Mach Translat (2011) 25:107–126
DOI 10.1007/s10590-011-9091-z

OpenLogos machine translation: philosophy, model, resources and customization

Anabela Barreiro · Bernard Scott · Walter Kasper · Bernd Kiefer

Received: 1 August 2010 / Accepted: 11 May 2011 / Published online: 10 June 2011

Abstract This paper reviews the OpenLogos rule-based machine translation system, and describes its model architecture as an incremental pipeline process. The paper also describes OpenLogos resources and their customization to specific application domains. One of the key aspects of rule-based machine translation systems intelligence is the symbology employed by these systems in representing natural language internally. The paper offers details about the OpenLogos semantico-syntactic abstract representation language known as SAL. The paper also shows how OpenLogos has addressed classic problems of rule-based machine translation, such as the cognitive complexity and ambiguity encountered in natural language processing, illustrating how SAL helps overcome them in ways distinct from other existing rule-based machine translation systems. The paper illustrates how the intelligence inherent in SAL contributes to translation quality, presenting examples of OpenLogos output of a kind that non-linguistic systems would likely have difficulty emulating. The paper shows the unique manner in which OpenLogos applies the rulebase to the input stream and the kind of results produced that are characteristic of the OpenLogos output. Finally, the paper deals with an important advantage of rule-based machine translation systems,

A. Barreiro (✉)
CLUP, Rua do Campo Alegre, 1055, 4150-180 Porto, Portugal
e-mail: barreiro_anabela@hotmail.com

A. Barreiro · B. Scott
Logos Institute, Tarpon Springs, FL, USA
e-mail: logos.institute@gmail.com

W. Kasper · B. Kiefer
DFKI GmbH, Stuhlsatzenhausweg 3, 66123 Saarbrücken, Germany
e-mail: kasper@dfki.de

B. Kiefer
e-mail: kiefer@dfki.de

namely, the customization and adaption to application-specific needs with respect to their special terminology and transfer requirements. OpenLogos offers users a set of comfortable customization tools that do not require special knowledge of the system internals. An overview of the possibilities that these tools provide will be presented.

Keywords OpenLogos · Machine translation system · Open source · SAL representation language · Semantico-syntactic abstract language · Rule-based machine translation · Linguistic knowledge system

1 OpenLogos background

OpenLogos is an open-source port of the Logos System, a rule-based machine translation system designed by Bernard Scott and described comprehensively in (Scott 2003). Before its transition to OpenLogos, the Logos system, currently owned by GROUP Business Software, had been, along with Systran, the longest running commercial machine translation system (1970–2001). The commercial product was considered high-end and was used successfully in many parts of the world. OpenLogos was created by DFKI and is available from the official website at http://logos-os.dfki.de and from Sourceforge at http://openlogos-mt.sourceforge.net/ under a GPL license. OpenLogos provides the same functionalities and language pairs as the commercial system, using the same linguistic resources. The only restriction is that OpenLogos currently is limited to plain text input/output and cannot handle other document formats such as MS-Office documents. One reason is that the MS-Office libraries presupposed in the commercial Windows version are not available on the Linux platform that OpenLogos was developed for.

Because OpenLogos derived from an enterprise-level machine translation system, OpenLogos can also be set up as a multi-user system in a distributed environment. The full release of OpenLogos consists of a number of components: (1) Core code *libraries* of the server side system and basic executables to start and run the system (*APITest, logos_batch*); (2) *Resources*, such as analysis and transfer grammars for source and target languages, and a multi-language dictionary database; (3) *Tools*: *LogosTermBuilder*, User administration (*LogosAdmin*), Command line tools (*APITest, openlogos*), and multi-user GUI for initiating and inspecting translation jobs and results (*LogosTransCenter*).

Logos, being one of the oldest machine translation systems in history, with over 30 years of and continual development and commercial use, is a system that cannot be fairly described in such a short paper; therefore, we limit ourselves to presenting the most general aspects for people who have little or no knowledge of the system. Whenever appropriate we point out some of its strengths with the purpose of motivating its use and exploitation. Among several unique characteristics of the OpenLogos system that will be described in this paper, perhaps the most important to the system's success was the richness of its linguistic knowledge base. Notwithstanding the computational advances provided by statistical machine translation, this paper aims to draw attention to the need to complement statistical machine translation systems with richer syntactic and semantic information presented in rule-based machine translation systems like OpenLogos. We hope that knowledge-based machine translation systems will realize their potential for high quality output by the use of semantico-syntactic information, and in particular that OpenLogos' extensive semantic knowledge, together with its other unique characteristics will be found helpful to that end. This is the prime motivation to write about OpenLogos, describe its strengths and elucidate the value that this system can offer to researchers and developers of future machine translation systems, and other distinct-purpose natural language processing tools.

2 Philosophy behind the system

The Logos commercial machine translation system from which OpenLogos derived, was based on an original view of how people process language, i.e., how the human brain functions with regard to language. It presumes that human-language processing is non-algorithmic and eschews any formalistic approach to language processing on the grounds that language is too complex for any formalism to be applied successfully. Given this complexity, the key to effective natural language processing and human language technology in a computer can be said to lie in how the *cognitive* complexity of natural language is handled, that is, the complexity relating to the difficulties humans experience in maintaining logic in maturing systems as that logic becomes increasingly more complex. This is a fundamental challenge and computational linguistics has yet to solve the problem, or even at times to sufficiently recognize it as a problem that needs to be solved. The brain deals with natural language with apparent ease. Why? We find it adequate to attempt to figure out what the mental process is and seek ways to profit from that knowledge. That at least was the philosophy behind the Logos system, and results were promising. The Logos system represents an attempt to implement some hypotheses about human sentence processing and the brain's language decoding function. The first hypothesis that proved fundamental to the design of the Logos model defines language processing as opportunistic, deterministic, and algorithm-free, i.e., not relying on supervisory logic controlling processes and decisions. Language analysis emerges in unpredictable ways from stored memory associations reacting opportunistically and deterministically, producing a single analysis of input signals (language strings, sentences). The input stream controls the process, similarly to the way the brain reacts to language input and seeks to assimilate its significance. The most important implication of this assumption for the Logos model is that associative memory networks, being non-algorithmic, would not reach logic saturation as language coverage development increased and the linguistic rulebase greatly expanded. The second hypothesis presupposed a procedure of incremental sentence-processing, i.e., analysis of sentences in different stages across a series of modules, comparable to the brain's visual pathway. The third hypothesis assumed that syntax and semantics constitute a representational *continuum* and that human sentence analysis must necessarily require the integration of syntax and semantics at every decision point along the linguistic representation pathway. For the Logos model this assumption led to the creation of an ontology-based, semantico-syntactic representation language (described in Sect. 7) exhibiting such integration. Finally, the fourth hypothesis contemplated the use of abstraction to deal with natural language complexity. This abstraction is suggested by the structure of the brain cell (a neuron with many dendrites for input and typically a single axon, with collaterals, for output). The prevalence of fan-in circuitry in the brain further suggests a structure designed for abstraction. For the Logos model, this requires that the semantico-syntactic representation language consists of *second-order* abstractions, one level higher than the first-order abstractions that natural language represents. Although these hypotheses were largely intuitive and pre-scientific, it might be said that experience in developing the system did not prove the falsity of any of these hypotheses. Logos' experience was that the system never suffered at any point in its 30-year history from developmental dead-end. The possibilities for system improvement seemed to be unlimited, just as they are in the brain, suggesting that, to an effective extent, the system, in some modest way, was mirroring how the human brain copes with complexity in decoding natural language. A more detailed description of these hypotheses can be found in (Scott 1977, 2003).

3 Other characteristics of the system

OpenLogos is a multi-target system, which means that once a source language has been developed, any number of targets can be added, requiring only that the linguistic knowledge base for that target has been developed (lexicon, morphology tables, and semantico-syntactic rules). Currently, the OpenLogos system includes seven different language pairs: German source with English and Italian targets, and English source with French, German, Italian, Portuguese and Spanish targets. The architecture of the system is in the form of a pipeline. The software of the system is language neutral, where all linguistic knowledge is kept in data files and in a relational database. Adding a new target to a source merely consists of attaching new target data files to the source.

The question of how to represent natural language inside a computer was answered in the Logos model by its symbolic representation language, the Semantico-syntactic Abstract Language, known as SAL (Scott 1977, 2003) (Scott and Barreiro 2009), that will be described in detail further ahead. The first step in the translation process of the OpenLogos system is to convert a natural language sentence to a SAL sentence, thus representing the sentence internally at a more abstract semantic level. It is this SAL sentence that the rules subsequently operate on. Another interesting feature of OpenLogos is its Semantic Table (also known as SemTab, for short), a database containing thousands of transformational rules. In conjunction with SAL, SemTab has given OpenLogos the modest potential to process text semantically as well as morpho-syntactically, as will be illustrated.

4 Classic problem: complexity and ambiguity of natural language

Natural language is very complex. This complexity is caused by natural language's fuzzy richness, an unlimited combination of linguistic units, many of which are without univocal meaning or function, except as provisionally established by the context, and context itself is often also ambiguous. Dealing with complexity and ambiguity touches on the heart of the difficulty with natural language processing, as ALPAC (Pierce et al. 1966) correctly recognized from the earliest days in machine translation history. Every machine translation/natural language processing developer recognizes the horns of the computational dilemma, namely, how attempts to deal with ambiguity by enriching the knowledge base increases complexity, and how attempts to avoid complexity by lessening the size of the knowledge base weakens the power of disambiguation. An associated problem is the issue of system performance, which degrades as the needs of disambiguation cause rulebases to grow in size. Rulebase growth in turn leads to the classic dilemma: (i) increase the rulebase and you increase complexity issues; (ii) decrease the rulebase and you weaken disambiguation. In effect, efforts to deal with either one of these problems tends to exacerbate the other.

Another of the more intractable problems for a rule-based machine translation system is the question of how to apply the rules. Typically, as a system matures and deals with natural language at an increasingly more comprehensive level, the rulebase can grow unmanageably large, which poses the question of how all these rules are to be applied to the input stream (i.e., sentences) in an efficient way. Solutions to this problem, whether through meta-rules or discrimination networks can often lead to logic saturation, resulting in a developmental dead-end.

In sum, output quality cannot improve without effective disambiguation. Disambiguation requires large rulebases. Large rulebases introduce complexity issues that are cognitively unwieldy. Large rulebases will also impact performance if not handled well. These were the issues explicitly addressed in the formation of the Logos model, and the solutions arrived at remain perhaps the more interesting aspects to be shared with the machine translation research community. The Logos solution to this classic dilemma lay in a computational design that allows the source language text and the knowledge base to be related in the same way natural language and a dictionary are related. SAL lies at the heart of this arrangement. Basically, both the natural language input stream and the rulebase are expressed as homogeneous SAL patterns, allowing the SAL input stream to serve as search argument to the SAL pattern rulebase, similarly to the way natural words are search arguments to natural language dictionaries. This arrangement is what allows the rulebase to be accessed like an indexed dictionary, and to grow without performance degradation. Thus, Logos followed an approach that permitted unlimited growth in the knowledge base, allowing it in turn to address ambiguity without incurring such issues as logic saturation and development dead-end.

5 OpenLogos uniqueness and advantages regarding distinct approaches

The Logos approach was to avoid any attempt to algorithmically wrap logic around natural language. The SAL input stream itself drives the system. Also, SAL was designed to be straightforward and easy to work with. One can easily map between natural language and SAL, so a SAL stream is equivalent to a natural language stream, only at a semantically more abstract level. As the SAL input stream is looked up segment by segment in the pattern rulebase, the matched-upon rule contributes to the building of a source parse tree and, when a source constituent is fully formed, linked target rule components (for any number of target) make notations regarding an equivalent target tree. In a very real sense, then, the SAL input stream itself has become the driving algorithm, much like it was thought to be the case in the brain's handling of language.

A common experience is that rule-based machine translation is not improvable beyond a certain point. This may be due to the fact that most commonly rules are applied to the input stream by meta-rules, limiting the power of the system to the inherent power of the controlling algorithm. Any such logic sequence is bound to break down at some point when dealing with the complexities and anomalies of natural language. In the case of OpenLogos, because its rulebase is accessed like a dictionary, no such supervisory algorithm needed to control the application of rules to the input stream and therefore logic congestion limiting the ability of the system to grow and improve is unlikely. OpenLogos may be unique in that respect, and in some unexpected way may be closer in spirit to the non-algorithmic, pattern-based methods of statistical machine translation.

Summarizing, OpenLogos claims to have two advantages that a typical rule-based machine translation system may lack: (1) a unique data-driven process that frees the system from logic-saturation and all its attendant problems; and (2) the natural language input stream and the linguistic rules are both based on the SAL representation of the natural language patterns. Because of the greater generality allowed by this higher order, the SAL representation, these more abstract natural language patterns may afford an advantage over statistical machine translation when it comes to processing text for which there has been little or no training (e.g., in minimizing the sparseness problem).

We may also indicate some other benefits and strengths of grammar-based systems in general over against statistical machine translation. A common problem experienced in most statistical machine translation systems translations is the lack of gender and number agreement between nouns and verbs, nouns and adjectives. Other problems include misresolution of *–ed* and *–ing* homographs; word order problems of many different types; problems with the translation of multiword units, among others. Even though grammar-based machine translation systems may themselves not always perform well in these cases, these kinds of errors are not typical of these systems and may often be resolved by the addition of linguistic data to the rulebase. In comparison to purely statistical machine translation systems that make no use of linguistics, rule-based machine translation systems and OpenLogos in particular generally perform significantly better when it comes to elisions, such as elided pronouns, conjunctions and punctuation. Purely statistical machine translation systems have no knowledge of grammar and therefore know nothing about clauses and their transitions and the need for commas that some target languages require, often messing up target clause construction as a result. The translation of relative clauses with implicit relative pronouns and *that* conjunctions seems to constantly get mixed up in German, for example. Similar poor behavior happens when there is lack of punctuation separating dependent and main clauses. The results are often quite unpredictable and seem to depend essentially on what aid that the system has gotten from parallel corpora and translation memories. Absent translation memory for a particular sentence of this kind, a statistical machine translation system may not perform well on the basis of parallel alignment alone (particularly if the training corpora is deficient in some way). With certain types of text, such as less formal texts, oral-type texts, etc., for which no parallel corpora is available, a purely statistical machine translation system tends more obviously to perform less well than a grammar-based machine translation system.

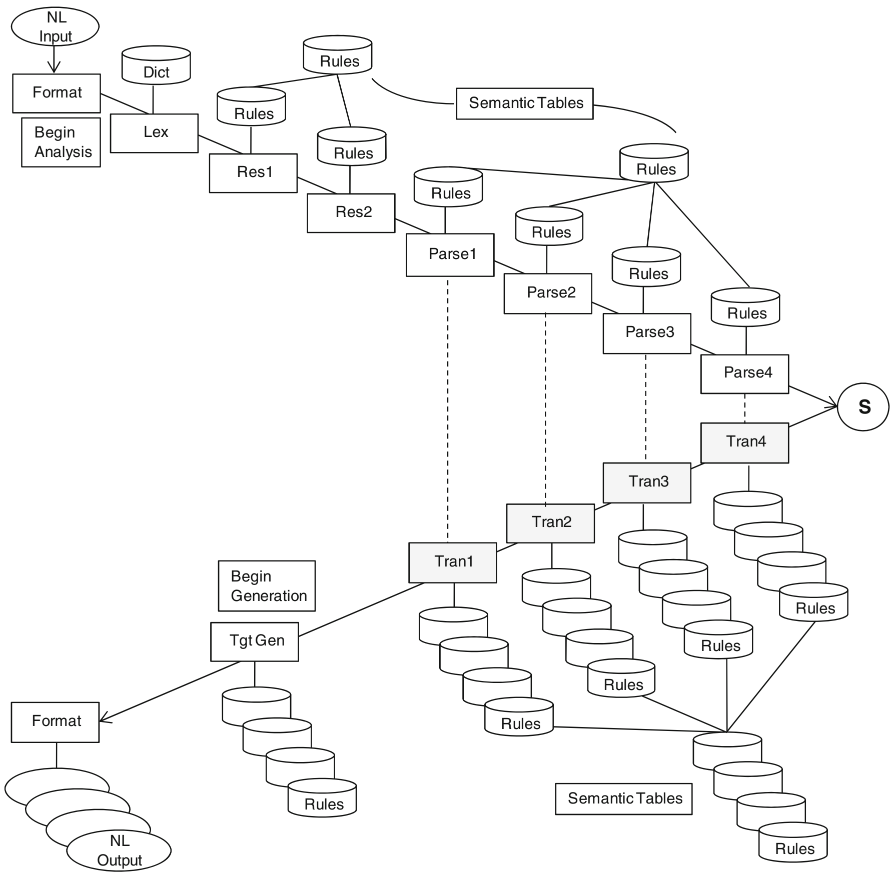


Fig. 1 Architecture of the OpenLogos system as an incremental pipeline process

6 Architecture model

The architecture model of OpenLogos may be seen as an incremental pipeline process, with the various modules performing incremental tasks of source language analysis and target language generation, as displayed in Fig. 1.

At the beginning of the pipeline, the Format Module strips document formatting codes from the input text, to be re-applied later to the target text. When the natural language input enters the pipeline, the first module converts the natural language input stream to a SAL stream. This SAL input stream moves down through the pipeline modules. Modules RES1 to Parse4 modules in Fig. 1 represent the pipeline's source analysis components. These software modules are language neutral. All linguistic functions take place in the interaction these software modules effect between the SAL input stream and the SAL rules. In the SAL rulebase, the rules are SAL patterns that match portions of the SAL input stream. There is further interaction between the SAL rules and the SemTab rules that will be described in Sect. 9. Tran1 to Tran4 display pipeline target components. During source analysis, notations are made about target structure and transfers on a constituent by constituent basis, i.e., when analysis of the source constituent is done, notations for a target equivalent is made, in tree-to-tree fashion, for both target sentence structure and word transfers. When source analysis is complete, at the end of Parse4, target generation takes place. The Target Format Module allows source document formatting to be re-applied to the target output. In brief, the pipeline process is highly modular and uses incremental processing. Source analysis is bottom-up and deterministic, meaning one parse is generated rather than a parse forest. Deterministic parse is made possible because of the use of semantics at every stage of analysis, to aid in disambiguation.

Modules RES1 and RES2 are purely source-related. The rules associated with these modules resolve part-of-speech homographs, and identify clausal transitions (clausal segmentation, including embedded clauses), giving in effect a top-down macro picture of the sentence for the benefit of the subsequent bottom-up micro parse in Parse1–Parse4. Deterministic parsing requires that all part-of-speech ambiguities are resolved by the end of RES2. OpenLogos has proven to perform well in this regard, achieving about 98% accuracy with regard to homograph resolution in previously unseen text of reasonable quality.

Parse1 rules effect a parse of simple noun phrases. Any semantic issues within the noun phrase are addressed by means of SemTab rules. Adjective and common noun polysemy are areas where OpenLogos remains weak. Parse2 rules concatenate constructions like [NP PREP NP] in cases where the prepositional phrase is seen to complement the head noun of the noun phrase. Thus, e.g., *the book on the presidency* would be concatenated in Parse2 as a noun phrase. SemTab rules would recognize and effect this concatenation and would also recognize that the preposition *on* in this construction has the sense of *about*, or *concerning*. The interaction of Parse and SemTab rules easily distinguishes the sense and grammatical function of the prepositional phrases in each of the following constructions: *the book on the presidency*, *the book on the table* and *place the book on the presidency on the table*. Parse2 also deals with constructions like relative clauses, parenthetical and clausal complements, extracting them from the clause they appear in and placing them at the end of the sentence for separate, independent treatment through Parse2–Parse4, leaving behind a placeholder for the benefit of target generation. Parse3 rules deal with simple clauses and the semantics of verbs and verb arguments. Parse4 rules deal with complex sentences. The rules of this last source analysis module sort out clausal order in complex sentences, particularly for the benefit of targets that require a different clausal order. Even though the system has a strong semantic component, obviously not all semantic issues are successfully dealt with. However, with the exception of common noun polysemy, most of its deficiencies are addressable by the addition of new SemTab rules.

To illustrate the potential power of the approach taken, we present sentences (1)–(5) (from (Scott 2003)) that involve the word *as*, and where the function and sense of *as* in each case is triggered by a variety of complex contextual clues. The German translations (1′)–(5′) are offered as indications of effective complexity handling. These translations are unedited output of the current English-German system.

(1) **As** you can see, he is sick.
(1′) ***Wie*** *Sie sehen können, ist er krank.*

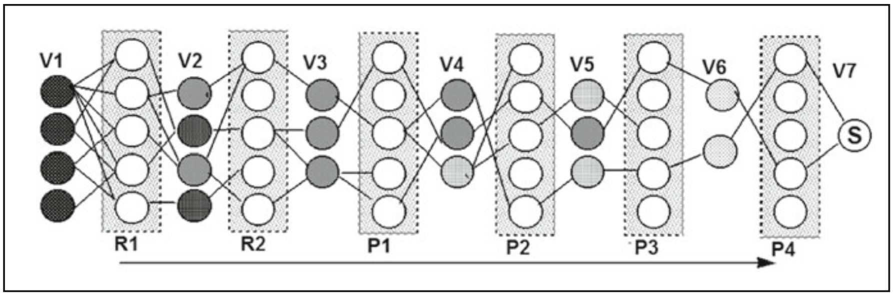


Fig. 2 OpenLogos pipeline depicted as a biological neural net

(2) **As** he is sick, we cannot ask him to work.
(2′) ***Well*** *er krank ist, können wir ihn nicht bitten, zu arbeiten.*
(3) **As** he was being given his medicine, he began to choke.
(3′) ***Während*** *ihm seine Medizin gegeben wurde, fing er an, zu ersticken*
(4) **As** he began to recover his health, he realized that his wife had stood by him through difficult times.
(4′) ***Als*** *er anfing, seine Gesundheit zurückzubekommen, erkannte er, dass seine Frau ihm durch schwere Zeiten beigestanden hatte.*
(5) **As** a patient, he was very cooperative.
(5′) ***Als*** *Patient war er sehr kooperativ.*

There are 80 patterns (rules) indexed on *as* in Parse1, 52 in Parse2, 5 in Parse3, and 11 in Parse4. The examples (1)–(5′) were chosen because they are handled relatively successfully, especially compared to the performance of a statistical machine translation system, such as Google Translate. It is quite easy however to find other *as* sentences that translate poorly, and that would require additional rules somewhere in the pipeline. Nevertheless, these examples are indicative of the strength of rule-driven systems when endowed with a strong ontology-based representation language such as SAL.

7 Analogy of the OpenLogos model to a biological neural net

Another way of viewing the OpenLogos architecture model is through the metaphor of a biological neural net (bionet), as illustrated in Fig. 2. In the graphic, the vectors labeled V1–V6 represent the SAL input stream of the pipeline, working its way progressively from beginning to end. The cells in these input vectors are SAL elements or words to which the natural language input stream has been converted. These cells become fewer in number as the process progresses, signifying the increasingly more abstract nature of the parse as it leads toward S (Sentence Parse). In addition, the SAL elements in these vectors become lighter as analysis progresses, signifying semantic disambiguation. In this network, R1 through P4 are the hidden layers. R1 represents RES1, P1 represents Parse1 and so on. Each hidden layer contains between 2 and 4,000 rules, organized by their SAL pattern, as in a dictionary.

The metaphorical similarity between OpenLogos and a biological neural net lies in the way the SAL input stream and the rules of the hidden layers interact. When some string of cells in the input vector and certain of the rules in the hidden layer are found to have matching SAL patterns, such rules become active and compete for the right to fire. The winner will be that particular rule whose syntactic and semantic correspondence to the SAL input string is greatest (assuming rule constraints have been satisfied). Because these hidden layers are well indexed, like in a dictionary, interaction between cells in the input vector and rules in the hidden layers is very efficient. In short, only those rules which should be examined are ever accessed. This efficiency in rule matching is a key defining aspect of the OpenLogos design and affords many benefits, among them: (1) rule size is no longer a factor in system performance (since only relevant rules are ever accessed); (2) rules are self-organizing, i.e., find their own place in the rulebase; (3) the developer has no need for algorithms (e.g., meta-rules or discrimination networks) to achieve efficiency in rule-matching. Noteworthy detail on this metaphor is found in (Scott 2003), Sect. 7, pp. 47–63.

8 Semantico-syntactic Abstract Language

As mentioned, the representation language in OpenLogos is called Semantico-syntactic Abstraction Language (SAL). SAL includes the word part-of-speech ("word class" in Logos terminology), which is one of the syntactic elements of the natural language word, plus the word semantico-syntactic properties at several levels of abstraction. SAL represents the point where syntax and semantics seem to intersect, capturing the implications that semantics has for syntax, placing both meaning (semantics), and structure (syntax) in a continuum, with the benefit that both syntax and semantics are available at every stage of analysis. This syntax-semantic continuum is the factor that allows the parse in OpenLogos to be deterministic.

As a language, SAL currently has over 1,000 elements or words (expandable), organized in a hierarchical taxonomy consisting of Supersets, Sets, and Subsets, distributed over all parts-of-speech. For example, SAL has 12 supersets for nouns: Concrete (CO), Mass (MA), Animate (AN), Place (PL), Information, Abstract (AB), Process (intransitive) (PI), Process (transitive) (PT), Measure (ME), Time (TI), Aspective (AS), and Unknown (UN). Figure 3 shows the Abstract noun Superset. In the Abstract noun Superset, there are two principal Sets: the non-verbal Abstract nouns, and the verbal Abstract nouns, both with their own Subsets. The Subset *Classifications* is a member of the non-verbal Abstract noun Set. It includes nouns such as *category*, *class*, *kind*, *make*, *nature*, *rank*, *type*, among others. The Subset *Methods/ Procedures* is a member of the verbal Abstract noun Set. It includes nouns such as *technique*, *means*, *mode*, *pattern*, among others. The complete taxonomy can be viewed at the Logos Archives website http://logossystemarchives.homestead.com/.

The power of SAL depends upon its synergy with the system's Semantic Table, i.e., with the SemTab rules that provide context. Just as the literal English word *German* has two meanings: one for language and one for people, so does the corresponding SAL element. In other words, SAL does not attempt to label the different meanings of words, and just as the brain uses context to resolve meaning of ambiguous natural language words, SAL words also need context to be properly understood, a context provided in OpenLogos by relevant SemTab rules. So in many respects SAL is

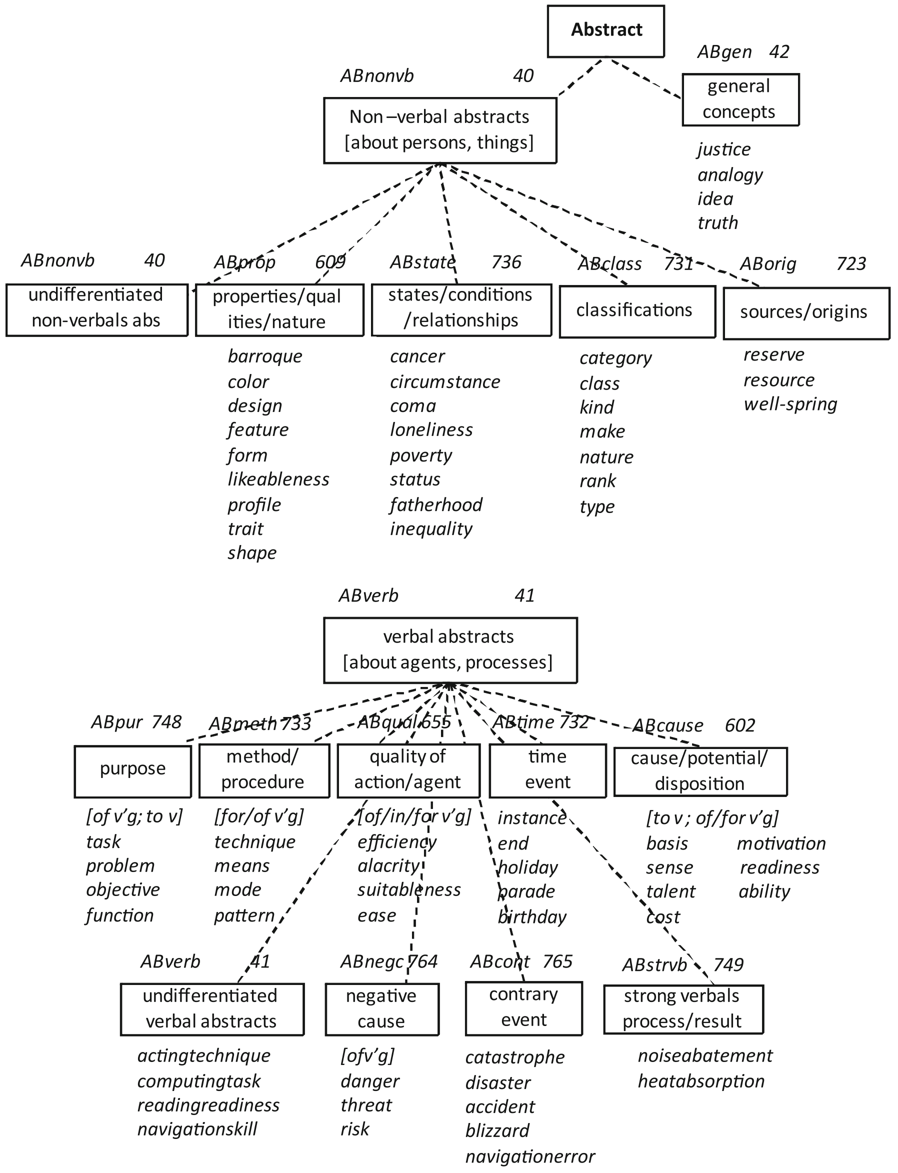


Fig. 3 *Abstract* Noun superset (non-verbal and verbal), its sets and subsets

similar to natural language, but at a greatly simplified and more abstract level. This level of abstraction is its principal advantage in machine processing of natural language because it helps minimize the complexity factors associated with such a process. To be sure, in comparison to human competence for resolving meaning in context, the OpenLogos system is an infinitely weaker model. Nevertheless, it can be claimed that for verbs and prepositions, and to a lesser extent for adjectives (1) the system copes with meaning to a degree not thought possible in earlier systems, and (2) the system is open-ended and therefore, in principle, capable of absorbing information about context without practical limit. It should be mentioned that contextual clues for resolving word ambiguity are initially provided at the dictionary level, by means of the subject matter codes that can be optionally associated with entries. Thus, for example, the term *figure*, when subject-matter-coded in the OpenLogos dictionary for Mathematics, will have an appropriately different target transfer than the one given for the default dictionary entry for this term. The system's undoubtedly most egregious limitation pertains to common noun polysemy, principally because of the difficulties in defining context for common noun resolution.

9 The Semantic Table and its use of SAL

The Semantic Table is a very large body of rules that serve as OpenLogos' principal means for dealing with semantics-related problems. SemTab is accessed by regular pipeline rules that ask it to deal with both source and target issues requiring a closer look at the semantics. Such access can be initiated by any rule at any point in the pipeline process. Like all rules in OpenLogos, SemTab rules consist of SAL patterns with constraint and action components. When a SemTab rule matches on a SAL pattern in the input stream, the SemTab rule may variously: (1) override dictionary transfers to provide translations more appropriate to the semantic context; (2) attach prepositional phrases to the verbs they complement; (3) resolve scoping problems; (4) capture and handle expressions that cannot be lexicalized, e.g., *take them away* or *pay it off*, or *pay her a visit to* or *bring it under control*. SemTab is extremely efficient at assigning secondary senses to verbs. On the negative side, as stated, SemTab is ineffective for resolving common noun polysemy. The handling of common nouns is, therefore, a problem that requires an entirely different approach than the one used for verbs, for example.

We will now illustrate how SemTab effects *syntactic* homograph resolution. For example, in the strings (6) and (7) below, the word *revolving* is a verb in (6) and an adjective in (7). In either instance, the term *revolving* would have *both* parts-of-speech as it comes out of the dictionary. Once in the pipeline, a SemTab rule will use the SAL code of the noun preceding the ambiguous term, to resolve its part-of-speech. In (6), for example, *ways* has the SAL code N(ABmethod), and in (7) the word *types* is SAL-coded N(ABclass). When a pipeline rule sends strings (6) and (7) to SemTab, in the case of string (6), a SemTab rule with the pattern N(ABmethod) + V would match and resolve *revolving* to the verb. In the case of string (7), a SemTab rule with the pattern N(ABclass) + Adj would resolve *revolving* to the adjectival form.

(6) new *ways* of revolving credit
(7) new *types* of revolving credit

The SAL codes in SemTab rules can be as specific or general as the situation requires. SemTab rules can also contain hash codes to represent natural language strings where needed. A hash code is a shortened numeric equivalent to the literal string.

We illustrate in Table 1 below a partial list of the SemTab rules for the verb *place*, and how the various SAL contexts affects the French and Portuguese transfers. In each

Table 1 SemTab rules comment lines for the verb*place*

Semantic Table (SemTab) rule comment line	Transfer
place(vt) N(advertisement, announcement,ad)	FR:*placer N*
	PT: *colocar N*
place(vt) N(order)	FR:*passer N(commande)*
	PT: *fazer N(encomenda)*
place(vt) N(restriction,constraint) on	FR: *imposer N à*
	PT: *impor N a*
place(vt) N(importance) on	FR: *attacher de l'N à*
	PT: *dar N a*
place(vt) N(pressure) on	FR: *faire N(pression) sur*
	PT: *exercer N(press ao) sobre*
place(vt) N(confidence,trust) in N(AN)	FR: *placer N dans N*
	PT: *depositar N(confiança) em N*

case, the rule overrides the *default* French (FR) or Portuguese (PT) dictionary transfers for *place*.

The translations effected by the SemTab rules listed in Table 1 occur through the interaction of a Parse3 rule with SemTab.

SAL transformational rules were never used by the OpenLogos system in a purely monolingual context, but they can also play an important role in monolingual paraphrasing. These rules allow rephrasing or finding synonyms that are suitable only when applied to words or classes of words that share identical syntactic and semantic properties, such as in (8).

(8) [*bring up* a N(child/son/daughter/baby/offspring) → *raise* a N(same)]

The objects of the phrasal verb *bring up* in (8) are all classified with the SAL code, superset [AN], which stands for animate, human and non-human beings, designated singly or by groups (AN also includes spiritual entities) (cf. (13)). SemTab rules also allow transformation of prepositions (Prep), such as *on* into *about*, in the phrase [book *on* political satire > book *about* political satire] (cf. (9)), where [INdata] stands for the superset and set combination 'information data', of which the noun *book* is an example. The SemTab rule comment line in (10) illustrates the transformation of the transitive phrasal verb (vt) *mark down* into the single transitive verb *reduce*, when it occurs in the context of the common noun *price* (its direct object), and also in the context of nouns classified under the superset concrete ([CO]), such as [COcloth], for clothes; [COvehic], for vehicles; [COsoft], for software; and [COmach], for machinery, i.e., objects that are typically sold. Note that in (9) and (10), SemTab rules are applied to words represented by their subset. However, SemTab rules can be applied at any level of abstract representation or at the literal word level. Thus, in examples (11) and (13) below the rules are used at the superset level to transform the transitive phrasal verbs *bring off* and *bring up* into the single verbs *rescue* and *raise*, respectively, when occurring with any and all animate nouns [AN]. In example (12), the SemTab rule operates at the literal word level, where the transitive verb *bring* followed by the nouns *charge* and *action* can be transformed into the verb *present* ((idem) signifies generation of the same nouns).

(9) [N1(INdata) Prep(on) N2(idem) → N1 Prep(about) N2]
(10) [mark down(vt) N1(price) Prep(of) N2(COcloth; COvehic; COsoft; COmach) → reduce(vt) N1(price) Prep(of) N2(idem)]
(11) [bring off(vt) N(AN) → rescue(vt) N(idem)]
(12) [bring(vt) N(charge; action) → present(vt) N(idem)]
(13) [bring up(vt) N(AN) → raise(vt) N(idem)]

A SemTab rule has five components: the SAL pattern, a comment line, constraints, source actions, and optional target actions. The chief component is the SAL pattern. (14) is a Parse2 SAL rule for the string *a book on the presidency*.

(14) [N(INdata;u) Prep("on";u) N(u;u)]

The (INdata) in the SAL rule for *book* stands for Information noun Superset, and recorded data Set. The *u*stands for undifferentiated morphology, singular or plural or possessive. Every rule has a comment line. The comment line for example (14) is [NP(info) Prep(*on*) NP → N1 *about* N2], where the example [*book on* political *satire* → *book about* political *satire*] can be also inserted. The comment line illustrates the effect of SemTab on the rule, where the preposition *on* has been resolved to the meaning *about*. Constraints match only if conditions are satisfied. If the condition is not satisfied, the rule will not fire. Source actions are taken at RES rulebase, which resolves the syntactic ambiguity, at PARSE rulebase, which creates the parse tree, and SemTab rules effect semantic disambiguation. The Target actions (optional) effect syntactic and/or semantic transfer.

All SemTab rules are deep structure rules. In effect, a single deep structure rule can match a variety of surface structures. For example, a SemTab rule such as [meet(vt) N] will allow for the transformations represented in (15)–(18) for the Italian target.

(15) He met the goal → *ha raggiunto l'obiettivo*
(16) The meeting of the goals → *il raggiungimento degli obiettivi*
(17) The goal, met by... → *l'obiettivo raggiunto da...*
(18) The goal meeting is... → *il raggiungimento degli obiettivi...*

In (15), the deep structure rule matches the verb *meet* and its object (the surface structure is similar to the rule). In (16), the deep structure rule matches *meeting* as gerund. The Italian target is also no longer a verb. In (17), the same deep structure rule matches *met* as a participial adjective. And finally, in (18), the same one deep structure rule matches *meeting* as a noun.

In the examples (19)–(24), the passive voice in English is variously transformed to the active voice in Spanish, because in this target language the passive is used much less often, especially when the agent is not expressed, as in these examples, where a direct translation would be considered very unsatisfactory. Such transformations are made possible by the incremental pipeline approach of OpenLogos, and its relatively strong semantic sensitivity.

Table 2 OpenLogos Core dictionary

Morphology/Stems	630000
Transfer	1400000
SemTab rules	120000

(19) The handle is released when… → *la manivela se suelta al…*
(20) English is spoken here… → *Aquí, se habla inglés…*
(21) He was disturbed. → *Se quedó perturbado.*
(22) This house was built by John. → *Esta casa la construyó Juan.*
(23) The books are written in English. → *Los libros son escritos en inglés.*
(24) The meeting was considered to be of no interest… → *Se consideró que la reunión no tenía ningún interés…*

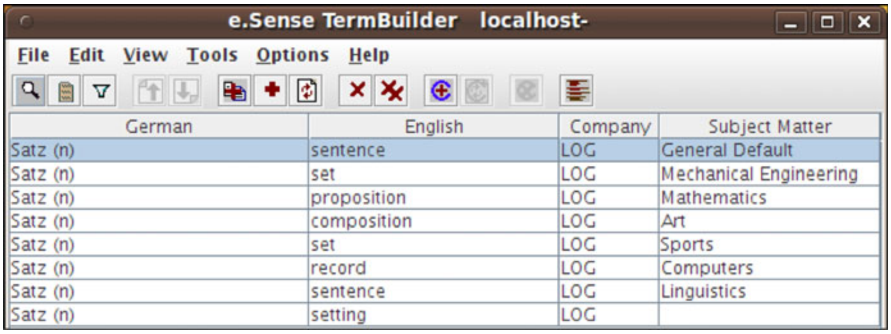
10 Resources and customization

A common mistake of inexperienced users of machine translation systems is the assumption that the system will translate their specific kinds of documents with reasonable quality *out of the box* without further ado. Seasoned users know better. No machine translation system can know all terms and all their possible meanings they might have in all possible domains. While the syntax of a language usually can be considered as being rather stable across domains and time, the lexicon is highly dynamic and variable across domains and time. New terms appear every day as well as new uses and combinations of existing terms. Different domains and users have their own specific terminology and their own preferences of how terms should be translated. For example, a German *Engländer* is not only an *Englishman,* as OpenLogos considers it, but is also a *type of wrench*. For such reasons, the possibility to adapt and extend the system to specific vocabularies, domains and translation requirements is crucial for the practical use of a machine translation system. Rule-based machine translation systems usually offer tools that allow users to extend the system and customize the translations to their needs, especially at the level of terminology.[1] This customizability constitutes one of the strengths of rule-based machine translation systems, allowing many translation problems to be corrected quickly, efficiently and with predictable result.

OpenLogos provides *LogosTermBuilder* as a powerful tool for browsing, modifying and extending the terminological database of the translation system. In its released form, OpenLogos comes with a large core dictionary that covers the general vocabulary of the supported languages and provides default translations for them. The size of that dictionary (for all currently covered languages) is indicated by the figures in Table 2.

The *LogosTermBuilder* allows users to define their own dictionaries on top of the core dictionary. Additionally different uses of terms can be assigned to different

[1] The grammars often are not open to customization by users but can only be adapted by developers since modifying the grammars requires deep linguistic knowledge as well as deep knowledge of the grammar formalisms for encoding grammar rules and an understanding of their interaction.


e.Sense TermBuilder localhost-
File Edit View Tools Options Help

German	English	Company	Subject Matter
Satz (n)	sentence	LOG	General Default
Satz (n)	set	LOG	Mechanical Engineering
Satz (n)	proposition	LOG	Mathematics
Satz (n)	composition	LOG	Art
Satz (n)	set	LOG	Sports
Satz (n)	record	LOG	Computers
Satz (n)	sentence	LOG	Linguistics
Satz (n)	setting	LOG	

Fig. 4 Entries for the German term *Satz*

domains. When triggering a translation process the user can specify a preference order for using dictionaries and domain-specific translations: higher ranked dictionaries and domains will override possibly competing translations of lower ranked dictionaries and domains.

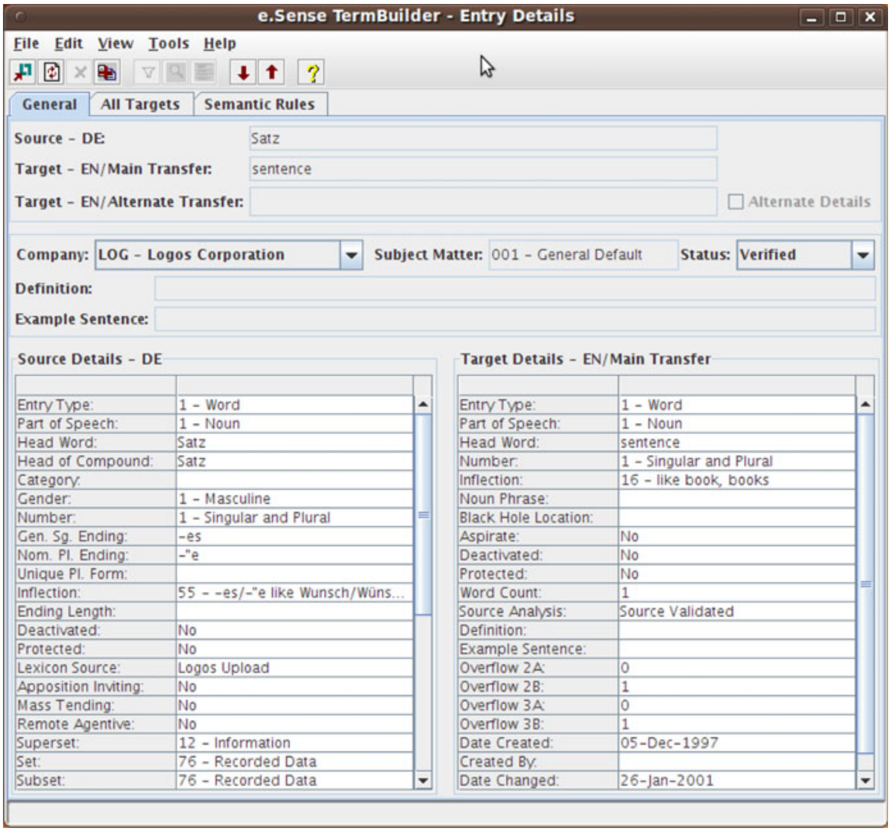
Figure 4 illustrates different translation possibilities for the German term *Satz* in English. Displayed are the part of speech ((n) standing for noun), the particular dictionary the terms belongs to (LOG in this illustration), and various translations for various domains. In the Logos system, dictionaries are called *Companies* (reflecting its commercial use), and domains are termed *subject matter*. The dictionary *LOG* represents the OpenLogos core dictionary. Subject matters/domains can be defined in inheritance hierarchies. The full subject matter tree can be viewed from within *LogosTermBuilder*. One will notice in Fig. 4 that the translation of *Satz* as *sentence* is marked as the "General Default" translation. This translation will be used if no domain-specific entry is found in the dictionary or if that domain is not specified at run time. Predictably, this will result in wrong translations when the input for translation is actually from the *Sports* domain, as illustrated in (25) and (25′).

(25) Federer gewann das Endspiel im Wimbledon-Turnier *in 3 Sätzen*.
(25′) *Federer won the final play in the Wimbledon tournament *in 3 sentences.*

Although there are a number of domain-specific entries in the LOG dictionary, most are classified *General default*. It is expected that domain-specific meanings of words would be defined by users based on their particular requirements. A domain-specific meaning for the German term *Satz* in a user selected *Sports* domain would turn (25′) into a perfect English translation.

Figure 4 illustrates the browser for the dictionary database. The menus and buttons give access to functionalities such as inspection of rules and linguistic information associated with an entry, creation of new dictionaries, new terms, rules, importing and exporting sets of entries, etc.

Figure 5 illustrates the details of an entry. The editor for new entries looks very similar. The many details might seem deterrent. But the interface provides excellent support for selecting the right values for mandatory fields without presupposing specialized linguistic knowledge. Moreover, there is a sophisticated auto-completion functionality. All a user must do is to specify the source and target terms and the


e.Sense TermBuilder - Entry Details
File Edit View Tools Help
General
All Targets
Semantic Rules

Source - DE:	Satz
Target - EN/Main Transfer:	sentence
Target - EN/Alternate Transfer:	
Alternate Details	☐
Company:	LOG - Logos Corporation
Subject Matter:	001 - General Default
Status:	Verified
Definition:	
Example Sentence:	

Source Details - DE

Entry Type:	1 - Word
Part of Speech:	1 - Noun
Head Word:	Satz
Head of Compound:	Satz
Category:	
Gender:	1 - Masculine
Number:	1 - Singular and Plural
Gen. Sg. Ending:	-es
Nom. Pl. Ending:	-"e
Unique Pl. Form:	
Inflection:	55 - -es/-"e like Wunsch/Wüns...
Ending Length:	
Deactivated:	No
Protected:	No
Lexicon Source:	Logos Upload
Apposition Inviting:	No
Mass Tending:	No
Remote Agentive:	No
Superset:	12 - Information
Set:	76 - Recorded Data
Subset:	76 - Recorded Data

Target Details - EN/Main Transfer

Entry Type:	1 - Word
Part of Speech:	1 - Noun
Head Word:	sentence
Number:	1 - Singular and Plural
Inflection:	16 - like book, books
Noun Phrase:	
Black Hole Location:	
Aspirate:	No
Deactivated:	No
Protected:	No
Word Count:	1
Source Analysis:	Source Validated
Definition:	
Example Sentence:	
Overflow 2A:	0
Overflow 2B:	1
Overflow 3A:	0
Overflow 3B:	1
Date Created:	05-Dec-1997
Created By:	
Date Changed:	26-Jan-2001

Fig. 5 Detail view for an entry

part-of-speech. The auto-completion function will fill in morphological and semantic properties derived from an expert subsystem. The automatic encoding of semantic (SAL) properties is not infallible, obviously, and users may override any incorrectly assigned codes.

The *LogosTermBuilder* also has provision for defining contextual SemTab rules that will alter the standard translation of a term in specified linguistic environments as well as the definition of rules for fixed phrases and regular patterns.

To assist users in detecting terminological gaps, the OpenLogos system offers an option to automatically mark any unknown terms in translation output. These then can be easily extracted. A valuable option for customizing input to the system is to mark up parts of the text as *not* to be translated. This is interesting, for example, for marking embedded quotes in other languages than the source language. An interesting use of this facility is to mark up automatically named entities, such as names of persons, organizations, music groups, etc., before the text is submitted to translation, as to prevent translation of, for example, the name *Bin Laden* to *Sortierfach* (pigeon hole for sorting things), *Hip Hop music* to *Hüfthopser-Musik* (not a recognized music style), and so on. In general, the set of possible named entities is too dynamic to be added to a dictionary in a static manner. Most machine translation systems suffer from the problem of not properly recognizing and handling proper names.

11 Exploitations

Several universities and individuals have been using OpenLogos to create new natural language processing resources and applications, namely a new machine translation system. The Anusaaraka group at LTRC, IIIT-Hyderabad is working on integrating OpenLogos in their English to Hindi 'Language accessor cum machine translator' system (Chaudhury et al. 2010). The approach adopted is to extract the parse information from the OpenLogos diagnostic file and represent it as CLIPS[2] (C language Integrated Production System) facts. CLIPS is a public domain software tool for building rule-driven expert systems. A CLIPS expert system comprises a list of *facts* and a set of *rules* which operate on them. In a machine translation application, the facts could be natural language sentences, or SAL sentence equivalents, and the rules could perform source analysis and target synthesis. This production system, or expert system shell, allows any Indian language to be easily generated. An OpenLogos-based English–Hindi machine translation prototype is already functional, but needs considerable refinement before it can be released.

OpenLogos resources have been adapted and enhanced to create new linguistic resources and new paraphrasing-based applications, and are also being used by a multilingual corpora management tool for pattern searching. Port4NooJ is a set of publicly available linguistic resources for the automated processing of Portuguese, fully described in (Barreiro 2008a). Port4NooJ contains several dictionaries of lemmas, originally OpenLogos data converted into NooJ dictionary format (Silberztein 2007) and enhanced with new morpho-syntactic and semantic properties. These dictionaries interact with a new inflectional and derivational system to generate inflected and derived forms and then are applied to local grammars to process multi-word units and generate paraphrases. Eng4NooJ is a set of linguistic resources for the English language processing, developed using the same methodology as the one used to produce Port4NooJ. SPIDER (formerly ReWriter, based on its Portuguese version, ReEscreve) and ParaMT are two new automated software tools which use the newly created resources developed for English and Portuguese to recognize, paraphrase and translate multi-word units, and are being extended to the processing of distinct linguistic phenomena. SPIDER is a System for Paraphrasing in Document Editing and Revision. SPIDER can be used as a writing aid to change, simplify and clarify text by reducing ambiguity and wordiness and also as a machine translation pre-editor or a linguistic quality assurance tool. The conceptual model (under the name ReWriter) is described in (Barreiro 2008b), but the tool has evolved since then (Barreiro and Cabral 2009), and mostly in (Barreiro 2011), where a detailed description of the system's linguistic intelligence can be found). A new version of SPIDER, named EXPERT (Expert Paraphrasing for Editing and Revision of Texts) is under development to assist with

[2] In addition to production system, CLIPS also supports object oriented and procedural programming styles.

the writing of technical language and use of terminologies, while covering a wider range of linguistic phenomena that can be paraphrased. ParaMT is a bilingual/multilingual paraphraser based on the same methodological principles as those established in the development of SPIDER, but it operates as an integrated function for machine translation. A description of the initial ParaMT application prototype can be found in (Barreiro 2008c). In addition, Corpógrafo (Maia and Matos 2008) is currently using Port4NooJ resources to help with pattern search in corpora.

12 Future direction

Initial efforts are under way to apply statistical techniques to OpenLogos resources in order to create new hybrid machine translation systems and to develop new paraphrasing techniques. The OpenLogos parser can be used in combination with other parsers and applied to monolingual or multilingual corpora. Parse trees can be generated from the different levels of OpenLogos analysis, from RES1 to TRAN3. At each level of analysis, natural language strings (words or expressions) are represented by SAL constructs, which constructs might then be used in statistical mapping for purposes of paraphrasing and translation. Thus, n-grams can evolve from literal strings to SAL strings (i.e. of words or expressions with semantico-syntactic properties), allowing mapping to take place at a more abstract level. The probability of synonyms and semantically equivalent expressions having similar semantico-syntactic properties is high, thus increasing the mapping score and adding to the probability of finding paraphrases and more adequate translations. In addition, the use of SemTab rules for monolingual and multilingual transformation should considerably help to improve statistical gray areas.

13 Conclusions

This paper has described the OpenLogos machine translation model and its loose analogy to a biological neural net, its philosophy, resources, customization, and various examples of exploitation of its linguistic resources.

The paper has discussed classic problems of rule-based machine translation and indicated that future machine translation systems must address design requirements that solve the cognitive complexity issue regarding *representation*, *storage* and *application* of vast quantities of linguistic knowledge (rules included), and that it is the computational approach that will ultimately determine how good a machine translation system will be. OpenLogos copes optimally with these three fundamentals. We have focused on the model's computational methodology relating specifically to the question of rule application, viz., how an exceedingly rich knowledge store is to be applied, effectively and efficiently, to an unconstrained input stream without giving rise to complexity issues. Key to this methodology is the SAL representation language that is used both for the input stream and for the rulebase that must interact with that stream. We have exemplified SAL, SemTab rules and OpenLogos output that show the computational approach to be robust and capable of producing high quality machine translation. The paper concludes by illustrating customization and optimization tools that will enable users to optimize the machine translation system for their specific needs, and to make use of OpenLogos linguistic resources to create new tools and applications. We trust that this paper will help the research community in understanding the benefits of including semantic and syntactic knowledge in future generations of machine translation systems, and we hope that OpenLogos methodology and resources may prove helpful in the much-desired achievement of better quality output.

References

Barreiro A (2008a) Port4NooJ: Portuguese linguistic module and bilingual resources for machine translation. In: Blanco X, Silberztein M (eds) Proceedings of the 2007 International NooJ Conference June 7–9, 2007, Cambridge Scholars Publishing, Barcelona, Spain, pp 19–47

Barreiro A (2008b) Make it simple with paraphrases. Automated paraphrasing for authoring aids and machine translation. PhD dissertation. Universidade do Porto, Portugal

Barreiro A (2008c) ParaMT: a paraphraser for machine translation. In: Teixeira A, Strube de Lima VL, de Oliveira LC, Quaresma P (eds) Computational processing of the Portuguese language, 8th International conference, Proceedings (PROPOR 2008) vol. 5190, (8–10 de Setembro de 2008), Lecture Notes in Computer Science, Springer Verlag, Aveiro, Portugal, pp 202–211. ISSN: (Print) 1611-3349 (Online)

Barreiro A (2011) SPIDER: a system for paraphrasing in documente editing and revision: applicability in machine translation pre-editing. Computational linguistics and intelligent text processing. Proceedings of the 12th CICLing International Conference 6609 (2011), Part II, Lecture Notes in Computer Science, Springer, pp. 365–376. ISSN: 0302-9743. e-ISSN: 1611-3349. doi:10.1007/978-3-642-19400-9.

Barreiro A, Cabral LM (2009) ReEscreve: a translator-friendly multi-purpose paraphrasing software tool. In: Goulet M-J, Melançon C, Désilets A, Macklovitch E (eds) Proceedings of the Workshop Beyond Translation Memories: New Tools for Translators, The Twelfth Machine Translation Summit Château Laurier, Ottawa, ON Canada, pp 1–8

Chaudhury S, Rao A, Sharma DM (2010) Anusaaraka: an expert system based nachine translation system. In: Proceedings of 2010 IEEE International Conference on Natural Language Processing and Knowledge Engineering (IEEE NLP-KE2010), Beijing, China

Maia B, Matos S (2008) Corpógrafo V4: tools for researchers and teachers using comparable corpora. In Pierre Z, Éric G, Pascale F (eds), LREC 2008 Workshop on Comparable Corpora (LREC 2008), European Language Resources Association (ELRA), Marrakech, pp 79–82

Pierce JR, Carroll JB et al (1966) Language and machines: computers in translation and linguistics. ALPAC report, National Academy of Sciences, National Research Council, Washington.

Scott B, Barreiro A (2009) OpenLogos MT and the SAL representation language. In: Pérez-Ortiz JA, Sánchez-Martínez F, Tyers FM (eds) Proceedings of the First International Workshop on Free/Open-Source Rule-Based Machine Translation, Alicante, Spain, pp 19–26

Scott B (2003) The Logos model: an historical perspective. In: Machine Translation 18:1–72

Scott B (1997) Linguistic and computational motivations for the Logos machine translation system. http://logossystemarchives.homestead.com

Silberztein M (2007) An alternative approach to tagging. Invited Paper. In: Proceedings of NLDB 2007. LNCS series, Springer-Verlag, Berlin, Heidelberg, pp 1–11