Lojban Machine Translation

Zachary Nielsen
zen5195@rit.edu

Benjamin Cohen
bjc4213@rit.edu

Lee Avital
lja5234@rit.edu

1 Introduction

Our project is to create a machine translator for the synthetic language Lojban. Lojban is a language derived from the language Loglan, which was developed by linguists starting in the 1950s. The goal of Lojban overall is to enhance communication with other humans, as well as with computers. The language accomplishes this in many ways. First of all, every attempt is made in to remove all ambiguity from sentences, a notorious problem in many natural languages. For example, the English sentence "They are hunting dogs". This sentence is syntactically ambiguous as it is unclear whether there are people hunting dogs or the dogs being referred to hunt. Second of all, the language has no exceptions to any rules, unlike any natural languages. For example, in English we have rules like I before E, except after C, but these rules have **weird** exceptions. These two features make it both very hard and very easy to think of a formulaic way to go between Lojban and natural languages. On one hand, the rules of Lojban will always hold, so once the rules have been coded, no exceptions will have to be accounted for; on the other hand, in Lojban the meaning of a sentence is unambiguous, which makes translation into a language with ambiguity difficult. Our goal with this project is not to provide one hundred percent accurate descriptions of all Lojban sentences into English, but to be able to take simpler Lojban sentences and translate into non-ambiguous English as closely as possible. This is why our translator contains many options with some translations (ie this is a (curve/turn/bend) in that).

2 Lojban Structure

The core of the Lojban sentence is the bridi, which is simply a phrase that describes a relationship between two things. The simplest form of a bridi is one that consists of one selbri, a word that describes the relationship between things, and one or more gismu, which simply are words which describe objects. For example, in English the equivalent of a simple bridi is:

Joe	is the father of	Jill
subject	predicate	subject

. in Lojban is the father of would be considered the selbri, and joe, and jill would be gismu. The actual Lojban phrase for this is "Joe patfu Jill". It is important to note that in Lojban, since the selbri only describes the relationship of the gismu, the positioning of the selbri in or around the gismu is unimportant, the only thing that matters is the ordering of the gismu.

The unimportance of the positioning of the selbri inside the bridi seems like a negligible concern when programming for bridi that only contain a selbri and gismu, and indeed it is. Bridi do not have to contain only a selbri and gismu however; the positions that the gismu fill are called sumti, and are only required to describe some object, which can be done with another selbri. For example Joe is the father of that girl who talks to Jim uses the same predicate, the first argument remains "Joe "but we replaced the second argument with that girl who talks to jim. The Lojban phrase for this is Joe patfu

nixli tavla Jim; in this case it is unclear whether Joe is talking to Jim , or if Joe's daughter is talking to Jim without more in depth knowledge of Lojban. In this Lojban sentence, due to the fact that the left most selbri uses the closest sumti to it (unless otherwise indicated), this phrase actually means that Joe is talking to Jim, with Joe being the father of the girl. This example also shows how difficult it can be to avoid ambiguity when translating from Lojban to English.

These sentences can be modified to mean anything, using cmavo, which are nothing more than logical operators. The example given on page 95 of The Complete Lojban Language is about co, which inverts tanru; a tanru is a grouping of a modifier and an object.

ta blanu zdani	ta zdani co blanu
that is a blue type of house	that is a house of type blue
that is a blue house	that is a blue house

This obviously makes is it complicated to create a fluid English translation if two Lojban phrases can mean the same thing, like the example above. Thankfully even if we cannot quite get to the acceptable English translation, the psuedo-English translation still makes sense to native speakers of English, and it is reflective of exactly what the Lojban phrase is. This was a rather simple example, but many cmavo are related to grouping tanru, and modifying the relational structure of selbri. They even make parsing more difficult because cmavo are found near selbri and between tanru where one would hope to find the related word, but instead what is found is an operator that modifies meaning without having any visible presence in the English translation.

Cmavo are among a long list of things that our group is not fully committed to implementing. While they are extremely important to the meaning of Lojban sentences, and surely many ideas cannot be expressed without them, they pose a major parsing issue and there are other more important concepts of the language that need to be addressed, for example, selbri, gismu, and tanru, which can be used to express a massive amount of ideas without the complexity of cmavo added.

Lojban was designed to be culturally ambiguous, and not favor one grammatical style of one language over another. This is especially apparent in the phonetics of the language. Lojban, if one examines it carefully, has many consonant pairs that native speakers of English are not accustomed to, as well as many that native speakers are accustomed to. Also, the ability to place subjects anywhere around the predicate so long as the order remains the same and no cmavo modify meaning because of it makes it trivial to mimic the structure of English, Spanish, or Hebrew, for example. Especially important to note is that the meaning of the phrase will not be modified for the person interpreting the Lojban no matter their native language.

3 Resources

The Lojban community has already created a translator similar to the goal of this project which can be found at (http://www.Lojban.org/jboski/index.php). It takes a Lojban sentence and parses it into its logical form, and the output given is that logical Lojban directly translated into English. Observe when the input mi tavla do (i talk to you) is given, the response from the translator is

$[_1(_2$[tavla$_1$ (talk-er(s)) :] mi I, me$)_2$ [is, does] $_3$tavla talk-ing$_3$ ($_4$[tavla$_2$ (talked to thing(s)) :] do you$)_4]_1$

This is an interesting translation for multiple reasons, first of all it leaves the Lojban words inside of the translation, with their definitions attached to them. It also places parenthesis and other grouping symbols so that any meaning carried with groupings in the Lojban is not lost in the English, and helps new users of Lojban to understand how the language works.

While this translator is great for showing the inner workings of Lojban, and keeping the logical grouping together, it does not function as a plain

Lojban to English translator. To interpret these translations one must observe all of the possible translations of each word, for example, I/me, and the translator does not always include all of the correct articles; an equivalent sentence in English could be I am talking to you, but the am article is not included. Another issue with these translations is the use of grouping symbols. The translation uses nested parenthetical grouping symbols to relate certain entities to each other. To interpret the translation, the reader first has to figure out how to interpret the grouping symbols. Unless one desires to understand the details of how Lojban conveys meaning, this translation is quite a burden. Nevertheless, this translator was very useful to us in testing our programs functionality.

4 Our project

Our teams goal is to create a similar translator where perfect English is not returned, but the translation is still fully intelligible to a native English speaker. To illustrate this simply, we say our project "translates Lojban to cave man." Unlike the translator at at Lojban.org, however, we want to provide the user with output that does not have to be analyzed before they can gather any sort of meaning. The end goal of this project is to provide the user with output that can be immediately understood without the need of any deep analytical thought.

One of the obstacles for this project was the fact that none of the members of the team are considered experts on Lojban, nor can they be assessed as proficient in it. Therefore we will be relying on the book The Complete Lojban Language by John Woldemar Cowan. This book is published by the Logical Language Group, and is considered the Lojban bible. Following the logical progression of the book, which teaches the reader the language of Lojban in a methodical way that starts with simple statements then fully fleshes out each of the complexities of the language with great detail, we plan to implement general rules first, then add in more subtle and obscure rules.

For our implementation we are planned on using the pyparsing package to parse and lexically analyze the Lojban sentences. Pyparsing allows a programmer to define rules about how a sentence is parsed, and into what entities the sentence is parsed. For example, to parse a simple bridi sentence with three arguments we used four different rules to account for the four placements for the selbri. (Recall that a bridi with three arguments consists of three sumti and one selbri).

When we tried to parse one such bridi sentence (mi vecnu ta ti) we got a list with four elements–one for each word. When we used a sentence with a nested bridi (mi vecnu ta mi vecnu ta ti) we got a list with four elements, one of which was nested list. The nested list was the nested bridi (mi vecnu ta ti).

To use pyparsing, we had to create several rules or constraints to correctly parse the sentence. To do this, we used an extended Backus-Naur representation. Backus-Naur representations, originally invented by computer scientists John Backus and Peter Naur to represent non-ambiguous programming languages, are a useful construct in representing Lojban.

After toiling with a handful of the rules of Lojban and trying to implement them in pyparsing we knew that we were in much over our heads and searched around for other solutions. Fairly quickly we came across an open source python parser for Lojban that returns a tree of words with their parts of speech. This has become the key to our current success in the project, because with this tree we can hard-code rules for some of the more common cmavo that perform mechanical operations on the meaning of the phrase by manipulating the tree structure.

The current stage of our project is this: we can translate basic bridi into intelligible English with most common cmavo that perform mechanical operations as well as the cmavo that (mean things like I, that, this). Currently tanru, phrases that describe a complex object pose a problem to our system. The root cause of this is that tanru are gismu, and in turn our system wants to translate tanru not as one word describing the other, but as one relationship modifying another with no

arguments. All of the translations that succeed have resulted in intelligible, while not grammatically correct, English.

An interesting side effect of our implementation method is we have immense scalability and upgradeability. Since we read all of our dictionaries from text files when the program is launched, any changes or additions to the Lojban dictionary get immediately reflected in our program. This is why our project would also work if given a dictionary from Lojban to another language. This proved very useful in testing, as we could load in parts of the dictionary at once, and test specific rules and sentences without using the whole language.

Lojban has very few corpuses to work from, indeed some of the only corpuses are Alice in Wonderland by Lewis Caroll, the Christian Bible, and The Declaration of Independence. Obviously this would prove to be a miniscule corpus to base a statistical machine translation off of. Also, if one looks into Alice in Wonderland, the Lojban is incredibly complex and precise, which while in the domain of literature is important to convey complex and precise ideas, isnt as important as brevity or clarity in spoken language. This makes the best corpus available the text files that is the dictionary which describes all of the words in the Lojban language. The benefit of this is that the exact same dictionary is supported in Japanese, Spanish, and Esperanto. Spanish, however, is the only language that particles are are defined for other than English. This means that we could easily port our translator in its current state to translate from Lojban to Spanish, and we could translate a vast portion of Lojban sentences to Japanese or Esperanto. That is the real power of our Lojban translator, the ability for nearly identical code to be used for any language with a dictionary defined in the same format as the English dictionary.

In short, while our translator does not give output for long, complex sentences, it excels at giving readable translations of simple sentences. It couldnt translate a book, but would be a very useful tool in getting the gist of a conversation, or looking up a word. As proof of success, our translator will get most of the simple phrases outlined at http://www.lojban.org/tiki/tiki-index.php?page=Simple+phrases correct.

5 Future Work

As of now our translator works for most simple sentences, and the obvious next step is to make it work on more complex sentences, such as ones that contain recursion. As example, our translator currently works on the Lojban phrase meaning I sell this to you, but not the phrase I sell the thing that you sell to him, as that has two levels of gismu.
Another logical step is to make our translator output in other languages. This could be implemented rather easily if we were to have dictionaries from Lojban to these languages like we had for English. This really outlines the strength and utility of the language.
Another thing that would improve our translator would be to add in more support for modifiers. For example, we support the modifier se which inverts two arguments to a selbri, but we dont support modifiers like color. The reason for this is that we were unable to find an online list of these formatted in a way that we could process it, and time did not allow us to hand encode all of these modifiers.
Lastly, future work would likely include another means of communicating with our translator. Perhaps we would put it up on the web, or develop an API so that one does not need to run our code on their machine, but simply send a request to a server or go to a website to get a translation.

References

Alfred V. Aho and Jeffrey D. Ullman. 1972. *The Theory of Parsing, Translation and Compiling*, volume 1. Prentice-Hall, Englewood Cliffs, NJ.

American Psychological Association. 1983. *Publications Manual*. American Psychological Association, Washington, DC.

Association for Computing Machinery. 1983. *Computing Reviews*, 24(11):503–512.

Ashok K. Chandra, Dexter C. Kozen, and Larry J. Stockmeyer. 1981. Alternation. *Journal of the Association for Computing Machinery*, 28(1):114–133.

Dan Gusfield. 1997. *Algorithms on Strings, Trees and Sequences*. Cambridge University Press, Cambridge, UK.