Conversational Semantic Standards

Kurt Fuqua, Cambridge Mobile
kurt@cambridgemobile.com

Abstract Conversation involves processing natural-language at the conceptual level. Conversational applications must share high-level information across apps. To accomplish this in real-world applications, there is a need for a standard formalism to represent natural-language statements and concepts. A semantic representation also opens the door to creating applications that can be readily adapted for multiple natural languages.

In a conversational system, there must be a unified front-end. In stand-alone applications, the application assumes a *monopoly* on the speech resources. It alone controls the speech recognizer, the parser, the lexicon, and the speech synthesizer. But, in a conversational application, all of these resources must be *shared*. The syntactic grammar must be an aggregate of each application's syntax requirements. No application performs its own morphology. The speech recognition is under control of the front-end. The front-end must processes each sentence. Then, based upon the semantics of the statement, it must route the sentence to the appropriate application.

Semantic Routing

Routing cannot be done by keywords. Routing based upon matching syntax fragments fails. Such techniques are far too superficial for conversation. The only reliable means for routing is to examine the semantics of the statements. Semantic routing presumes that there is a standardized representation of the semantics. The applications must convey to the front-end what semantic statements the app is capable of processing. The front-end must sort out what semantics the statement contains. Both sides must use the identical semantic representation.

Processing Steps

During analysis by the front-end, the sentence is first parsed. The resulting parse tree is then translated into a semantic representation of the sentence. The representation must richly encompass the full meaning of the sentence, and the translation must be done automatically. This will require full morphologic decomposition and therefore the lexicon must support grammatical feature tags. The lexicon entries must also be given an interlingual semantic tag. The tagging must be consistent across languages and applications.

Conversational applications must frequently resolve anaphoric references from earlier portions of the conversation. The front-end must use the parse trees to resolve these to the semantic level.

The applications must make available to the front-end, the semantic expressions that it is capable of handling. The front-end then can inference the user's statements against each application's capabilities in order to find a match for the routing.

Within the app, inferencing becomes far simpler if the app is supplied with an inferenceable semantic expression. This functionality radically improves what a typical mobile application developer is capable of providing. With the right libraries and tools, a developer can directly manipulate semantics. This cannot be said of the equivalent natural-language text. Tasks that were nearly intractable become trivial.

During synthesis the application can directly compose replies in semantic representation. This will then be translated into the natural-language parse tree and rendered (morphologically and phonologically). Why compose in semantics rather than in a specific natural language? If the app composes in semantics, the application can be language-independent. This is a dramatic step. The application can be readily adapted to multiple languages with little effort by the app developer.

Requirements

So then, what are the requirements of a semantic representation? It must represent a broad range of common natural-language constructs including interrogatives, imperatives and declarative statements. It must represent verbs well including valency, aspect, tense and negation. There must complete coverage for the range of all the closed-class categories such as pronouns, prepositions, conjunctions and determiners. It must represent the open class categories of nouns, adjectives and adverbs. The defined vocabulary must be sufficiently large that it includes all of the close classes and a large portion of the open class terms. The notation must be language- independent and avoid a strong bias toward any one language. There should be little or no morphology in the notation itself. Most important of all, inferencing must be possible with simple manipulation of the semantic expressions.

InterLing

InterLing is a moderately-large-vocabulary commercial interlingua which is a component of the Scalable Language API. It is a higher-order intentional predicate logic loosely based upon that of Richard Montague. Like Montague's formalism, it is strongly typed however the lambda calculus model has been simplified making it easier to read and manipulate. Unlike Montague's academic formalism, InterLing has a comprehensive defined vocabulary. The remainder of the paper discusses the feature of InterLing and its use in conversational natural-language applications.

The InterLing has both a textual and a binary representation; the two are logically equivalent. The examples will use the textual form but applications will generally communicate the binary form. The binary form is extremely compact and efficient. Verbs, adjectives, and adverbs are represented as predicates. Nouns are terms. Variables are used to link together predicates.

Here is an example:

When can he meet me?
When(y) & TPres(y) & MInd(y) & MoCan(y) & VTrans(y, Vmeet, z, w) & PI(w) & Phe(z)

There are 7 predicates linked with the '&' symbol; the order is unimportant. At the center of the expression is the verbal frame VTrans(). This frame containing a transitive verb and each of its related verb arguments; there are 4. The first parameter is a variable (y) used to index the complete NL predicate phrase. The second is the InterLing verb term itself. The third (z) represents the subject of the verb (he). The forth represents the direct object of the verb (me = I).

The expression in essence reads, when is y such that y represents z meeting w. Z is 'he'; w is me/I. Note that since InterLing has no morphology the pronoun 'me' is equivalent to 'I'; pronouns are not inflected for case. The variable y ties the predicate phrase (he meet me) to the tense, aspect and sentential adverbials. In this sentence, y is in the present tense (TPres), the indicative mood (Mind). The main verb is also modified by the modal 'can' (MoCan). It is not rocket science but it takes a bit to get used to reading it.

When is my meeting?
When(y) & TPres(x) & MInd(x) & VCopuV(x, Vbe, z, y) & Is(z, Nmeeting) & Singl(z) & Pmy(z)

Here, the verbal frame is not a transitive but is instead a copula which takes an adverbial (y). Variable z represents 'my' singular meeting.

When is my meeting with Chris?
When(y) & TPres(y) & MInd(y) & VCopuV(y, Vbe, z, w) & Prep(w, pMwith, Chris) & Is(z, Nmeeting) & Singl(z) & Pmy(z)

Here, the adverbial is represented by the variable 'w' which is an index to a prepositional frame. The InterLing preposition is 'pMwith' (preposition-of means-with). The object of the prep is 'Chris'.

When is my meeting on Friday?
When(y) & TPres(y) & MInd(y) & VCopuV(y, Vbe, z, w) & Prep(w, pTPon, DefTrn(DwFriday)) & Is(z, Nmeeting) & Singl(z) & Pmy(z)

Now the preposition is 'pTPon' (preposition-temporal-position-on). The prepositional object is the definite time range (DefTrn) of Friday i.e. 'this Friday'.

Valency, Aspect and Tense
English uses 10 verbal valencies. Three additional valencies have been identified in other world languages. Altogether 13 valencies are supported. Each valency is supported by a distinct verbal frame. This systematic framework for verbal valency tremendously simplifies the inferencing.

Core Vocabulary

In natural languages, prepositions are highly polysemic. The English preposition 'over' has at least 8 distinct meanings. In InterLing, there is a distinct InterLing preposition for each semantic sense. During analysis, the system is forced to resolve the sense used. InterLing contains a large inventory of distinct prepositions. The distinct prepositional senses were determined through a careful analysis of the prepositions of several languages. While the vocabulary does not claim to be exhaustive, the approach used does result is a fairly comprehensive vocabulary.

The same approach was taken for determining InterLing pronouns, conjunctions, determiners, and, of course, a proper treatment of quantifiers. Word frequency lists were used to compile the most common adverbs, adjectives, and verbs. The vocabulary is continually expanding.

Practical Example

Let's step through a simple practical example using Lingua. The user asks a question.

When is my meeting with Chris on Friday?

This sentence is parsed by the front-end and translated into Lingua.

When(y) & TPres(y) & MInd(y) & VCopuV(y, Vbe, z, w) & Prep(w, pMwith, Chris) & Is(z, Nmeeting) & Singl(z) & Pmy(z) & Prep(w, pTPon, DefTrn(DwFriday))

The front-end compares the semantics of the user's sentence against the semantics submitted by the 6 different loaded conversational applications and selects the Scheduler App. It sends the InterLing expression (in binary form) to that app.

The Scheduler App, using simple inferencing rules, maps the InterLing expression to its internal function in order to find the requested appointment. The InterLing terms are passed directly to that function.

Result = FindAppt(Chris, DefTrn(DwFriday), *)
Result = June 25, 2010 15:00

The function finds every appointment with Chris on this Friday and returns a binary timestamp. The timestamp is then fed back to the front-end to generate the reply. The front-end converts the timestamp into an InterLing expression using some practical rules concerning the expression of time (June 25, 2010 15:00 -> tomorrow at 15:00).
Is(z, DefTrn(.DYtomorrow)) & Prep(z, pTPat, Time(15))

In a simple lambda-calculus operation, it then combines the expression with the components of the original sentence expression to create a full InterLing reply.

Is(y, .Nmeeting) & Singl(y) & Pyour(y) & TPres(x) & MInd(x) & VCopuN(x, Vbe, y, z) & Is(z, DefTrn(.DYtomorrow)) & Prep(z, pTPat, Time(15))

Finally, the front-end translates the InterLing expression into a parse tree. The parse tree is then morphologically rendered with proper grammatical concord, and phonologically rendered with appropriate Sandhi rules.

Your meeting is tomorrow at 3:00.

Because all the rules and processing by the app are done language-independently in InterLing, the application's processing would be identical for any language. As long as the front-end can process a particular natural language, the app will function in that language. Only cosmetic localization needs to be done on the application level to adapt it to another language.

Conclusion

A rich, predicate-logic based interlingua enables proper expression of natural language. If this is standardized, applications and the conversational front-end can cooperatively exchange information. This allows sentences to be semantically routed to the appropriate application. It also allows for semantic resolution of anaphoric reference. Inferenceing operations can be performed on the semantic expressions. This dramatically reduces the complexity for the application. At the same time, a language-independent semantic representation allows applications to become language-independent. Development time is reduced and the potential market is expanded for conversational applications.

Bibliography

The Scalable Language IPA, Technical Reference Manual, January 2009
(http://slapi.sourceforge.net/)