Computational modeling of Force Dynamic terms

Jonathan Segal

May 21, 1995

In this paper I will attempt to:

- briefly describe a bodily-based, experiential, cultural, metaphoric theory of meaning.
- Discuss how (and why) we try to model aspects of said theory computationally
- Discuss the preliminary model for force dynamic terms I am working on, based on an Aspect model of Srini Narayan

1 Introduction

The main question we are all trying to answer is this: "How does language (and thought) work?"

This is a complicated process, and to begin to attempt to explain it, we break it down into several pieces. The first thing we do is discuss what we mean when we say a person thinks something, or, what it means that a word means something. These are different, yet related questions. We will focus on the latter for the remainder of the paper.

2 Experientially Based Meaning

What do we mean when we say that a word (or grammatical construction) "means" something? As do (Lakoff 1987), (Johnson 1987), (Quinn 1987), *etc.*, we view language use and language acquisition, and thus word and construction meaning, to be inextricably interwoven with world knowledge and bodily and cultural experience, and how we store and access this knowledge. Let us consider:

From the day they are born (perhaps earlier), people have experiences from sensing and interacting with with the world. They slowly make order out of them – they make generalizations, notice patterns, generate expectations, categorize, *etc*. We are genetically predisposed to noticing certain types of patterns, and reacting in a certain way (*e.g.*, newborn babies will smile when smiled at), and though this is not important

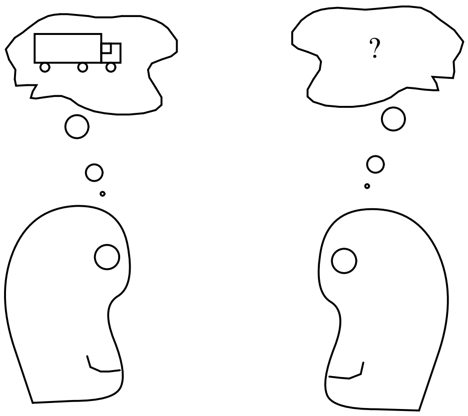

Figure 1: What we're trying to answer: How does this work? How does person A transmit his idea to person B?

from a meta-theoretical point of view, When we actually are forming our theory of meaning, we will need to know what structures are "built-in," and in fact an important part of our theory is that what these structures are is essential for determining a theory of (human) meaning[1], for it is this "built-in" or "bodily-based" knowledge which provides the structure around which all other knowledge is acquired, stored, and referred to. This will be discussed in more detail later.

Let us consider experiences which people generalize over: At the "lowest" level, we have sensory stimuli – i.e. visual, tactile, kinesthetic sensations which we generalize over forming things which Lakoff and Quinn call "Schemas" (sometimes "Image Schemas"). To avoid terminological confusion throughout this paper, we will refer to these experiential/gestaltish schemata (or experiential representations) as K-schemas (for Knowledge schema). This same generalization/categorization process is used to generate these bodily-based schemas as the "higher level," experiential or cultural schemas. For example, based on our personal bodily experience interacting with the world, we form a general K-schema for the concept CONTAINMENT, or for PUSH. When we see lots of images of dogs, over time, we form a general K-schema for DOG. After a number of years, if we live in America, we will most likely form a K-schema for THANKSGIVING. If we are a practicing computer programmer, we will form a K-schema

[1]In some sense the expression "theory of meaning" is ill-defined, for in itself it presumes that "meaning" can exist in a void. One of our premises is that meaning does *not* exist in a void – it is not enough to say that word A means B: we must say that word A means B *to person* C. Perhaps we should say "theory of *human* meaning."

for COMPILER. Note: a non-practicing computer scientist is certainly capable of knowing what a compiler is and does (simply, it translates a computer program from a form that a person can understand into a form that a computer can understand). But if one does not ever use a compiler, one will not truly *understand* the concept in the same way that one who uses it will. (This view of K-schemas is very much like Quinn's view of the word cultural model (e.g. in (Quinn 1987)), and is not unlike Lakoff's use of the word "concept.") But we can also generalize over more abstract things. For example, we notice that whenever we want to do something, perform some action, these actions have (usually) a beginning, and an end. Some actions happen again and again, and so on. From these generalizations, too, we form K-schemas (these particular, *aspectual* K-schemas will be discussed in more detail below).

Where, then, does language fit into this view? Individual language elements (be they words, specific morphemes, or grammatical constructions) index into this "K-schema storehouse" which we have. As we learn these K-schemas, we also learn constructions[2 3] which map on to them. Since we have largely shared experiences (and culture and biology) (particular experiences for the most part are not "shared" between people, but experience *types*, e.g. EATING, are) and a large shared language base to refer to these representations of shared experiences, it is possible to communicate. By this view, communication happens roughly like this: the speaker has some idea she wants to convey to the listener. The speaker has a large base of K-schemas (some conceptual, some linguistic, many of them shared) which, when put together, can describe the situation. Each of the K-schemas has an associated construction, and there is a shared understanding of how the constructions are linked to the shared K-schemas, and how the way in which the constructions are combined relates to the way in which the K-schemas are combined. The hearer hears the linguistic utterance, and can, using the same shared knowledge base, construct a representation not unlike the idea the speaker had.

We should also note that this whole language/concept learning process is rife with feedback – language provides one means for informing us what information we should be focusing on when forming these K-schemas. As we learn language for describing the world, we notice what sorts of things the language we use distinguishes, and what it doesn't.

2.1 Metaphor

K-schemas are not isolated from each other. As discussed in detail in, among other places, (Lakoff & Johnson 1980), (Lakoff 1987), and (Quinn 1987), we use knowledge/experience/K-schemas from some domains to help structure (i.e. form K-schemas) and reason about, and thus choose words (constructions) for other domains. In the prototypical case of this, K-schemas from more basic, more grounded domains

[2]For the purposes of this paper, I will use the general term "construction" to include both words, morphemes, and grammatical construction, akin to Langacker (Langacker 1991)

[3]Note that language itself is something we experience, so we of course form generalizations and K-schemas about language, too.

(e.g. bodily experience) help structure K-schemas from more complex domains. In describing this, Lakoff uses the expression "we understand the more abstract domain in terms of the more concrete domain," but perhaps it would be more accurate to say (and this is more in line with what Quinn says) that if we notice (consciously or unconsciously) partial similarities in structure between K-schemas in different domains, we can both: a) posit extensions in the more simply-structured/less well organized [4] K-schema (i.e. the "target domain," in Lakoff's terminology) which are structured similarly to the corresponding existing structure in the "more organized" K-schema (the "source domain") and concomitantly b) use words which index the source domain K-schema to index the corresponding target domain K-schema. The latter is an instance of "linguistic" metaphor, while the former "conceptual," but they both derive from the same process.

Note that there is nothing terribly extraordinary about noticing similarities between K-schemas – K-schemas themselves are formed as we note similarities and make generalizations. Just as we note similarities and form generalizations in formation of K-schemas, we can note similarity between K-schemas in different domains, and form generalizations thereby. A strong claim (which I am not necessarily making here, but which may be true) would be that it is exactly the same mechanism by which we form categories (and their representative K-schemas) with which we also make metaphorical links and form metaphors ("meta-Ks?").

2.1.1 Space

Space is a domain we are very familiar with – from the day we are born perceive it, move within it, push through it, get pulled or carried along through it, have things fly through it towards us, throw things though it away from us, or towards someone/something else, etc. We thus have a large knowledge base of "space-based" K-schemas which we reason with and about, and it provides a rich source of "reasoning material" which we can generalize/metaphorize to other domains.

2.1.2 Action

One additional key point on our theory of semantics, which follows from what is said above but bears repeating – we believe (and in fact claim that it is obviously true), that since action is a large part of our experience (i.e. we are constantly acting on and being acted upon by the world), K-schemas for action form a prevalent grounding for semantics.

[4] This terminology is unsatisfactory, since there as of yet exists no good theory saying what it means for a K-schema to be "more well organized" or "less simply structured."

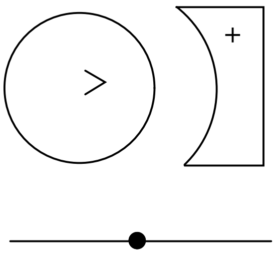

Figure 2: A
Talmy-esque Force Dynamic Diagram (after (Talmy 1988))

3 Computational modeling

The issue of computer modeling is independent of the particular theory of semantics being espoused (e.g. the one summarized above) – it is really a methodological rather than theoretical paradigm. Modeling, though is especially important when the theories under question are inherently difficult to model. Why do we try to model? There are two main reasons:

- We are forced to be precise. Many of theories of cognitive semantics (for what I outline above is really a generalized description of a large amount of current cognitive semantics research), though they provide incredible insights into meaning, are so "high-level" that it is impossible to say what they theorize is *really* going on in the brain. Though we can gain great insights from a Talmy-esque force dynamic drawing like that of figure 2, unless we specify enough what is going on to run a program which can take new, heretofore un-thought-of data and produce correct (i.e. the same as a person) results, we can't really say that we have completely described the situation. [5]

- Equally importantly, computational models provide a test-bed for our theories, and allow us to see what areas or which data our theories do not account for, and thus in which areas they need to be emended. They can show us where our theories are just plain wrong, and also show us where we are on the right track. [6]

[5] I do not want to detract at all from Talmy's work, or the work of any other cognitive semanticist in this discussion. I am merely saying that computational modeling must be a direction these ideas are carried for them to be truly complete.

[6] Note that this is superficially at odds with computer engineering, which has the paradigm: you are given a task, and want to find the most efficient way possible to solve this task, without worrying about how similar the solution is to what people may or may not do. I say superficially because the tasks we are trying to do are precisely the ones which people do naturally and effortlessly, and at which modern computers (traditional

As I said above, the whole *idea* of computer modeling is orthogonal to the type of theory being modeled. For example, a "traditional objectivist," feature-based theory of semantics is actually quite simple to model computationally – the theoretical primitives have close correlates with well-known and well-understood computational data structures. This is perhaps part of the reason for these theories' prevalence. The primitives of the class of theory outlined above (i.e. K-schemas), on the other hand, have no obvious computational correlate. This helps explain the dearth of computational cognitive semantic models. For our task is significantly harder – not only do we need to figure out the details of the theory so that all of the data can be accounted for, we also need to figure out (or invent) the computational objects and methods which best suit the theory. Then, when we look at our results and need to adjust the model or the theory to properly account for the data, we have a difficult time determining whether the errors lie in the theory itself, or in the implementation of the model.[7]

4 Modeling Aspect and Force Dynamics

Now that we have established the broad theoretical and methodological framework under which we are working, we can describe the particular problem we are trying to solve, and the preliminary attempts we have to solve them.

We are attempting to construct a computational model of the semantics of force dynamic terms, *à la* (Talmy 1988). It is based on the model Srini Narayan proposed in (Narayan 1994) and (Narayan 1995), and is part of the L-zero project, described in (Feldman *et al.* 1993).

We begin by giving a brief description and summary of work Srini Narayan (Narayan 1994), describing a computational model for the semantics of aspect. Then we will talk about force dynamics, and how as a preliminary attempt to model the semantics of force dynamic terms we can extend Narayan's model.

4.1 Aspect

Consider any action. Usually it has a beginning, when the action starts. It usually has an end, after which the action is completed. It may or may not be repeated. It may be habitual, *etc.* According to our general theory, people can notice these different phases of action, and can generalize across actions (i.e. starting can be generalized to any action, etc.). Many of these generalizations are described linguistically, either grammatically or lexically. For example, consider the differences between the following sentences (which are all in the past tense):

1. India fell into recession.

2. India was falling into recession.

AI, etc.) fail miserably.

[7]Needless to say, this is not an easy task.

3. India had fallen into recession.

4. India used to fall into recession.

5. India was on the verge of falling into recession.

6. India fell back into recession.

Each of these sentences codes the same action (that of India falling into recession), but they each focus on different aspects of the action. Sentence 1 focuses the action as a whole. Sentence 2 focuses to the process of the falling while it was happening. Sentence 3 focuses on the fact that the action was completed. Sentence 4 focuses on the fact that the action was habitual, and so on.

Different languages code different aspectual states differently – some languages (like English) require all verbs in all sentences to have some sort of aspectual marking, others (like Hebrew) don't syntactically mark aspect at all. (and thus, due to the feedback nature of language and K-schema formation mentioned above, perhaps those speakers don't make those generalizations across actions with as much "strength" as do English speakers). The key point is that our aspectual model, as any model, must, when complete, be able to capture all the generalizations which might be made in any language, and, when in the context of a compete system, be able to be properly "tuned" to capture whatever generalizations any particular language needs.[8]

Let us consider this in light of the communication story described above: a speaker has a thought she wants to get across, and has a (shared) storehouse of aspectual distinctions. She chooses which aspect of the action she wants to focus on, and can draw from her (shared) linguistic knowledge how to specify this so that the hearer will get it. The hearer decodes the speech, notices the aspectual marking, and computes the "aspectual meaning" of the phrase.

But what is the "aspectual meaning" of a phrase? This we have a theory of: Our working theory states something stronger than simply that we have some representation of the generalization of aspectual states across actions. We hold that this representation is in some way verisimilitudinous to the action itself. In other words, we claim that the representation is itself at some level an active, moving representation. This is part of our "body-based" claim. We can't claim to know the exact details of how this is done (in fact, we know ourselves to be a long way from this knowledge), but we can try to model the general idea of "active" or "motive" schemas, and determine what inferences we can derive from the model which coincide with inferences people naturally draw.

This approach is appealing because the inferences we draw about the actions "come for free." That is, by the very act of the schema executing itself, inferences which people naturally and easily draw are obvious, or rather, inherent to the schema. For example, if our model for the semantics of "India used to fall into recession" in some way actively performed an action over and over again, by the pure fact of this operation we could infer that the action occurred more than once (for it is in fact infelicitous to

[8]Further, when we extend the model to a learning model, it must be able to properly learn which generalizations to make given appropriate input. See, for example, (Regier 1992) or (Bailey 1994).

use the English "used to" construction to refer to an action which only occurred once). This is as opposed to a purely declarative, feature-based model of aspect, where one would need rules and such to manipulate feature values to compute results. In the above example, the sentence would have some aspectual marker [+habitual], and in order to perform the inference there would need to be some sort of rule system, with a rule "[+habitual] implies *happened more than once*".

We clearly have no clue as to how this is really done in the brain, but we can do is to try to test this general hypothesis of active schemas with an active model as a representation for aspect.

4.2 A situational model

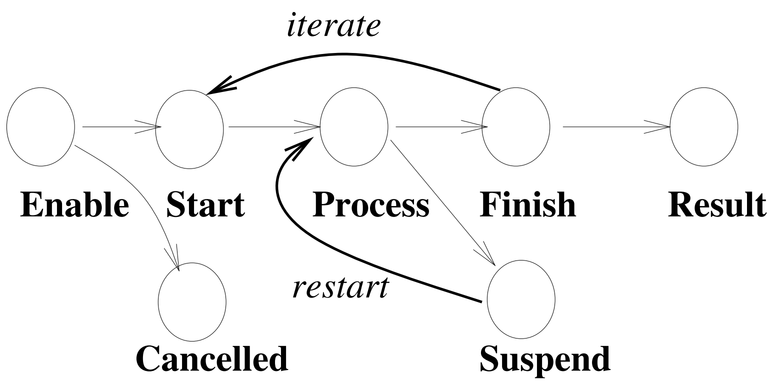

Figure 3: A generalization over actions

Figure 3 shows a abstraction of the generalizations we make over actions. Any action we undertake has a start, a finish, a result. It may be cancelled, suspended, resumed, or iterated. Of course, each action has its own, specific nature, listed as the "process." For the sake of the aspectual distinctions, which generalize over all actions or processes, we are not interested in what the particular process is. In modeling, we assume that for each verb, the process is some model or representation of the particular semantics of that verb. Working on this, (Bailey 1994) talks about modeling the semantics of a particular class of verbs, and (Narayan 1995) talks about metaphoric semantics.

So what do we need from a computational model for aspect? We want the model to be active, *i.e.* it must execute, and from this execution we can derive inferences. We also need to be able to embed the model in a system where we can bind appropriate parts of the model to linguistic material, thus effecting constructions. In addition, if possible we would like the model to have well understood computational properties. I will now briefly describe the model proposed in (Narayan 1994).

4.3 Petri nets

To model computationally aspectual semantics, we use Petri nets[9]. Petri nets are models for "procedural systems that are characterized as concurrent, asynchronous, distributed, parallel, nondeterministic, and/or stochastic" (Murata 1989). They are finite automata, composed of **places**, **transitions**, (possibly **weighted**) directed **arcs**. Each place or transition can contain zero or more **tokens**. The arcs link places to transitions and transitions to places. A place with an arc leading to a transition t is said to be an *input place* of t, while a place with an arc leading from t is said to be an *output place* of t. Formally, we say that Petri net is a 5-tuple $\{P, T, F, W, M_0\}$ where:

- $P = \{p_1 \ldots p_m\}$ is a finite set of places
- $T = \{t_1 \ldots t_n\}$ is a finite set of transitions
- $F \subseteq (P \times T) \cup (T \times P)$ is a set of arcs
- $W : F \rightarrow \{1, 2, 3, \ldots\}$ is a weight function
- (optionally) $M_0 : P \rightarrow \{0, 1, 2, \ldots\}$ is the initial marking
- $P \cap T = \{\}$ and $P \cup T \neq \{\}$.

The operation of a Petri net is quite simple, based on the following rules (as illustrated in Figure 4):

1. A transition t is *enabled* if each input place p of t contains (or "is marked with") at least $w(p, t)$ tokens, where $w(p, t)$ is the weight of the arc from p to t.
2. An enabled transition may or may not *fire* (depending on whether or not the event actually takes place)
3. A *firing* of an enabled transition t removes $w(p, t)$ tokens from each input place of t, and places $w(t, p)$ tokens to each output place of t. (note that $w(t, p)$ need not equal $w(p, t)$).

From these building blocks, we can construct a detailed Petri net model of Figure 3, as is shown in Figure 5.

Here we have a computational *schema*, represented by a Petri net, which can abstractly represent action operations. If we specify how we can link certain configurations of the net to aspectual states and these to linguistic constructions, we will have a computational model for the semantics of aspect. Work is being done (in the spirit of (Narayan 1995) and (Jurafsky 1992)) on the implementation of this linking, which we represent with the Saussurian form/meaning diagram of Figure 6. We will be focusing on the schematic representations of the semantics, and how those should correspond to the K-schemantics of the constructions. For example, consider Figure 6. The entire

[9] For a complete, formal discussion of Petri nets, see (Murata 1989) or (Reisig 1982)

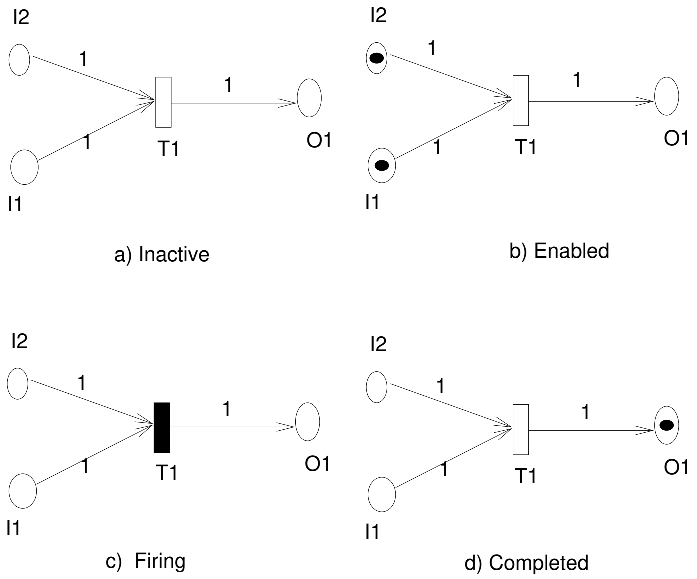

Figure 4: Basic Execution States of a Petri Net

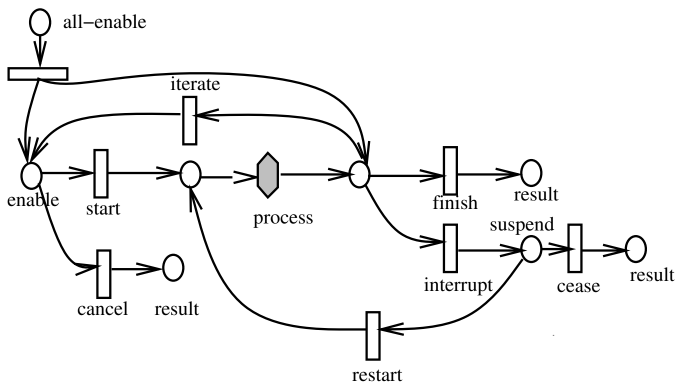

Figure 5: Generalized Aspectual Petri Net

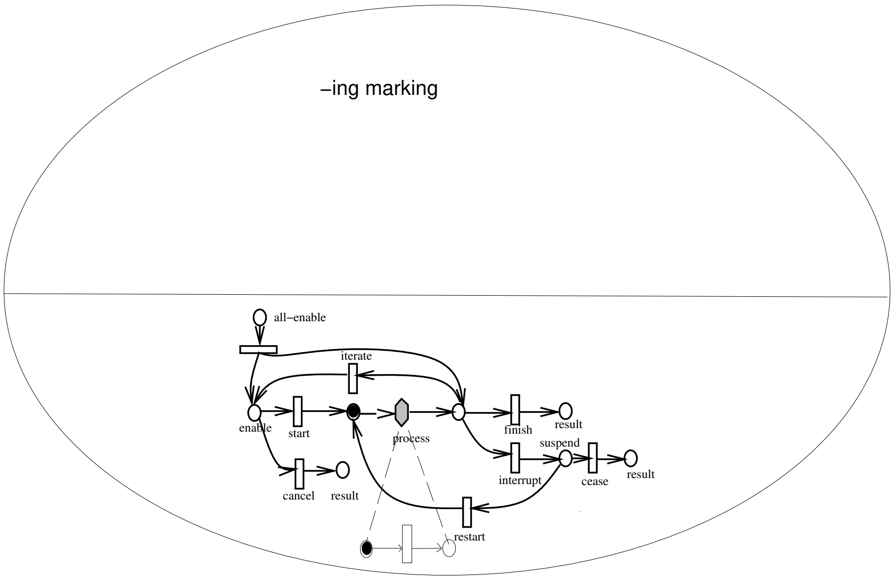

Figure 6: English Progressive Construction

diagram signifies a representation for a construction, the "English Progressive" construction. The top half is shorthand for the syntactic information necessary to represent, in this case, the English progressive. We notate that with **-ing marking**, though the actual syntactic implementation is left for later work (see (Jurafsky 1992) for a sample of a syntactic computational model). We will focus our examination on the bottom half of the diagram, which represents the semantic information.

4.4 Representing the Semantics

We adopt a unification model, as per (Langacker 1991) for our description of the semantics portion of the constructions. The actual computational details of how to implement this have not yet been worked out, but the general idea is as follows:

We assume that the semantics portion of every verbal construction contains, as part of its action schema, a Petri net like the one displayed in Figure 5. The schematic specifications in the grammatical constructions (e.g. the aspectual constructions, like Figure 6) denote structural patterns (usually patterns of token marking) which need to match the markings as the entire complex schema is being built. For example, the construction of Figure 6 denotes that the action schema should have tokens marking that the process is under way, corresponding to the semantics of the -ing suffix. Figure 7 shows an example of the specification of a schema for sentence with perfect aspect, like "India has fallen into recession" or "Mary has written the paper." This indicates that the action has been completed, and that the action's result are still present – or, in the token-flow model, that the *result* node has a token.

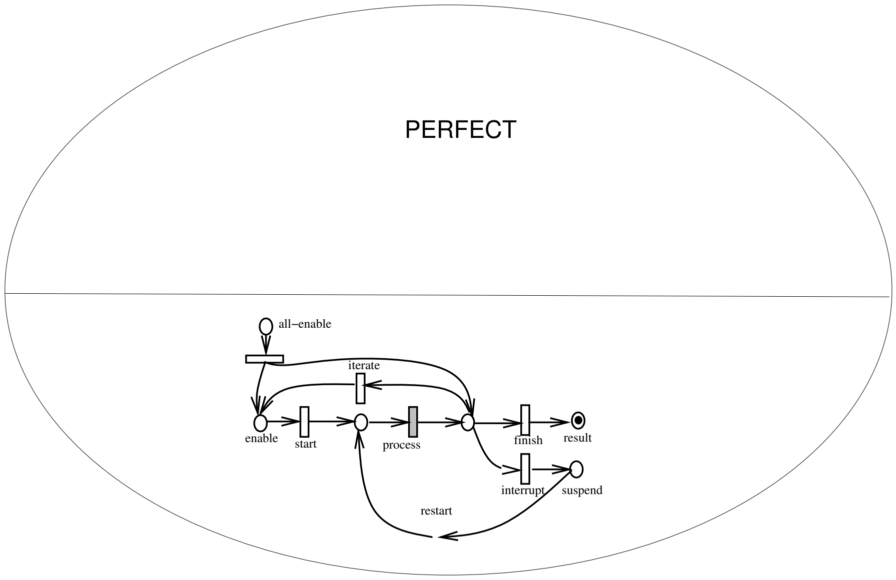

Figure 7: Perfect Aspect (Construction)

To summarize, we use the Petri Net formalism to build a schematic model of generalized action, which we use as the basis for a computational semantics for verbs. Constructions link grammatical information to token configurations of the Petri nets. For the case of Aspectual constructions, they specify configurations of the generalized aspectual schema.

5 Extensions to the model – Negation and Force Dynamics

There are two main extensions to this model will be discussed: representing "simple" negation, and representing the semantics of force dynamic terms (force dynamic terms as a linguistic category are generalizations of the causative, as described by (Talmy 1988)). We will discuss the modeling of simple negation first, then of force dynamics.

5.1 The General Approach

Recall that Token-flow is the "life-blood" of Petri nets. Tokens flow through the nets, simulating the actions. It is via token flow that actions are simulated, and linguistic representations are derived, therefore it will be through manipulating token flow that we represent negation and the force-dynamic/causative terms. [10]

5.1.1 Negation

What is the semantics of a negative sentence? *I.e.* if we say "Bill did not go to the store", how does this differ from saying nothing? Asserting that an action did *not* occur is qualitatively different from not asserting that the action did occur. Mentally, we set up the potential action of Bill's going to the store, and then deny that this action actually happened. To model this in the Petri-net token-flow paradigm, this means that we set up the schema for Bill's going to the store, but do not allow tokens to flow into and thus activate the process. Technically, we can do this with a "token sink," as illustrated in Figure 8

The construction indicates that when the negation node is present, acting as a token sink preventing a given process's schema from being activated, that schema exhibits semantics corresponding to being syntactically negated. In the figure, if the "process" node refers to the action of Bill's going to the store, when the "negation node" is present, the actual representation of the action Bill going to the store has been summoned[11], yet the action is not activated. Thus we have an active model emulating the semantics of the negation.

5.1.2 Force Dynamic Terms

We want to model force dynamic terms, i.e. terms of generalized causation, as described in (Talmy 1988). Sample sentences whose semantics we want to capture with the model are as in figure 9.

In all of these sentences, there is an *Agonist* John for whom an action of "going to the store" is posited, and an *Antagonist* Bill who exercises (or potentially exercises, or, if negated, fails to exercise) force over the Agonist. The Antagonist may prevent, cause, help, or hinder the Agonist (or fail to, if negated). "Let" has an additional twist: the Antagonist has the power to prevent the Agonist from performing the action, but does not exercise that power.

Before we examine the schematic representation of the force dynamic terms, we need to extend the Petri net formalism a little. We need to add *inhibitory links*. These can be considered to be negative weights on output arcs of transition nodes. We need to modify the net transition rule thusly to account for these links:

[10] Also recall that this is all *very* preliminary – in working on this model, we hope to uncover all the situations where it fails, and thus learn what we need to do. Also recall that at this stage we are still only developing what the *representation* will be, and not (yet) any process-level description (e.g. how the structures are built, filled in, etc.)

[11] Precisely what this means is part of the forthcoming implementation work of this project

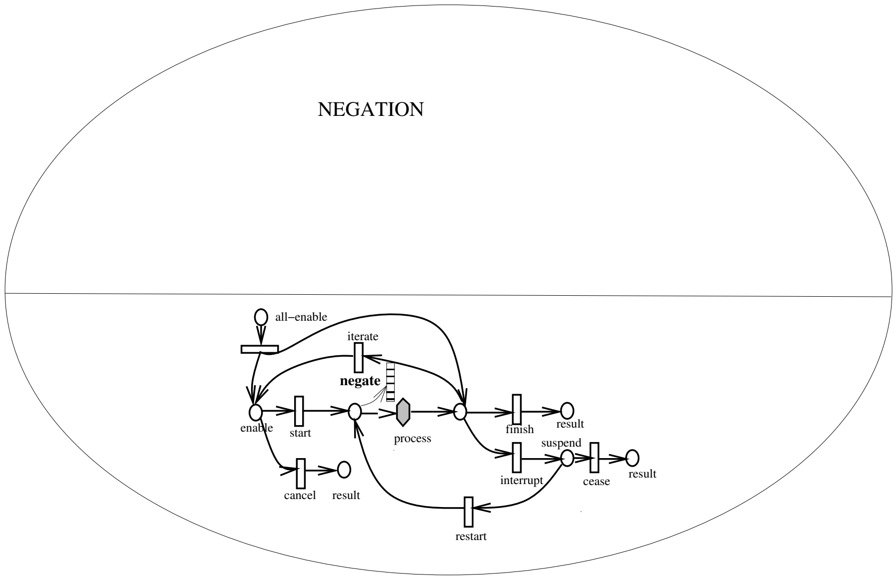

Figure 8: Negation as A Token Sink. The indicated node acts as a token sink. If the tokens flow to that node, they can not active the process sub-net. This is ensured by giving it a higher priority activation than the process node.

- Bill made John go to the store
- Bill did not make John go to the store
- Bill let John go to the store
- Bill did not let John go to the store
- Bill helped John go to the store
- Bill did not help John go to the store
- Bill prevented John from going to the store
- Bill did not prevent John from going to the store
- Bill hindered John in going to the store
- Bill did not hinder John in going to the store

Figure 9: Sentences with force-dynamic (generalized causative) terms

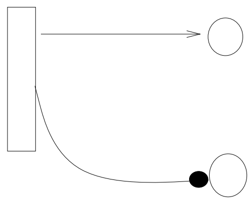

Figure 10: An inhibitory link.

- A *firing* of an enabled transition t removes $w(p,t)$ tokens from each input place of t, and places $w(t,p)$ tokens to each output place of t. If $w(t,p)$ is negative (i.e. it is an *inhibitory* link), then $|w(t,p)|$ tokens are removed from each output place. We allow token counts to become negative.

Inhibitory links allow us to model an obstruction (or a barrier) to the activation of the net.

The main insight towards extending the generalized aspectual model to account for force dynamic terms is this: We use the force-dynamic verb schemas (e.g. those which model semantics of words like "let," "prevent," "make," *etc.*) as types of "gating" schemas, which can be used to modify the firing patterns of the action of the schema for the subordinate clause (*e.g.* the "making" schema affects the execution of the "going" schema for the sentence "Bill made John go to the store") Figure 11 shows the general form the model takes. Since the clauses contain two verbs, the pattern includes two action schemas. In all of the diagrams here, the lower schema will correspond to the force-dynamic verb, and the upper schema will represent the subordinate verb. We see that the force-dynamic verb can affect the subordinate verb in one of two places: at the enable node, or at the beginning of the process node. For the force dynamic terms in which the Antagonist has power over the Agonist (e.g. let, make, prevent), the gating occurs directly at the "process" stage of the sub-schema, while for the less-powerful terms, such as "help" and "hinder", the gating occurs at the "enable" node of the net.

For example, consider Figure 12,[12] which shows the semantics portion of the construction for the causative verb MAKE. The bottom schema is the schema for the verb itself – it, too has the full aspectual schema mechanism surrounding the "process" subnet, in which the actual internal semantics of the verb is recognized. But this semantics relates to the action of the subordinate verb – thus there is a link leading from the process portion of the make schema to the input place of the process of the subordinate verb. The representational semantics of this is as follows: since the subordinate process node is enabled not of its own accord, but via the process of the MAKE schema, this indicates that the process was, in fact, caused as a result of

[12] For the remainder of these figures, for convenience we will not show the entire construction, just the semantic schema half of the constructions.

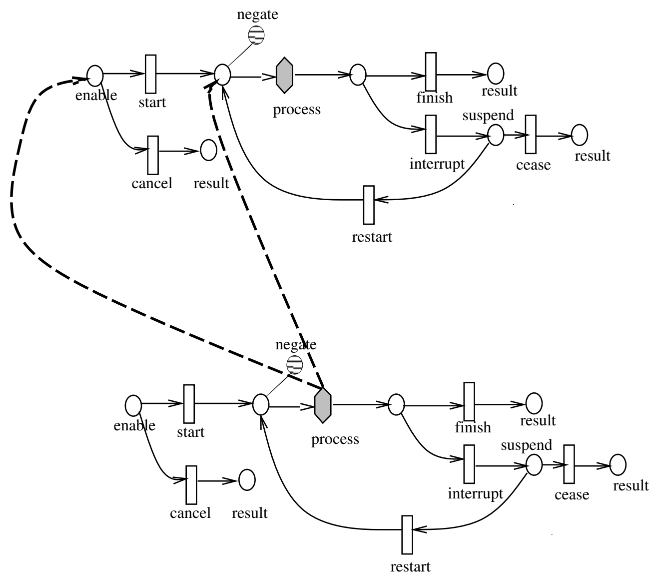

Figure 11: General paradigm for force-dynamic gating nets

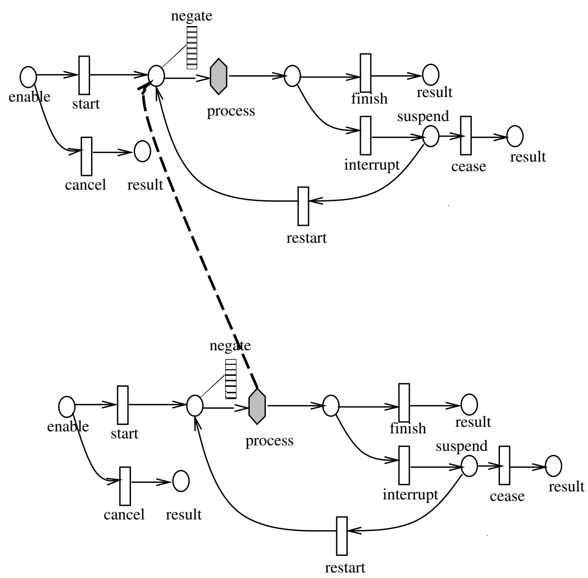

Figure 12: Petri net semantics for MAKE.

the agent of the *make* clause, not as a result of the agent of the subordinate clause. Computationally, the link from the MAKE process to the subordinate process places tokens into the input place of the subordinate process, thus causing it to fire. Again, we have a verisimilitudinous representation of the semantics of the clause we are modeling.

Consider the case when the *make* clause is negated – *e.g.* in a sentence such as "Bill did not make John clean the car." In this case, the negation of the *make* clause would act as a token sink, preventing the process of the *make* clause from being activated. Since the MAKE process is not activated, it will not send tokens to the subordinate processes. Therefore, the subordinate process will not be activated by the MAKE process. Note that just from the sentence "Bill did not make John clean the car" we cannot infer whether or not John did in fact clean the car – we only know that Bill did not make John do so. Similarly, the semantic model of the sentence does not specify whether or not John cleaned the car – it will construct a semantic state consistent with either "So it didn't get cleaned" or "but he cleaned it anyway" as following clauses.

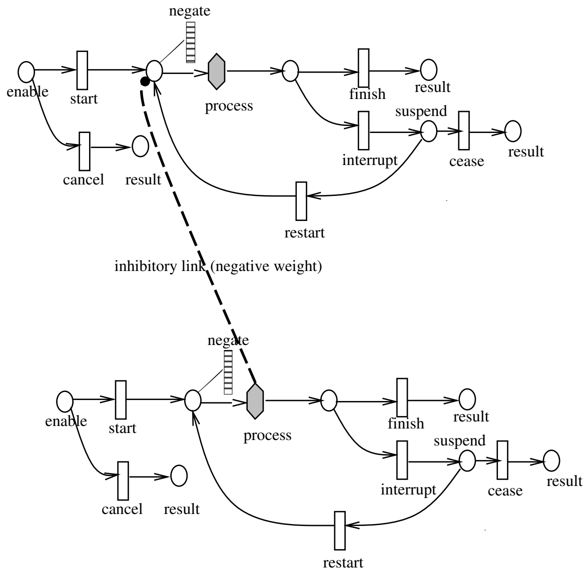

Figure 13: Petri net semantics for PREVENT.

PREVENT (Figure 13) is modeled similarly to MAKE, except that the link is an inhibitory link. Thus, the input link to the process of the subordinate action is prevented from obtaining a token. Thus the process of the subordinate action can not be activated. Again, our model emulates the semantics of clause. If the *prevent* clause is negated, we have the same effect as when the *make* clause is negated: since the PREVENT process is not activated, it does not send any tokens down the inhibitory link, so the subordinate action can occur or not occur – the key is that it is not prevented.

To model HELP and HINDER (see Figure 14 and Figure 15) we need to take advantage

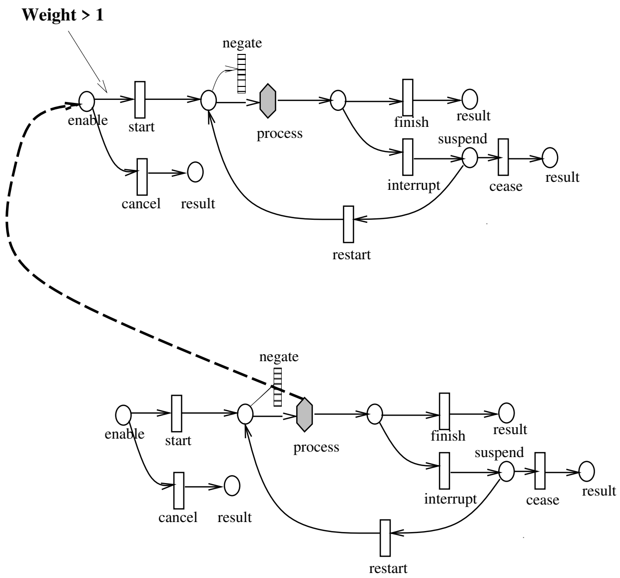

Figure 14: Petri net semantics for HELP.

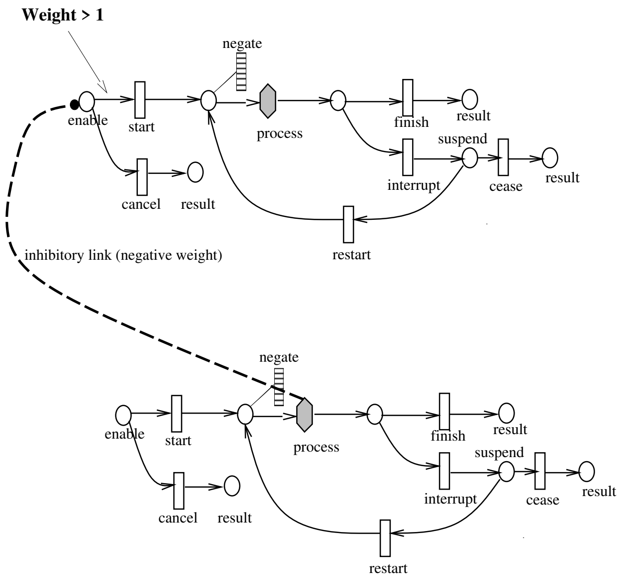

Figure 15: Petri net semantics for HINDER.

of the fact that it is possible to specify that transitions can have weights, meaning that it requires multiple tokens in an input place in order for a transition to be enabled to fire. With the link weight greater than one, let us see what happens: in the case of HELP, the HELP process supplies additional tokens to allow the event to be enabled – thus, fewer tokens directly from the subordinate event are needed to enable the process to activate. The model emulates the semantics of the clause. We get the same effect, in reverse for the HINDER action. The implication is that the Agonist of the subordinate clause either needed to expend additional effort in order to complete the action (as in "Bill hindered me but I managed to move my car anyway"), or did not succeed ("Bill hindered me so much I couldn't do it"). As in the case of PREVENT and MAKE, if the superordinate clause is negated, whether the subordinate action occurs in entirely independent of the Antagonist, and this is represented by the lack of token flow from the Antagonist schema to the Agonist, subordinate schema.

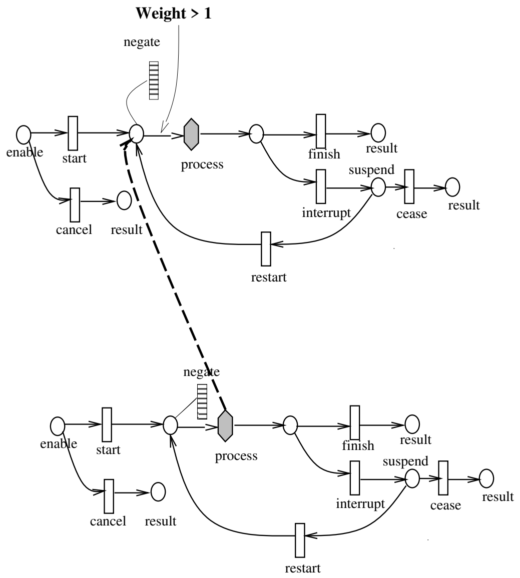

Figure 16: Petri net semantics for LET.

Modeling LET is a little tricky. The semantics of LET are as follows: in a LETTING action, the Antagonist potentially opposes the Agonist, but refrains from exercising this opposition, allowing the Agonist to act. When the *let*-clause action is negated, as in "Alice did not let Bill perform the action," there are two possible interpretations. The most natural one is that the action was not performed, as Alice prevented Bill from performing it. Alternatively, Bill performed the action anyway, despite Alice's failure to allow it (as is "Alice didn't *let* Bill do it, but he did it anyway). We attempt to capture these semantics by positing a weighted link into the subordinate process sub-net. When the LET schema is active (see Figure 16) an extra token is applied to the subordinate action's process's input place, but this token is not in itself enough to the activate the process – nor is the token generated by the subordinate process alone ordinarily enough to activate the process sub-net. Both being present is the case for the expression "A let B (action)". In the more natural case where there is negating letting "A didn't let B (action)," the token representing B's attempt (or desire) to perform the action, and the result is that the subordinate action does not occur. The other interpretation is possible, though, if the Agonist supplies enough "energy," represented by the accumulation of sufficient tokens, for the process sub-net to be activated. The model again yields a verisimilitudinous representation of the semantics of the actions being modeled.

5.2 Discussion

As was mentioned in the introduction, this paper is describing the preliminary directions the research is taking. Many details are still to be worked out, including:

- working the semantic representation into a system to combine with the syntax. (i.e. modeling the full linguistic constructions)
- actually implementing the Petri net model
- how to link the representation of more tokens to more force
- more fully tying in the aspect model (for instance, in the HINDER action, repeated attempts by the Agonist may allow the action to succeed, though precisely how needs to be worked out).
- the "potential" aspect of the LET action is tricky, and it is not clear that the "token threshold" approach is enough to capture it. More likely, we need to model potentiality through a computational model of Fauconnier's mental spaces (Fauconnier 1985), which would need to be ties in to the syntactic and semantic mechanism.
- once the model is built, with all the many details worked out and fixed, we need to test it, in a comprehension mode (i.e. parse sentences and construct semantic models) and possibly in a generation mode (given a model of the situation, generate sentences which can describe it).

There is still much to be done!

References

BAILEY, DAVID, 1994. Getting a grip: Body-based acquisition of verb semantics for hand actions. Thesis Proposal. UC Berkeley, Department of Computer Science.

FAUCONNIER, GILLES. 1985. *Mental spaces*. Cambridge, MA: MIT Press.

FELDMAN, JERRY, GEORGE LAKOFF, DAVID BAILEY, SRINI NARAYANAN, TERRY REGIER, & ANDREAS STOLCKE. 1993. L0 – the first four years. *AI Review* 8. To Appear in AI Review Volume 8.

JOHNSON, MARK. 1987. *The body in the mind: the bodily basis of meaning, imagination, and reason*. Chicago: University of Chicago Press.

JURAFSKY, DANIEL. 1992. An on-line computational model of human sentence interpretation: A theory of the representation and use of linguistic knowledge. Technical Report 92/676, University of California at Berkeley dissertation, also available as Computer Science Division, Berkeley, CA.

LAKOFF, GEORGE. 1987. *Women, fire, and dangerous things*. Chicago: University of Chicago Press.

——, & MARK JOHNSON. 1980. *Metaphors we live by*. Chicago: University of Chicago Press.

LANGACKER, RONALD. 1991. *Concept, image, and symbol: The cognitive basis of grammar*. Berlin: Mouton de Gruyter.

MURATA, TADAO. 1989. Petri nets: Properties, analysis and applications. *Proceedings of the IEEE* 77.

NARAYAN, SRINI, 1994. A computational semantics for aspect. Draft. UC Berkeley, and International Computer Science Institute.

——, 1995. One more step: An on-line model of metaphoric interpretation. Thesis Proposal. UC Berkeley, Department of Computer Science.

QUINN, NAOMI. 1987. Culture and cognition. In *Cultual models in language and thought*, ed. by Dorothy Holland & Naomi Quinn, 3–42. Cambridge, England: Cambridge University Press.

REGIER, TERRY, 1992. *The acquisition of lexical semantics for spatial terms: A connectionist model of perceptual categorization*. U.C. Berkeley dissertation.

REISIG, WOLFGANG. 1982. *Petri nets: An introduction*. New York: Springer-Verlag.

SWEETSER, EVE. 1982. A proposal for uniting deontic and epistemic modals. In *Proceedings of the 5th Annual Meeting of the Berkeley Linguistics Society*, Berkeley, CA.

TALMY, LEONARD. 1988. Force dynamics in language and cognition. *Cognitive Science* 12.