Support Profile Leads to a Pattern among Natural Languages

Anfal ALGharabally, Bala Kalyanasundaram, and Mahe Velauthapillai

***Abstract*—Given a collection of texts from different spoken languages, this paper investigates the fundamental question of discovering a common pattern among these languages. Considering the fact that orthography differs, amongst many other things, should there even exist a pattern among many natural languages? Further, will the pattern change if we choose a different collection of texts? Can we concisely characterize the pattern and possibly associate a meaning to the pattern? This paper introduces a concept called support profile for any collection of strings. A simple yet intuitive hypothesis that predicts a hidden pattern among support profiles of individual natural languages is presented. The pattern has an elegant mathematical representation and it can be explained by a limitation on sound production of the speakers of the language. Languages from six different families are chosen to validate the hypothesis. They are Arabic, English, Finnish, Greek, Latin, Maltese, Swahili, Tagalog and Turkish. The hypothesis is called The Universal Support Hypothesis for Natural Languages. Intuitively, the pattern predicted by the hypothesis is the existence of a small support set. This set differs from one language to another but it happens to coincide with the set of vowels of the natural language.**

***Index Terms*—Approximation algorithm, k-letter, universal support set, vowels.**

I. Introduction

A language is a string based system to represent an underlying structure. In general, natural language refers to a human language such as English, French, Hebrew, Latin, Sanskrit, Tamil, etc. [1]-[5]. It is often thought of as a naturally evolved system as opposed to an artificially created system such as computer languages with predesigned underlying grammar [5].

Given a collection of valid strings from a language, it is reasonable to expect a pattern among the strings. Any pattern that we learn can possibly reveal some useful information about the underlying grammar or structure of the language. On the other hand, consider a collection of sentences from different languages such as Arabic [3], Hindi [6], English [3], Latin [3], Turkish [7], and Greek [8].

البَيْتُ كَبيرٌ جدًّايَأكلُ أحْمَدُ تُفّاحة
Αποφάσισαν να κάνουν ό,τι τους είπε ο Προμηθέας αλλά με ένα όρο.
The wedding was like a big celebration.
agricolae provinciam in fossam collocant.
Avrupa ile Asyayı bölen1 2boğazı karşıdan karşıya geçen
मैं काउफ़ी प्यार कर्ती पीना हू।

Fig. 1. Pattern among a collection of strings.

Manuscript received September 15, 2016; revised November 8, 2016.

Anfal ALGharabally is with the Public Authority for Applied Education and Training, Kuwait (e-mail: anfalqw8@gmail.com).

Bala Kalyanasundaram and Mahe Velauthapillai are with the Computer Science Dept., George-town University, Washington DC, USA (e-mail: @kalyan@cs.georgetown.edu, $mahe@cs.georgetown.edu).

Is it possible to observe a pattern among such collection of strings from different natural languages [9] (see Fig. 1)? It is hard to believe that a meaningful pattern may exist among strings from different languages. The question we consider in this paper is even more intriguing. Suppose a random string from the set of valid strings is chosen for each language. Among such collection of random but valid strings from different natural languages, can we observe [10]-[13] a meaningful pattern? Can the observed pattern be the same if we swap random valid string with another from the same language? Or, equivalently, is there a meaningful pattern among different natural languages?

Given a string one can search for a pattern in the string. This pattern may reveal something about the string. On the other hand, a pattern among a collection of strings reveals something about the collection and less about individual strings. Suppose you have more than one collection of strings. Can we find something common among many such collections? This paper is about finding a pattern among such collections. Each collection may have infinitely many strings. We have developed a new measure, called support profile, to represent each collection. Using this profile, we look for a pattern. Finding the support profile of a natural languages is computationally very hard. Without the full knowledge of the support profiles of natural languages, this paper predicts and empirically validates a pattern among the support profiles. Since the support profiles are hard to compute for natural languages, we present the pattern as a hypothesis called Universal Support Hypothesis.

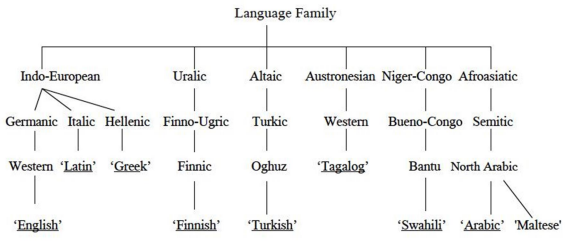

Fig. 2. Language family tree.

Intuitively, this hypothesis predicts the existence of a small support set for every natural language. We test the hypothesis on nine different natural languages, from six different families, shown in Fig. 2. (See [3]-[5], [12]-[15]). It is amazing that the small support set coincides with the set of vowels for all nine languages.

The following definitions are useful in defining the main hypothesis. Let $\sum$ be a finite alphabet. Let $V \subset \sum^*$ be a set of strings w from $\sum^*$. The set L is called a language with alphabet $\sum$. The alphabet $\sum$ is called non-redundant if there is no letter in $\sum$ that can be replaced by another letter or sequence of letters in $\sum$.

doi: 10.18178/ijlll.2016.2.4.92

In one example, ∑ = {a, b, c... z} and L consists of all the words, sentences and collection of sentences in English. In another example, let ∑ be the set of twenty amino acids and L be the set of naturally occurring proteins where each protein is a sequence of amino acids.

Given an alphabet ∑ and a language L over ∑*, we define a weight function w whose domain is ∑* and its range is positive real numbers.

A weight function w is called non-decreasing with respect to substrings, if w(s) ≥ w(t) for any two strings s and t where t is a substring of s.

A weight function is called natural if it is non-decreasing and the weight of the empty string is 0.

There are many weight functions, and one natural example is the string length function *len*

For any string s ε ∑* we define *len(s)* to be the length of the string s measured as the number of letters in s.

II. Universal Support Hypothesis

Let us first consider an example. Let L be a natural language with non-redundant alphabet ∑ = {a, b, c.. z}. Now, consider a small subset V = {a, e, I, o, u} of letters from ∑. For ease of presentation, let us call the letters in V vowels. Consider a string σ = this is an example. Let w be a natural weight function that assigns a non negative weight to any substring from ∑*. We now highlight the appearance of vowels in string σ by thIsIsAnExAmplE. Now imagine vowels as pillars or support characters that carry the weight of the entire string s as defined by the weight function w.

This represents a natural way to define the weight of string s carried by V. Using letters in V as scissors, cut the string s into substrings th, s, s, n, x, and mpl. The weight of s carried by V is

$$\text{Max}\{\ w(th);w(s);w(s);w(n);w(x);w(mpl)\}: \qquad (1)$$

Given a string s ε ∑*, a set V ⊂ ∑ is said to be a support set for s, if the weight of s supported by V is the smallest among all subsets of S of size |V|. For ease of presentation, a simpler version of the hypothesis is presented first.

Universal Support Hypothesis (USH for short).

Given a natural language L for a non-redundant alphabet ∑, there exists a small subset V of ∑ and a nontrivial natural weight function w such that for every long string σ in L, the support set for σ is V.

Observe that the simpler version of the hypothesis does not specify the numerical quantities small and long. It is because there is no easy answer to this and these quantities may change from language to language. Also, it is possible that there are some pathological cases, i.e., some long strings that spoil a true support set V. That is, a true support set V may carry the smallest weight in almost all long strings and possibly close to the smallest weight in other long strings.

The following version makes the hypothesis more robust by replacing long by an error parameter ε.

Universal Support Hypothesis (more formal definition). Let ε > 0 be a small constant. Given a natural language L for a non-redundant alphabet S, there exists a small subset V of ∑ and a non-trivial natural weight function w such that the probability that V is the support set for a **σ** *in L chosen uniformly at random is at least 1 - ε.*

The following argument shows that the hypothesis holds trivially for carefully chosen but very uninteresting weight function.

Let L be the language under consideration with alphabet ∑. Take any non-empty subset V of ∑. We now define a non-decreasing function that satisfies the hypothesis. Every letter in V gets a weight of 1 while all other letters get a weight of 0. For any string s, we define w(s) to be the sum of the weights of the letters in s. It is now easy to see that the support set is V for every string in L.

It is obvious that such weight functions are not interesting at all. They do not reveal anything interesting about a natural language. In the next section, we introduce some weight functions that reveal meaningful support set for the following five natural languages: Arabic, English, Greek, Latin, and Swahili.

III. Weight Functions

What type of natural weight functions should we consider? The weight function should not bias any support set. Instead, the strings in the language must determine the support set. There are two types of natural functions. In the first case, the function assigns weight indepen-dent of the natural language under consideration. An example of such a natural function is the length function *len* where each letter in the alphabet gets weight 1. This weight function obviously does not bias any particular subset of ∑ as a support set.

In the second type, the weight function takes the language into consideration while assigning weights to a string. For instance, one would like to assign less weights to substrings that occurs often. In English, the substring "th" occurs often and so must be assigned weight less than 2. Based on bi-gram frequency count, a new weight function is defined below.

Let L be the natural language under consideration. Without loss of generality, let ∑ = {a,b,c…z,*space*} Suppose σ be a long string $\varepsilon_1\varepsilon_{2\ldots.}\ \varepsilon_n$. First we calculate the following:

For each α in {a,b,c..z} do

Count the number of occurrences of α in σ.

Call the count *Freq(*α).

For each α_1 in {a,b,c…z, *space*} do For each α_2 in {a,b,c…z } do

Count the number of occurrences of $\alpha_1\alpha_2$ in σ. Call the count BiGramFreq(a_1a_2).

For each α_1 in {a,b,c…z, *space*} do For each α_2 in {a,b,c…z} do

Set weight function

Pair Weight (α_1; α_2) =

(Freq(α_2) – Bi Gram Freq (α_1; α_2)) / Freq (α_2)

We now define the Markov weight of σ which is de-noted as w(σ) to be:

$$PairWeight(space,\varepsilon_1)+\sum_{i=2}^{n} PairWeight(\varepsilon_{i-1},\varepsilon_i).$$

Observe that the weight contributed by each character depends on the previous character. The only exception is the first character ε_1 which contributes a weight equal to PairWeight(*space*; ε_1). Space does not contribute to the weight and thus PairWeight(ε_i ,space) = 0 for all ε_i.

The third type of weight function is based on raw frequency of letters in a text. Let total be the number of letters in the text. Also, let *freq*(α) be the number of occurrences of a in the given text. We set the weight(α) to be the ratio: (total - *freq*(α))/ total.

IV. Data Curation

Ideally, given some sample text from a natural language, we want to test the Universal Support Hypothesis and possibly identify the support letters. In this paper, we present our results for the natural languages English, Greek, Latin, Arabic, and Swahili. Since languages are constantly changing, it is possible that we may get caught in the flux and jump to the conclusion that the hypothesis does not hold or end up with in-correct support letters. For instance, with the introduction of instant messaging, many new words such as gtg (stands for got to go) were introduced. There are over two million acronyms and this list continues to grow. Acronyms appear all over text documents. Some examples are CPU, RSVP, and BYOB. In addition, text also carry foreign letters, symbols, names, abbreviations, punctuation, and numbers. In some languages, such as Arabic, certain letters, such as short vowels, are omitted and the native speaker knows where to add such letters.

In order to make the playing field even, we apply the following data curation procedure to every text.

1. Do not distinguish between upper and lower-case letters. Every letter is an upper-case letter.
2. Remove foreign letters, symbols, punctuation and numbers.
3. Eliminate hyphenation, subscripts, and superscripts.
4. Remove parenthesis while retaining the content inside the pair.
5. Expand or remove abbreviations and acronyms.
6. Choose text so that there are no hidden or implicit letters.
7. Diacritics are removed from letters except for Quran text.

In the case of the Semitic language Arabic, passages from Quran are chosen since vowels are not eliminated from the text. Since text under consideration is huge, and sometimes foreign to the authors, some abbreviations or acronyms may still be present in the text. However, the result is inspected to make sure that the acronyms and abbreviations do not influence the final outcome. For each language, approximately 25 to 40 documents are selected from literature, religious documents, and classics to modern text from Internet to test the hypothesis. It is expected that the results will be more accurate if the files are not too small. File sizes ranges from 20,000 characters to a few hundred thousand characters. with AEIOU is the winner 97 % of the time.

If a d-letter combination consistently wins across al-most all documents, we can declare this combination as a potential support set. It is possible that a potential support set exists for many different values of d. In this case, which support set is the true support set? Theoretically it is possible that such potential support sets need not share any letter in common. But it turns out that the potential sets share a structural property. In almost all the cases, the support sets happened to be a subset of vowels and as the size of the support set increases, a fuller display of vowels emerge to more fully represent the support set. For Arabic (Fig. 3), English (Fig. 4), Greek (Fig. 5), Latin (Fig. 6) and Swahili (Fig. 7) vowels emerge as the support set [10], [16].

We now explain the results for English (Fig. 4). For size 1, the high frequency letter E wins at size 1 while E along with other high frequency letters including vowels win at size 2. At size 4, the vowel combination AEIO wins in 90% of all the documents, thus this set is a potential support set. So far, as expected, the letter U barely showed up as a potential support letter. But, at size 5, AEIOU wins in 95% of all documents. On the other hand, at size 6, AEIOUY wins at or above 70% of the documents. Based on these results, we can conclude that AEIOU is a strong support set while Y is a weak support.

The following Greek-to-English mapping is used to handle Greek documents. Greek vowels need not map to corresponding English vowels (Fig. 5).

$$\{A,B,\Gamma,\Delta,E,Z,H,\Theta,I,K,\Lambda,M,N,\Xi,O,\Pi,P,\Sigma,T,\Upsilon,\Phi,X,\Psi,\Omega\} = \{A,B,G,D,E,Z,H,Q,I,K,L,M,N,X,O,P,R,S,T,U,F,C,Y,W\}$$

V. Results

Markov weight function is the only function considered in this presentation. USH claims that there is a small support set for each natural language. The size of the support set varies from language to language. In or-der to test the hypothesis, algorithms were run on each text assuming the size of the support set to be 1,2..d where d is 8. For a given d and a given text, a set V of d-letters is said to be the winner if the weight of the text carried by V is minimal among all d-letter combinations We used bar charts to summarize our results. For each d, the winning combination is collected for approximately 25 different texts and plotted in a bar chart. The x-axis displays different size cuts and the y-axis displays the percentage of the text that were winners for the specific cut. For each d we used different colors. For English (Fig. 4), at top of the bar chart we display legends.

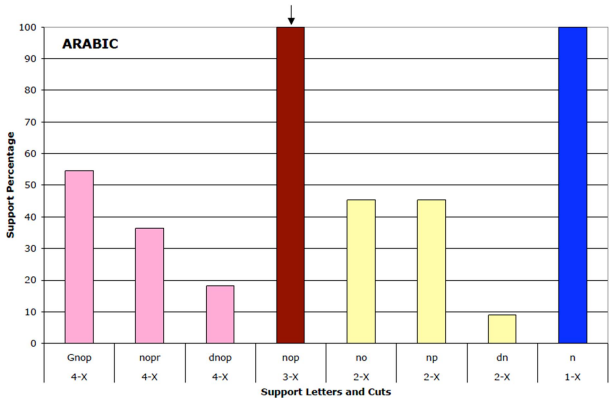

Fig. 3. Arabic.

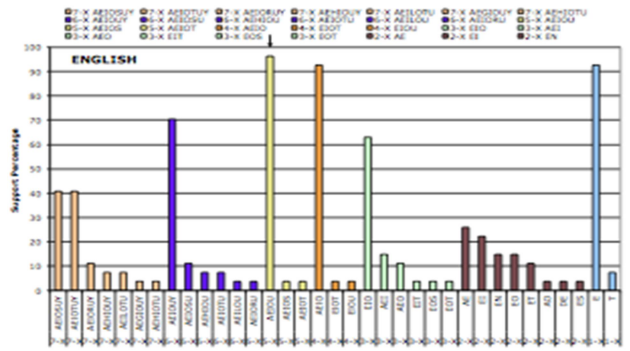

Fig. 4. English.

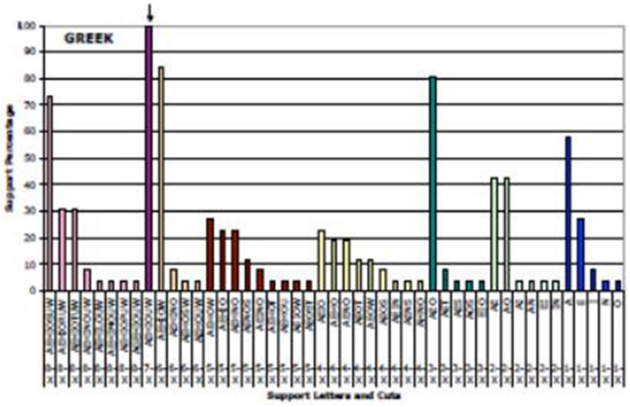

Fig. 5. Greek.

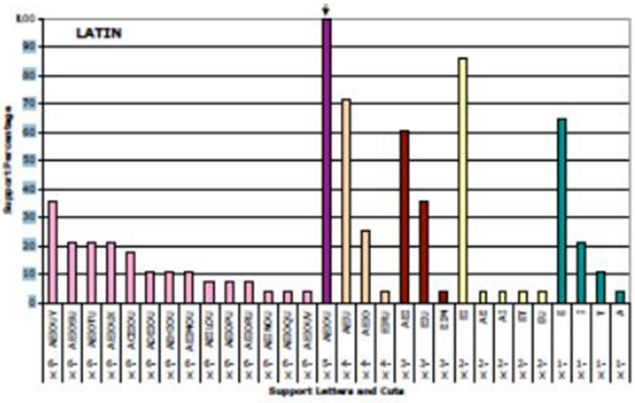

Fig. 6. Latin.

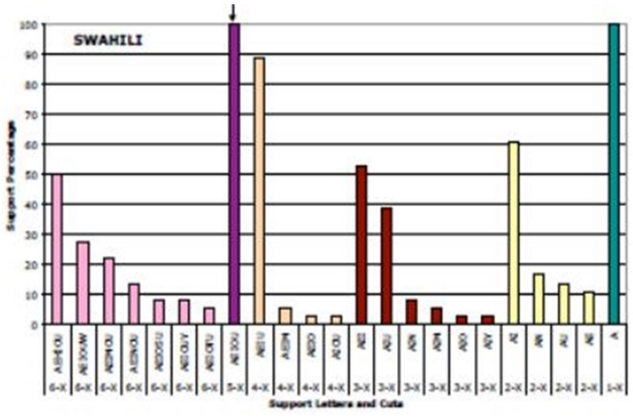

Fig. 7. Swahili.

For Greek (Fig. 5), observe that for support set of size 1, the top four frequent letters A, E, I and O win. But the winner changes from document to document. For size 2, AE or AO wins with AE having slight advantage over AO. Since the winning set depends on the document, neither set is a potential candidate for support. As we increase the size to 3, AEO combination seems to win 80% of the documents. Other potential winners do not repeat significantly many times, thus making letters AEO prime candidates for support. So far, letter H is nowhere to be found. As we move to size 4, AEHO emerges as the winner for 40% of the documents with AEIO as the close second winner. For the first time, W or U shows up in 10% of the documents. Obviously, the winner of size 3, namely AEO, is the local maxima. Continuing to increase the set-size to 4, 5, and 6 we observe that AEHO, AEHOW and AEHIOW wins. But the margin of victory changes from document to document. Finally at size 7, the winner AEHIOUW emerges by winning in all documents (i.e., 100%). Clearly, this is a primary candidate for support set. In order to check if it is the local maxima, we ran our algorithms for support set of size 8. The winner AEHIOSUW appears in 70% of the documents. AEHIORUW and AEHIOTUW appears as possible alternate for AEHIOSUW. Based on this, we conclude AEHIOUW as the support set for Greek.

The Semitic language Arabic (Fig. 3) has three short vowels in the graphic representation. These vowels are not typically written in the body of the word, but they are added above of under the consonant that they represent. In order to facilitate the string processing of our algorithms, we represent the Arabic letters in the standard English letters. The three short vowels are Fathah, Dammah, and Kasrah which are represented by the English letters n, o and p. Based on the chart given above, it is clear that the nop combination is the support set for Arabic.

For Latin (Fig. 6), support set of size 1, high frequency letters E, I, A or T wins. At size 2, the two most frequent letters E and I win as the support set. EIU or AEI wins almost 50% at size 3. So after letters E and I, either A or U appears as a support. But the actual support letters changes from document to document. Also, for the first time, O showed up as a potential support in approximately 10% of the documents. At size 4, AEIU combination provides support in 70% of the documents while AEIO supports in 30% of the documents. Finally, at size 5, AEIOU wins 100% of the documents, making this set as a potential support set. This is con-firmed when we find that the winner of the support set for size 6 varies from document to document where the highest winner AEIOSU ranges from 60% to 20%.

The Swahili language [17]-[18] is of Bantu (African) origin. Swahili is spoken in many countries of Eastern Africa. Swahili has become the primary language for East African region. The alphabet of Swahili is almost identical to that of English. The letters Q and X are not used. There are five vowels, A, E, I, O and U in Swahili. Unlike in English where two vowels merge together to form a single sound, each vowel is pronounced separately in Swahili. There is clearly no surprise in the results. At size 4, the vowel combination AEIU wins in 90% of the documents (Fig. 7), thus making this set a primary candidate for support set. However, at size 5, AEIOU wins in all documents while at size 6, the sup-port changes from document to document. Based on these results, it is clear that AEIOU is a strong support set for Swahili.

VI. Conclusion

USH holds for nine natural languages from six language families. In this paper, we presented our investigation for five natural languages. However for all natural languages we have tested so far, the support set happens to be the set of vowels for these natural languages. The authors believe that the hypothesis will hold for other languages if alphabet redundancy is removed. Also, the support graph for various sizes seems to provide a fingerprint/profile of the corresponding language. What other information do they convey? For instance, after correctly identifying the set of vowels, the algorithm selects the weak vowel Y as the sixth support for English, whereas the consonant S is selected as the eighth support for Greek. However, challenges as well as future discoveries await those who attempt to expand this idea to new frontiers. For instance, experiments are underway to test the hypothesis on the text representation of natural proteins. There is strong belief that this hypothesis can be applied to music as well. Can this idea be applied to any other natural phenomenon, say animal or biological communication, which can be represented as strings?

References

[1] D. Abondolo, *The Uralic Languages*, Rout-ledge, 1998.
[2] B. Comrie, *Language Universals and Lin-guistic Typology: Syntax and Morphology*, University Of Chicago Press, 1989.
[3] B. Comrie, *The World's Major Languages*, USA: Oxford University Press, 1990.
[4] Boer de Bart, *The Origins of Vowel Systems: Studies in the Evolution of Language*, USA: Oxford University Press, 2001.
[5] P. T. Daniels and W. Bright, *The World's Writing Systems*, USA: Oxford University Press, 1996.
[6] R. G. Gordon, *Ethnologue: Languages of the World*, SIL International, 2005.
[7] P. Ladefoged, *Vowels and Consonants*, UK: Wiley Blackwell, 2005. Lewis, Geoffrey. Turkish Grammar, USA: Oxford University Press, 2001.
[8] H. W. Smyth and G. M. Messing, *Greek Grammar*, Harvard University Press, 1956.
[9] J. Tore, *Speak: A Short History of Lan-guages*, USA: Oxford University Press, 2003.
[10] P. Ladefoged and I. Maddieson, *The Sounds of the World's Languages*, UK: Wiley Blackwell, 1996.
[11] I. Mackay, *Phonetics: The Science of Speech Production*, Allyn and Bacon, 1991.
[12] P. Lieberman and S. E. Blumstein, *Speech Physiology, Speech Perception, and Acoustic Phonetics*, Cambridge University Press, 1988.
[13] G. Sampson, *Writing Systems: A Linguistic Introduction*, USA: Stanford University Press. 1990.
[14] R. Hetzron, *The Semitic Languages*, Rout-ledge, 2006.
[15] E. Konig, *The Germanic Languages*, Routledge, 2002.
[16] D. B. Fry, *The Physics of Speech*, Cambridge University Press, 1979.
[17] P. Wilson and S. Swahili, United Kingdom: Longman Group, 1983.
[18] R. M. Vago, *Abstract Vowel Harmony Systems in Uralic and Altaic Languages*, Indiana University Linguistics Club, 1972.

Anfal Algharabally obtained her undergraduate degree in linguistics from Kuwait University in 1999. She joined Georgetown University in 2000 and obtained her MS in computational linguistics in Dec. 2001. Subsequently she joined the Public Authority for Applied Education and Training, Kuwait. Her research involves automatic detection and correction of errors in Arabic, machine learning, natural language Processing and computer aided learning.

Bala Kalyanasundaram received his PhD degree in computer science in 1988. He was a faculty of University of Pittsburgh from 1988-2000. He joined Georgetown University in 2000. He is currently Craves Family Professor of Computer Science. His research interests include algorithms, computational complexity, computational learning theory and computational science.

Mahendran Velauthapillai joined Georgetown University in Fall 1986 after obtaining his Ph.D. degree in computer science from University of Maryland. He is currently hold the McBride Family endowed chair. His research interest includes computational learning theory, wireless networks and computational science.