With all final consonants other than kinzi (င်), instead of the final consonant being forced up and over the next consonant, it is the next consonant that is forced down and under the final consonant. So, if we have a pair of syllables like သန့် and တာ and they appear in a word that requires them to be compressed, then force the main consonant of second syllable တ under the န်, and write as သန္တော. The same consonants can be stacked and it is known as consonant repetition.
- (3) **Loan Words.** Loan words mean the words which adopt the English pronunciation directly and sometimes adding Myanmar pronunciation together with English pronunciation. For example, English word “car” is written as “ဘတ်စ်ကား” where the first syllable “ဘတ်စ်” is English pronunciation and the second syllable “ကား” is Myanmar pronunciation.
- (4) **Great SA.** Myanmar consonant Great SA “သ” is commonly used in Myanmar words . For example, the English word “problem” in Myanmar language is written using Great SA as “ပြဿနာ”.
- (5) **Contraction.** There are usages of double-acting consonants in Myanmar and as the name give which acts both the final consonant of one syllable and the initial consonant of the following syllable. For example, the word လောက္ခ which means

“man” is the contracted form of ယောက်ျား. There are also some words which can be written in both in standard syllable structure and contracted form, for example, the word “daughter” is written as သမီး and သွီး. (Tin Htay Hlaing and Mikami 2013)

3.4.2 Example of Syllabification

As discussed in the above section, irregular Myanmar words have different systems of writing and syllabification of these words requires special handling and is different from those of other words. This section explains how syllabification of irregular words is processed.

Types of Irregular Words	Input Words	Correct orthographic syllabification	Meaning	Remark
Kinzi	အင်္ဂလန်	အင် #ဂ#လန်	England	Words with reduced form of “င”
	သင်္ဘော	သင် #ဘော	Ship	
	အင်္ဂါနေ့	အင် #ဂါ #နေ့	Tuesday	
Consonant Stacking	သန္တာ	သန် #တာ	Coral	
Great SA	ပြဿနာ	ပြ သ်#သ#နာ	Problem	
Loan Words	ဘတ်စ်ကား	ဘတ်စ် #ကား	Bus	Syllable with two ending consonants
Contraction	ယောက်ျား	ယောက် #ကျား	Husband	
	သွီး	သ #မီး	Daughter	

3.5 Problem Statement

In Myanmar language, syllable is the smallest linguistic unit and one word consists of one or more syllables. Generally, Myanmar words can be classified into (1) Standard words (i.e, words with standard syllable structure) and (2) irregular words (i.e, words with abbreviated characters or words written in special traditional writing forms which is discussed in detailed in early chapter). And each word type needs different orthographic ways of syllabifications as follows.

Word Type	Example Word	Character Sequence	Meaning	Syllabified Output	Syllabified Character Sequence
Standard Word	သမီး	CCVD	Daughter	သ # မီး	C#CVD
<i>Irregular Word</i>	တက္ကသိုလ်	CC <i>Vi</i> CCVF	University	တက်# က # သိုလ်	CC <i>A</i> #C#CVF

In this example, the word “Daughter, သမီး” has two syllables, သ and မီး where the former syllable has only one sub-syllabic element, consonant သ but the latter has three sub-syllabic elements of consonant မ, vowel ိ and diacritic ့.

For the word “University, တက္ကသိုလ်” has three syllables and it is written in one of the special traditional writing formats known as consonant stacking. The symbol *Vi* stands for Myanmar Sign VIRAMA (U+1039) in the sequence. Consonant stacking is syllabified by extra insertion of Myanmar Sign ASAT, U+103A (represented as letter A in the sequence) between two stacked consonants to get the syllable boundary. For instance, the first character တ is combined with the upper character from the stack and form one syllable and the lower character in the stack itself becomes a syllable. Thus such kind of language-specific features make Myanmar syllable segmentation task complicated. Myanmar language has five different forms of *irregular* words which are explained in early chapter.

Besides, many Pali words and English loan words that refer to people, places, abbreviation of foreign words, currency units etc., can be found in Myanmar texts and we need to tackle segmentation of such words.

In handling such *irregular* words in under-resourced language, Myanmar, rule-based approach and corpus-based approach have shown some failures. Thus, we develop FST based approach which works for syllabification of both standard and irregular words with high accuracy.

3.6 Literature Review

Syllable segmentation of most Alphabetic and Arabic scripts is done phonetically, i.e, the input surface string is first converted into phonetic symbols and then syllabify by using suitable approach such as rule-based or statistical approach. For the Indian-based script Abugida, syllabification can be done either phonetically or orthographically.

There are many approaches tackling the syllable segmentation task. Generally, these can be divided into two broad categories namely rule-based and data-driven approaches. The rule-based method effectively embodies some theoretical position regarding the syllable, whereas the data-driven paradigm infers “new” syllabifications from examples assumed to be correctly-syllabified already. However, it is difficult to determine correct syllabification in all cases and so to establish the quality of the “gold-standard” corpus used either to quantitatively evaluated the output of an algorithm or as the example-set on which data-driven methods crucially depend (Marchand and et al 2007). Besides these two categories, statistical methods and finite state methods are also applied for automatic syllabification and we will introduce previous approaches briefly.

In syllabification of written Uyghur (Maimaitimin Saimaiti , Zhiwei Feng 2007), a rule-based approach that uses the Principle of Maximum Onset is applied. Experiment on a random sample shows that the syllabification algorithm achieves 98.7 percent word accuracy on word tokens, 99.2 percent on word types, and 99.1 percent syllable accuracy. For Sinhala syllabification (Ruvan and eta al 2005), the authors described rule based

syllabification algorithm for Sinhala after analyzing the syllable structure and linguistic rules for syllabification of Sinhala words. Rule-based syllabification algorithm for Malay is proposed based on Maximum Onset principle for text to speech system in (El-Imam and Don 2000).

In (Marchand and eta al 2007), one rule-based approach, and three data-driven approaches are evaluated (A Look-up Procedure, an Exemplar-based generalization technique and the syllabification by Analogy (SbA)). The results on the three databases show consistent and robust patterns: the data-driven techniques outperform the rule-based system in word and juncture accuracies by a very significant margin and best results are obtained with SbA.

In (Kiraz, G.A., M'obius, B. 1998), a weighted finite-state-based approach to syllabification is presented. Their language-independent method builds an automaton for each of onsets, nuclei, and codas, by counting occurrences in training data. These automata are then composed into a transducer accepting sequences of one or more syllables. They do not report quantitative results for their method. Syllabification of Middle Dutch texts is done by the method which combines a rule-based finite-state component and data-driven error-correction rules. The authors adapt an existing method for hyphenating (Modern) Dutch words by modifying the definition of nucleus and onset, and by adding a number of rules for dealing with spelling variation (Bouma and Hermans xxxx).

Regarding Myanmar syllabification, two of the above mentioned approaches have already been done. In the corpus-based longest matching approach (Hla Hla Htay and Murphy 2008), the authors collected 4,550 syllables from different resources. The input texts are syllabified by using longest matching algorithm over their syllable list. They observed that only 0.04% of the actual syllables were not detected and described their failures as three facts:

- (1) differing combinations of writing sequences
- (2) loan words borrowed from foreign languages

(3) rarely used syllables not listed in their syllable list

Rule-based Myanmar syllable segmentation is done by (Zin Maung Manung and Mikami 2008) in which input text strings are converted into equivalent sequence of category form (e.g. CMCACV for the word “Myanmar”) and compares the converted character sequence with the syllable rule table to determine syllable boundaries. The authors tested 32,238 syllables in the Myanmar Orthography (Myanmar Language Commission 2006) and the experimental results show an accuracy rate of 99.96% for segmentation. However, their approach cannot solve for the segmentation of *irregular* words with traditional writing forms namely kinzi, consonant stacking, great SA and English loan words with *irregular* forms as shown in table 1. However, in our approach, these kinds of failures in (Hla Hla Htay and Murphy 2008) and (Zin Maung Manung and Mikami 2008) are addressed the syllabification of *irregular* words and managed correctly.

Table 6. Syllable Segmentation Examples and Results³

Myanmar Text	Letter Sequence	Segmented Letter Sequence	Segmented Result
အတ္တုရသရက်	CCSCCSCCCCA	[CCSCCSC CC CCA]	[အတ္တုရ သ ရက်]
ဥတ္တရယဉ်စုန်းတန်း	ECSCCCACMCAFCFAF	[ECSC C CCA CMCAF CCAF]	[ဥတ္တရ ယဉ်စုန်း တန်း]
ဣစ္ဆာသယ	ECSCVCC	[ECSCV C C]	[ဣစ္ဆာ သ ယ]
ဧကရာဇ်	ICCVCA	[I C CVCA]	[ဧ က ရာဇ်]
ဝင်္ကဇာသိ	CCASCCSCCVCA	[CCASCCSC CVCA]	[ဝင်္ကဇာ သိ]
မားစ်ဂြိုဟ်	CVFCACMVVCA	[CVFCA CMVVCA]	[မားစ် ဂြိုဟ်]
မနုဿိဟ	CCVGVCA	[C CVGV C]	[မ နုဿိ ဟ]

In (13) Unicode Technical Note, diacritic storage order of Myanmar characters in Unicode (which we refer as sub-syllabic components) is explained in detail and it is highlighted that diacritic storage order does not define a phonetic syllable. Further, automated syllable breaking approach for Myanmar script is mentioned. It is said that the syllable break may occur before any character cluster so long as the kinzi, asat and stacked slots remain empty in the cluster following the possible break point. It is mentioned that their algorithm does not require dictionary but still needs more refinement, for example,

³ This table is directly taken from the original manuscript of (Zin Maung Manung and Mikami 2008).

sequence of digits should be kept together and visible virama needs more complex analysis. The author also stated that the result of syllable breaking can be applied for line breaking and the same syllable breaking rules can be applied for lexicographic sorting.

Finite state methods have been applied in syllabification of alphabetic scripts but it has not yet been done in Myanmar script. Therefore, in our study, we describe syllable structure model in regular grammar and write regular expressions which are used as input to Finite State Transducer. In our experiment, we use 32,238 syllables covering all possible syllable structure in standard words and *irregular* words from Myanmar Orthography published by Myanmar Language Authority (Myanmar Language Commission 2006) and our transducer correctly syllabified all the tested words. We implemented the syllabification system by using the programming language for finite state transducer, SFST-PL.

3.7 Finite State Transducer for Myanmar Syllabification

3.7.1 Constructing Regular Grammar and Transducer

Automatic syllabification of word is challenging, not least because the syllable is not easy to define precisely. Consequently, no accepted standard algorithm for automatic syllabification exists (Le Hong and et al 2008). However, syllabification can be achieved by writing a declarative grammar of possible locations of syllable boundaries in polysyllabic words (John Okell 2002). On the other hand, Finite state machines are widely used in the field of natural language processing. Finite state transducers (FSTs) have been used ubiquitously in the domain of phonology as well as in morphology for all sorts of string mapping between string descriptions. Again, Finite state automata and transducers have been used in natural language processing of Asian languages for example, morphological analysis of Urdu (Hassain 2004) and likewise, formal grammar is used to express Sinhala computational grammar (Chamila and et al 2012). This study proposes application of FST for Myanmar syllabification though the general use of FST is for Morphology.

In our approach, first we write syllable structure in regular grammar and then converted it into regular expression as FST program which is a set of regular expression. We adopt two-level morphology approach in which the surface string, for example, “boys” is analyzed and produced together with lexical information as “boy<Noun><Plural>”. In our case, our syllabification transducer accepts input string/surface string and outputs the string together with syllable boundary information.

Grammar rules for independent vowels, digits and abbreviated syllable can be described as

$$S \rightarrow \text{က} \mid \text{ဤ} \mid \text{ဥ} \mid \text{ဦ} \mid \text{ဧ} \mid \text{ဩ} \mid \text{ဪ}$$

$$S \rightarrow \text{ဝ} \mid \text{၁} \mid \text{၂} \mid \text{၃} \mid \text{၄} \mid \text{၅} \mid \text{၆} \mid \text{၇} \mid \text{၈} \mid \text{၉}$$

$$S \rightarrow \text{ဦ} \mid \text{ဦ} \mid \text{င} \mid \text{၏}$$

and the syllable with five sub-syllabic elements can be written as

$S \rightarrow \text{က} X \mid \dots \mid \text{အ} X \quad \# 33 \text{ rules for all consonants}$ $X \rightarrow \text{ချ} A \mid \dots \mid \text{ဇွ} \quad \# 11 \text{ rules for all medials}$ A $X \rightarrow \text{ာ} B \mid \dots \mid \text{ိ} \quad \# 12 \text{ rules for all vowels}$ B $X \rightarrow \text{က} T \mid \dots \mid \text{အ} T \quad \# 33 \text{ rules for ending consonants (consonant + vowel killer)}$ $X \rightarrow \varepsilon$ $A \rightarrow \text{ာ} B$ $A \rightarrow \text{က} T \mid \dots \mid \text{အ} T \quad \# 33 \text{ rules for all consonants}$ $A \rightarrow \varepsilon$ $B \rightarrow \text{က} T \mid \dots \mid \text{အ} T \quad \# 33 \text{ rules for all consonants}$ $B \rightarrow \text{း} \mid \text{း} \quad \# 2 \text{ rules for tones}$ $B \rightarrow \varepsilon$ $T \rightarrow \text{်} D$
--

$$\begin{array}{l} D \rightarrow \text{◌} | \text{◌} \quad \# \text{ 2 rules for tones} \\ D \rightarrow \varepsilon \end{array}$$

Thus, we developed the transducer which accepts input Unicode strings and then output the strings with correct syllable boundaries.

Firstly, based on the regular grammar as mentioned above, we write the regular expression to recognize Myanmar syllables and then construct the orthographic automata for a Myanmar syllable A_{mm} as

$$A_{mm} = C \text{ Opt}(M) \text{ Opt}(V) \text{ Opt}(CK) \text{ Opt}(D) \mid I \text{ Opt}(CK)$$

where Opt is the just acronym for “optional”.

The above automaton accepts one syllable at a time and it can check the combinations of sub-syllabic elements orthographically.

Then, we construct the syllabification automaton, denoted by A_{syl} , which accepts a sequence of syllables and finds the syllable boundaries correctly. This is achieved by the expression

$$A_{syl} = A_{mm} (\# A_{mm})^*$$

In this expression, syllable structure represented by A_{mm} is followed by zero or more occurrence of the boundary marker (#) and a syllable form A_{mm} . The automaton A_{mm} accepts the sequence of syllables but we need to transform this into a transducer which inserts a boundary marker `#` after each syllable but not after the last syllable. This is simply achieved by computing the identity transducer for A_{mm} and replacing `#` with a mapping $\varepsilon : \#$ in A_{mm} . Now, the syllabification transducer becomes

$$T_{mmsyl} = \text{Id}(A_{mm}) ((\varepsilon : \#) \text{Id}(A_{mm}))^*$$

The syllabification Transducer for words with standard syllable structure is shown in figure 5.

For *irregular* words, the stored character sequence is special. It usually uses invisible Myanmar sign *VIRAMA* (U+1039) in the input sequence encoding but it is required to output different character or characters according to the types of *irregular* words in section 3.5. Though *irregular* words are written in tradition forms and complicated, the result of their syllabification turned into standard word syllable structure.

We construct the finite state transducers (FST) for each type of syllabification of *irregular* words respectively and we also show finite state transducer for standard syllable structure in figure below as an example.

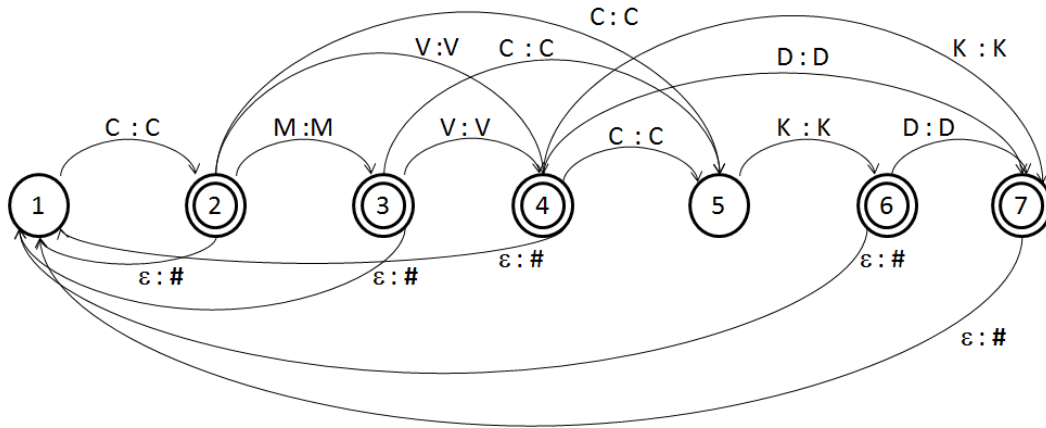


Figure 5 Finite State Transducer for Myanmar syllabification

3.7.2 Implementation

Based on the regular grammar which we constructed manually, we implemented syllabification transducer and set up the experiment by using Stuttgart Finite State Transducer Tool (SFST) although it is primarily concerned with morphology, for example, SMOR (a large German Morphological Analyzer). The specification of our syllabification

transducer is written in SFST-PL, the programming language of the SFST tools which is a set of regular expressions.

During the experiment, it is found that the default Myanmar keyboard layout in Ubuntu is in Unicode 4.1 based “Myanmar 1” Font and the code point values of some characters are different from the Unicode 6.1 Myanmar character code table. Thus we customized the keyboard layout file namely `/usr/share/m17n/my-kdb.mim` so as to get correct syllabification result. Part of the updated keyboard layout for Myanmar 3 Unicode font is as follows.

Table 7. Updated Keyboard Layout in my-kdb.mim

Original Layout	Updated Layout	Represented Characters
("s" 0x1039 0x101A) ("S" 0x1039 0x101F)	("s" 0x103B) ("S" 0x103E)	Myanmar Consonant Sign Medial YA (s-key) and HA(S key)
("j" 0x1039 0x101B) ("J" 0x1032) ((A-j) 0x104E)	("j" 0x103C) ("J" 0x1032) ((A- j) 0x104E)	Myanmar Consonant Sign Medial RA (j-key) and Myanmar Vowel Sign AI (J- key)
("k" 0x102F)	("k" 0x102F) ("K" 0x1012)	Myanmar Vowel Sign U (k-key) and Myanmar letter DA (K-key)
("f" 0x1039) ("F" 0x200D)	("f" 0x103A) ("F" 0x1039)	Myanmar Sign ASAT (f-key) and Myanmar Sign Virama or killer (F-key)
("g" 0x200C) ("G" 0x1039 0x101D)	("g" 0x102B) ("G" 103D)	Myanmar Vowel Sign TALL AA (g-key) and Myanmar Consonant Sign Medial WA (G-key)

Then, we constructed the finite state transducer using finite state programming language (Schmid 2007). We first implement FST for single syllable and find that it recognizes every syllable structure. Then, we upgraded our FST for polysyllabic-words and insert the

syllable boundary marker (we use # sign as our boundary sign). Our FST program is as follows.

```

$c$=က | ခ | ဂ | ဃ | င | စ | ဆ | ဇ | ဈ | ည | ဋ | ဌ | ဍ | ဎ | တ | ထ | ထ | ဖ | ဖ | န |
ပ | ဖ | ဖ | ဘ | မ | ယ | ရ | လ | ဝ | သ | ဟ | ဌ | အ | ဉ
$m1$= ချ | င | ဝ | ဟ
$m2$= (ချ | င )? ဝ
$m3$= (ချ | င )? ဟ
$m4$= ဝ ဟ
$m5$= (ချ | င )? ဝ ဟ

$v1$= တ | ဝ | ဝီ | ဝဲ | ဝါ | ဝု | ဝူ | ဝံ | ဝိ
$v2$= ဝေ | ဝေါ
$v3$= ဝော် | ဝေါ
$v4$= ဝီ
$v5$= ဝု ဝံ | ဝံ ဝု
$standalone$= က | ဤ | ဤ | ဉ | ဇ | ဇြာ | ဇြာ | ငါ | ဤ
$annuvsara$= ဝ
$f1$= $c$ ဝ
$asat$= ဝ
$THA$= သ

$d1$= : | ဝ

$virama$= ဝ
$standcombine$= $standalone$ ($v1$ | $v2$ | $v3$ | $v4$ | $v5$ )? ($f1$)?
($d1$)?
$tincorpus$= "mmcorpus2"

$normal$=$c$ ($m1$|$m2$|$m3$|$m4$|$m5$)? ($v1$ | $v2$ | $v3$ | $v4$ | $v5$
)? ($f1$)? ($d1$)?
$kinzi$= $c$ ($m1$|$m2$|$m3$|$m4$|$m5$)? ($v1$ | $v2$ | $v3$ | $v4$ | $v5$
)? $c$ ဝ ဝ:<>

$stack$= $c$ ($m1$|$m2$|$m3$|$m4$|$m5$)? ($v1$ | $v2$ | $v3$ | $v4$ | $v5$
)? ($c$) ဝ: ဝ | $standalone$ ($m1$|$m2$|$m3$|$m4$|$m5$)? ($v1$ | $v2$
| $v3$ | $v4$ | $v5$ )? ($c$) ဝ: ဝ | ($c$) ($v1$ | $v2$ | $v3$ | $v4$
| $v5$ )? ($c$) ဝ: ဝ

$greatsa$=$c$ ($m1$|$m2$|$m3$|$m4$|$m5$)? ($v1$ | $v2$ | $v3$ | $v4$ | $v5$
)? သ: {သ်#သ} ($f1$)? ($v1$ | $v2$ | $v3$ | $v4$ | $v5$)?

```

```

$loan$= $c$ ($m1$|$m2$|$m3$|$m4$|$m5$)? ($v1$ | $v2$ | $v3$ |$v4$ |$v5$
)? ($f1$) ($d1$)?$c$ ဝ်
$conone$= $c$ $c$ ဝ်:{ ဝ်#} ချ : ဝ်$v1$
$contwo$= $c$ ($m1$|$m2$|$m3$|$m4$|$m5$) $c$ ဝ်:{ ဝ်#} ချ : {ခ ခ }$f1$
$conthree$= $c$ ဝ်:# $c$ ($v1$ | $v2$ | $v3$ |$v4$ |$v5$ )$d1$
$confour$ = $c$ ဝ်:{က ဝ်#} $c$ $f1$
$confive$ = $c$ ($v1$ | $v2$ | $v3$ |$v4$ |$v5$ ) $f1$
($m1$|$m2$|$m3$|$m4$|$m5$) ($v1$ | $v2$ | $v3$ |$v4$ |$v5$ ) $d1$
$standgreatsa$=$standalone$ ဝ်: {ဝ်#ဝ်}
$standgreatsavowel$ = $standalone$ ဝ်:{ဝ်#ဝ်} $v1$ <>:{#} $stack$
$greatsalast$ = $normal$ဝ်: {ဝ်#ဝ်}($c$) ဝ်: ဝ်
$annuv$= $normal$ $annuvsara$
$mmsyllable$= $normal$ | $kinzi$ | $stack$ | $greatsa$ | $loan$ |
$conone$ | $contwo$ | $conthree$ | $confour$ |$confive$ | $standalone$ |
$standcombine$ | $standgreatsa$ | $standgreatsavowel$ | $greatsalast$ |
$annuv$
$complex$ = $mmsyllable$ ( (<>:#) $mmsyllable$)*

$stincorpus$ || $complex$

```

In the experiment, the data from Myanmar Orthography are fed into the corpus named “mmcorpus” in our program. Then, finite state transducer reads the words from the corpus one by one and produces the output with correct syllable boundaries. Some results of syllabification on standard words are shown below.

```

thh@ubuntu:~/SFST/src$ fst-compiler-utf8 syltesting.fst syltesting.a -
s
thh@ubuntu:~/SFST/src$ fst-infl syltesting.a mmcorpus
reading transducer from file "syltesting.a"...
finished.
> ကကုသန်ဘုရား
က#ကု#သန်#ဘု#ရား
> ကကူရံငါး
က#ကူ#ရံ#ငါး
> ကခွန်
က#ခွန်
> ကျွန်းမြေကိုင်းမြေ
ကျွန်း#မြေ#ကိုင်း#မြေ

```

The following results are examples of syllabification for *Irregular Words* covering consonant stacking, consonant repetition, Kinzi, Great SA, contraction.

```

thh@ubuntu:~/SFST/src$ fst-infl complexsylEightIdn.a mmcorpus2
reading transducer from file "complexsylEightIdn.a"...
finished.

> ဗျူမဟော်ဂနီ           #Consonant Stacking
ဗျိုက်#ဃ#မ#ဟော်ဂနီ
> ပုလ္လိုင်                 #Consonant Repetition
ပုလ်#လိုင်
> တနသ်ာရီတိုင်း         #Kinzi
တ#နင်#သာ#ရီ#တိုင်း
> ဝေဿန္တရာ               #Great SA
ဝေသ်#သန်#တ#ရာ
> ပြဿနာ
ပြသ်#သ#နာ
> ကျွန်ုပ်                 #Contraction
ကျွန်#နုပ်
> သွီး
သ#မီး
> လက်ျာ
လက််#ယာ
> လွက်
လက််#ဘက်

#Word with free standing vowels က္လ (U+1023) and consonant stacking
> က္လန္တေဆည်
က္လ်#ဒြေ#ဆည်

#Word with vowel Sign I (U+102D) and Myanmar Sign Anusvara (U+1036)
> တာဝတိသာ
တာ#ဝ#တိ#သာ

```


3.7.3 Experimental Result

For the test data set in our experiment, we use Myanmar Orthography published by Myanmar Language Commission which is a standardized system to write Myanmar words including rules of spelling (Myanmar Language Commission 2006). Based on the regular grammar for syllable structure of Myanmar, we can identify correct syllable boundaries in the given texts. We tested all 11,732 distinct words contained in Myanmar Orthography corpus yielding 32,283 syllables covering standard and *irregular* words. Details of the results based on the types of word can be found as follows.

Table 8. Details of Experimental Results

No.	Type of Words	No. of words	Correctly Syllabified Words	% of correct syllabification for each word type
1.	Standard Words	11,092	11,092	100%
2.	Irregular Words	640		
	2.1 Consonant Stacking	253	245	96.83%
	2.2 Consonant Repetition	266	266	100%
	2.3 Kinzi	71	71	100%
	2.4 Great SA	26	26	100%
	2.5 Contraction	3	3	100%
	2.6 Loan Words	21	21	100%
	TOTAL	11,732		

Initially, we found that only syllabification of 8 stacked words out of 11,732 words are erroneous. Therefore, we received 99.93% of overall accuracy covering both types of word, standard and *irregular* words. By doing error analysis, the errors are caused by those words in which free standing vowels ဖ (U+1023), consonant stacking and other sub-syllabic elements are mixed and our FST cannot find correct syllable boundaries for these words.

However, this kind of error is improved in our later experiment. And as a result, all tested words in the Myanmar Orthography can be syllabified correctly.

We also analyze the syllabification results of our approach and other existing approaches on *irregular* words.

Table 9. Comparison of Syllabification Results on Irregular Words

<i>Irregular</i> Words	Corpus-based Method	Rule-based Method	Finite State Transducer Method
1. Consonant Stacking	NO	NO	YES
2. Consonant Repetition	NO	NO	YES
3. Kinzi	NO	NO	YES
4. Great SA	NO	NO	YES
5. Contraction	NO	NO	YES
6. Loan Words	YES but with some failures	YES for regular words	YES

Further, we summarize and compare the accuracy of the developed methods on Myanmar syllabification as follows.

Table 10. Summary of Myanmar Syllabification Methods

Method	Source Data	Total no. of Syllables	Syllabification of Standard Words	Syllabification of <i>Irregular</i> Words	Accuracy (%)
Corpus-based Longest Matching Method	11 Short Novels	70,384	YES	Plases refer to Table 4.	99.96%
Rule-based	Myanmar Orthography	32,238	YES	Please refer to	99.96%

Method				Table 4.	
Finite State Transducer Method	Myanmar Orthography	32,238	YES	YES	100%

According to the above table, our approach covers both standard and *irregular* words.

3.8 Contributions of the Research

Although automatic syllabification is an important component in several natural language processing tasks, little has been done in deploying formal approach. This FST-based approach is expected to be a significant contribution to the field, and especially to researchers working on various aspects of Myanmar language and other Asian scripts.

For automatic syllabification of alphabetic languages, spelling does not have well-defined structure. Thus, the input texts are transliterated into phonetic symbols and syllabification is done on these transliterated texts, not on the input texts directly. Moreover, it is necessary to apply additional information using dictionary or annotated corpus and even FSA-based approach could be applied in statistical way.

Myanmar script is Indic script in origin like Lao and Thai. Myanmar syllable structure is well-defined and has structural regularity. And thus, it could be represented in finite state model. In general, finite state language processing becomes popular because of their simple, elegant and efficient computational power. From computational point of view, finite state based methods achieve a good performance and they are suitable for application more interested in speed and memory footprint. Many works have been done for Myanmar syllabification but FSA approach has not yet been applied to Myanmar.

This research proved that FSA approach gives significant solution for syllabification of Myanmar language as we achieve correct syllabification without applying step-by-step rules and the need of annotated corpus. This is major achievement of our study because

syllabification of other languages, for example English, could be done with the availability of lexicon containing marked syllable boundaries for both spelling and pronunciation domains. And we hope that it could be applicable to automatic syllabification of other syllabic writing systems.

4 NORMALIZED CODE SEQUENCE FOR MYANMAR LANGUAGE : FSA BASED CODE SEQUENCE CHECKER FOR SAFE IDNS

3.1 Introduction

Now-a-days, not only online contents but also domain names can be represented in local languages and this makes human society with better communication, better education and better business. However, the introduction of what are called internationalized Domain Names (IDNs) amplifies both the difficulty of putting names into identifiers and the confusion that exists between scripts and languages. The introduction of Unicode support in operating systems and applications has lead to a vastly increased number of available homoglyphs and a new threat arose from the use of characters which are visually indistinguishable from western characters but belong to a non western script (ICANN, 2005) (Peter and Bolan 2009). Given the added complications of using a much broader range of characters than the original small ASCII subset, precautions are necessary in the deployment of IDNs in order to minimize confusion and fraud. Considering that a large number of users are not scholars of the language and hence can be easily cheated by homographs, and spoofing, and phishing will occur to a large extent in languages. This calls for great care and caution in supporting local languages and scripts in the domain names (Department of IT, Government of India 2009).

Phishing attacks can also occur to Myanmar language as it has different types of combining marks, similar-looking characters and different ways of combination of character code sequence. Possible threats on IDNs have been discussed and also, guidelines and recommendations for IDN based attacks are developed as mitigating strategies by international organizations. Though general guidelines and mitigating methods are stated to cover IDN implementation in all scripts, there are some scripts like Myanmar, which need language specific mitigation methods.

Therefore, this chapter covers the implementation of finite state automata to check Myanmar character code sequence and the case study to apply FSA is Myanmar IDN (Internationalized Domain Name) strings to make them as secure as possible. We construct the FSA by analyzing the language specific characteristics of Myanmar language and our developed FSA is able to check the format of Myanmar IDN strings as well as the correct combination of sub syllabic elements in a syllable.

4.1.1 Phishing and Internationalized Domain Names (IDNs)

Phishing is a criminal mechanism employing both social engineering and technical subterfuge to steal consumers` personal identity data and financial account credentials. Other techniques typically used in phishing attacks include content spoofing and spamming, the former referring to their resemblance to original Web sites and the latter to fake mails sent to potential victims. Phishing has been already discussed one decade ago under the more general term Web spoofing in literature (Viktor Krammer 2006).

IDNs mean web addresses represented by local language characters which enable more web users to navigate the Internet in their preferred scripts. The Internet Corporation for Assigned Names and Numbers allows users of the internet are able to utilize any Unicode character in constructing a domain name. For example, <http://වෙබ්.පාර්ලිමේන්තුව.ලංකා//> , the home page for the Parliament of Sri Lanka.

Different kinds of phishing attacks are possible and among them, the attack which use Internationalized Domain Names (IDNs) strings to spoof the users are increasing now-a-days. Although the authors of IDNA (Internationalized Domain Names in Applications) had security in mind, Gabrilovich and Gontmakher discovered a possible attack scenario, called the homograph attack, in which the characters might be replaced by visually identical ones (homographs) found in Unicode. The grouping single character homoglyphs with either other homoglyphs or other characters to form a visual twin of a known word is referred to as a homograph. Homographs may be constructed across a single or multiple character set or script and thus the differential terms single-script and multi-script homograph are found throughout the literature (Hanary and Bolan 2010).

4.2 Problem Statement

After internationalized domain names (IDNs) had been introduced in 2007, possible threats on IDNs have been discussed and also, guidelines and recommendations for IDN based attacks are developed as mitigating strategies by international organizations, for example, ICANN. Similarly, each registrar sets the rules and guidelines to handle IDN based attacks, i.e, look alike character issues.

For alphabetic language, the solution is simple. It is just necessary to prepare character tables and check one-to-one letter similarity. However, in Myanmar language, it has complicated combining characters and homographs. In this case, letters are not just similar but have exactly the same as shown in Table 15 though underlying code sequence is completely different. Thus, we need special treatment to handle this issue and normalized code sequence order is necessary.

Though general guidelines and mitigating methods are stated to cover IDN implementation in all scripts, there are some scripts like Myanmar, which need language specific mitigation methods. Thus in this section, Myanmar homoglyphs and possible spoofings for Myanmar domain names are described and types of spoofing which need language specific checking method for Myanmar is highlighted.

Having look-alike characters in any script are prone to phishing attacks especially when they are displayed in a default address bar of browsers. A careful visual inspection of Myanmar letters shows that some letters open for visual spoofing because of their visual similarities. Our proposed Myanmar homoglyphs, look-alike consonants, and digits are listed in the following tables.

Table 11. Myanmar look-alike digits

No.	Glyph	Unicode Value	Name
1-a	၆	U+1046	Myanmar Digit Six
1-b	၉	U+1049	Myanmar Digit Nine

Table 12. Myanmar homoglyphs

No.	Glyph	Unicode Value	Name
2-a	o	U+101D	Myanmar letter WA
2-b	o	U+1040	Myanmar Digit Zero

Table 13. Myanmar look-alike consonants

No.	Glyph	Unicode Value	Name
3-a	ၓ	U+1000	Myanmar letter KA
3-b	ၔ	U+1018	Myanmar letter BA
3-c	ၕ	U+101A	Myanmar letter YA
3-d	ၖ	U+101E	Myanmar letter SA
3-e	ၗ	U+101F	Myanmar letter HA
4-a	ၘ	U+101B	Myanmar letter YA
4-b	ၙ	U+1047	Myanmar Digit Seven
5-a	ၚ	U + 1025	Myanmar Vowel U
5-b	ၛ	U+1009	Myanmar letter NYA
6-a	ၜ	U + 1010	Myanmar letter TA
6-b	ၝ	U + 1011	Myanmar letter THA

Development of applications in multilingual environment is brought by UCS/Unicode. Unicode contains a large number of characters, and incorporates the varied writing systems of the world, incorrect usage can expose programs or systems to possible security attacks. Unicode consortium also listed seven possible spoofings which are described as visual security issues namely

- (1) Mixed-script Spoofing
- (2) Single-script Spoofing
- (3) Whole-script Spoofing
- (4) Inadequate Rendering Support
- (5) Bidirectional Text Spoofing
- (6) Syntax Spoofing

(7) Numeric Spoofs

Like other scripts, Myanmar script is highly vulnerable for different types of spoofing and some spoofing cannot be solved by using general guidelines and procedure as summarized in the Table 14.

Table 14. Summary of types of spoofing, existing mitigation methods and threats for Myanmar IDNs

Types of Spoofing	Description	Existing Mitigating Methods	Possible Threats in Myanmar IDNs
Mixed-script spoofing	the existence of visually confusable characters across scripts	Mixed-script spoofing detection procedure in Unicode Security Mechanisms	“ο “ (U+03BF – Greek omicron), “ο” (U+006F – Latin small letter o), “ο” (U+043E - Cyrillic small letter o) “ο” (U+101D - Myanmar letter WA) and “ο “ (U+1040 - Myanmar Digit Zero)
Single-script spoofing	Spoofing with characters entirely within one script, or using characters that are common across scripts	IDN implementation guidelines by ICANN	“ο” (U+101D - Myanmar letter WA) and “ο“ (U+1040 - Myanmar Digit Zero) can lead to single-script spoofing.

Combining mark order spoofing	Spoofing by reordering of character	Not established yet as it is language specific issue	Highly vulnerable and need language specific checking method (please refer to table 15)
Inadequate rendering support	a font or rendering engine has inadequate support for characters or sequences of characters	it is mentioned in Unicode Security Considerations	Myanmar Unicode characters (U+1039 – Myanmar Sign Virama) is invisible, and it may affect the rendering of the characters around them.
Bidirectional text spoofing	visually confusable characters obtained by mixing inherent right-to-left and left-to-right writing directions	Unicode bidirectional Algorithm	Impossible to happen in Myanmar language
Syntax spoofing	Spoofing syntax characters eg. U+2044 (/) FRACTION SLASH can look like a regular ASCII '/' in many fonts	visual distinguishing is suggested in Unicode Security Consideration.	Impossible to happen in Myanmar language
Numeric spoofing	Individual digits may have the same shapes as digits from other scripts, even digits of different values.	IDN implementation guidelines by ICANN	“၀” (U+1040 - Myanmar Digit Zero) has visual similarity with Basic Latin digit zero (U+0030).

In most cases, two sequences of accents that have the same visual appearance are put into a canonical order. This does not happen, however, for certain scripts in Southeast Asia, so reordering characters may be used for spoofs in those cases (Mark and Michel 2013) as shown in Table 15.

Table 15. Combining Mark Order Spoofing in Myanmar

String	Unicode sequence	Punycode
၂	U+101C <u>U+102D</u> U+102F ၂ ၵ ၵ	xn--gjd8ag.com
၂	U+101C U+102F <u>U+102D</u> ၂ ၵ ၵ	xn--gjd8af.com

For Myanmar language combining diacritic marks shown in the above table, visual appearances of the given two strings are same though they have different underlying coded sequences and so as their punycode values. Current example is the dependent vowels combinations within a syllable and likewise, other Myanmar vowels and medial consonants can bring such kind of inconsistent ordering. Therefore, it is necessary to check underlying code sequence and propose an efficient code checking method to check such kind of language specific problem.

Moreover, complex and traditional writing styles of Myanmar language such as kinzi, consonant stacking, consonant repetition, and contractions could lead to combining mark order spoofing.

Myanmar has not yet been described specifically in the standards and guidelines by Unicode and ICANN. Up to our knowledge, this kind of issue is language-specific issue and should be managed by respective language authorities. Therefore, the main purpose of this study is to express possible threats in Myanmar domain names including combining mark order spoofing and propose finite state automata (FSA) based coded sequence checking to prevent mal use of Myanmar domain names.

4.3 Implementation of Myanmar Domain Names: Linguistic Issues

Among Asian countries, Myanmar having population of 54 millions, has only 1% of penetration of Internet as of 2012 and this result shows more efforts and attempts are necessary to be carried out for the IT development. Implementation of domain names in Myanmar language has become a necessity in order to popularize the use of Internet among the rural masses of Myanmar because the majority of 50 million population of Myanmar count Myanmar language (Burmese) as their first language. Thus, as a first part in this study, linguistic issues to be considered for the implementation of Myanmar domain names are proposed briefly.

- (1) **Encoding and Character Set:** Different encodings are available for Myanmar script. However, Myanmar character set has been standardized under Unicode with the range from U+1000 to U+109F. To implement domain names, it is necessary to define encoding standard and what types of characters should be allowed for Myanmar.
- (2) **Allowed and Disallowed Characters:** Unicode being a script based encoding standard groups all letters across all languages which use the same script. Thus, language specific conventions need to be given for controlling which characters may be allowed within and across scripts for a particular language. According to ICANN's PVALID characters in IDNA2008, except punctuation marks and various signs group II, all encoded characters for Myanmar script are allowed for IDN labels. These code points are summarized in the following table.

Table 16. ICANN's Myanmar character in IDNA2008

Unicode Code	Type	Description
U+1000 ~U+1049	PVALID	Myanmar letter KA ~ Myanmar Digit Nine
U+104A ~ U+104F	DISALLOWED	Myanmar letter Sign Little Section

		~ Myanmar Symbol Genitive
U+1050 ~U+109D	PVALID	Letter Shan ~ Vowel Sign Aton AI
U+109E,U+109F	DISALLOWED	Myanmar Symbol Shan One, Shan Exclamation

However, character set selection should be done based on the feedback of the native speakers because every language could have some characters which are technically possible to use but impossible to use from linguistic view.

(3) Some Spelling Variants

Some Myanmar words can be expressed in two different ways and this may cause serious IDN implementation problems. For example, the word “daughter, သွီး “ as “သမီး”. Such kind of words should be listed and it is also necessary to establish a policy with regards to words with multiple forms.

(4) gTLD and ccTLD in Myanmar language

The process namely gTLD translation process plays important role for implementation of Myanmar domain names. Myanmar language should have its own gTLD set and separate namespace but it is more likely to access existing namespace into Myanmar by using direct mapping or translation. The process should be done in the following steps:

- (1) Collecting Myanmar terms as possible as for English gTLD
- (2) Language authority and linguistic experts should consult for the best word or usage for naming gTLD
- (3) Suitable short forms or abbreviation for gTLD should be decided
- (4) We also suggest the possible words to use as Myanmar gTLD for direct mapping of some English gTLD as examples.

(5) Label Separators

In IDNA2003, three characters are listed to treat as label separators. In IDNAbis, only ASCII period is allowed to be used. If other characters are required for any language,

they are expected to be mapped before using or storing the domain name. Some language specific delimiters identified in are as follows.

Language	Character	Unicode Value
Dzongkha	%	U+0F14
Urdu and Pashto	-	U+06D4

Language interface handling these particular languages will handle these delimiters in addition to FULL STOP U+002E. It is noted that all languages did not decide to use their sentence separator marks to be used as label separators. For example, in Nepali U+0964 is used as sentence marker but they decided to use FULL STOP U+002E as label separator in domain names (Hussain and Karamat xxxx). Therefore, label separator for Myanmar domain names should be decided like other Asian scripts.

To sum up, the above mentioned five steps are crucial and the implementation of these steps could be accomplished by co-operation of technical experts, linguists and language authority.

4.4 Finite State Automata (FSA) for Myanmar Code Sequence Checking

In this section, Myanmar character combination sequences are discussed in detail and our proposed FSA based code sequence checking method is explained.

4.4.1 Myanmar Combining Marks in a Syllable

Myanmar is a syllabic script and syllable is a smallest unit in Myanmar language. As introduced in earlier chapters, most Myanmar syllables are in the structure containing at most 5 sub-syllabic groups namely consonant (C), medial consonant/ dependent consonant sign (M), dependent vowel (V), *Asat* or a vowel killer (K) and tones (D) (which are taken from the group of Various Signs) and these groups can appear in a syllable as one of the following combinations. These combinations can be described as a regular expression as

$$X = C M^* V^* F^+ D^+$$

where the symbol “ + ” stands for 0 or 1 occurrence of the character and “*” for 0 or more occurrences. In this expression, the combination of consonant and *Asat* or vowel killer (CK) is called ending consonant (F) and the syllables end with this combination are known as closed syllables.











For dependent vowels (V) and medials (M) groups, some of the members are formed by the combination of others. For instance, Myanmar script encoded in Unicode has eight dependent vowels through code values U+102B to U+1032, however, new three vowels obtained by combining some of these individual vowels and Myanmar sign ASAT (U+1039). Two or more Myanmar attached vowels are combined and formed new three members { , ,  } in the vowel set as shown in the table below.

Table 17. Vowel combining marks

Glyph	Unicode Values	Description
 + 	U+1031 + U+102C	Vowel sign E , Vowel sign AA
 +  + 	U+1031 + U+102C +U+103A	Vowel sign E, Vowel sign AA, ASAT
 + 	U+102D + U+102F	Vowel sign I, Vowel sign UU











Similarly, 4 basic Myanmar medials combine each other in some different ways and produce new set of medials { , , ,  }.

Table 18. Medial combining marks

Glyph	Unicode values	Description
 + 	U+103B + U+ 103D	Consonant Sign Medial YA + WA
 + 	U+103C + U+103D	Consonant Sign Medial RA + WA
 + 	U+103B + U+103E	Consonant Sign Medial YA + HA

ꠌ + ꠌ	U+103C + U+103E	Consonant Sign Medial RA + HA
ꠌ + ꠌ	U+103D + U+ 103E	Consonant Sign Medial WA + HA
ꠌ + ꠌ + ꠌ	U+103B + U+103D +U+ 103E	Consonant Sign Medial YA+WA + HA
ꠌ + ꠌ + ꠌ	U+103C + U+103D + U+103E	Consonant Sign Medial YA+WA + HA

4.4.2 FSA for checking code sequence in Myanmar Domain Names

Languages can be presented as entities generated by a computation. This is a very common situation in formal language theory: many language families are associated with computing machinery that generates them. The simplest computation device is Finite State Automata (FSA) which can be thought of a finite set of states, connected by a finite number of transitions.

Finite state automata are efficient computational devices for generating regular languages. An equivalent view would be to regard them as recognizing devices. Further, most of the algorithms one would want to apply to finite-state automata take time proportional to the length of the word being processed, independently of the size of the automaton. In computational terminology, this is called *linear time complexity*, and is as good as things can get (Shuly Wintner 2002). Therefore, many NLP applications use FSA and we also apply FSA for code sequence checking of Myanmar domain name strings to mitigate combining mark order spoofing.

For Myanmar language domain names, it is also necessary to follow the standard procedures established by Unicode, ICANN and IDN working teams. And by doing this, most of the potential attacks can be reduced. For single-script spoofing and mixed-script spoofing, script-level specifications are necessary to define allowable and disallowable characters, identifying similar-looking characters and homoglyphs between Myanmar scripts and other scripts. Further, it is also necessary to develop variant tables and the results can be achieved through discussions between technical and linguistic experts.

For combining mark order spoofing, language-specific checking method is not proposed yet and thus finite state based approach is proposed here. Our method firstly analyses the orthographic rules for combining mark in Myanmar language and also Myanmar canonical ordering. Secondly, we develop FSA to check correct code sequence of input Myanmar characters according to Myanmar canonical order. Then, the final step, we suggest to incorporate our FSA in the validation step of IDN registration process.

Finite state diagram for generalized Myanmar domain name is expressed as follows. In the following FSA, state 1 is the initial state and state 6 and 8 are the final state.

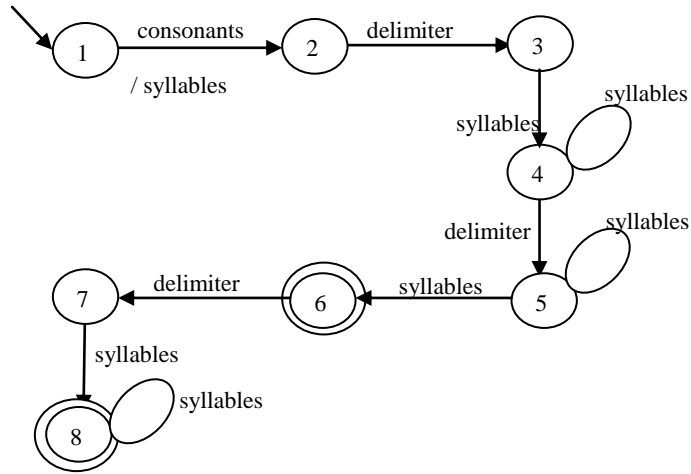


Figure 6. FSA for generalized Myanmar domain names

To check the coded sequence for syllables, we propose another FSA for correct combination of sub-syllabic elements within a syllable based on the regular expression

$$X = C M^* V^* (CK)^+ D^+$$

where X = syllable

C = consonant

M = medial consonant/dependent consonant sign

V = dependent vowel

K = asat or vowel killer

D = tone

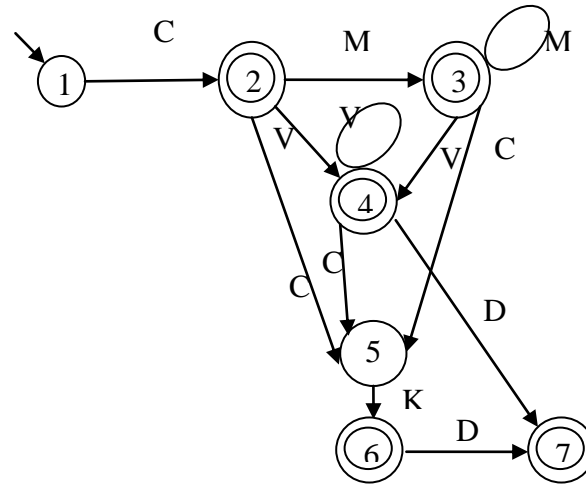


Figure 7. FSA for Myanmar syllable with sub-syllabic element in correct order

It is necessary to include FSA module in the registration process. There are two registration processes managed at the registrar and the registry. We propose to use our FSA module in the registration process at the registrar. We refer Verisign's IDN registration process in our example and a brief explanation about the process is given here. A registrant requests an IDN from a registrar that supports IDNs. The registrar converts the local language characters into a sequence of supported letters using an ASCII-compatible encoding (ACE). The registrar submits the ACE string to the Verisign® Shared Registration System (SRS) where it is validated. The IDN is added to the .com and .net TLD zone files and propagated across the Internet. This process is shown in the following figure.

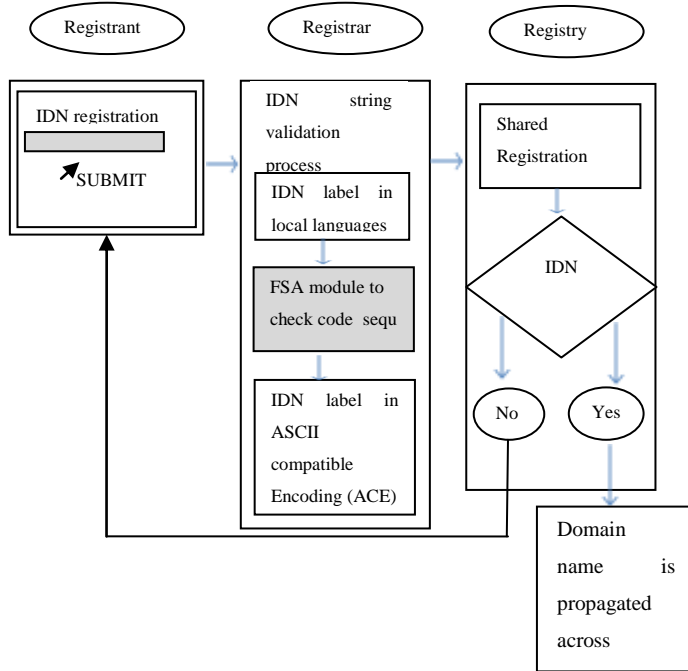


Figure 8. Registration process with proposed FSA module

4.4.3 Experimental Results and Discussion

Preliminary Experiment: To show proper working of our proposed FSA, we set up an preliminary experiment to check code sequence order in Myanmar domain names. Most of Myanmar domain names are in English names and so far, no Myanmar IDNs have been implemented. However, some romanized domain names are found, for example, <http://www.zawtika.com/> which means in Myanmar language www.ဇောတိက.com. And, also there are some domain names which are the combination of Myanmar words and English words, for example, <http://www.hlagabarfurniture.com/> in which address, “hlagabar” is Myanmar word “လှူဒါန” and “furniture” is of course English word. Further, there is no complete web directory for Myanmar yet. For the above difficulties, we could set up a preliminary experiment using 25 active commercial websites with romanized domain names in our current study (Please refer to Table 14).

All collected domain names are in romanized form and we convert them into equivalent Myanmar words. Converted words are either in regular syllable structure or irregular

writing practices such as consonant stacking, consonant repetition, and kinzi. Based on our tested results, it is found that all 24 Myanmar domain names are correctly recognized by our proposed FSA.

Discussion: IDN is a societal issue as well as technical challenge and it calls for great care and caution in supporting local languages and scripts in domain names. IDN based phishing attacks are surveyed by Anti-Phishing Working group (APWG) as follows. According to the APWG IDN Phishing Report 1H2009, from January 1, 2007 to June 30, 2009 only 85 IDNs were used for phishing. The majority were .HK domain names apparently used by the Rock Phish gang early in 2008. Again in 2H2010 survey, it is sated that since January 2007, only one true homograph attack was found.

According to the global phishing survey (2011) by APWG, only 10 of the 42,624 domain names they studied were IDNs and only one was a homograph attack. In survey 2H2012, they stated that since January 2007, they have found only five homographic phishing attacks, and none since 2011. In July 2012, there were two interesting attacks. They were not homographic attacks, but were malicious IDN registrations (APWG 2013).

Though a considerable amount of attacks has not been found yet for IDNs, we should prepare safety measures for respective scripts in advance as the number of IDNs are growing obviously and the attacks on IDNs are the threats which we have to face in near future.

Some of the problems of visual spoofing can be best handled on the user agent side, i.e, browsers, emailers, and other programs that display and process URLs. Thus, some works have been documented about implementation of anti-spoofing functions as a browser plug-in. In (Krammar 2006), the authors explores the various types of address spoofing attacks focusing on IDN, and presents a novel client-side web browser plug-in Quero to protect the user against visually undistinguishable address manipulations. Likewise, a client-side solution in the form of Firefox plug-in is developed (Al Helou and et al 2010).

Contributions: For some Asian languages, domain names implementations in their languages have already documented by stating language specific characteristics, for example, Sinhala (Wijayawardhana and et al 2008) and Urdu (Sarmad and Durrani 2006). However, for Myanmar, any kind of attempt for such kind of work has not been documented yet so far. Further, Myanmar use different combining marks and it may bring an additional challenge to implementation of IDNs. And, there has not been provided any mitigating method for combining mark order spoofing which has high potential to happen in Asian scripts. Thus, this could be a risk to Asian script IDN based phishing so far no visible mal use of Asian IDN is reported.

Therefore, our study will cover this gap by proposing normalized code sequence for Myanmar strings. And FSA based code sequences checking module is developed so that cyber threats using IDNs could be reached to a tolerable level and thus the Internet community would be safe. Secondly, it is reported that most of the syllabic writing systems can be described by using finite state automata (FSA) and we expect other scripts derived from Indic script can be checked by using our proposed method to prevent malicious registration of IDN labels.

Our current work is a preliminary stage and it requires collaboration between language authorities and technical experts for completion. It is expected that this work is just a proposal for Myanmar IDN work which has not been initiated yet.

Table 19. Romanized URLs for Myanmar

No.	URLs in Romanization	URLs in Myanmar
1.	http://www.zawtika.com/	ဇောတိက (example : www.ဇောတိက.com)
2.	http://www.aungyadana.com/	အောင်ရတနာ
3.	http://www.mingaladon.com	မင်္ဂလာဒုံ
5.	http://www.kaytumadi.com/	ကေတုမတီ
6.	http://www.seinpan.com/	စိန်ပန်း
7.	http://www.annawar.com/	အဏ္ဏဝါ
8.	http://www.kyaikhtesaung.org/	ကျိုက်ထီးဆောင်း
9.	http://www.mingalarbar.com/	မင်္ဂလာပါ
10.	http://www.shwete.com/	ရွှေတီ
11.	http://kabakyaw.com/	ကမ္ဘာကျော်
12.	http://ziwaka.com/	ဇီဝက
13.	http://www.wincherry.com	ဝင်းချယ်ရီ
14.	http://www.hlamyintzu.com/	လှမဇူ
15.	http://www.mahatoe.com/	မဟာတိုး
16.	http://www.mauriya.com/	မောရိယ
17.	http://yeeshin.com/	ရီရှင်း
18.	http://www.kumudara-bagan.com/	ကုမုဒြာပုံ
19.	http://www.chinsu.com/	ချင်စု
20.	http://cherryoo.com/	ချယ်ရီဦး
21.	http://www.heinsi.com/	ဟိန်းစည်
22.	http://www.kyarnyopann.com/	ကြာညိုပန်း
23.	http://thamadamonkrobe.com/	သမ္မတသင်္ကန်း
24.	http://www.minthila.com/	မင်းသီလ

5 FORMAL DEFINITION OF LEXICOGRAPHIC ORDER AND SORTING ALGORITHM

5.1 Introduction

Collation is the general term for the process and function of determining the sorting order of strings of characters. Among many tasks in the localization, collation is the fundamental process which is necessary to be developed as a first priority. There are more than 3000 languages spoken in Asia but collation order remains undefined for most of these languages. This greatly inhibits their processing online. Some of these languages even have an official status and/or significant speaking populations. However, lack of awareness, linguistic research and resources are some of the factors delaying the process. Working on many of these languages to define their sorting sequences not only presents great commercial opportunities but also large socio-economic benefits, as many of these languages are spoken by developing populations (Sarmad, Sana and Afifah xxxx).

Collation comprises two important steps namely defining collation order or lexicographic order and implementation of collation/sorting algorithm.

Defining the standard collation order of a language is difficult because it is necessary to solve language specific issues such as the existence of different forms of the same word, and so on. Further, some differences could exist among scholars, dictionaries and government/private organizations with respect to standard lexicographic order.

Collation implementations must deal with the complex linguistic conventions for ordering text in specific languages, and provide for common customizations based on user preferences. Furthermore, algorithms that allow for good performance are crucial for any collation mechanisms to be accepted in the marketplace. Regarding implementation of lexicographic sorting, many people expect the characters in their language to be in the "correct" order in the UCS/Unicode code charts. Because collation varies by language and not just by script, it is not possible to arrange the encoding for characters so that simple

binary string comparison produces the desired collation order for all languages. Because multi-level sorting is a requirement, it is not even possible to arrange the encoding for characters so that simple binary string comparison produces the desired collation order for any particular language (Mark and Ken 2013).

This study applies set theory based formal model to describe Myanmar lexicographic order and then deduce an algorithm based on the formal expression. The order in a given character set and among different character sets encoded in UCS/Unicode can be presented by using formal notations. By proposing this approach, we hope that other Asian languages having complex lexicographic order could be described in a systematic and unambiguous way of expression. Our proposed algorithm works directly only on the sub-syllabic element present in the whole word and without a preprocessing step known as syllabification. The algorithm is based on detail analysis of Myanmar orthographic rules. Then collation element table with 92 elements are used as collation weights in our algorithm. The correct results are achieved by simply giving the collation order on the types of sub-syllabic elements in the given input UCS/Unicode strings.

5.2 Problem Statement

Myanmar script is encoded in UCS/Unicode and code points of characters cannot be used as sorting key obviously. Different algorithms have been reported for Myanmar in (Yuzana and Tun 2008a) (Yuzana and Tun 2008b) (Martin 2012) (Tin and Mikami 2011) and all these works reported syllabification process is compulsory. By analyzing the previous algorithms on Myanmar lexicographic sorting, the following points are noticed:

- (1) Collation works on 5 collation elements within a syllable (i.e, consonants, vowels, medials, final consonants and tone marks)
- (2) Reordering process for vowel and finals is required
- (3) There are syllables which do not have 5 sub-syllabic elements because all are optional except consonant. But collation weights are given for these 4 optional elements by using empty elements

- (4) Fixed length key is generated for every syllable in a given word
- (5) Final sort key for a given word is generated by concatenation of sub-sort keys for each syllable if UCA is applied.

Different algorithms have been implemented for Myanmar lexicographic ordering and this could make non-native speakers feel difficult to understand Myanmar sorting order. Multilevel sort on different number of character groups brings additional complexity to Myanmar sorting algorithm and thus simple, unambiguous and formal model of lexicographic order is necessary. Furthermore, like other Asian languages, Myanmar language collation needs pre-processing tasks such as reordering, contraction and syllabification. For these reasons, an algorithm which gives better performance by removing pre-processing steps as much as possible becomes a requirement in Myanmar lexicographic sorting.

5.3 Formal Description of Order

In this section, formal definitions to several key concepts relating to the subject of this paper are given and lexicographic orders of English, Japanese are given to demonstrate the usefulness of this approach.

Definition 1 – Partial Ordering

A partially ordered set (P, \leq) , POSET for short, is a set P equipped with a binary relation \leq , called the ordering relation, such that for all x, y, z in P we have;

(Reflexibility) $x \leq x$

(Transitivity) if $x \leq y$ and $y \leq z$ then $x \leq z$

(Antisymmetry) if $x \leq y$ and $y \leq x$ then $x = y$.

Example: $R = \{(x, y): x \text{ divides } y\}$ is a partial order relation on the set N of natural numbers.

Definition 2 – Complete Ordering

A complete ordered set (C, \leq) , COSET in short, is a set C equipped with a binary relation \leq such that for any x, y, z in C we have;

(Transitivity) if $x \leq y$ and $y \leq z$ then $x \leq z$

(Antisymmetry) if $x \leq y$ and $y \leq x$ then $x = y$

(Totality) $x \leq y$ or $y \leq x$

Example: $R = \{(x,y): x \text{ less than or equal to } y\}$ is a complete order relation on the set N of natural numbers.

COSET satisfies all properties of POSET. In order to develop a sorting algorithm of a given language, COSET should be defined because totality is crucial to complete sorting.

Robert and et.al defines lexicographic order as follows in his book (Robert and et.al 1987).

Let X be a finite alphabet, and X^* be the set of all finite-length strings over X . We can impose various partial ordering on X^* . Suppose $X = \{a_1, a_2, \dots, a_k\}$. We obtain lexicographic order \leq_{lex} on X^* by first imposing an ordering on the symbols of X , say $a_1 \leq_{\text{lex}} a_2 \leq_{\text{lex}} \dots \leq_{\text{lex}} a_k$. Then for all $x = x_1..x_m$ and $y = y_1..y_n$, where $m, n \geq 0$ and $x_i, y_j \in X$, we set $x \leq_{\text{lex}} y$ if and only if:

(1) $x_i = y_i$ for $i = 1, \dots, m$ and $m \leq n$; or

(2) there is $j \geq 1$ such that $x_i = y_i$ for $i = 1, \dots, j-1$ and $x_j <_{\text{lex}} y_j$.

If $X = \{a, b, c, \dots, z\}$, the Roman alphabet in its usual order, then the lexicographic order on X^* is the familiar "alphabetical order" used in dictionaries and telephone directories.

Definition 3 - Union of Orders: Let (X, \leq_X) and (Y, \leq_Y) be POSETs, and Z be the union of X and Y , then a binary relation \leq_Z on Z is defined:

$u \leq_Z v$, where $u, v \in Z (=X \cup Y)$, if and only if

(1) $u, v \in X$ and $u \leq_X v$; or

(2) $u, v \in Y$ and $u \leq_Y v$; or

(3) $u \in X$ and $v \in Y$

Note that the operation is not *reversible* and \oplus symbol is used to represent union of order in our definition. For example, if you define union of Latin alphabet complete order and Greek alphabet complete order, $(L, \leq_L) \oplus (G, \leq_G)$ gives an order over the union of L and G, then Latin letters will be sorted in front of all Greek letters.

Note: Different terminologies for the above definition can be found in the literature. The terminologies, “The cardinal sum” or “Disjoint Union” is used by (Carpineto and Romano 2004).

Definition 4 - Product of Orders: Let (X, \leq_X) and (Y, \leq_Y) be COSETs, then a product of them is defined on the set of strings made of elements of X and Y, or $Z = X \times Y$, or $Z = \{xy, x \in X, y \in Y\}$. In other words, $(Z, \leq_Z) = (X, \leq_X) \otimes (Y, \leq_Y)$. Let $u_i \in X, v_i \in Y$ and X, Y : COSET, then $u_1 v_1 \leq u_2 v_2$ if and only if either of the following conditions is satisfied:

$u_1 = u_2$ and $v_1 \leq v_2$ or

$u_1 = u_2$ and $v_1 = \phi$ (where ϕ means null) or

$u_1 \leq u_2$

Note that the operation is not *reversible* and \otimes symbol is used to represent product of order throughout this paper. Further, since this relation satisfies associative property, we can write either $((A, \leq_A) \otimes (B, \leq_B)) \otimes (C, \leq_C)$, or $(A, \leq_A) \otimes ((B, \leq_B) \otimes (C, \leq_C))$. When the same orders are multiplied m-times, we can write it in the form $(A, \leq_A)^m$. Once union and

product are introduced, the lexicographic order given by Robert could be rewritten in more simple manner.

Note: The different terminologies, “The direct product of two disjoint ordered sets” for the above definition is used by (Carpineto and Romano 2004). Also, the terminology “ Direct product ordered set” is used by (Iwanami 2006).

Definition 5 - Lexicographic Order in Set Order form:

Suppose N be the maximum length of strings in X^* and X be extended to include null character ϕ ($\phi \leq x_i$ for any $x_i \in X$). And we set x' be the lengthened x, by adding trailing ϕ to make it fixed length string, i.e. $x' = x_1 \dots x_n \phi \dots \phi$. By adding null character ϕ to the set X, all strings in the set X^* will be of fixed length N. Then, the lexicographic order of X^* with strings of fixed length N can be obtained by using product of order of the set X as follows.

$$\begin{aligned} (X^*, \leq_{\text{LEX}}) &= (X^N, \leq_{\text{LEX}}) \\ &= (X, \leq) \otimes (X, \leq) \otimes \dots \otimes (X, \leq) \text{ (N times)} \\ &= (X, \leq)^N \end{aligned}$$

As shown in this case, COSET form representation gives a simple and formal ways of definition of order. We give another example to demonstrate its usefulness.

In a multi-level sort, the primary comparison is made using the strings, which represent only primary differences. Suppose $Y^* = \{y\}$ be a set of the primary difference strings of $x \in X^*$, and \leq_Y be an order defined on Y, the target order of the primary comparison is $(Y, \leq_Y)^N$.

In the secondary comparison, only the secondary differences of the strings are compared. Let $Z^* = \{z\}$ be a set of strings, which shows the secondary differences of $x \in X^*$, then the

target order of the secondary comparison is given by $(Z, \leq_Z)^N$, where \leq_Z is an order defined on Z . Here, we can get the final, lexicographic order as

$$(Y, \leq_Y)^N \otimes (Z, \leq_Z)^N.$$

Example 1: English Lexicographic Order

Lexicographic order of English is used as an example to show multilevel sorting. Difference in case is treated as the secondary difference in the normal lexicographic order of English. Suppose $X = \{aa, aA, ab, Ab\}$, then the primary comparison is made case-insensitively, and the secondary comparison is made with $Z = \{ss, sc, ss, cs\}$, where "s" and "c" represent small and capital letter respectively. The final result of sort is shown in Table below. In this case, the combined target order is given by;

$$(X^1, \leq_1)^N \otimes (X^2, \leq_2)^N$$

where $X^1 = \{\phi, a, b, \dots, z; \phi <_1 a <_1 \dots <_1 z\}$ and $X^2 = \{s, c; s <_2 c\}$.

Table 20. Comparison of Multi-level Sort and Simple Sort

Multi-level Sort			Simple Sort
Primary Comparison	Secondary Comparison	final result	($a < b < A < B$)
Aa	ss	aa	aa
Aa	sc	aA	ab
Ab	ss	Ab	aA
Ab	cs	Ab	Ab

Example 2: Japanese Lexicographic Order

The typical sorting practice found in modern Japanese dictionaries is also multi level sorting. Although there is no single authoritative definition for the lexicographic order of Japanese, one of the most popularly used Japanese dictionary, KOJIEN, gives a three-level sorting order (Niimura 2011).

At the primary comparison, simple letter ordering is made like a case of English. At the secondary comparison⁴, variety of presentations are sorted into the sequence; small letter form, base letter form, letter with a diacritical sign daku-on (゜) and letter with another diacritical sign handaku-on (゚). At the third level comparison, imported words come in front of vernacular words. It means that a word spelt with katakana comes in front of hiragana. As a result, the final lexicographic order of KOJIEN dictionary is given by the product of three COSETs, (S, \leq_S) , (D, \leq_D) , (H, \leq_H) .

(S, \leq_S) represents a simple syllable order defined on the Japanese basic syllable set $S = \{a, i, u, e, o, ka, ki, ku, ke, ko, \dots, wa, wo, n; a \leq_S i \leq_S u \leq_S e \dots \leq_S n\}$, also known as Sei-on. (D, \leq_D) represents *yo-on* (y) / *soku-on* (t) / *sei-on* (φ) / *daku-on* (b) / *handaku-on* (p) order, $D = \{y, t, \phi, b, p; y \leq_D t \leq_D \phi \leq_D b \leq_D p\}$. Katakana / hiragana order is represented by (H, \leq_H) where $H = \{h, k; k \leq_H h\}$.

Any element of Kana character set K is identified in three-dimensional space $S \times D \times H$. For example, the first letter of Hiragana あ is identified by (a, ϕ, h) ; the 6th letter of Katakana カ is identified by (ka, ϕ, k) ; a Yo-on, small Hiragana letter や is identified by (ya, y, h) ; a Soku-on, small Katakana letter ツ is identified by (tu, t, k) ; a Daku-on Hiragana letter ば is identified by (ha, b, h) ; a Handaku-on Katakana letter ぱ is given by (ha, p, k) ; and so on. Then the lexicographic order on Kana strings K^* , (K^*, \leq_{LEX}) , is defined as:

$$(K^*, \leq_{LEX}) = (S, \leq_S)^N \otimes (D, \leq_D)^N \otimes (H, \leq_H)^N$$

⁴ It is not the main subject of this paper to give precise details of Japanese case, but let us introduce briefly. Firstly, while there is no *case* in Japanese writing in its exact sense of meaning, several letters take small form such as (や/ゃ, づ/ぢ, ょ/ょ, っ/っ, etc.). These small form letters are no more independent syllable, but work as sub-syllabic elements to modify the phonetic value of the syllables. Yo-on has an effect of yotization, introduction of <y> sound, Soku-on has an effect of <tsu>. Secondly, Japanese writing has two diacritical signs, daku-on sign (゜) and handaku-on sign (゚). When daku-on sign is attached to the right shoulder of specific syllable letters, it has an effect of changing /k/ to /g/ or /h/ to /b/. And when handaku-on sign is attached to the right shoulder of specific letters, it has an effect of changing /h/ to /p/.

Table 21. Example of Japanese Lexicographic Sorting

Kana	Kanji	Primary	Secondary	Tertiary
K*		$(S, \leq_S)^N$	$(D, \leq_D)^N$	$(H, \leq_H)^N$
さっか	作家	sa/tu/ka	$\varphi/t/\varphi$	h/h/h
ざっか	雑貨	sa/tu/ka	b/t/ φ	h/h/h
サッカー	サッカー	sa/tu/ka/a	$\varphi/t/\varphi/\varphi$	k/k/k/k
さっき	殺気	sa/tu/ki	$\varphi/t/\varphi$	h/h/h
さつき	五月	sa/tu/ki	$\varphi/\varphi/\varphi$	h/h/h
ざっき	雑記	sa/tu/ki	b/t/ φ	h/h/h
ざつき	座付き	sa/tu/ki	b/ φ/φ	h/h/h
ざつぎ	雑技	sa/tu/ki	b/ φ/b	h/h/h

Example 3: German Lexicographic Order

Diacritical marks and accents are treated as the secondary or tertiary difference in some languages. In the German standard lexicographic order, as given in the German Standard DIN 50075, the umlaut sign is neglected in the primary and secondary comparison, and is considered only in the tertiary comparison. Letter ß, which characterizes German alphabet uniqueness, should be included in X1 at its proper position. As the secondary comparison is made with case, the combined target order is defined as;

$$(X^1, \leq_1)^N \otimes (X^2, \leq_2)^N \otimes (X^3, \leq_3)^N$$

⁵ DIN 5007 retrieved from <http://www.cabeweb.de/html/din5007.htm>

where $X^1 = \{\varnothing, a, b, \dots, r, s, \beta, t, \dots, z; \varnothing <_1 a <_1 \dots <_1 z\}$

$$X^2 = \{s, c; s <_2 c\}$$

$$X^3 = \{\varnothing, u; \varnothing <_3 u\}. \quad \text{"u" means umlaut sign here.}$$

Example 4: Swedish Lexicographic Order

The Swedish standard lexicographic order does not neglect diacritical marks in the primary comparison. An alphabet "ä" is considered to come after "z", and the secondary comparison is made on case. So the Swedish lexicographic order is defined as follows (Hufflen 2007).

$$(X^1, \leq_1)^N \otimes (X^2, \leq_2)^N$$

where $X^1 = \{\varnothing, a, b, \dots, z, \ddot{a}; \varnothing <_1 a <_1 \dots <_1 \ddot{a}\}$

$$X^2 = \{s, c; s <_2 c\}.$$

Contrary to the above cases, multiplicity of comparison takes a bit different form in the case of languages which use syllabic writing systems, including Myanmar. There is no case in most syllabic writing systems including Myanmar. The multiplicity appears in syllabic order itself. In syllabic writing systems, syllables are composed of several sub-syllabic elements, such as consonants, vowels, tone marks, ending consonants, and so on. And the syllable order is given by products of orders of these sub-syllabic elements which will be discussed in detail in the following section.

5.4 Myanmar Lexicographic Sorting Algorithm

5.4.1 Primary Lexicographic Order Elements

Myanmar script is encoded in UCS/Unicode. The categories of characters defined in UCS/Unicode are shown in the first column of the Table 17. Every character shown in the Table 17 is non-ignorable character for sorting. But not all groups are COSET. It is necessary to add several elements and redefine orders to make these sets work as COSETs. For this purpose, we introduce five functional groups, consonants, vowels, final consonants, medials, and tone marks. These are slightly modified versions of character groups and are shown in the second column of the Table 17. On these groups, complete orders can be defined and these complete orders work independently at Myanmar sorting process.

Table 22. Categories of Myanmar Characters

UCS/Unicode Character Category with number of elements	Proposed Functional Groups with number of elements	Remark
Consonant letters (33)	Consonants C (34)	Myanmar vowel letter A (U+1021) is added
Dependent vowel signs (8) Independent vowel letters (8)	Vowels V (11)	3 combinations of vowel signs are added
Various Signs (5)	Final consonants F (34)	Each member of F is a combination of consonant C and Myanmar sign Asat K and F is isomorphic with C
	Tone marks D (2)	
Dependent consonant signs(4)	Medials M (11)	7 combinations of medial signs are added
Various Signs (4) and Myanmar Consonant Great SA (1)	ဉ်, ြ်, င, ျ, သ	They are not covered in this work.

Complete Consonant Order

Let C be the set of consonants composed of 33 consonant letters and one vowel letter အ, i.e,

$C = \{က, ခ, ဂ, ဃ, င, ဇ, ဈ, ည, ဋ, ဌ, ဍ, ဎ, ပ, ဏ, တ, ထ, ဒ, ဓ, န, ဖ, ဖ, ဗ, ဘ, မ, ယ, ရ, လ, ဝ, သ, ဟ, ဉ, အ\}$. Then C becomes a COSET if normal consonant

order is given for all consonants and vowel letter အ is added at the end of it. The order defined on C is written as (C, \leq_C) in this paper. The reason why a vowel letter is added is explained later.

Complete Vowels Order

The set of Myanmar dependent vowel signs in UCS/Unicode contains 7 elements {◌့, ◌ိ, ◌ီ, ◌ု, ◌ူ, ◌ေ, ◌ဲ}. It is not an exhaustive set of vowels. As shown in Table 4, three vowel signs take combined glyph forms composed of vowel signs and Myanmar sign *Asat*. So, three vowels are added and the vowel order is defined on extended vowel set V composed of total eleven elements {◌့, ◌ိ, ◌ီ, ◌ု, ◌ူ, ◌ေ, ◌ဲ, ◌ော, ◌ော်, ◌ံ, ◌ိ့ ◌ု}. Elements in the parenthesis are arranged in the order of the set. The vowel order COSET is written as (V, \leq_V) in this paper.

Table 23. List of Added Vowels

Glyph	Code sequence representation	Description
◌ေ + ◌့	1031+102C	Vowel sign E + AA
◌ေ + ◌့ + ◌ံ	1031+102C+103A	Vowel sign E + AA + <i>Asat</i>
◌ိ + ◌ု	102D + 102F	Vowel sign I + UU

Let us recall the structure of a Myanmar syllable which is introduced in earlier section, $S := C(M)(V)(F)(D) \mid I(F)$. Here a set I is an independent vowel set and independent vowel can represent a syllable. The set of Myanmar independent vowels in UCS/Unicode contains 8 elements {အ, ဣ, ဤ, ဥ, ဦ, ဧ, ဩ, ဪ} and can be introduced as one of functional groups. For sorting purpose, however, the independent vowel is replaced by the combination of Myanmar letter A (U+1031) and respective dependent vowel sign, namely {အ, အိ, အီ, အု, အူ, အေ, အော, အော်}. A point to note is Myanmar letter A (U+1021), “အ” can be counted as vowel as well as consonant but, in sorting process, it will be treated as consonant. Thus, the sorting process can be described by only 5 sub-syllabic functional

elements. The syllable structure formula is then written in simpler manner, $S := C (M) (F) (V) (D)$.

Complete Medials Order

There are four basic medial signs {ချ, ငြ, ဝ, ျ}, which combine and produce additional seven medial signs as shown in Table 5. After normalization of these 7 medials, the set of medials have 11 elements with a complete order and which is described as (M, \leq_M) . The complete list of Myanmar medials are {ချ, ငြ, ဝ, ျ, ချဝ, ငြဝ, ချၻ, ငြၻ, ဝၻ, ချဝၻ, ငြဝၻ}.

Table 24. List of Added Medials

Glyph	Code sequence representation	Description
ချ + ဝ	103B + 103D	Consonant Sign Medial YA + WA
ငြ + ဝ	103C + 103D	Consonant Sign Medial RA + WA
ချ + ျ	103B + 103E	Consonant Sign Medial YA + HA
ငြ + ျ	103C + 103E	Consonant Sign Medial RA + HA
ဝ + ျ	103D + 103E	Consonant Sign Medial WA + HA
ချ + ဝ + ျ	103B + 103D + 103E	Consonant Sign Medial YA+WA + HA
ငြ + ဝ + ျ	103C + 103D + 103E	Consonant Sign Medial YA+WA + HA

Complete Finals Order

When there is a consonant at the end of a syllable, it carries a visible mark called Myanmar sign *Asat* (ြ) to indicate that the inherent vowel is killed. It usually comes with a consonant letter but sometimes comes with an independent vowel letter. The set of finals F contains elements which are the combinations of each consonant and *Asat*, $F=\{\text{က်, ခ်, ဂ်, ဃ်, င်, ဖ်, ဆ်, ဇ်, ဈ်, ည်, ဋ်, ဌ်, ဍ်, ဎ်, ဏ်, တ်, ထ်, ဒ်, ဓ်, န်, ပ်, မ်, ဖ်, ဘ်, မ်, ယ်, ရ်, လ်, ဝ်, သ်, ဟ်, ဠ်, အ်}\}$ and described as (F, \leq_F) .

Complete Tones Order

Tone marks alter the vowel sounds of accompanying consonants and they are used to indicate tone level. There are 2 diacritics or tone marks {း, ့} in Myanmar script and their order is (D, \leq_D) .

5.4.2 Conventional Multi-level Sorting of Syllables

After defining primary COSETs for different functional groups, syllable order can be defined using these primary orders. In the conventional sorting processes of Myanmar language, firstly target strings are syllabified and a multi-level string is generated. Let $s^*=s_1s_2\dots s_n$ be the target string of comparison. The generated string takes the form like $c_1m_1f_1v_1d_1c_2m_2f_2v_2d_2 \dots c_nm_nf_nv_nd_n$. When some functional elements are missing in the target syllable, null filler ϕ is inserted in place of missing element. If vowel and tone elements of the first syllable are missing, then $c_1m_1f_1\phi\phi$ is generated for the first syllable. If all elements other than consonant are missing in the n-th syllable, then $c_n\phi\phi\phi\phi$ is generated for the n-th syllable. At the secondary step, two generated strings are compared syllable by syllable, and the comparison of a syllable is made at first by consonant level, then by medial level, then by final level, then by vowel level, and lastly by tone level. It means that the syllable level order is defined by

$$(C, \leq_C) \otimes (M, \leq_M) \otimes (F, \leq_F) \otimes (V, \leq_V) \otimes (D, \leq_D),$$

and the final complete order applied for this comparison is written as

$$((C, \leq_C) \otimes (M, \leq_M) \otimes (F, \leq_F) \otimes (V, \leq_V) \otimes (D, \leq_D))^N.$$

This gives the formal description of multi-level, syllable-wise sorting for Myanmar words. The multiplicity of sorting process appears here in different form from examples given in the previous section of the paper. Here in the case of Myanmar, the multiple comparisons are made at each syllable, and then the comparison process moves to the next syllable and continues until the last syllable. But in English, German and Japanese examples the primary comparison continues until the last characters, and then the comparison process moves to the next level and continues until the last level.

5.4.3 Sub-syllable level Sorting

When Myanmar sorting process is analyzed at sub-syllable level, another definition of the order should be devised. Again let us recall Myanmar syllable structure formula $S := C [M] [F] [V] [D]$. From this expression, theoretically, the following 16 combinations can be created; C, CD, CV, CVD, CF, CFD, CFV, CFVD, CM, CMD, CMV, CMVD, CMF, CMFD, CMFV, CMFVD. Among the resultant combinations, two combinations, CD (consonant and tone mark) and CMD (consonant, medial, and tone), do not exist in Myanmar language and thus total number of possible sub-syllabic element combinations becomes 14. Monosyllabic order in Myanmar language is given in (Myanmar Orthography Dictionary 2006). It is shown in the table below.

Table 25. Monosyllable order defined by the Myanmar Orthography Dictionary

Order	Syllable Composition	Order	Syllable Composition
S ₁	(C)	S ₈	(C, M)
S ₂	(C, V)	S ₉	(C, M, V)
S ₃	(C, V, D)	S ₁₀	(C, M, V, D)
S ₄	(C, F)	S ₁₁	(C, M, F)
S ₅	(C, F, D)	S ₁₂	(C, M, F, D)
S ₆	(C, F, V)	S ₁₃	(C, M, F, V)
S ₇	(C, F, V, D)	S ₁₄	(C, M, F, V, D)

When comparison should be made at sub-syllabic element level, we have to compare elements which belong to different functional groups, on where no complete order is defined. Based on Table 20, multi-syllable order is produced. After the last entry of specific ending patterns, always longer entry word comes. In Table 21, such longer words are denoted by adding C- symbol at the end of syllable, because following syllable always begins with a consonant.

Table 26. Analysis of multi-syllable order

1	2	3	4	5	6	7	8	9	10	11	12
Syllable	Sub-syllable	u	v	u	v	u	v	u	v	u	v
S ₁	C	C	null								
S ₁ S-	CC-	C	C								
S ₂	CV	C	V					CV	Null		
S ₂ S-	CVC-	C	V					CV	C		
S ₃	CVD	C	V	CVD	null			CV	D		
S ₃ S-	CVDC-	C	V	CVD	C			CV	D		
S ₄	CF	C	F			CF	null				
S ₄ S-	CFC-	C	F			CF	C				
S ₅	CFD	C	F	CFD	null	CF	D				
S ₅ S-	CFDC-	C	F	CFD	C	CF	D				
S ₆	CFV	C	F			CF	V	CFV	null		
S ₆ S-	CFVC-	C	F			CF	V	CFV	C		
S ₇	CFVD	C	F	CFVD	null	CF	V	CFV	D		
S ₇ S-	CFVDC-	C	F	CFVD	C	CF	V	CFV	D		
S ₈	CM	C	M							CM	null
S ₈ S-	CMC-	C	M							CM	C

after u (V): null < C < D (See column 10 in Table 26)

after u (M): null < C < V < F (See column 12 in Table 26)

This result means that if we define order between functional groups as $C < D < V < F < M$, which satisfies all above requirements, then simple sub-syllabic level comparison will produce a correct Myanmar lexicographic order over strings composed of sub-syllabic elements. In the formal description form, a complete order defined on a set of combined sub-syllabic elements are given by COSET

$$(\mathbf{C}, \leq_{\mathbf{C}}) \oplus (\mathbf{D}, \leq_{\mathbf{D}}) \oplus (\mathbf{V}, \leq_{\mathbf{V}}) \oplus (\mathbf{F}, \leq_{\mathbf{F}}) \oplus (\mathbf{M}, \leq_{\mathbf{M}}),$$

and the sorting of multi-syllabic strings can be done by COSET,

$$((C, \leq_C) \oplus (D, \leq_D) \oplus (V, \leq_V) \oplus (F, \leq_F) \oplus (M, \leq_M))^N \quad (N \text{ is the maximum syllable length})$$

Here we notice that all functional group elements are arranged in a single strain of order now. It is not guaranteed to have this kind of single strain of order between different functional group elements. It happens in case of Myanmar because conditions in Table 26 just allow it. Also note that the order of functional groups in the above formula is different from the pronunciation order in Myanmar syllable $S := C [M] [V] [F] [D]$.

5.5 Demonstration and Discussion

In this section, a sub-syllable level approach for Myanmar lexicographic sorting is presented. The algorithm works directly on input UCS/Unicode texts and syllabification process is not required. But preprocessing process of conversion from UCS/Unicode characters to the functional group elements is required. Further, generation of sort key for the whole input word is not necessary like other UCA based algorithms. The steps in our algorithm are as follows.

- Step 1 Input UCS/Unicode String
- Step 2 Preprocessing: Convert UCS/Unicode character (sequence) to functional group elements like C1, M1, F1, V1, D1. And put # to show the end of each input word.
- Step 3 Reordering: replace two sub-syllabic elements, vowels and finals (combination of consonant and *Asat*), when necessary
- Step 4 Sort the elements using the order

$$((C, \leq_C) \oplus (D, \leq_D) \oplus (V, \leq_V) \oplus (F, \leq_F) \oplus (M, \leq_M))^*$$

The collation element table with 92 elements are used in our algorithm, for instance, 34 elements for consonant order, 11 elements for medials order, 34 for final consonants order, 11 for dependent vowels order and 2 elements for tone marks respectively. The demonstration of sorting process using multisyllabic words is shown below.

Step - 1 Input of UCS/Unicode strings

Correct Order	Words	Comments
1.	ကိုးရိုး	Word with 2 syllables
2.	ကုတ်ကလံ	Word with 3 syllables
3.	ကျပန်း	Word with 2 syllables
4.	ကျေးငှက်	Word with 2 syllables
5.	ကျော်ကြား	Word with 2 syllables

Step - 2 Preprocessing

Input	Words								
5.	ကျော်ကြား	C1	M1	V9	C1	M2	V1	D2	#
4.	ကျေးငှက်	C1	M1	V6	D2	C5	M4	F1	#
3.	ကျပန်း	C1	M1	C22	F21	D2	#		
2.	ကုတ်ကလံ	C1	V4	F1	C1	C26	V10	#	
1.	ကိုးရိုး	C1	V11	D2	C27	V11	D2	#	

Step 3 – Reordering of vowels and final consonants

Reordering	Words								
5.	ကျော်ကြား	C1	M1	V9	C1	M2	V1	D2	#
4.	ကျေးငှက်	C1	M1	V6	D2	C5	M4	F1	#

3.	ကျပ်နိုး	C1	M1	C22	F21	D2	#
2.	ကုတ်ကလံ	C1	F1	V4	C1	C26	V10 #
1.	ကိုးရိုး	C1	V11	D2	C27	V11	D2 #

Step 4 – Sort each column by using the collation order of C<D<V<F<M.

Result	Words	Collation Weights for sub-syllabic elements						
1.	ကိုးရိုး	C1	V11	D2	C27	V11	D2	#
2.	ကုတ်ကလံ	C1	F1	V4	C1	C26	V10	#
3.	ကျပ်နိုး	C1	M1	C22	F21	D2	#	
4.	ကျေးငှက်	C1	M1	V6	D2	C5	M4	F1 #
5.	ကျော်ငြား	C1	M1	V9	C1	M2	V1	D2 #

In our demonstrated example, the syllable structure in the input Myanmar words is in the form of S: = C [M] [V] [F] [D]. And collation weights are assigned to the sub-syllabic elements present in the given words. The authors have tested the lexicographic ordering algorithm with the random data collected from the Myanmar Orthography published by (Myanmar Language Commission 2006). It is found that our algorithm works correctly for all selected words except the irregular words.

5.6 Conclusions and future work

Lexicographic ordering is an essential standard requirement for information processing in any language. This paper explains some of the linguistics and orthographic factors applicable for defining collation order of Myanmar language. And, a formal description method for defining collation orders is presented and Myanmar lexicographic order is defined using order set notation. It is observed that different sub-syllabic element groups of Myanmar can be sorted in one linear order. Further, a sorting algorithm for Myanmar is developed based on the formal order and the algorithm works without syllabification process which is significant for sorting multi-syllabic words as compared with previous approaches.

Our proposed algorithm handles all multisyllabic words except a limited number of irregular words and a preliminary experiment is set up to check the proper working of our algorithm. In order to process irregular words, pre processing for code sequences are

necessary because the underlying code sequence to represent irregular words contains Myanmar sign Virama (U+1039). Since Virama does not convey any value for sorting of Myanmar language, it should be removed for sorting process. Again, Myanmar Sign Asat (U+103A) should be included in the code sequence explicitly. With such preprocessing, the code sequence becomes the form of $S = C [M] [V] [F] [D]$ and our proposed method becomes fully applicable. Such needs of pre-processing can be eliminated during a simple syllabification process. However, since the main purpose of this study is to develop sorting algorithm without syllabification process, including such process is out of our scope. Therefore, as a future study, the lexicographic order which can handle both regular as well as irregular words should be done.

In this work, only preliminary experiment is set up to demonstrate the proper working of our algorithm. Thus, to set up the experiments using the data collected from different resources covering all possible forms of Myanmar words will be our future work. Further, comparison of our proposed algorithm with other algorithms could be done to show accuracy and correctness of different algorithms on Myanmar lexicographic sorting.

6 CONCLUSIONS

There is little work on syllable structure in orthographic domain for Alphabetic languages using formal approaches. Lack of structural regularity of syllables in these languages gives favor to implementation of applications in pronunciation domain. In contrast, syllable structure information of Myanmar language could be given in both pronunciation and orthographic domains. Further, Myanmar syllable model in orthographic domain is well-defined and unambiguous. There are critical and basic applications where orthographic model of syllable structure is applied. Therefore, this research proposed implementation of fundamental language processing tasks using formal model of orthographic syllable structure.

We began this study with the following objectives:

- (1) To apply formal approaches without using language resources such as annotated corpus or lexicon
- (2) To represent syllable structure information in orthographic domain in formal grammar
- (3) To develop generic applications where orthographic syllable model is important
- (4) To propose formal description of lexicographic order of Myanmar language
- (5) To implement Myanmar sorting algorithm without syllabification process

We developed a theoretical framework for dealing with these objectives. In this framework, formal generative and recognition approaches are used to represent Myanmar language and then this framework is applied to real-world application areas: automatic syllabification, normalized code sequence checking and formal definition of lexicographic order.

As a formal generative method, regular grammar is used to describe Myanmar syllable structure. And two recognition approaches namely finite state automata and finite state transducers are used to implement applications because automata are the most important computational model for language processing. Moreover, set theory based formal

definition of Myanmar lexicographic order is proposed and sorting algorithm is implemented based on the proposed formal model.

The significant achievements on Myanmar language processing using formal model of orthographic syllable structure are stated as follows.

(1) Automatic Syllabification of Myanmar Texts using FST

Generally, finite state transducer (FST) for automatic syllabification is not new for alphabetic languages in pronunciation domain. But, automatic syllabification of Myanmar texts in orthographic domain using FST is the first known documented work. For instance,

- (1) syllabification is done on input texts directly, i.e, no conversion to phoneme strings.
- (2) the structure of a syllable is represented using formal grammar and no statistical information or weight is necessary to apply for correct syllabification

Formal grammar and automata applied in Myanmar syllabification gives unambiguous results and the computing process is simple as compared with rule-based or corpus-based method. Moreover, our proposed FST can syllabify both standard and *Irregular* words which has not been completed yet by previous methods.

(2) FSA based Normalized Code Sequence Checking

In this application area, finite state automata (FSAs) are used to check the order of combination of sub syllabic elements in a given string. To show the importance of code sequence order, we introduce a case about combining mark order spoofing in Myanmar IDNs (Internationalized domain Names). Combination of sub syllabic elements in a syllable is first described in regular expression (RE) notation. As FSA is the mathematical device used to implement regular expressions, the combining mark order of input strings can be validated on FSA. Though a preliminary experiment is set up to show the proper working of our method, it shows the importance of describing syllable structure in formal grammar.

This work is another area of application where orthographic syllable structure model is used and it is found that FSA approach works well on checking character code sequence for secure IDN strings.

(3) Formal Definition of Lexicographic Order and Sorting Algorithm

The term ‘formal methods’ is used to refer to any activities that rely on mathematical representations of software. The branch of mathematics used is discrete mathematics and the mathematical concepts are drawn from set theory, logic and algebra. According to the above definition of ‘formal methods’, we proposed a formal model of general lexicographic order and applied it to Myanmar language as an example.

We introduce some fundamental definitions such as partial order set (POSET), complete order set (COSET) and lexicographic order in traditional form. Then, we propose new definitions of union of orders, product of orders and lexicographic order in set order form. Then, multi level sorting order on four languages namely Japanese, English, Swedish and German are described and finally, formal definition of lexicographic order for Myanmar is proposed using primary lexicographic order elements. Further, Myanmar sorting process is analyzed at sub-syllable level and a sorting algorithm is implemented which gives correct lexicographic order over strings directly.

Formal way of describing lexicographic order is considered as an important achievement of this study and sorting algorithm with fewer preprocessing tasks, is a significant contribution to Myanmar language processing.

In conclusion, even though there are many areas of applications to be developed using formal methods in Myanmar language processing, this research can serve as a good ground for future formal language processing of Myanmar. As our future work, we explore more about application areas of Myanmar language using formal approach, for example, finite state transducer (FST) for Morphological Analysis of Myanmar language. Last but

not least, we believe that our study could be a significant contribution to natural language processing of Asian languages.

REFERENCES

Al Helou, Johnny, and Scott Tilley. (2010). Multilingual web sites: Internationalized Domain Name homograph attacks, 12th IEEE International Symposium on Web Systems Evolution (WSE), IEEE.

Antiphishing Working Group. Phishing Attack Trend Reports. Available at <http://www.antiphishing.org/resources/apwg-reports/> (Accessed May 2013)

Balaram Prasain. (2011). A Computational Analysis of Nepali Morphology: A Model for Natural Language Processing. Doctoral Disseration, Tribhuvan University.

Carpineto, C., & Romano, G. (2004). Concept data analysis: Theory and applications. John Wiley & Sons.

Chamila, L., Randil, P., Dulip, L. Herath and Ruwan W. (2012): A Computational Grammar of Sinhala. In the proceeding of 13th International Conference on Computational Linguistic and Intelligent Text Processing (CICLING), Mumbai, India.

Connie R. Adsett. (2008) : Automatic Syllabification in European Languages: A Comparison of Data-driven Methods, Master Thesis Dissertation, Dalhousie University, Halifax, Nova Scotia.

Department of IT, Ministry of Communication and IT, Government of India. (2009) : Internationalized Domain Names in Indian Languages, A Draft policy document: Policy

Framework and Implementation Plan. Available at <http://www.docstoc.com/docs/74240340/India--IDN--Policy> (Accessed June 2013)

Dinu, LIVIU P. (2006). "On the quantitative and formal aspects of the Romanian syllables." *Revue Roumaine de Linguistique*, LI (3-4), pp. 477-498.

Gillman, R., Unicode Demystified. (2003). A Practical Programmer`s Guide to the Encoding Standard, Addison-Wesley, Boston, USA.

Gosse Bouma, Ben Hermans. Syllabification of Middle Dutch. http://alfclul.clul.ul.pt/crpc/acrh2/ACRH-2_papers/Bouma-Hermans.pdf

Hannay, Peter, and Christopher Bolan. (2009) : Assessment of Internationalised Domain Name Homograph Attack Mitigation, In the Proceedings of Australian Information Security Management Conference, Perth, Australia.

Hassain, S. (2004): Finite State Morphological Analyzer for Urdu. Master Thesis, National University of Computer and Applied Science, Lahore, Pakistan.

Hussain, Sarmad, and Nadir Durrani. (2006). Urdu Domain Names. In Multitopic Conference, INMIC'06. IEEE, pp. 299-304.

Hla Hla Htay, Murphy Kavi Narayana. (2008). Myanmar Word Segmentation using Syllable level Longest Matching. In Proceedings of the 6 th Workshop on Asian Language Resources, January 11-12, Hyderabad, India.

Hufflen, Jean-Michel. (2007). Implementing Language-Dependent Lexicographic Orders in Scheme. Proceeding of the 8th Workshop on Scheme and Functional Programming.

ISO/IEC 14651 (2001). Information Technology – International String ordering and

Comparison – Method for Comparing character Strings and Description of Common Template Tailorable Ordering, ISO Geneva, Switzerland ([www. iso.ch](http://www.iso.ch)).

ISO/IEC 10646 (2012). Information technology - Universal Coded Character Set

Jarkko Kari. (2013). Automata and Formal Languages. Lecture Notes, University of Turku. Available at <http://users.utu.fi/jkari/automata/fullnotes.pdf>

John Okell. (1994): Burmese, An Introduction to the Script. Northern Illinois University Press.

John Okell. (2002): *Burmese By Ear*, Audio-Forum, Sussex Publications Limited, Microworld House, 4 Foscoate Mews, London W9 2HH. Available at <http://www.soas.ac.uk/bbe/> (Accessed on February, 2013)

Jurafsky, Daniel and James H. Martin. (1998). Speech and language processing: an introduction to natural language processing, computational linguistics and speech recognition. Pearson Education.

Keith Stribley. (2007). Collation of Myanmar in Unicode technical report. Available at <http://my.duniakitab.com/ThanLwinSoft/ThanLwinSoft/MyanmarUnicode/Sorting/MyanmarCollation20080424.pdf>

Kiraz, G.A., M'obius, B. (1998): Multilingual syllabification using weighted finite-state transducers. Proceedings of the Third International Workshop on Speech Synthesis. Jenolan Caves, Australia, pp. 71–76.

Krammer, Viktor. (2006) : Phishing defense against IDN address spoofing attacks. In the Proceedings of the 2006 International Conference on Privacy, Security and Trust: Bridge

the Gap Between PST Technologies and Business Services. ACM.

Lauri Karttunen. (2001). Applications of Finite-State Transducers in Natural Language Processing, Implementation and Application of Automata Lecture Notes in Computer Science Volume 2088, pp. 34-46.

Le Hong P., Nguyen Thi M.H., Azim R., Ho Tuong V. (2008): A Hybrid approach to Word Segmentation of Vietnamese Texts. In Proceeding of the 2nd International conference on Language, Automata Theory and Application, LATA 2008, pp. 240-249.

Marchand, Y., Connie A., Damper R. (2007) : Evaluation of automatic syllabication algorithms for English. In Proceedings of the 6th International Speech Communication Association (ISCA) Workshop on Speech Synthesis.

Maimaitimin Saimaiti , Zhiwei Feng, A Syllabification Algorithm and Syllable Statistics of Written Uyghur. Available at ucrel.lancs.ac.uk/publications/CL2007/paper/153_Paper.pdf

Mark Davis and Michel Suignard. Unicode Security Considerations. Available at <http://www.unicode.org/reports/tr36/> (Accessed March 2013)

Mark Davis and Ken Whistler. (2013). Unicode Collation Algorithm, Version 6.0.0. Available at <http://www.unicode.org/reports/tr10/>

Mark Davis and Michel Suignard. Unicode Security Mechanisms. Available at <http://www.unicode.org/reports/tr39/tr39-1.html> (Accessed June 20)

Martin Hosken. (2012). Unicode Technical Note # 11, Representing Myanmar in Unicode, Version 4. Available at <http://www.unicode.org/notes/tn11/>

Martin Hosken. (2012): Representing Myanmar in Unicode, Details and Example. Available at <http://www.unicode.org/notes/tn11/> (2013, February 23)

Mohri, Mehryar. (1997). Finite-state transducers in language and speech processing. Computational linguistics, pp. 269-311.

Myanmar Language Commission. (2006): Myanmar Orthography. Third Edition, University Press, Yangon, Myanmar.

Niimura Izuru ed. Kojien 6th edition, Iwanami Shoten Publisher, 2011.11-13.

O.G. Kakde. (2007). Theory of Computation. Laxmi Publication, New Delhi, India.

Peter T. Daniels, William B. (1996): The World Writing Systems, Oxford University Press.

Roark, Brian, and Richard William Sproat. (2007). Computational approaches to morphology and syntax. Oxford University Press.

Robert N. Moll, Michael A. Arbib and A.J. Kfoury. (1987). An Introduction to Formal Language Theory, Springer-Verlag, New York.

Ronald M. Kaplan and Martin Kay. (1994). Regular models of phonological rule systems. Computational Linguistics.

Ruvan W., Asanka W., and Kumudu G. (2005): A Rule Based Syllabification Algorithm for Sinhala, Proceedings of 2nd International Joint Conference on Natural Language Processing (IJCNLP-05), Jeju Island, Korea, pp. 438-449.

Sarmad Hussain and Nadir Darrani. (2008). A Study on Collation of Languages from

Developing Asia, National University of Computer and Emerging Sciences, Lahore, Pakistan.

Sarmad Hussain , Nayyara Karamat. Internationalized domain names: Feedback of PAN L10n project on IDNAbis for Languages of Developing Asia, Center for Research in Urdu Language Processing, Lahore, Pakistan.

Samad Hussain, Sana Gul, and Afifah Waseem. Urdu Encoding and Collation Sequence for Localization. Center for Research in Urdu Language Processing National University of Computer and Emerging Sciences.

Schmid, H. (2005): A programming language for finite-state transducers. In Finite-State Methods and Natural Language Processing, FSMNLP 2005.

Shuly Wintner. (2002). “Formal language theory for natural language processing”. In the Proceedings of the ACL-02 Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics. Morristown, NJ, USA: Association for Computational Linguistics, pp. 71–76.

Susanna S.Epp. 2011. Discrete Mathematics: An Introduction to Mathematical Reasoning, Cengage Learning, USA.

Tin Htay Hlaing. (2012): Manually Constructed Context-Free Grammar for Myanmar Syllable Structure. In the proceeding of the European Chapter of the Association of the Computational Linguistics(EACL), Student Research Workshop.

Tin Htay Hlaing and Yoshiki MIKAMI. (2011). Collation Weight Design for Myanmar Unicode Texts. In the proceedings of International Conference on Human Language and Technology, Alexandria, Egypt, pp. 1-6.

Tin Htay Hlaing and Yoshiki MIKAMI. (2013). Automatic Syllabification of Myanmar Texts using Finite State Transducer, International Journal on Advances in ICT for Emerging Regions, Volume 6, Number 2.

Unicode Consortium, Default Unicode Collation Element Table. (2013). <http://www.unicode.org/Public/UCA/6.3.0/allkeys.txt>

Verisign. The IDN registration Process. Available at http://www.verisigninc.com/en_GB/products-and-services/domain-name-services/value-added-products/idn-domain-names/why-are-idns-important/index.xhtml?loc=en_GB

Wijayawardhana, Harsha, et al. (2008) : Implementation of Internet Domain Names in Sinhala, International Symposium on Country Domain Governance. Nagaoka, Japan, pp. 20-23.

Win, Kyawt Yin. (2011): Myanmar Text-To-Speech System with Rule-based Tone Analysis. PhD Dissertation, University of Ryukyus, Okinawa, JAPAN.

Y.A. El-Imam and Z.M. Don. (2000) Text-to-Speech conversion of Standard Malay. International Journal of Speech Technology 3, Kluwer Academic Publishers, pp. 129-146.

Yoshiki Mikami.: World of Scripts in Asia. Available at <http://gii2.nagaokaut.ac.jp/ws/indic.html> (2013, January 23)

Yuzana and Tun, K.M. (2008a). A comparison of collation algorithm for Myanmar language, Third International Conference on Digital Information Management, ICDIM 2008, pp. 538-543.

Yuzana and Tun, Khin Marlar. (2008b). Collation Strategy Based on Heuristics Chart for Myanmar Language. Ninth ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing, (SNPD'08), IEEE, pp. 640- 645.

Zin Maung Maung, Mikami Yoshiki. (2008): Rule-based Syllable Segmentation of Myanmar Texts. In Proceedings of the 6th Workshop on Asian Language Resources, January 11-12, Hyderabad, India.