An Extensible Multilingual Corpus of DFA Construction Problems

Aditya VISHWANATHAN[a], Mallika Pushpa BHAVATARINI[a], Namratha RAVI[a], Sneha Umapathi BHUVANESHWARI[a], Srilalitha Krishnan MURTHY[a], & Viraj KUMAR[b*]
[a]*Department of Information Science and Engineering, PES Institute of Technology, India*
[b]*Department of Computer Science and Engineering, PES University, India*
*viraj.kumar@pes.edu

Abstract: In a country with linguistic diversity, academic instruction may occur in a language in which learners and instructors lack fluency. In such a situation, learners may struggle to answer problems posed by instructors not because they lack the requisite technical skills, but because they are unable to precisely comprehend what is being asked. In this work-in-progress paper, we demonstrate a semi-automated technique to translate a specific category of problems in the undergraduate Computer Science curriculum (the construction of deterministic finite automata) into multiple languages. We show how a corpus of problems (specified in a recently proposed mathematical representation) can be extended in two ways. First, new languages can be targeted by manually translating elements of this mathematical representation into each new language. This is a one-time effort per target language. Second, new problems can be added to the corpus in the mathematical representation at any time, and these are translated into all target languages automatically. We are currently evaluating our technique with English (the language of instruction at our institution) as well as Kannada and Hindi (a majority of our instructors and learners are fluent in at least one of these), and we describe the results of a small-scale ($N = 38$) study which shows promising results.

Keywords: Language translation, Computer Science education, Deterministic Finite Automata (DFA).

1. Introduction and Related Work

English has emerged as the *lingua franca* in technical domains such as engineering (Riemer, 2002). In an increasingly global work environment, a premium is placed on graduates with strong English communication skills, and fluency in this language can result in a competitive advantage (Kapur and Ramamurti, 2001). It is therefore common for technical educational institutions around the world to use English as a medium of instruction, and successful students require proficiency in English (Rauchas et. al, 2006; Qian and Lehman, 2016). In a linguistically diverse country such as India, a substantial fraction of people are not native English speakers[1], and poor educational standards mean that command over English is weak even for those with formal training in the language. As a result, instructors and learners may be compelled to communicate in a language in which neither group is fluent. India's National Knowledge Commission recommends that "translation should play a critical role in making knowledge available to different linguistic groups" (Pitroda, 2009). Although translation cannot compensate for poor English fluency, we believe that "making knowledge available" in a familiar language will benefit learners who otherwise need to cross two mental hurdles: (1) parsing English, and (2) understanding technical concepts. Learners who cannot cross the first of these two barriers often have no chance to demonstrate their technical understanding. For pedagogical purposes, it may therefore make sense to provide learners with "training wheels" in the form of translation tools to help them cross former barrier and hone their technical skills, and then gradually withdraw such tools.

Machine Translation (MT) has made rapid advances over the past few years, with translations to and from English receiving the bulk of attention. Existing techniques appear to perform well with

[1] In India, the census of 2001 recorded 29 Indian languages with more than 1 million native speakers each, but less than 0.25 million native English speakers.

simple sentences, and translations between linguistically proximate languages. For technical content, Chen et. al (2016) examined the quality of Google Translate on English-language medical educational materials and found that “the likelihood of incorrect translation increased when the original sentence required higher grade levels to comprehend”, and that humans significantly outperformed MT when the target language was Chinese, but not when it was Spanish.

In this paper, we consider problems from the undergraduate Computer Science curriculum that ask learners to create deterministic finite automata (DFA) that accept a particular set of strings. As an example, consider the following problem: Construct a DFA for the regular language *L* consisting of the set of all binary strings *w* that *start with* ‘10’ *and end with* ‘01’. The unique minimum-state DFA for this problem is shown in Figure 1.

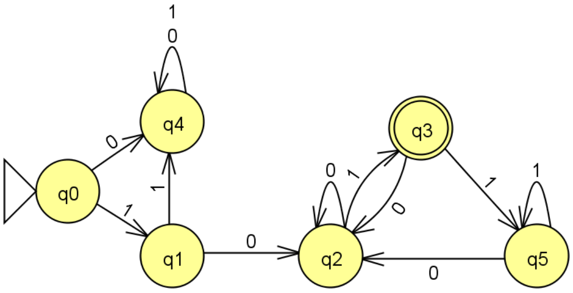
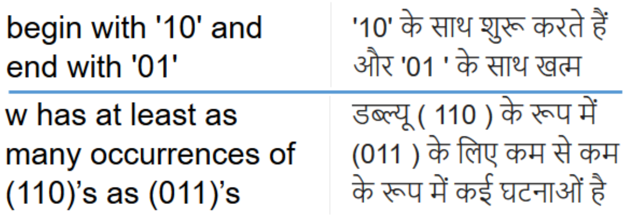


Figure 1. A DFA for the given language *L*.



begin with '10' and end with '01'	'10' के साथ शुरू करते हैं और '01 ' के साथ खत्म
w has at least as many occurrences of (110)'s as (011)'s	डब्ल्यू (110) के रूप में (011) के लिए कम से कम के रूप में कई घटनाओं है

Figure 2. The Google translation of two DFA construction problems from English to Hindi. The simpler example (top) is translated accurately, whereas the translation for the more complex example (bottom) is almost wholly meaningless.

In order to solve such a problem, it is crucial for learners to parse the italicized description unambiguously. This sentence is easy enough for Google Translate to generate accurate Hindi (Figure 2, top). In contrast, consider a more complex problem: Construct a DFA accepting all binary strings *w* such that *w has at least as many occurrences of* (110)’s *as* (011)’s[2]. If a learner uses Google Translate to convert this into Hindi (Figure 2, bottom), the result is impossible to comprehend. For example, the term “occurrences” refers to the number of times a pattern appears, but this is incorrectly translated into घटनाओं, the Hindi word for “incidents”. Our goal in this paper is to demonstrate a simple and accurate method for translating DFA construction problems into several languages. Instead of translating from English, our technique relies on a mathematical representation of DFA problems that has recently been proposed (Shenoy et. al, 2016). The remainder of this paper is organized as follows. In Section 2, we review this representation for DFA construction problems and explain how problems can be easily translated from this representation into other languages. Next, we present results of a small-scale pilot study to assess the effectiveness of providing translations for learners in Section 3. Finally, we discuss classroom applications and limitations of this approach in Section 4, together with our plans for future work.

[2] This problem was adapted from the Graduate Aptitude Test in Engineering (Computer Science), 2014. This high-stakes test is taken annually by approximately one million students in India.

2. Representation and Translation of DFA Construction Problems

Shenoy et. al (2016) describe a tree-representation of DFA construction problems. Internal nodes of this type of tree correspond to functions, and leaves refer to the input string, constants, etc. The root of this tree is a Boolean-valued function. As an example, Figure 3 shows this mathematical form of the DFA construction problem for the language *L* defined in Section 1.

The set of functions is rich enough to represent most of the DFA construction problems found in popular textbooks for the relevant undergraduate Computer Science course, and a corpus of 17,537 such problems has been automatically generated (Shenoy et. al, 2016). Each such problem is persisted as a file in JSON (JavaScript Object Notation) format, which captures the hierarchical tree-structure.

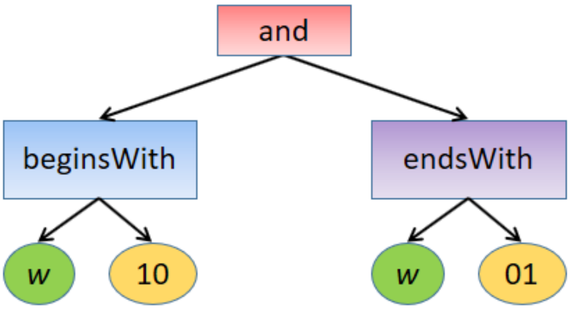


Figure 3. The DFA construction problem for the language *L* represented mathematically.

Each of the functions in the mathematical representation above leads to a simple translation rule in every target language. For example, **beginsWith**(*w*, *x*) translates to "*w* begins with *x*" in English. In Hindi, the translation has a slightly different structure: "*w x* से शुरू होता है". In contrast, **and**(*x*, *y*) has the same structure in both English ("*x* and *y*") and Hindi ("*x* और *y*"). There are about 40 functions that need to be translated (additional functions can always be added to enhance the expressiveness of the problems), so the one-time effort to translate from the tree representation to any new language is manageable. Also note that our translation mechanism allows problems to be translated in "segments" that can vary from individual functions to whole sub-trees. We are particularly interested in this capability, because several of our instructors and learners are (at least) bilingual, and constructions that are somewhat ambiguous to them in one language may be clearer in another.

For rapid translation, our implementation persists the rules for translating functions in a locally stored JSON file (one for each target language). For each new target language, it is only necessary for users to download this small, language-specific JSON file and update the list of supported languages.

3. Experimental Evaluation

In order to assess the benefit of such translations, we performed a small-scale pilot study with $N = 38$ undergraduate volunteers (18 male, 20 female; all were fluent in Kannada but English fluency varied).

Table 1: Three regular languages for which learners were asked to classify strings.

Language	Binary strings *w* such that…	Kannada translation (English transliteration)	Strings to classify
L_1	*w* starts with '01' or ends with '10'	*w* '01' rinda shuruvagabahudu athava *w* '10' rinda anthyagolabahudu	001, 010, 011, 100
L_2	*w* contains exactly three 1's and *w*'s length is at least 4	*w* nalli '1' mooru saari maatra barabeku mattu *w* vina udda 4 kintha kadime ira baradu	111, 1111, 0111, 010101

Language	Binary strings *w* such that…	Kannada translation (English transliteration)	Strings to classify
L_3	*w* has an equal number of 0's and 1's and *w*'s length is at most 5	*w* nalli '0' mattu '1' ra sambhavisuva enike samavaagirabeku mattu *w* vina udda 5 kintha hecchu ira baradu	empty string, 01, 1100, 011100

All volunteers had taken the appropriate "Theory of Computation" course in the prior semester. Based on self-reported grades in this course, the volunteers were split into four groups of approximately equal size so that each group had nearly the same distribution of A, B and C grades in this course. Next, each volunteer was asked to indicate which of a given set of four strings belonged to each of the given languages, as shown in Table 1. The 9 volunteers in group A were given only English descriptions for each language. The 10 volunteers in group B were additionally given Kannada descriptions for languages L_1 and L_2, the 9 volunteers in group C were given additional Kannada descriptions for languages L_1 and L_3, and the 10 volunteers in group D were given additional Kannada descriptions for all languages. Thus, for each language, some volunteers had only the English text whereas other volunteers had both English and Kannada text. Each volunteer's answers were evaluated by assigning a score of 1 for each correctly classified string. Thus, a volunteer could obtain a maximum score of 4 for each language. The scores obtained are listed in Table 2.

Table 2: Scores obtained by volunteers on the three languages.

Language	English-only text	English + Kannada text
L_1	Average score: 2.89 Standard deviation: 1.17 Count: 9	Average score: 3.72 Standard deviation: 0.53 Count: 29
L_2	Average score: 2.72 Standard deviation: 0.83 Count: 18	Average score: 2.85 Standard deviation: 0.81 Count: 20
L_3	Average score: 3.11 Standard deviation: 1.39 Count: 19	Average score: 3.05 Standard deviation: 1.27 Count: 19

We note that the English-only and English + Kannada groups both scored lowest on language L_2. The English-only group performed better on language L_3 than L_1, but this difference is not statistically significant ($p > 0.66$). In contrast, the English + Kannada group performed much better on language L_1 than L_3 ($p < 0.04$). This initial finding suggests that the benefit of additional translations may depend on the problem being solved.

There is no significant difference in scores between the English-only and English + Kannada groups for languages L_2 and L_3, but for language L_1 the result is marginally significant ($p < 0.067$). We were somewhat surprised by this, so we analyzed the results more carefully for individual strings. Notice that strings in language L_1 satisfy the disjunction of two conditions (*w* starts with '01' or *w* ends with '10'). The string '010' satisfies both these conditions, and therefore belongs to the language. For the English-only group, 6 out of 9 volunteers answered this question correctly. In contrast, 28 out of 29 volunteers answered this question correctly in the English + Kannada group, a statistically significant difference ($p < 0.03$). This finding suggests that the availability of translations for certain words ("or" in this case) may be more important than wholesale translations. These findings need to be investigated more carefully, however.

We make one final observation regarding the "empty string", which belongs to the language L_3 since it contains an equal number (zero) of 0's and 1's, and has length zero which is at most 5. Based on our experience, we know that learners often find it difficult to reason correctly about the empty string, and we hypothesized that translating "empty string" into Kannada (or any other language) would not provide learners much help. Our results bear this out: 14 out of 19 volunteers in the English-only group answered this question correctly, whereas 12 out of 19 volunteers answered it correctly in the English + Kannada group ($p > 0.45$).

4. Discussion and Future Work

We are presently exploring ways in which to present translations to instructors and learners. As stated in Section 2, it is possible for our system to perform piecemeal translations. Our initial findings presented in Section 3 suggest that learners may find it helpful to view the entire question in the language of instruction (English, in our case) and request translations only for certain "tricky" portions of the question. Instructors, too, may find this facility useful in a computer-based examination environment. Here, translations can be made available to learners for a small penalty to ensure that they attempt to understand problems as originally presented, but do not get entirely stuck merely because they are struggling with the language.

Our automated translations are fairly rudimentary, and certain problems in our mathematical representation are poorly translated. For example, a tree whose root and several immediate descendants are **or**() functions represents a disjunction of several sub-properties. A human may translate this as: strings satisfying at least one of the properties P_1, P_2, …, P_n. Our translation will be more cumbersome with several nested parentheses that are unnecessary in this case, but are required to specify precedence in the presence of other functions such as **and**(). Our system would produce a translation of this form: strings that satisfy ((P_1) or (P_2 or P_3)) or (…). Our initial tests have revealed that learners find this form extremely difficult to follow. Therefore, we are considering ways to implement new types of translation rules that apply at a larger granularity than individual functions. In our example above, such a new rule would apply to the entire sub-structure of **or**() functions below the root of the tree.

We are also investigating ways in which our approach can extend to other problem domains (not just DFA construction problems). Note that our only requirement is to map a representative collection of domain problems to a suitably rich mathematical representation. From this representation, the same approach as ours can be used to translate problems into other natural languages. We believe that it may be possible to specify problems from other introductory courses in Computer Science in this way, and we are specifically examining CS1 and CS2 courses.

References

Chen, X., Acosta, S. and Barry, A.E. (2016). Evaluating the Accuracy of Google Translate for Diabetes Education Material, JMIR Diabetes 2016; 1(1):e3.

Kapur, D. and Ramamurti, R. (2001). India's emerging competitive advantage in services. The Academy of Management Executive, 15(2), 20–32.

Pitroda, S. (2009). National Knowledge Commission Report to the Nation 2006-2009. National Knowledge Commission, Government of India.

Qian, Y. and Lehman, J. D. (2016). Correlates of Success in Introductory Programming: A Study with Middle School Students. Journal of Education and Learning, 5(2), 73–83.

Rauchas, S., Rosman, B., Konidaris, G. and Sanders, I. (2006). Language Performance at High School and Success in First Year Computer Science. Proceedings of SIGCSE'06, 398–402.

Riemer, M. J. (2002). English and communication skills for the global engineer. Global Journal of Engineering Education, 6(1), 91–100.

Shenoy, V., Aparanji, U., Sripradha, K. and Kumar, V. (2016). Generating DFA Construction Problems Automatically. In Proceedings of the 4th International Conference on Learning and Teaching in Computing and Engineering (to appear).