Probabilistic Language Networks:

Integrating Word Grammar and Link Grammar in the Framework of Probabilistic Logic

Ben Goertzel
December 26, 2008

This is a preliminary, rough-draft article representing some ideas-in-progress. Expect it to be substantially revised and expanded as the train-of-thought continues.

Introduction

This document briefly describes a partially-new theory of language formed via combining ideas from three sources: Hudson's Word Grammar (1990, 2007), Sleator and Temperley's link grammar (1993), and the Probabilistic Logic Networks (PLN) framework (Ikle and Goertzel, 2007; Goertzel et al, 2008) created for use within the Novamente design for integrated cognition (Goertzel et al, 2008). Reflecting its origin in these three sources, we have named the new theory PROWL grammar, meaning PRObabilistic Word Link Grammar.

We believe PROWL has value purely as a conceptual approach to understanding language; however, it has been developed largely from the standpoint of computational linguistics – as part of an attempt to create a framework for computational language understanding and generation that

- yields broadly adequate behavior based on hand-coding of "expert rules" such as grammatical rules, combined with statistical corpus analysis
- integrates naturally with a broader AI framework that combines language with embodied social, experiential learning, that ultimately will allow linguistic rules derived via expert encoding and statistical corpus analysis to be replaced with comparable, more refined rules resulting from the system's own experience

As such, PROWL is part of the larger Novamente project oriented toward artificial general intelligence, and its role in that project is described in (Goertzel, 2008); but, it is presented here mostly outside that context, and is intended to be independently evaluable (and valuable).

As an integration of three existing frameworks, PROWL could be presented in various different ways One could choose any one of the three components as an initial foundation, and then present the combined theory as an expansion/modification of this component. Here we choose to present it as an expansion/modification of Word Grammar, as we feel this is the most natural approach for readers with a linguistics background. From this perspective, to simplify a fair bit, one may describe PROWL as consisting of Word Grammar with three major changes:

- Word Grammar's network knowledge representation is replaced with a richer PLN-based network knowledge representation.
    - This includes, for instance, the replacement of Word Grammar's single "isa" relationship type with a more nuanced collection of logically distinct probabilistic inheritance relationship types
- Going along with the above, Word Grammar's "default inheritance" mechanism is replaced by an appropriate PLN control mechanism that guides the use of standard PLN inference rules
    - This allows the same default-inheritance based inferences that Word Grammar relies upon, but embeds these inferences in a richer probabilistic framework that allows them to be integrated with a wide variety of other inferences
- Word Grammar's small set of syntactic link types is replaced with a richer set of syntactic link types as used in Link Grammar
    - The precise optimal set of link types is not clear; it may be that the link grammar's syntactic link type vocabulary is larger than necessary, but we also find it clear that the current version of Word Grammar's syntactic link type vocabulary is smaller than feasible (at least, without the addition of large, new, and as yet unspecified ideas to Word Grammar)

In the following sections of this document we will review these changes in a little more detail. Basic familiarity with Word Grammar, Link Grammar and PLN is assumed.

Note that in this document I will focus only on those issues that are somehow problematic! This means that a whole of of very important topics that come along with the Word Grammar / PLN integration are not even mentioned. The way Word Grammar deals with morphology, semantics and pragmatics, for instance, seems to us quite sensible and workable – and doesn't really change at all when you integrate Word Grammar with PLN, except that Word Grammar's crisp isa links become PLN-style probabilistic Inheritance links. These topics are extremely important, but we will focus here only on areas where there are problems to be addressed:

1. turning Word Grammar's approach to syntax parsing into something computationally tractable (via integration with Link Grammar)
2. validating that the Word Grammar approach to logical inference (using default inheritance) is achievable with PLN, using an appropriate control mechanism. Note that these two issues are quite

as already noted above. Note that two areas are fairly unrelated to each other, so that the rest of this document consists of two disconnected sections: one on logic and the other on syntax. To see how they connect together you'll have to read *Language Networks* ... or use your imagination.

Word Grammar's Logical Network Model

Word Grammar presents an elegant framework in which all the different aspects of language are encompassed within a single knowledge network. Representationally, this network combines two key aspects:

- Inheritance (called is-a) is explicitly represented
- General relationships between n-ary predicates and their arguments are explicitly represented

Dynamically, the network contains two key aspects:

- An inference rule called "default inheritance"
- Activation-spreading, similar to in a neural network or standard semantic network

The similarity between Word Grammar the Novamente Cognition Engine is fairly strong. In the latter, inhiertance and generic predicate-argument relationships are explicitly represented; and, a close analogue of activation spreading is present in the "attention allocation" subsystem. As in Word Grammar, important cognitive phenomena are grounded in the symbiotic combination of logical-inference and activation-spreading dynamics.

At the most general level, the reaction of the Word Grammar network to any situation is proposed to involve three stages:

1. Node creation and identification: of nodes representing the situation as understood, in its most relevant aspects
2. Where choices need to be made (e.g. where an identified predicate needs to choose which other nodes to bind to as arguments), activation spreading is used, and the most active eligible argument is utilized (this is called "best fit binding")
3. Default inheritance is used to supply new links to the relevant nodes as necessary

Default inheritance is a process that relies on the placement of each node in a hierarchy (dag) of isa links. The basic idea is as follows. Suppose one has a node N, and a predicate f(N,L), where L is another argument or list of arguments. Then, if the truth value of f(N,L) is not explicitly stored in the network, N inherits the value from any ancestor A in the dag so that: f(A,L) is explicitly stored in the network; and there is not any node P inbetween N and A for which f(P,L) is explicitly stored in the network. Note that multiple inheritance is explicitly supported, and in cases where this leads to multiple assignments of truth values to a predicate, confusion in the linguistic mind may ensue. In many cases the option coming from the ancestor with the highest level of activity may be selected.

My suggestion is that Word Grammar's network representation may be replaced with PLN's logical network representation without any loss, and with significant gain. Word Grammar's network representation has not been fleshed out as thoroughly as that of PLN, it does not handle uncertainty, and it is not associated with general mechanisms for inference. The one nontrivial issue that must be addressed in porting Word Grammar to the PLN representation is the role of default inheritance in Word Grammar. This is covered in the following subsection.

The integration of activation spreading and default inheritance proposed in Word Grammar, should be easily achievable within the NCE assuming a functional attention allocation subsystem.

Default Inheritance in PLN

The notion of "default inheritance" as used by Word Grammar is closely related to the one used in various species of nonmonotonic "default logic (Reiter, 1980; Delgrande and Schaub, 2003). Hudson draws various distinctions but they are not relevant to the issue of achieving default inheritance in the PLN framework.

To exemplify the notion of default inheritance, consider the case of penguins, which do not fly, although they are a subclass of birds, which do fly. When one discovers a new type of penguin, say an Emperor penguin, one reasons initially that they do not fly – i.e. one reasons by reference to the new type's immediate parent in the ontological hierarchy, rather than its grandparent. In some logical inference frameworks, the notion of hierarchy is primary, and default inheritance of this nature is wired in at the inference rule level. But this is not the case with PLN – in PLN, correct treatment of default inheritance has got to come indirectly out of other mechanisms.

Consider the two inferences

```
A)
penguin --> fly <0>
bird --> penguin <.02>
|-
bird --> fly

B)
penguin --> bird <1>
bird --> fly <.9>
|-
penguin --> fly
```

The correct behavior according to the default inheritance idea is that, in a system that already knows at least a moderate amount about the flight behavior of birds and penguins, inference A should be accepted but inference B should not. That is, evidence about penguins should be included in determining whether birds can fly – even if there is already some general knowledge about the flight behavior of birds in the system. But, evidence about birds in general should not be included in estimating whether penguins can fly, if there is already at least a moderate level knowledge that in fact penguins are atypical birds in regard ot flight.

But how can the choice of A over B be motivated in terms of PLN theory? The essence of the answer is simple: in case B the independence assumption at the heart of the deduction rule is a bad one. Within the scope of birds, being a penguin and being a flier are not at all independent. On the other hand, looking at A, we see that within the scope of penguins, being a bird and being a flier are independent. So, the reason B is ruled out is that, if there is even a moderate amount of knowledge about the truth value of (penguin

--> fly), this gives a hint that applying the deduction rule's independence assumption in this case is badly wrong.

On the other hand, what if a mistake is made and the inference B is done anyway. In this case the outcome could be that the system erroneously increases its estimate of the strength of the statement that penguins can fly. On the other hand, the revision rule may come to the rescue here. If the prior strength of (penguin --> fly) is 0, and inference B yields a strength of .9 for the same proposition, then the special case of the revision rule that handles wildly different truth value estimates may be triggered. If the 0 strength has much more confidence attached to it than the .9, then they won't be merged together, because it will be assumed that the .9 is an observational or inference error. Either the .9 will be thrown out, or it will be provisionally held as an alternate, non-merged, low-confidence hypothesis, awaiting further validation or refutation.

What is more interesting, however, is to consider the implications of the default inference notion for inference control. It seems that the following may be a valuable inference control heuristic:

1. Arrange terms in a hierarchy, e.g. by finding a spanning dag of the terms in a knowledge base, satisfying certain criteria (e.g. maximizing total strength*confidence within a fixed limitation on the number of links)
2. When reasoning about a term, first do deductive reasoning involving the term's immediate parents in the hierarchy, then ascend the hierarchy, looking at each hierarchical level only at terms that were not visited at lower hierarchical levels

This is precisely the "default reasoning" idea – but the key point is, in PLN it lives at the level of inference control, not inference rules or formulas. In PLN, default reasoning is a time-saving heuristic, not an elementary aspect of the logic itself. Rather, the practical viability of the default-reasoning inference-control heuristic is a consequence of various other elementary aspects of the logic, such as the ability to detect dependencies rendering the deduction rule inapplicable, and the way the revision rule deals with wildly disparate estimates.

Link Grammar Parsing vs Word Grammar Parsing

Coming at it from a Novamente/PLN point of view, perhaps the most striking original contribution of Word Grammar is in the area of syntax parsing. The treatment of morphology and semantics is, basically, exactly what one would expect from representing such things in a richly structured semantic network. PLN adds much additional richness to Word Grammar via allowing nuanced representation of uncertainty, which is critical on every level of the linguistic hierarchy. Regarding syntax processing, however, Word Grammar makes some quite specific and unique hypotheses, which if correct are very valuable contributions.

The conceptual assumption we make here is that syntax processing, while carried out using generic cognitive processes for uncertain inference and activation spreading, also involves some highly specific constraints on these processes. The extent to which these constraints are learned versus inherited is yet unknown, and for the subtleties of this issue the reader is referred to (Elman et al, 1997). Word Grammar and Link Grammar

are then understood as embodying different hypotheses regarding what these constraints actually are.

It is interesting to consider the contributions of Word Grammar to syntax parsing via comparing it to Link Grammar.

Note that Link Grammar, while a less comprehensive conceptual theory than Word Grammar, has been used to produce a state-of-the-art syntax parser, which has been incorporated into a number of other software systems including the RelEx natural language comprehension system (Goertzel, 2006). So it is clear that the Link Grammar approach has a great deal of pragmatic value. On the other hand, it also seems clear that Link Grammar has certain theoretical shortcomings. It deals with many linguistic phenomena very elegantly, but there are other phenomena for which its approach can only be described as "hacky."

Word Grammar contains fewer hacks than Link Grammar, but has not yet been put to the test of large-scale computational implementation, so it's not yet clear how many hacks would need to be added to give it the relatively broad coverage that Link Grammar currently has. Our own impression is that to make Word Grammar actually work as the foundation for a broad-coverage grammar parser (whether standalone, or integrated into a broader artificial cognition framework), one would need to move it somewhat in the direction of link grammar, via adding a greater number of specialized syntactic link types (more on this shortly). There are in fact concrete indications of this in (Hudson, 2007).

The Link Grammar framework may be decomposed into three aspects:

1. The link grammar dictionary, which for each word in English, contains a number of links of different types. Some links point left, some point right, and each link is labeled. Furthermore, some links are required and others are optional.
2. The "no-links-cross" constraint, which states that the correct parse of a sentence will involve drawing links between words, in such a way that all the required links of each word are fulfilled, and no two links cross when the links are depicted in two dimensions
3. A processing algorithm, which involves first searching the space of all possible linkages among the words in a sentence to find all complete linkages that obey the no-links-cross constraint; and then applying various postprocessing rules to handle cases (such as conjunctions) that aren't handled properly by this algorithm

In PROWL, what we suggest is that

1. The link grammar dictionary is highly valuable and provides a level of linguistic detail that is not present in Word Grammar; and, we suggest that in order to turn Word Grammar into a computationally tractable system, one will need something at least halfway between the currently minimal collection of syntactic link types used in Word Grammar and the much richer collection used in Link Grammar
2. The no-links-cross constraint is an approximation of a deeper syntactic constraint ("landmark transitivity") that has been articulated in the most recent formulations of Word Grammar. Specifically: when a no-links-crossing parse is found, it is

correct according to Word Grammar; but Word Grammar correctly recognizes some parses that violate this constraint

3. The Link Grammar parsing algorithm is not cognitively natural, but is effective in a standalone-parsing framework. The Word Grammar approach to parsing is cognitively natural, but as formulated could only be computationally implemented in the context of an already-very-powerful general intelligence system. Fortunately, various intermediary approaches to parsing seem possible.

Using Landmark Transitivity with the Link Grammar Dictionary

An earlier version of Word Grammar utilized a constraint called "no tangled links" which is equivalent to the link parser's "no links cross" constraint. In the new version of Word Grammar this is replaced with a subtler and more permissive constraint called "landmark transitivity." While in Word Grammar, landmark transitivity is used with a small set of syntactic link types, there is no reason why it can't be used with the richer set of link types that Link Grammar provides. In fact, this seems to us a probably effective method of eliminating most or all of the "postprocessing rules" that exist in the link parser, and that constitute the least elegant aspect of the Link Grammar framework.

The first foundational concept, on the path to the notion of landmark transitivity, is the notion of a syntactic parent. In Word Grammar each syntactic link has a parent end and a child end. In a dependency grammar context, the notion is that the child depends upon the parent. For instance, in Word Grammar, in the link between a noun and an adjective, the noun is the parent.

To apply landmark transitivity in the context of the Link Grammar, one needs to provide some additional information regarding each link in the Link Grammar dictionary. One needs to specify which end of each of the link grammar links is the "parent" and which is the "child." Examples of this kind of markup are as follows (with P shown by the parent):

```
                              P
S: subject-noun ------- finite verb

      P
O: transitive verb ----- direct or indirect object

                       P
D: determiner ----- noun

      P
MV: verb ----- verb modifier

      P
J: preposition ----- object

     P
ON: on ----- time-expression

     P
M: noun ----- modifiers
```

In some cases a word may have more than one parent. In this case, the rule is that the landmark is the one that is superordinate to all the other parents. In the rare case that two words are each others' parents, then either may serve as the landmark.

The concept of a parent leads naturally into that of a landmark. The first rule regarding landmarks is that a parent is a landmark for its child. Next, two kinds of landmarks are introduced: Before landmarks (in which the child is before the parent) and After landmarks (in which the child is after the parent). The Before/After distinction should be obvious in the Link Grammar examples given above.

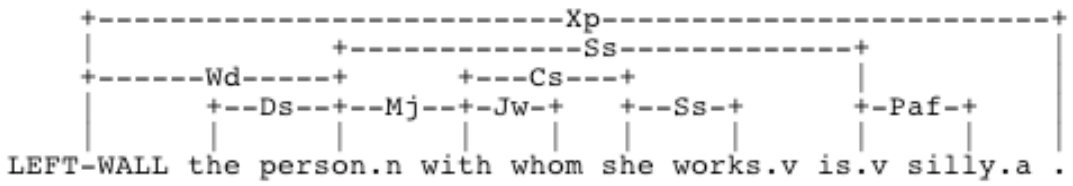
The landmark transitivity rule, then, has two parts. If A is a landmark for B, of subtype L (where L is either Before or After), then

- **Subordinate transitivity** says that if B is a landmark for C, then A is also a type-L landmark for C
- **Sister transitivity** says that if A is a landmark for C, then B is also a landmark for C

Finally, there are some special link types that cause a word to depend on its grandparents or higher ancestors as well as its parents. I note that these are not treated thoroughly in (Hudson, 2007); one needs to look to the earlier, longer and rarer work (Hudson, 1990). Some questions are dealt with this way. Another example is what in Word Grammar is called a "proxy link", as occurs between "with" and "whom" in

```
The person with whom she works
```

The link parser deals with this particular example via a Jw link



so to apply landmark transitivity in the context of the Link Grammar, in this case, it seems one would need to implement the rule that in the case of two words connected by a Jw-link, the child of one of the words is also the child of the other. Handling other special cases like this in the context of Link Grammar seems conceptually unproblematic, though naturally some hidden rocks may appear. Basically a list needs to be made of which kinds of link parser links embody proxy relationships for which other kinds of link parser links.

According to the landmark transitivity approach, then, the criterion for syntactic correctness of a parse is that, if one takes the links in the parse and applies the landmark transitivity rule (along with the other special-case "raising" rules we've discussed), one does not arrive at any contradictions (i.e. no situations where A is a Before landmark of B, and

The main problem with the landmark-transitivity constraint seems to be computational tractability. The problem exists for both comprehension and generation, but we'll focus on comprehension here.

To find all possible parses of a sentence using Hudson's landmark-transitivity-based approach, one needs to find all linkages that don't lead to contradictions when used as premises for reasoning based on the landmark-transitivity axioms. This appears to be extremely computationally intensive! So, it seems that Word Grammar style parsing is only computationally feasible for a system that has extremely strong semantic understanding, so as to be able to filter out the vast majority of possible parses on semantic rather than purely syntactic grounds.

On the other hand, it seems possible to apply landmark-transitivity together with no-links-cross, to provide parsing that is both efficient and general. If applying the no-links-cross constraint finds a parse in which no links cross, without using postprocessing rules, then this will always be a legal parse according to the landmark-transitivity rule. However, landmark-transitivity also allows a lot of other parses that link grammar either needs postprocessing rules to handle, or can't find even with postprocessing rules. So, it would make sense to apply no-links-cross parsing first, but then if this fails, apply landmark-transitivity parsing starting from the partial parses that the former stage produced. This is the approach suggested in PROWL, and as noted below, a similar approach is suggested for language generation.

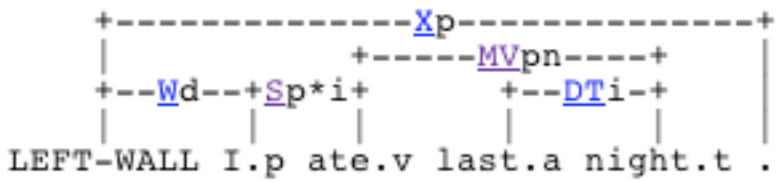
Overcoming the Current Limitations of Word Grammar

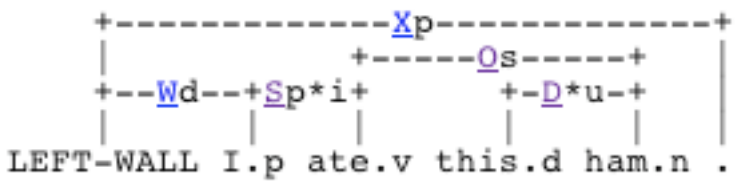
It is worth noting that expanding the Word Grammar parsing framework to include the link grammar dictionary, will likely allow us to solve some unsolved problems in Word Grammar. For instance, Hudson (2007) notes that the current formulation of Word Grammar has no way to distinguish the behavior of last vs. this in

```
I ate last night
I ate this ham
```

The issue he sees is that in the first case, *night* should be considered the parent of *last*; whereas in the second case, *this* should be considered the parent of *ham*.

The current link parser also fails to handle this issue according to Hudson's intuition:





However, the link grammar framework gives us a clear possibility for allowing the kind of interpretation Hudson wants: just allow *this* to take a left-going O-link, and (in PROWL) let it optionally assume the parent role when involved in a D-link relationship. There are no funky link-crossing or semantic issues here; just a straightforward link-grammar dictionary edit.

This illustrates the syntactic flexibility of the link parsing framework, and also the inelegance – adding new links to the dictionary generally solves syntactic problems, but at the cost of creating more complexity to be dealt with further down the pipeline, when the various link types need to be compressed into a smaller number of semantic relationship types for purposes of actual comprehension (as is done in RelEx, for example). However, as far as we can tell, this seems to be a necessary cost for adequately handling the full complexity of natural language syntax. Word Grammar holds out the hope of avoiding this kind of complexity, but without filling in enough details to allow a clear estimate of whether this hope can ever be fulfilled.

Contextually Guided Greedy Parsing and Generation Using Word Link Grammar

Another difference between Link Grammar and currently utilized, and Word Grammar as described, is the nature of the parsing algorithm. Link Grammar operates in a manner that is fairly traditional among contemporary parsing algorithms: given a sentence, it produces a large set of possible parses, and then it is left to other methods/algorithms to select the right parse, and to form a semantic interpretation of the selected parse. Parse selection may of course involve semantic interpretation: one way to choose the right parse is to choose the one that has the most contextually sensible semantic interpretation. We may call this approach *whole-sentence purely-syntactic parsing*, or WSPS parsing.

One of the nice things about Link Grammar, as compared to many other computational parsing frameworks, is that it produces a relatively small number of parses, compared for instance to typical head-driven phrase-structure grammar parsers. For simple sentences the link parser generally produces only handful of parses. But for complex sentences the link parser can produce hundreds of parses, which can be computationally costly to sift through.

Word Grammar, on the other hand, presents far fewer constraints regarding which words may link to other words. Therefore, to apply parsing in the style of the current link parser, in the context of Word Grammar, would be completely infeasible. The number of possible parses would be tremendous. The idea of Word Grammar is to pare down parses via semantic/pragmatic sensibleness, during the course of the syntax parsing process, rather than breaking things down into two phases (parsing followed by semantic/pragmatic interpretation). Parsing is suggested to proceed forward through a sentence: when a word is encountered, it is linked to the words coming before it in the

sentence, in a way that makes sense. If this seems impossible, consistently with the links that have already been drawn in the course of the parsing process, then some backtracking is done and prior choices may be revisited. This approach is more like what humans do when parsing a sentence, and does not have the effect of producing a large number of syntactically possible, semantically/pragmatically absurd parses, and then sorting through them afterwards. It is what we call a *contextually-guided greedy parsing* (CGGP) approach.

For language generation, the link parser and Word Grammar approaches also suggest different strategies. Link Grammar suggests taking a semantic network, then searching holistically for a linear sequence of words that, when link-parsed, would give rise to that semantic network as the interpretation. On the other hand, Word Grammar suggests taking that same semantic network and iterating through it progressively, verbalizing each node of the network as one walks through it, and backtracking if one reaches a point where there is no way to verbalize the current node consistently with how one has already verbalized the previous nodes.

The main observation I want to make here is that, while Word Grammar by its nature (due to the relative paucity of explicit constraints on which syntactic links may be formed), can operate with CGGP but not WSPS parsing. On the other hand, while Link Grammar is currently utilized with WPSP parsing, there is no reason one can't use it with CGGP parsing just as well. There is no objection to using CGGP parsing together with the link-parser dictionary, nor with the no-links cross constraint rather than the landmark-transitivity constraint (in fact, as noted above, earlier versions of Word Grammar made use of the no-links-cross constraint).

What we propose in PROWL is to use the link grammar dictionary together with the CGGP parsing approach. The WSPS parsing approach may perhaps be useful as a fallback for handling extremely complex and perverted sentences where CGGP takes too long to come to an answer – it corresponds to sentences that are so obscure one has to do really hard, analytical thinking to figure out what they mean.

Regarding constraints on link structure, the suggestion in PROWL is to use the no-links-cross constraint as a first approximation. In comprehension, if no sufficiently high-probability interpretation obeying the no-links-cross constraint is found, then the scope of investigation should expand to include link-structures obeying landmark-transitivity but violating no-links-cross. In generation, things are a little subtler: a list should be kept of link-type combinations that often correctly violate no-links-cross, and when these combinations are encountered in the generation process, then constructs that satisfy landmark-transitivity but not no-links-cross should be considered.

Arguably, the PROWL approach is less elegant than either Link Grammar or Word Grammar considered on its own. However, we are dubious of the proposition that human syntax processing, with all its surface messiness and complexity, is really generated by a simple, unified, mathematically elegant underlying framework. Our goal is not to find a maximally elegant theoretical framework, but rather one that works both as a standalone computational-linguistics system, and as an integrated component of an adaptively-learning AGI system. It that PROWL may fit this bill.

References

- Delgrande, J. and T. Schaub (2003). On the relation between Reiter's default logic and its (major) variants. In Seventh European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty (ECSQARU 2003), pages 452-463.
- Elman, Jeff et al (1997). Rethinking Innateness. MIT Press.
- Goertzel, Ben, Matt Ikle', Izabela Freire Goertzel and Ari Heljakka (2008). Probabilistic Logic Networks
- Hudson, Richard (2007). Language Networks. The new Word Grammar. Oxford University Press.
- Hudson, Richard (1990). English Word Grammar. Blackwell Press.
- Ikle, Matt and Ben Goertzel (2007). Indefinite Probabilities for General Intelligence. Advances in Artificial General Intelligence, IOS Press.
- Goertzel, Ben, Hugo Pinto, Cassio Pennachin et al. (2006). Using Dependency Parsing and Probabilistic Inference to Extract Relationships between Genes, Proteins and Malignancies Implicit Among Multiple Biomedical Research Abstracts. Proceedings of the BioNLP Workshop/HLT-NAACL 2006
- Reiter, R. (1980). A logic for default reasoning. Artificial Intelligence, 13:81-132
- Sleator, D. and Davy Temperley. 1993. Parsing English with a Link Grammar. Third International Workshop on Parsing Technologies.