LojLink:
A Novel Approach to Knowledge Management

Ben Goertzel
March 10, 2005

1 Introduction

We present here a plan for a novel knowledge management system called LojLink, which has the following properties:

- Knowledge encoders, once they undergo an extensive period of training, will be able to use the system to *rapidly* enter large amounts of arbitrarily complex, ambiguous, unstructured knowledge into a computer database
- The system will be able to automatically ingest simple information from free text in an unsupervised way (in English; or, with some additional work building linguistic preprocessing tools, in other languages as well)
- The knowledge, once entered, can be easily reasoned on using AI techniques
- Individuals with only a small amount of training can query this database using simple English queries and receive comprehensible results

Some of the terms used here require clarification:

- "an extensive period of training": we estimate that the first batch of knowledge encoders will require roughly 6 months of intensive training, whereas subsequent batches will require roughly 4 months of intensive training
- "rapidly enter … knowledge": we estimate that, for a trained user, the time required to enter a sentence worth of knowledge into this system should be roughly 3-5 times as long as the time required to type a sentence into a word processor.
- "a small amount of training": On the other hand, to train users for querying, a single day of training should be sufficient.
- "easily reasoned on": Reasoned on with roughly the same effectiveness as automated reasoning currently achieves on databases of predicate-logic relations
- "comprehensible results": Results will be displayed in an awkward dialect of computer-generated English, which will be comprehensible but not elegant, and will need human attention to be translated into text suitable for inclusion in reports

The downside of this plan is obvious: the extensive period of training for the knowledge encoders. Unfortunately, however, there seems no way to get around this point. This is the price one has to pay to get the other advantages, given the current state of technology in AI and computational linguistics.

2 Comparison with INLINK

It may be useful to draw a comparison between LojLink and the INLINK framework (Goertzel et al, 2005) currently under development at Novamente LLC and Object Sciences, in which knowledge encoders interact with a user interface to enter English sentences.

The most important point is that knowledge entered using LojLink will be much more amenable to automated reasoning (which may be done using many tools including the Novamente AI system under development by the author and his colleagues; see Looks et al, 2004). This will result in significantly more intelligent inferential conclusions being drawn by the LojLink system than by the INLINK system. This advantage is difficult to quantify, but we anticipate it will be a very big one, for reasons that will become clear a little later in this document.

A smaller but still very important point has to do with the speed of knowledge entry – which will be much faster with LojLink than with INLINK.

We anticipate that in a future version of INLINK, with a more refined interface, it will be possible to enter knowledge at a rate of roughly 5 minutes per English sentence. There is a tendency to use simple sentences with INLINK, so this probably amounts to about 10 minutes per average-complexity English sentence. On the other hand, with the proposed LojLink system, we estimate that knowledge entry will occur at a rate more like 1 minute per average-complexity English sentence: a speedup of 10 times. Furthermore, the proposed LojLink system will be much more enjoyable for trained knowledge encoders to use, whereas the use of INLINK will always be somewhat frustrating for the encoder.

Figuring 30 sentences per page, these estimates mean that INLINK encoders would encode about 1 page of information per day, whereas LojLink encoders would encode about 10 pages of information per day.

As an aside, getting the INLINK system to the point where encoding takes 5 minutes per complex sentence probably involves about 2.5 man-years of work, from this point. On the other hand, creating the new proposed framework probably involves about 2 man-years of work. On the other hand, making automated reasoning work well with INLINK requires the same work required to make it work well with LojLink, plus a significant amount of extra work – the amount of which is difficult to estimate, because it increases depending on how much inferential intelligence one requires.

3 The Basic Idea

OK, now that the advantages and disadvantages of the proposed new approach have been outlined – what is the new approach, exactly?

Put simply, the idea is to have knowledge encoders do encoding using the Lojban language (Cowan, 1998; Nicholas and Cowan, 2003), instead of the English language. (More on the Lojban language in the following section.)

Querying would be done in English, using an improvement of the current INLINK interface; and query results would be returned using English produced by machine-translation from Lojban to English. Query results would thus appear in somewhat awkward English, but would be comprehensible enough.

Results of automated, unsupervised mining of English texts would be automatically translated into Lojban and stored in the knowledge base in Lojban form.

Note that none of this involves automated translation of English into Lojban, which is a very hard problem – it only involves easier problems such as

- Translation of simple English queries into Lojban (simple queries are a very restricted subset of English)
- Sloppy translation of Lojban into English (which is far easier than translating the other way around)
- Translation of semantic information mined from text into Lojban (which is fairly easy because semantic patterns mined from text have a simple form that is easy to translate into Lojban expressions)

4 What Is Lojban?

Lojban is an intentionally-constructed human language whose semantics is explicitly based on formal logic.

One of the design goals in the creation of Lojban was to create a language with the minimum plausible amount of ambiguity. This minimal amount is not zero, which is why Lojban is not equivalent to formal logic. Communication using formal logic would be unrealistically cumbersome for human beings, and perhaps for AI systems as well. The construction of ambiguous expressions, together with the implicit assumption that the listeners/readers will be able to disambiguate based on shared contextual knowledge, is essential to the nature of language itself. But it's clear that all human languages possess a much larger amount of ambiguity than is necessary to achieve compact communication to context-savvy listeners/readers. Lojban appears to come close to the necessary minimum. For instance, each Lojban word has only one meaning (no lexical-level semantic ambiguity), and Lojban syntax is completely specified by a tractably-sized formal grammar.

One of the motivations underlying the creation of Lojban was to make communication between humans and computers easier. The conceptual basis here seems quite clear: computers deal well with formal logic and with unambiguous knowledge representations generally, so a language which is logic-like and relatively unambiguous should be particularly amenable to computational processing.

So far Lojban has not been used within AI research at all, so far as I know; and it's worth asking why. Part of the reason for this is simple lack of familiarity. Part of it is the practical, application-driven nature of much current AI research (in both academia and industry): it's a lot easier for funding sources to see the value of making AI systems understand languages with more than a few hundred speakers! And part of the reason, I believe, is the lack of any professional-quality computational linguistics resources for Lojban.

In (Goertzel, 2005) I have outlined a detailed plan for creating Lojban computational linguistics tools and resources, similar to resources that now exist for English. These tools and resources fall into two categories:

- Those aimed at making it easier to parse Lojban text into semantic relationships inside an AI system, or to generate Lojban text from semantic relationships inside an AI system
- Those aimed at automating translation between English and Lojban

The conclusion of that document is that the creation of these tools and resources will involve roughly two man-years of effort. About 10 months is for the Lojban-only work, and the rest for translation-oriented work. While this may seem like a lot of work, it's remarkably little compared to what would be necessary to do a similar job for any natural human language (or for a constructed language like Esperanto, which lacks Lojban's logical foundation).

The basic strength and weakness of the LojLink proposal should now be clear to the reader. The downside is that knowledge encoders need to learn a new language. Lojban is easier to learn than ordinary languages, but still requires several months of intensive study. On the other hand, once this initial obstacle is overcome, then all of a sudden everything else becomes simple: encoding is fast, automated reasoning is easy, and complex queries to the database will get intelligent answers. The price is big but so is the benefit.

5 LojLink

Finally, in this section we give some more detailed comments on the proposed LojLink system.

5.1 LojLink for Knowledge Entry

INLINK, as noted above, is a software system that interacts with the user to help the user enter English language sentences into a database, in a way that ensures the language processing software has correctly interpreted the sentences. A simplified and modified version of INLINK's knowledge encoding framework may be useful for Lojban.

Much of what the English INLINK knowledge encoding system does is not needed for Lojban. For instance, word sense disambiguation is not an issue for Lojban, and nor is parse selection a major problem (since Lojban parsing is straightforward). However, there are still significant portions of INLINK functionality that will be very useful as aids for communicating with computers using Lojban.

For one thing, humans may not always use Lojban correctly. LojLink could show the user the normalized Lojban version of his input, which could then quickly be stupidity-checked, to be sure that no grammar mistakes were made.

Next, realistically the user is not likely to know all Lojban vocabulary by heart. For this purpose, the Lojban-WordNet mapping described in (Goertzel, 2005) should be very useful. The user will be able to browse WordNet to find the concept he wants, and then see if there is any Lojban word corresponding to this concept or any other closely related concepts. Also, he can do a keyword search of the Lojban-English dictionary. If none of these yields results, then he can optionally create a new Lojban word and store it in the LojLink database for other users to see.

There is also the issue of reference resolution. Lojban reference resolution is simpler than in natural languages but it's still far from unambiguous. INLINK allows the user to explicitly specify referents for words, which allows unambiguous understanding of interreferential sentences, and also gives AI systems data from which to learn how to do reference resolution in a fully automated way.

Finally, LojLink should be a valuable resource for correcting errors in the Lojban-English translation process. The translator, if built along the general principles described in (Goertzel, 2005), will often output several possible translations for a given Lojban sentence. The user will be able to rank these in the LojLink knowledge encoding interface, and this feedback will allow both human and automated improvements to the translator.

5.2 Query and Response

Suppose one has built up a database of knowledge using automated inference on knowledge entered in LojLink. How then can one ask questions of this database?

Of course one can ask questions in Lojban, but that means one has to be fluent in Lojban to get information out of the database. Fortunately this isn't the only option.

Translation from general English sentences into Lojban is a hard problem, and is basically equivalent to full automated comprehension of English. Fortunately, however, this isn't really needed for LojLink, which can get by with more limited English/Lojban translation.

For querying, the current INLINK framework can be used (with an improved user interface). This means that submitting a query may take several minutes, but, this is acceptable because the main intended usage pattern of this kind of system is where users have a fairly small number of topics of interest, and they want to get an in-depth exploration of these issues. It's much more important that knowledge entry be fast than that querying be fast.

Inside LojLink, a query will be translated into the same logical expressions that Lojban is mapped into, so it can readily be matched against the knowledge in the database. Of course, many kinds of "matching" can be done here, ranging from the simple (dynamic programming query matching, as is implemented in the current INLINK system) to the highly advanced (inference-based query analysis, as would be enabled e.g. by plugging the Novamente AI Engine's higher-order inference module into LogLink).

How about query results? Well, producing results in Lojban is very simple, but this isn't going to be of much use to non-Lojban-speaking users. So clearly what's required is some Lojban-to-English translation. But the key is that this translation doesn't need to be perfect. It just needs to be comprehensible. This kind of Lojban-to-English translation will be easy to achieve once we have the Lojban computational linguistics resources described in (Goertzel, 2005). As an example of the type of translation I mean, consider the Lojban sentence

```
la Mark cu tavla be do bei le melbi ku vecnu
```

This can be almost immediately translated into

```
Mark is a (talker to you about beautiful things) salesperson
```

based Lojban grammar and the dictionary equivalences

```
tavla = talk
vecnu = sales
do = you
melbi  = beautiful
```

which is a crude but comprehensible translation. A collection of English syntax rules could be implemented in a specialized way to translate this sort of translation into nicer translations like

```
Mark is a salesperson who talks to you about beautiful things.
```

But even without this additional step of prettification, results will be comprehensible

5.3 Automated Inference and Semantic Transformation

Finally we come to the most important point – automated inference on the knowledge entered into the LojLink system.

In recent experimentation with the Novamente AI system, we have achieved quite interesting results doing probabilistic inference on commonsense knowledge. But this commonsense knowledge is presented to the system in formal-logic-type form. Appendix 2 at the end of this paper gives a simple example.

In order to reason on knowledge entered by humans via an interface like INLINK, it's necessary to translate the linguistic knowledge into "reasoning-friendly form." Some of the difficulties involved in this process are described in (Goertzel et al, 2005). Initial semantic analysis of English sentences results in logic expressions that capture sentence meaning, but in a format that is very awkward for logical inference. Transformations must then be applied to put these logic expressions in a form more easily amenable to inferential manipulation. In the human mind/brain such transformations are learned; but in a software system without embodiment and experiential interaction with other minds in a shared world, these transformations must be programmed explicitly. This is certainly not an unmanageable task, but it's a large task that will never be truly complete: there will always be more transformation required to deal with more peculiar English language constructions.

In the LojLink case, on the other hand, these difficulties are substantially diminished, because the output of Lojban parsing is basically immediately in inference-friendly form. To give a very simple example, consider the sentence

```
"Mark is Brad's friend."
```

The output of INLINK's semantic analysis of this sentence, in the current version, is a collection of predicate-argument relationships of the form

```
Inheritance(B,be)
Tense(B,%pres_ongoing)
objTARGET2(B,F)
subjDESCRIPTEE(B,O)
Inheritance(B1,Mark)
Inheritance(F,friend)
possFOCUS2(F,B1)
```

```
Inheritance(O,Brad)
```

For inference purposes, what one wants is something more like

```
friend_of(Mark,Brad)
```

(which is the form used in the example in the Appendix). Getting from the former to the latter requires applying a set of semantic transformation rules as outlined in (Goertzel et al, 2005) – a process that is moderately complicated, though definitely workable.

On the other hand, the semantically equivalent Lojban sentence

```
Mark pendo Brad
```

immediately translates into

```
pendo(Mark, Brad)
```

Given the Lojban dictionary mapping from "pendo" to "friend", this immediately maps into

```
friend_predicate(Mark, Brad)
```

which allows matching with English-language queries as well as being highly amenable to automated inference.

Not all examples are this simple, of course. Appendix 1 briefly explores some of the issues that arise when Lojban utterances contain significant ambiguity, and the ways this can be worked around using a modest amount of AI on the semantic-interpretation side.

References

- Cowan, John Woldemar (1998). *The Complete Lojban Language*, The Logical Language Group
- Goertzel, Ben (2005). *Potential Computational Linguistics Resources for Lojban*
- Goertzel, Ben, Moshe Looks, Michael Ross, Cassio Pennachin and Hugo Pinto (2005). *NL Comprehension via Integrative AI and Human-Computer Interaction*, submitted for publication
- Looks, Moshe, Ben Goertzel and Cassio Pennachin (2004). Novamente: An Integrative Architecture for Artificial General Intelligence. *Proceedings of AAAI Symposium on Achieving Human-Level Intelligence through Integrated Systems and Research*, Washington DC, August 2004
- Nicholas, Nick and John Cowan (2003). *What Is Lojban?* , The Logical Language Group

Appendix 1: Subtleties of Lojban Semantic Mapping

Here we briefly consider some of the trickier aspects of automated semantic interpretation of Lojban expressions. While Lojban is much closer to logic than English, it still isn't entirely formally specified – which is what makes it usable for human communication, of course. This means that the "semantic transformation" problems existing in inference on English-encoded knowledge still exist for Lojban, they're just dramatically decreased.

For example, in order to support simple automated reasoning, we would like to map the sentence presented above

```
la Mark cu tavla be do bei le melbi ku vecnu
```

into predicates of the form

```
Inheritance(Mark, $X)
Inheritance($X, le vecnu)
Inheritance($X, $Y)
Inheritance($Y, le tavla)
Patient($Y, do)
Topic($Y, le melbi)
```

All of these relationships can be immediately read from the sentence, using Lojban grammar – except for the fourth one, which asserts that Mark is a talker.

There is some real subtlety here, which has to do with the compound phrase

```
tavla be do bei le melbi ku vecnu
```

which is an expansion of the simpler compound phrase

```
tavla vecnu
```

which interpreted most directly just means "talk sell". This kind of Lojban compound phrase is called a "tanru," and its meaning is usually metaphorical rather than direct.

The semantic mapping given above can be obtained immediately if one uses the default heuristic that: If an item is an argument of a tanru, sometimes it's an argument of each element in the tanru as well. In Novamente node and link language this heuristic would be written as follows:

```
Implication <s>
      AND
```

```
        Inheritance($A, $X)
        TanruLink($X, ($Q,$R))
        PhraseSenseLink($W_X, $X)
        PhraseSenseLink($W_Q, $Q)
        PhraseSenseLink($W_R, $R)
    AND
        Inheritance($A, $Q)
        Inheritance($A, $R)
```

Note however that this heuristic is not always true by any means (hence the probabilistic truth value, $s < 1$). A simple counterexample is the tanru "nixli ckule" – girl school. If we say that

```
Janis cu le ve nixli ckule
```

i.e.

```
Janis attends a girls' school
```

then we are saying that

```
Janis attends a school
```

but not

```
Janis attends a girl
```

In this case, the correct interpretation of the tanru requires some commonsense reasoning based on background knowledge.

First of all, it doesn't make sense for Janis to attend a girl. So even if we applied the above heuristic and derived "Janis attends a girl" as a consequence, this conclusion would have a lower probabilistic strength than the commonsense knowledge "Janis does not attend a girl", and would result in the interpretation "Janis attends a girl" getting a very low strength. So there's not much risk of a reasonably intelligent AI system getting misled into a wrong interpretation, in this particular case

But what we really need is for our AI system to automatically infer that "nixli ckule" probably refers to a school that girls attend. Fortunately this doesn't seem to be a very hard task given the simplicity of the Lojban language. The simplest questions the AI system may ask are whether any of the following hold:

- nixli ckule (nixli goes in the first argument of ckule)
- nixli se ckule (… the 2n'd argument…)
- nixli te ckule (...3'rd…)
- nixli ve ckule (...4'th…)
- nixli xe ckule (...5'th…)

As it happens the meanings of the arguments of the Lojban predicate ckule are:

1. The school itself
2. What type of school
3. What's taught there
4. Who attends there
5. Who operates it

The only ones of these that "girl" could sensibly fit into are the 4'th and 5'th argument positions. Thus, likely meanings for the tanru "ckule nixli" are "school attended by girls" or "school operated by girls." To disambiguate between these requires further commonsense knowledge, which is that girls are children and children more often attend than operate schools.

Of course, this kind of problem could be worked around by using more precise phrasing. "A school attended by girls" can be stated as (among other possible phrasings)

```
le se xe ckule be nixli
```

or simpler and nicer)

```
le ckule be fo nixli
```

(the "fo" compactly marking nixli as going into the fourth argument-slot). And "a person who is a salesman and who talks to you beautifully about the things he is selling as well as sometimes other things" could be similarly reformulated, in a more complicated and loquacious way.

But much of the point of using Lojban instead of encoding knowledge in predicate logic directly is that one doesn't need to be completely precise – Lojban allows communicational ambiguity when appropriate and convenient, but doesn't introduce additional needless ambiguity like natural languages do. Handling the ambiguity that does exist in Lojban – which is almost entirely in the form of tanru such as the above – requires some semantic transformation rules and heuristics, but these are an order of magnitude simpler than the corresponding rules and heuristics required to do English semantic mapping.

Appendix 2: An Example of Novamente Inference on Commonsense Knowledge

In this Appendix we run through the inference steps that the Novamente AI Engine's PTL (Probabilistic Term Logic) inference component would use to answer the question "Are the two individuals Mark and Brad friends?" based on a database of other related knowledge, which includes the knowledge that both Mark and Brad are friends with James, and that friendship is sometimes transitive. While we have not given any real background on Novamente or PTL in this article nevertheless the spirit of PTL inference may be perceptible from the general flow of the inference steps in the example.

From a Lojban perspective, what this example illustrates is the kind of inference that can be done in an AI system once it has been fed knowledge in predicate logic form. The power of Lojban from an AI perspective is that it provides a relatively easy way for humans to put their commonsense knowledge in this form, so that AI's can easily reason on it.

A key point about PTL inference is that it handles uncertainty gracefully (hence the "probabilistic" part). This is critical in doing reasoning based on Lojban-encoded knowledge, because Lojban is not entirely unambiguous, so knowledge derived from ambiguous Lojbanic constructs like tanru is going to have a certain amount of uncertainty attached to it. In the following inference example, we'll see how PTL propagates uncertainty from conclusions to premises during the reasoning process. Uncertainty is represented in the notation used below via pairs like

```
(0.65,0.72)
```

which refers to "probability .65, based on evidential weight .72." Evidential weights are scaled into [0,1] where 1 means the associated probability is based on an infinite amount of evidence.

In this example, queries to Novamente are represented by the syntax "findtv R" where R is a relationship and Novamente is asked to evaluate the truth value of R; or the syntax "findExamples R" where R is a relationship involving variables and Novamente is asked to find values for the variables that will make the relationship as true as possible.

We begin with the findtv query

- `findtv friendOf(Mark,Brad)`

The inference control heuristic then searches in the knowledge base to find what relationships are known involving the terms in the query. It first finds the following relationship involving friendOf, which indicates the (probabilistic, not certain) symmetry of the friendOf relationship:

- `friendOf is symmetricRelation (0.65,0.72)`

The "equivalence to implication conversion" inference rule yields

- `if  R000 is symmetricRelation then if and only if R000(X007,X008) then`
- `R000(X008,X007) (0.995,0.985)`

PTL then applies this definition to the relationship friendOf, which involves a step of the variable-instantiation rule and a step of the deduction rule:

- `if friendOf is symmetricRelation then if and only if friendOf(X007,X008) then friendOf(X008,X007) (0.995, 0.985)`
- `if and only if friendOf(X007,X008) then friendOf(X008,X007) (0.654, 0.713)`

Now it applies a step of variable instantiation, seeking to match the new link it has learned regarding friendOf with the other terms that were there in the original query.

- `if and only if friendOf(Brad,Mark) then friendOf(Mark,Brad) (0.654,0.713)`

This gives it a way to assess the truth value of friendOf(Mark, Brad). Namely: it now realizes that, since friendOf is symmetric, it suffices to evaluate friendOf(Brad, Mark). Thus it submits a findtv query of its own creation.

- `The truth value of friendOf(Mark, Brad) is unknown:`
- `findtv friendOf(Brad, Mark)`

But unfortunately, this query finds no answer in the system's knowledge base. The exploitation of the symmetry of friendOf was a dead end, and the inference heuristic must now backtrack and try something else. Going back to the start, it looks for another relationship involving the terms in the original query, and finds this one, which indicates that friendOf is transitive:

- `friendOf is transitiveRelation (0.4, 0.8)`
- `if and only if R001 is transitiveRelation then if exists X010 such that AND(R001(X009,X010),R000(X010,X011)) then R001(X009,X011) (0.99,0.99)`
- `if R001 is transitiveRelation then if exists X010 such that AND(R001(X009,X010),R000(X010,X011)) then R001(X009,X011) (0.995,0.985)`
- `if friendOf is transitiveRelation then if exists X010 such that AND(friendOf(Mark,X010),friendOf(X010,Brad)) then friendOf(Mark, Brad)(0.995,0.985)`
- `if exists X010 such that AND(friendOf(Mark,X010),friendOf(X010,Brad)) then friendOf(Mark, Brad) (0.410, 0.808)`

In this case, the system is led to a findExamples query rather than a findtv query:

- `findExamples AND(friendOf(Mark,X010),friendOf(X010,Brad))`

This query finds an answer in the knowledge base:

- `X010={James}`

where

- `friendOf(Mark, James) (0.8, 0.3)`
- `friendOf(James, Brad) (0.7, 0.6)`

To resolve this find query, it must use the truth value formula for the AND operator

- `AND(friendOf(Mark, James),friendOf(James,Brad)) (0.56,0.3)`

Since this evaluation yields a reasonably high truth value for the find query, the system decides it can plug in the variable assignment

- `X010={James}`

And now the system can use the transitivity of friendOf to assess the degree to which Mark is a friend of Brad:

- `friendOf(Mark, Brad) (0.238, 0.103)`