Conceptual Engineering

Implementing Cognitive Semantics[1]

Kenneth Holmqvist
Department of Cognitive Science, Lund University

1. Introduction

Although Langacker's Cognitive Grammar (Langacker 1987) is a very powerful full-scale model of language, there have been few attempts to implement it on computers. We think that the reason for this is twofold. First, cognitive grammar is presented as a theory of language as a static entity. It does not focus on the linguistic and semantic processing that an individual has to perform when understanding and producing speech. (As opposed to, for instance, generative grammar, in which the language description, the language reception model and the language production models are all one and the same.) Therefore, anyone who wants to implement language understanding based on cognitive grammar must first devise a processual counterpart to it.

The second reason that implementations of cognitive grammar are and will remain rare is the general complexity and openendedness of the theory. It is simply impossible to make any larger computer implementation of cognitive grammar without first interpreting virtually all elements of the theory into some more rigid form. Domains, constructional schemata, schema types, scanning modes, predications, valence composition etc. must therefore be given computational counterparts. This is not only difficult, but it also involves the danger of completely rebuilding the theory so that it is adapted more to the needs of the computer than to the linguistic reality behind cognitive grammar.

The reason why a computer scientist would be interested in implementations of cognitive grammar is of course its well-founded integration of grammar and semantics with imagery, especially as there is now a rising interest in spatial imagery as a representation form among AI researchers (Glasgow 1993). A cognitive linguist would probably take most interest in the many implications of language processing that an implementation gives: incrementation, linear time complexity, the continuous and parallel adaptation to semantic and grammatical constraints etc. In our project, the cognitive linguist could find suggestions for problems that need the attention of linguistic research, but s/he could not take our solutions to be truths about the human language. We are *simulating* human language understanding in the computer, but also making necessary computational assumptions about that which is yet unknown.

Our project is based on the preliminary computational model of cognitive grammar developed in Holmqvist (1993). This computational model consists of three main parts: Representations of Activated Lexical Units, Semantic Composition Processes, and Mechanisms for Valence Suggestion and the Incremental Updating of the Schema Population. In the following, we will make a summary description of these mechanisms with examples to clarify them.

2. Representation of activated lexical units

First, we implemented a computer model of the lexical unit structure in so-called 'image schemata'. Different authors in the cognitive linguistics tradition attach different meanings to the term 'image schema'. The advantage of choosing Langacker (1987) rather than, for instance, Lakoff (1987, 1989) is simply that its account is clearer and more systematic (although Langacker never explicitly connects his heuristic diagrams to imagery). Above all, the confusion over Lakoff's ICMs and other meaning variation structures can then be avoided. Unfortunately, this clarity comes at the cost of a lack of description of lexical meaning variation in Langacker.

Our model of the semantic pole of an activated lexical unit has been implemented in the form of:

I. a matrix of domains, ordered by centrality values. Langacker ranks domains according to their centrality in the meaning of a term. We have interpreted centrality to be a numeral ordering.

II. a list of *parts* ordered by their saliences. Salience is the ordering of parts and wholes according to their relative importance to the meaning of a term. Langacker does not have this ordering, although it is closely related to the centrality of domains.
III. a list of *wholes* ordered by their saliences

Take the lexical unit [KNIFE] as an example, as depicted in Figure 1. [BLADE] and [HANDLE] are clearly *parts* of [KNIFE].

[KNIFE] has [SILVERWARE] as a *whole*: [KNIFE] is one of the parts in collections making up [SILVERWARE]. But [KNIFE] can also have [CUT] as a whole, because [KNIFE] can be the agent (TR) part of the cutting process.

Parts and wholes like [BLADE] or [CUT] are themselves described with the same three elements. Their saliences, as well as the centralities, should be lexically stored but changed on activation to fit a context where that whole, part or domain is salient (this mechanism is not included in our immediate implementation goals).

Of course, just saying that the [BLADE] is a part of [KNIFE] is not sufficient. We must characterize this part-whole relation closer. For instance, the relative sizes of the blade and the knife must not deviate outside certain limits. The relative spatial position of the blade on the knife must also be correct, i.e. the blade must be correctly attached. There are two possibilities of implementing these constraints, both involving the predication (below).

Most of the content substance of a lexical unit resides in its *domains*, which collectively form its *matrix*. Typical domains are: color, 3D space, material, age, temperature, profession, emotion, etc. The matrix of a lexical unit (like [KNIFE]) allows the unit to involve all these aspects simultaneously and to be identified as the same entity across the domains (that this spatial form belongs to that color etc.).

Domains are made up by *dimensions*. The color domain has three dimensions (hue, saturation and brightness), while temperature has one dimension. 3D space has three dimensions. Profession and emotion domains probably have rather abstract dimensions.

The lexical unit *predicates* in each domain in its matrix. In the spatial form domain, [KNIFE] predicates a 1D-directed form. In the color domain, [KNIFE] may predicate a brown color. A predication such as a 1D-directed form or a semancolor picks a subset of the total number of dimensions in the domain and attributes values to the dimensions in that subset. For example, a 1D-directed form attributes a value to the spatial extension dimension.[2]

Dimensions and their values need not refer to measure units in the external world. Their main function is to serve in a mathematical region activation model underlying the representation. In all domains, (the dimensional values of) a predication is (are) equated with an absolute point or a variable prototype. The simplest case is that of points for color sensations and regions for color predications in color space. The model thus allows for variability of predication: there is a range of brown colors just like a range of 1D-directed forms. The variability is constrained by there being other regions than the predicated region in the domain. If the color dimension values (i.e. the color point) of the knife would move outside of the brown region, another region will be more saliently activated and there would be a conflict between that region and the brown predication in the lexical unit. All semantic constraints regarding predications are intended to be withheld by this region activation model.

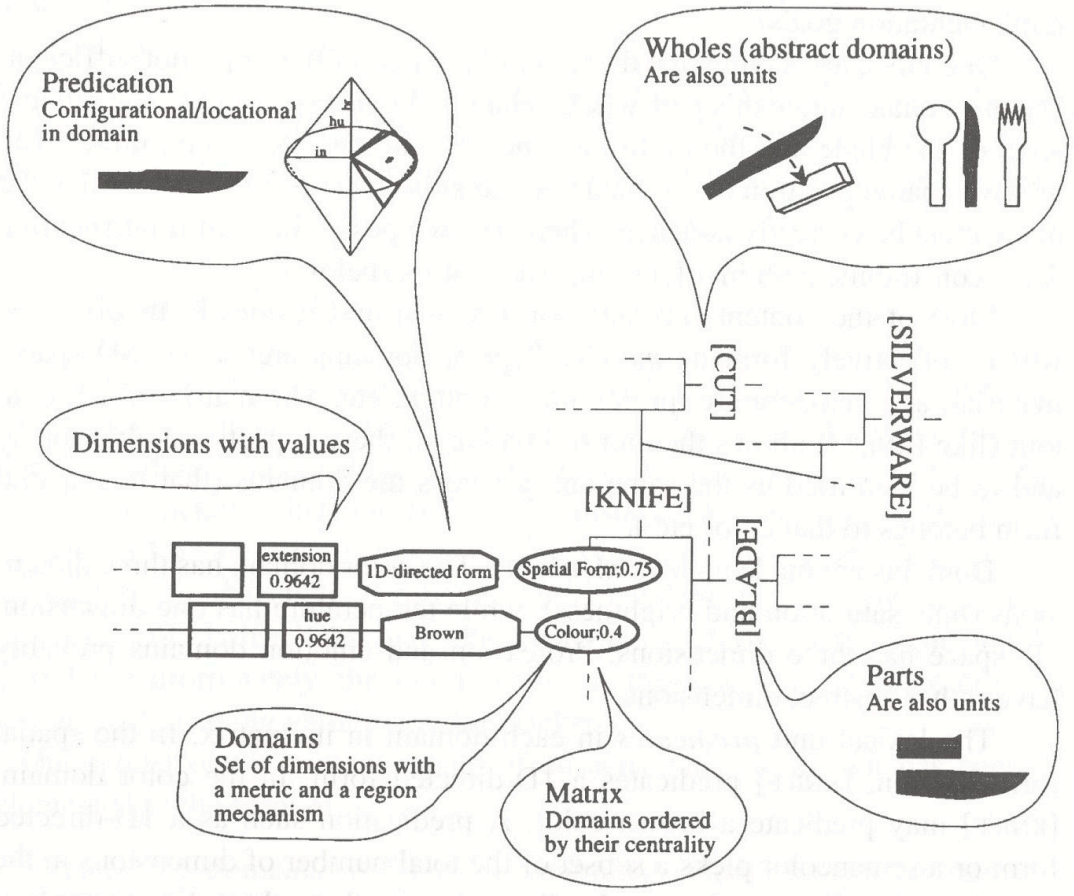


Figure 1.

The example here has concerned the representation of the thing unit [KNIFE], but the same representation form also applies to other schema types. A process like [CUT] also has its parts, wholes and domains. The domains of [CUT] are different from that of [KNIFE] in several ways, the most important being that [CUT] has temporal dimensions in its predications. It is these temporal dimensions that make [CUT] processual (which in turn has bearings on its possibility to combine with tense-forming lexical units).

The structures described this far are implemented, but we have not filled them with actual lexical content, i.e. the correct domains, predications and values of dimensions. In testing the composition processes to be described, we have used an 'unembodied' randomly generated filling in the lexical units. This is ultimately because we have no perceptual system in our computer, and hence nothing that can provide the correct domains and values of dimensions in the lexicon.

3. Semantic composition processes

The second part of the computational model consists of a semantic composition process (aiming at what Langacker [1987] calls 'accommodation'). The guiding principle behind the construction of the computational model is that understanders of human language perform semantic composition by means of image superimposition: individual lexical units are superimposed to form a composite structure.

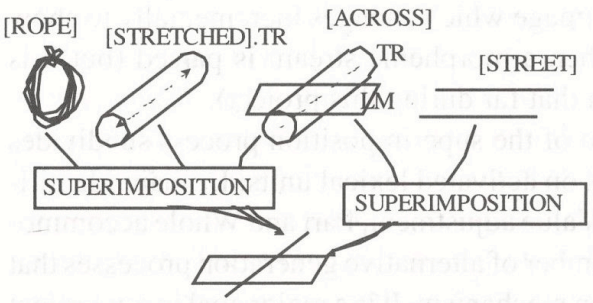


Figure 2.

Figure 2 shows how the single units [ROPE], [STRETCHED], [ACROSS] and [ROAD] are superimposed to form [(A) ROPE STRETCHED ACROSS (THE) ROAD]. It is those parts that have been set in a valence relation that can be superimposed, simply because they are then identical. For instance, the LM of [ACROSS] is identical to [STREET].

We can use image superimposition as the guiding principle, because lexical units having predications in different domains can be viewed as images. We normally think of images as something spatial, as in Figure 2, where the domain is 3D space and the predications are spatial configurations. But it is just as easy to think of 21° C as a point predication in the temperature domain or five hours as an extended predication in the time domain. In those domains, we can therefore use the same superimposition mechanism that is employed in the spatial domain.

Superimposition *constructs* the composite by making pieces from different lexical units identical and thus welding together the composite. Superimposition also *evaluates* the composite in that the pieces have to match. A bad match like "A ball stretched across the street" is detected by superimposition. This means that the content of a lexical unit is its own semantic constraints. During superimposition, constraint satisfaction is only a matter of seeing whether a superimposed unit violates the constraints of one of the other units that it is going to be identical to. (The most common case of valence formation is asymmetric: A relatively vacuous e-site is elaborated by a semantically richer elaborator. However, the existence of symmetric cases, as in Langacker [1991b:177], has led us to develop a model which allows for both symmetric and asymmetric valence relations.)

Of course, the computational model of the superimposition process operates on lexical unit structures of the sort presented above. Its goal is to mirror natural superimposition as closely as possible. One possible output from the implementation could be a screen image which changes incrementally to show how superimposition proceeds when a morpheme stream is parsed (but it is still unclear whether we can reach that far during this project).

The computational simulation of the superimposition process subdivides into a number of smaller processes on activated lexical units: Domain identification, Predication identification, Value adjustment, Part and Whole accommodation. Additionally, there are a number of alternative generation processes that alter the flow of the superimposition mechanism. It is a major goal in our project to investigate these processes and specify an invocation order between them.

The first of these operations is called *domain identification*. It takes the matrices of the input units and finds those domains that are present in all input units, such as 3D space in Figure 2 above. For instance, if we superimpose [SALLY] onto [UNDER].TR, the matrices of the two units and the composite would be:

[SALLY]: Emotional, Spatial, Age, Color, Profession, ...
[UNDER].TR: Spatial
→
[SALLY UNDER]: Spatial

That is, in the composite only the spatial domain remains.

[SALLY]: Emotional, Spatial, Age, Color, Profession, ...
[FURIOUS].TR: Emotional, Color
→
[SALLY FURIOUS]: Emotional, Color

If [SALLY] is superimposed onto [FURIOUS].TR instead, the composite probably contains the emotional and color domains.

[STONE]: Spatial, Material, Age, Color, ...
[FURIOUS].TR: Emotional, Color
→
[STONE FURIOUS]: Color

In Holmqvist (1993), a mathematical function for the calculation of the composite matrix is discussed. Here it is sufficient to point out that the domain centralities play the main role.

By reducing the number of domains in activated lexical units, domain identification helps in disambiguation, as the lower number of domains restricts the number of possible meanings.[3] But domain identification also serves for anomaly detection. If only a non-central domain remains in the composite, as in the third example, then the two input units are too different to be superimposed, and an anomaly will then be signaled.

Once the domains of the composite have been sorted out, the actual superimposition can start. In a common domain, each of the input units has a predication. These predications must now be joined into one, as in Figure 2. A number of operations combine to perform this welding task. Sometimes it is only a matter of seeing that two predications are both 1D-extended, as in the temporal domain examples of Figure 3.

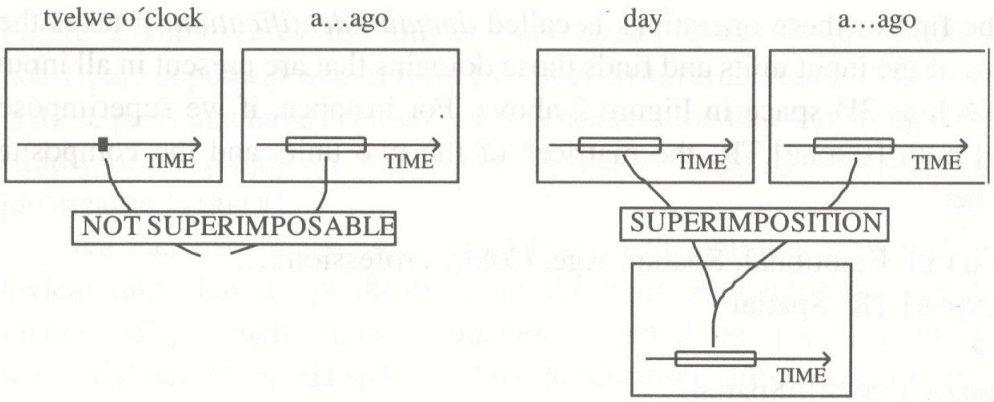


Figure 3.

However, it often happens that two predications cannot be superimposed until one of them has been turned properly, as in Figure 4.

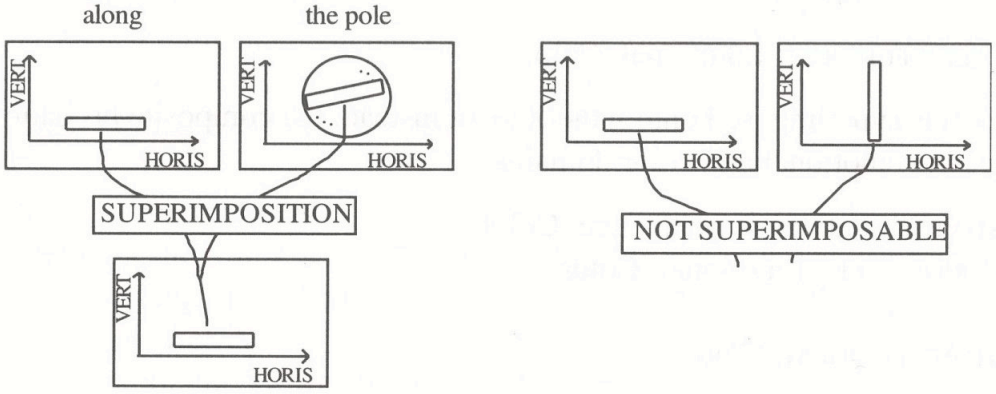
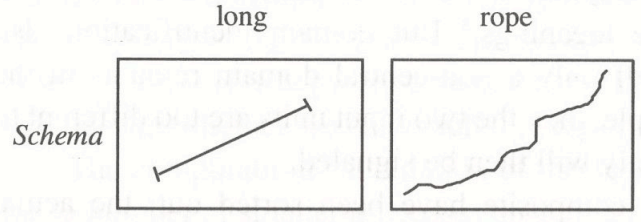


Figure 4.

Also, the superimposition must map the dimensions in one predication onto the proper dimensions of the other predication. When there is only one dimension of the same type, there is no problem. See Figure 5.



Dimensions	Extension	Extension Diameter Position(s) Direction

Figure 5.

But when there are several dimensions of that type in the other predication, as in Figure 6, another mapping principle must be adopted.

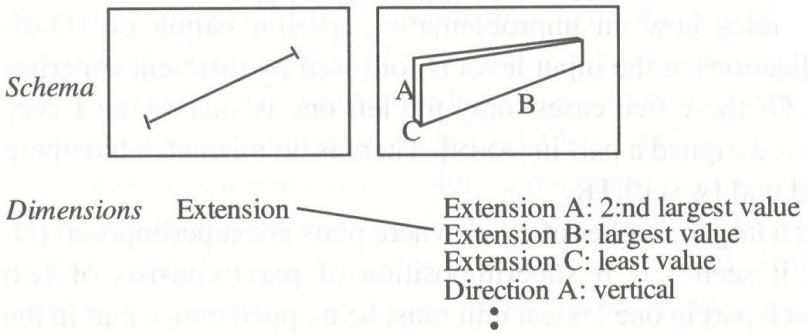


Figure 6.

The solution suggested in Holmqvist (1993) is based on the thorough investigation of the dimensional mapping principles constructed by Lang et al. (1991). Lang presents a complete model of spatial configurations based on how dimensional adjectives apply to them, and a part of our project is to translate Lang's model into a model that can function as a domain-general mechanism for superimposition of configurations (Holmqvist 1995). The turning and tilting mechanisms of Figure 4 are derived from this work. For figure 6, the application of Lang's model means that the extension dimension in [LONG] should map onto the extension dimension in [WALL] having the largest value.

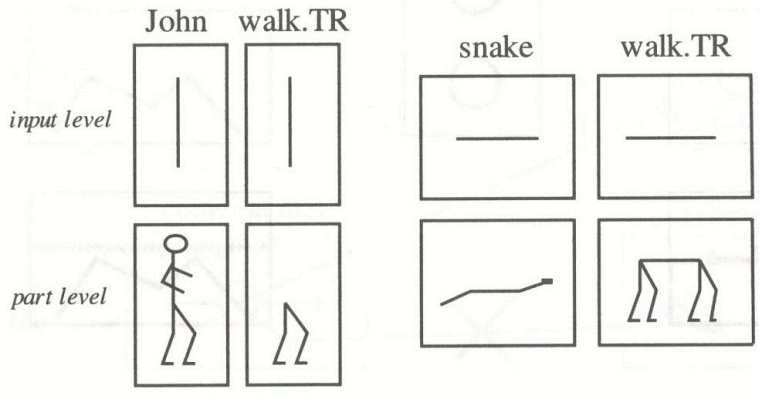


Figure 7.

Let us now turn to Part accommodation. Even if we are successful in mapping dimensions, turning and adjusting the predications at the input (matrix) level, there are also predications to be superimposed at the part level.

Figure 7 illustrates how an unproblematic superimposition of 1D-directed spatial predications at the input level is followed by different superimpositions of parts. Of these two cases, only the left one is successful: Every part of [WALK].TR is assigned a part in [JOHN]. There is no mismatch like there is between [SNAKE] and [WALK].TR.

If one looks at a larger number of cases where parts are superimposed (cf. Holmqvist 1993), it seems as if superimposition of parts consists of two operations: first, each part in one lexical unit must be mapped onto a part in the other lexical unit; second, each such pair of parts must be superimposable. Thus, the legs and feet of [JOHN] must be attached at the same place and have the same directionality and extension as the legs and feet of the generic walker [WALK].TR, as well as having basically the same domains in their matrices.

The wholes are in a way the opposites of the parts. Superimposing wholes in a similar way to that used above for superimposing parts is a mechanism for dealing with important but slightly subtle semantic effects. Take as an example the difference between "John used to row above the shoals" and "John used to row above the hills".

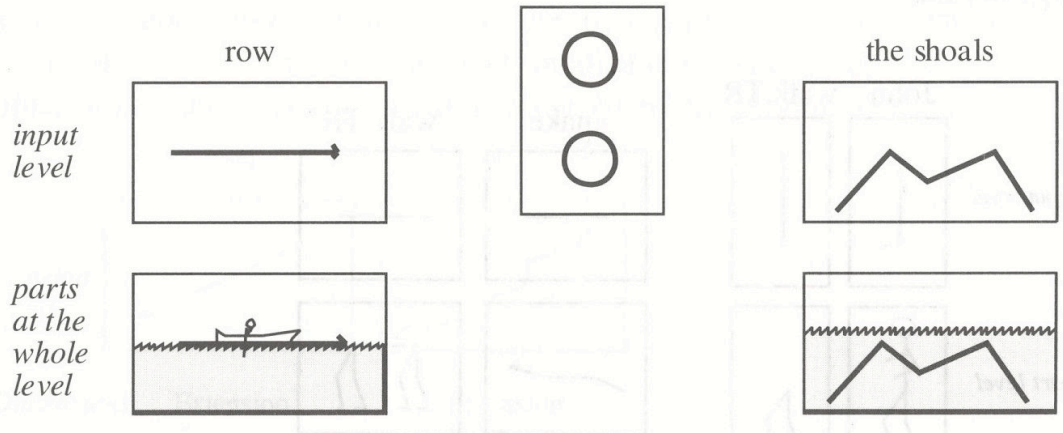


Figure 8.

Figure 8 depicts the situation. [ROW] predicates a processual motion path, placed spatially above the [SHOAL] or [HILL] predications by [ABOVE]. At the input level, this works fine in both cases, because there [HILL] and [SHOAL] have the same predications. However, when we enter the whole level of each, they differ. A whole could be said to be the surrounding of each of its parts. The whole of [SHOAL] thus also has [(SEA-)WATER] as a part (i.e. [SHOAL] and [(SEA)WATER] are siblings), while [HILL] has [AIR] as a sibling.

[ROW] has a number of sibling units, among them [(SEA-)WATER]. When [ROW], [ABOVE].TR and [SHOAL] superimpose, the [(SEA-)WATER] unit of [ROW] will coincide with the [(SEA-)WATER] unit of [SHOAL]. This is of course fine, since [(SEA-)WATER] easily superimposes onto itself. However, when [ROW], [ABOVE].LM and [HILL] superimpose, [(SEA-)WATER] will coincide with [AIR], and these two units do not superimpose easily.

The basic principle here is the same as when parts were superimposed (Figure 7): When lexical units are mapped onto each other, it is done because they coincide in the composite structure. When two lexical units coincide, it must be possible to superimpose them into being one unit. To simulate this is a central goal in our implementation project.

Along with the superimposition operations we have seen so far, Domain identification, Predication identification, Part and Whole accommodation, there are also a number of operations that alter the processing in various ways. They often appear after anomalies occur, but some seem to be invoked without having to be triggered by anomaly. In our project we mean to investigate the order of invocation between these semantic operations.

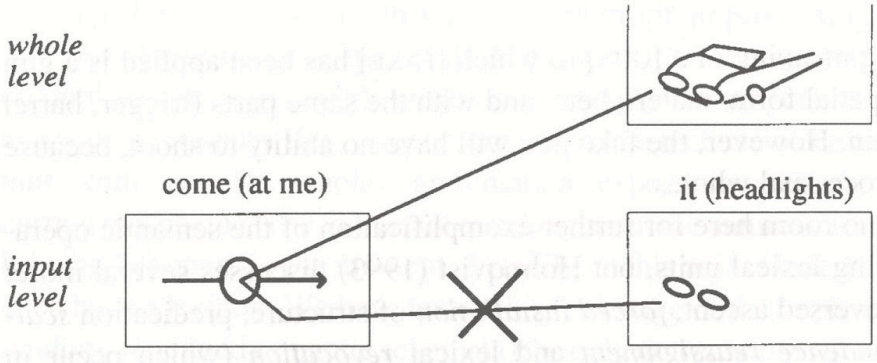


Figure 9.

One such operation is the *ascent*, which can be illustrated by the question what "it" in (1) refers to.

(1) I saw *headlights coming* straight at me, but I was able to get out of *its* way.

One answer would claim that it cannot be the headlights, because they superimpose badly onto [COME].TR. However, the whole [CAR] of [HEADLIGHTS] has the domains that allow it to superimpose onto [COME].TR. The ascent operation substitutes a lexical unit by its whole, as illustrated in Figure 9.

Another altering operation is the *split*. The splitting operation appears in different situations, where there is need to split or divide a lexical unit. The hedge [FAKE] performs a split of type 2 by cutting off the processual wholes of units, as in Figure 10.

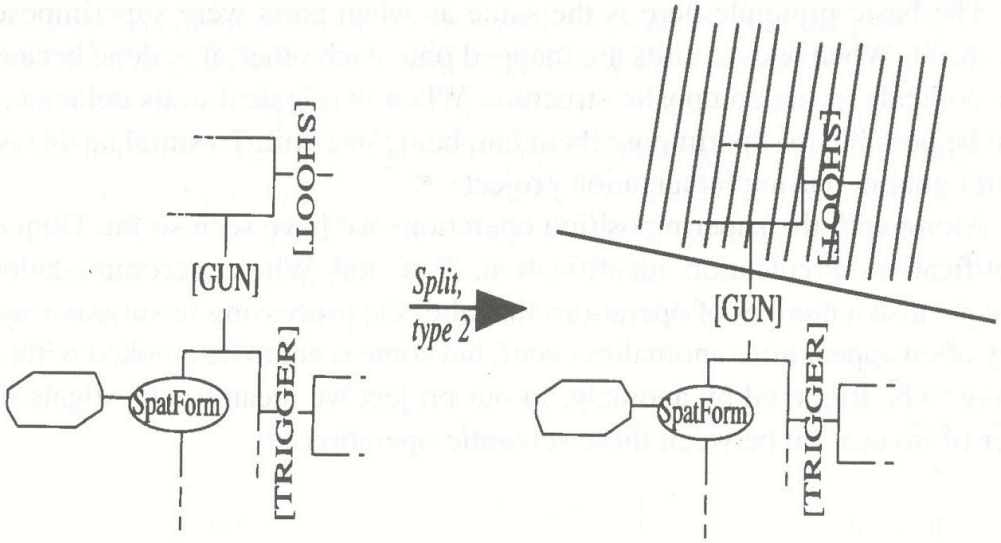


Figure 10.

The remaining meaning of a [GUN] to which [FAKE] has been applied is a gun of the same spatial form, material etc. and with the same parts (trigger, barrel etc.) as any gun. However, the fake gun will have no ability to shoot, because it lacks the processual wholes.

There is no room here for further exemplification of the semantic operations for altering lexical units, but Holmqvist (1993) discusses several more: *descent*, the reversed ascent, *forced installation* of structure, predication *scaling*, *correspondence reassignment* and lexical *revocation* (which occur in reinterpretations such as Garden-Path sentences) and *predication mapping metaphors*.

Let us finally point out that, in this process, there is no classical type-checking of the kind that the direct object of a verb must be a noun. An error like "Sam hit goes" is detected during *domain identification* because [GOES] has domains very dissimilar from the domains of [HIT].LM; the two units can not be superimposed easily.

4. Valence suggestion and the schema population

In order to put the semantic operations into a parser, we need mechanisms for incoming morphemes, suggestion of valence relations and management of the composite constructional schemata. These mechanisms, drawn from Holmqvist (1993a), have already largely been implemented and further described in Holmqvist 1994.

The input to the parser is a stream of morphemes. The morphemes may originate either from text or from speech, although at this stage we have not yet included phonetic information in the stream. It is important that the stream should be viewed as such, because the rate of incoming morphemes must be decided from the stream. It may not be the case that the parser decides the rate of incoming morphemes by what speed it is able to consume them at. The parser must consume morphemes at the *external* rate.[4]

Placing this simple demand on the time complexity of the parser has as a consequence that most present parser algorithms are excluded. One natural algorithm which satisfies the complexity demand is *incremental parsing* combined with *interrupts* caused by incoming morphemes. The algorithm that we are implementating can be described in the following way:

(t_0) A new morpheme just arrived. Interrupt all processing.

(t_1) Perform a generation shift. The result is that the processed lexical units up until the previous morpheme are kept in the previous generation. We set up a new generation for the current morpheme, in which we place only the lexical unit (with its parts, wholes, grammatical expectations etc.) evoked by the current morpheme. The important question about the nature of lexical evocation is being left open for the moment. Actually, at this point (June 1995) we do not even have a lexicon. We have tested this first stage of the implementation with random-generated semantic schemata (the only non-random structures were the grammatical expectations, which were entered along with the morpheme).

(t_2) The parser splits into two (pseudo-parallel) processes, of which the

first has been implemented: (A) Suggestion of new valence relations by grammatical expectations; (B) Accommodation of valence relations by means of simulated superimposition and, additionally, suggestion of new valence relations by semantic expectations. A suggested valence relation between two lexical units means that a new composite structure is entered into the current generation of the schema population, where it competes for space with a survivability value derived from how well it abides by grammatical *and* semantic constraints.

Section 2 of this chapter discussed how activated lexical units are represented. They are the structures undergoing processing. Section 3 described parts of the B-process in step t_2. We are now going to discuss the expectations that operate both in the A- and the B-processes. Their job is suggest valence relations, that is what (parts of) what units are identical to what (parts of) what units. Valence relations keep activated lexical units stuck to one another, that is they form composite structures.

Now let us sketch the valence suggestion mechanisms. There are two of them: *Semantic* and *grammatical expectations*, the latter being a special case of the former.[5]

Semantic expectations suggest a valence relation between two lexical units if their predications coincide in one or more central domains. We have already seen examples of this during part and whole superimposition (Figures 7 and 8): Parts and sibling units of the two inputs were mapped onto each other if they coincided.

The same kind of mapping seems to occur in many examples: In "John waded across the river", the liquid part of [WADE][6] will coincide with [RIVER] and the two will therefore superimpose, the result being that in "John waded across the river" there is only one liquid, not two.

As the examples show, an implementation of semantic expectations requires a previous implementation of the superimposition mechanism, in order for coinciding units to be possible to detect.

Grammatical expectations are a special case of semantic expectations in that the semantic expectation of, for instance a TR elaborator (agent) for [GIVE] is coupled with additional information that the TR elaborator can be found leftward of the <give> morpheme. Figure 11 shows a number of grammatical expectations stretched out in a morpheme stream.

The hexagons mark the position in the stream where the expectations have been evoked. The rectangles mark positions where it is likely that unit

can be found that suit the parts of the evocating lexical unit. For example, [MARY] seems proper to superimpose onto the LM1 ('recipient') part of [GIVE].

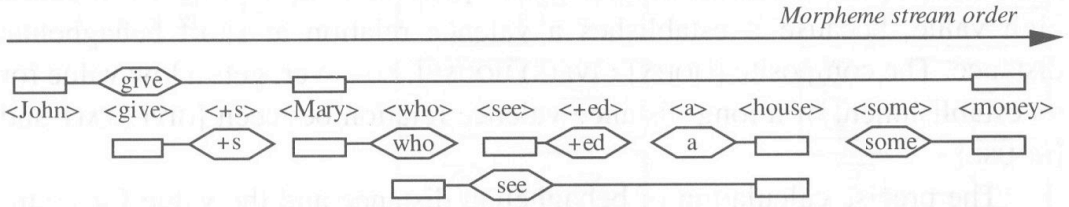


Figure 11.

Grammatical expectations are "stretchable". Finding the correct stretch is a crucial matter for suggesting the correct valence relations. The so-called Behaghel's principle claims a correlation between closeness of morphemes and closeness in valence relations. Based on this principle, Holmqvist (1993) develops an incrementally calculable measure describing the behaghelian distance between morphemes and lexical units that we have now implemented, Holmqvist (1994).

In parsing the morpheme stream of Figure 11, the distance from the evocating morpheme <give> evolves as in Figure 12. There we can see that, at the shortest distances, we find [GIVES] (distance 0), <+s> (1), <Mary> (1), [MARY WHO] (1) and [MARY WHO SAW A HOUSE] (1).

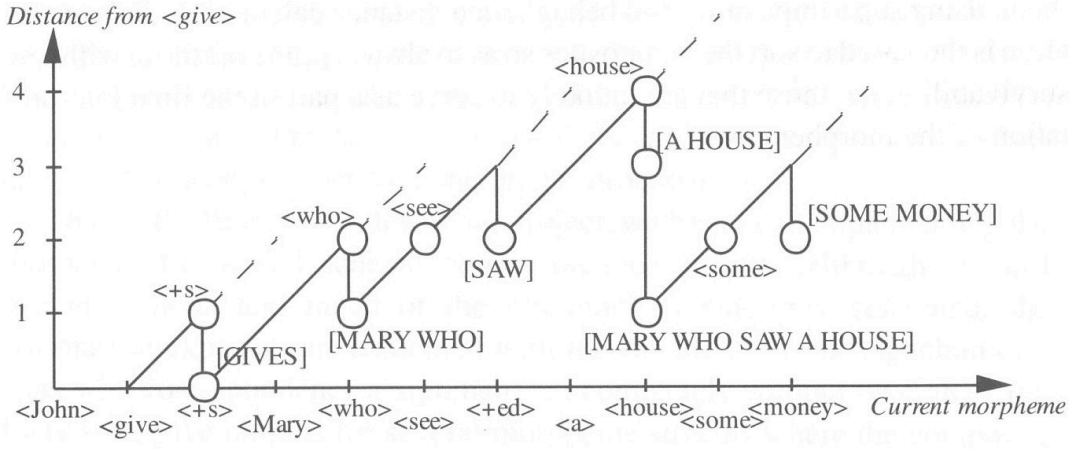


Figure 12.

The process A of t_1 below suggests all of those, and all other units as well, for valence relations to the LM1 of [GIVE]. The resulting composites are entered into the schema population, and a value for its grammatical obedience is calculated. For instance, the composite [(JOHN) GIVE(S) MARY WHO] gets a high value, because it establishes a valence relation at short behaghelian distance. The composite [(JOHN] GIVE(S) HOUSE], however, gets a low value for its establishment of a long-distance valence relation between [GIVE].LM1 and [HOUSE].

The precise calculation of behaghelian distance and the value for grammatical obedience can be found in Holmqvist (1994). However, once the value is calculated, it is used in the calculation of the survivability value of the composite. As the survivability value also draws on the semantic superimposability of the units, it weighs grammatical and semantic obedience against one another. The result is that the parser can not only make the correct grammatical assignments of valence relations for most morpheme streams, it will also be able to find many semantically correct but non-grammatical valence relations.

All lexically evoked schemata and composites are kept in a *schema population*. Together with grammatical expectations and the parameters (distance, density and superimposition result), each entry in the schema population models corresponds to what Langacker (1991a) calls a grammatical construction.

The schema population functions as a sort of semantic short-term memory, where composites are stored during composition (see Figure 13). Because of the large number of composites being suggested, the parser continuously evaluates them, using superimposition and behaghelian distance calculation. This evaluation is then used to sort the composites so as to always prune off those with low survivability, i.e. those that are unlikely to serve as a part in the final interpretation of the morpheme stream.

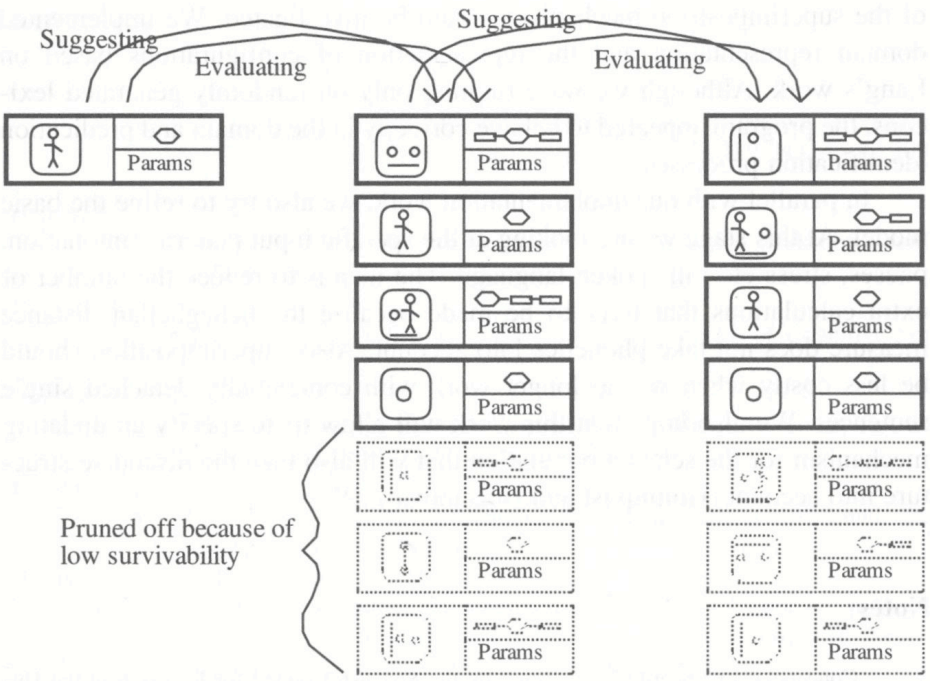


Figure 13.

5. Project goals and current project work

The project 'Conceptual Engineering' has as its goal an experimental implementation of the model described above. In particular, we are interested in investigating the processual order between the superimposition mechanisms in the implementation, such as domain identification, predication mapping, ascent etc. It is also highly relevant that the valence suggesting mechanisms and the schema population have the proper processuality.

During the first six months of our project, we have been implementing the morpheme stream and schema population mechanisms. Although we still have no lexicon and much of the semantics is randomly generated, the grammatical expectations combined with the calculation of behaghelian distance, with correspondence assignment but completely without type-checking of any sort, gave outputs for several morpheme streams where the composite with the highest survivability was composed of the correct valence relations.

This mechanism then served as the frame within which implementations

of the superimposition mechanisms could be investigated. We implemented domain representation and the representation of configurations based on Lang's work. Although we were running only on randomly generated lexicons, the program appeared to behave correctly in the domain and predication identification processes.

In parallel with our implementation work, we also try to refine the basic model. At this stage we are looking at the specific input patterns (intonation, pauses, stress etc.) in spoken language. The idea is to reduce the number of extra calculations that have to be made because the behaghelian distance measure does not take phonetics into account. Also, superimposition should be less costly when we no longer work with contextually detached single sentences. We also hope that this work will allow us to specify an updating mechanism for the schema population that will also take the discourse structure into account (Holmqvist and Holsánová 1995).

Notes

1. 'Conceptual Engineering', sponsored by the Swedish Council for Research in the Humanities and Social Sciences for 1993 – 1996. Currently the author and Jana Holsánová are working in the project.

2. Holmqvist (1993a) gives a survey of different predications and their dimensionality. For spatial form predications, Lang et. al. (1991) gives a similar and very systematic formalization, which can be translated into the more general framework of predications and region activation, see Holmqvist (1995).

3. This connects to a number of questions concerning lexicon and lexically activated structures involving schema polysemy that have been left out here for the sake of simplicity.

4. This means that the parser should have a roughly linear time complexity in terms of the number of incoming morphemes.

5. Both of these are based on Langacker's (1987) conceptual dependency. Harder (1993) suggests the existence of an additional functional dependency which could be developed to a similar valence suggestion mechanism.

6. For the existence of this liquid part, cf. "John waded across the street".

References

Glasgow, J.I.
1993 "The Imagery Debate Revisited: A Computational Perspective", *Computational Intelligence* (Taking issue position paper), Vol. 9, No. 4, pp. 309-333.

Harder, P.
1993 "Cognition, Interaction and Structure", presented at the workshop *Cognitive Semantics*. NAL 1993: Göteborg University

Holmqvist, K.
1993 *Implementing Cognitive Semantics*. Lund: Department of Cognitive Science.

Holmqvist, K.
1994 "Conceptual Engineering I: From morphemes to valence relations". LUCS 28. Lund: Department of Cognitive Science.

Holmqvist, K.
1995 "Two Dimensional Representations Compared", manuscript, Lund: Department of Cognitive Science.

Holmqvist, K., and Holsánová, J.
1997. "Focus Movements and the Internal Images of Spoken Discourse" in W.-A. Liebert, G. Redeker, L. Waugh. "*Discourse and Perspective in Cognitive Linguistics*", Amsterdam, Philadelphia; John Benjamins.

Lakoff, G.
1987 *Women, Fire, and Dangerous Things*, Chicago: University of Chicago Press.

Lakoff, G.
1989 "Some Empirical Results about the Nature of Concepts", *Mind & Language*. Vol. 4. Nos. 1 and 2 Spring/Summer 1989. 103–129.

Lang, E. with Carstensen, K.-U. and Simmons, G.
1991 *Modelling Spatial Knowledge on a Linguistic Basis*. Heidelberg: Springer-Verlag.

Langacker, R.
1987 *Foundations of Cognitive Grammar, Vol. I*. Stanford: Stanford University Press.

Langacker, R.
1991a *Foundations of Cognitive Grammar, Vol. II*. Stanford: Stanford University Press.

Langacker, R.
1991b *Concept, Image, and Symbol*, Berlin. New York: Mouton de Gruyter.