Natural Language Processing by Reasoning and Learning

(draft for comment)

Pei Wang
Department of Computer and Information Sciences
Temple University
pei.wang@temple.edu

Abstract

This paper proposes a new approach toward natural language processing in a general-purpose intelligent system. The major features that differs this approach from the traditional ones are: natural language understanding and production are not carried out by a separate module in the system, but by the reasoning-learning mechanism responsible for all kinds of cognitive functions; furthermore, when a sentence in a natural language is processed, its syntactic, semantic, and pragmatic aspects are taken into consideration altogether. This approach has been formalized in the form of a logic, and partially implemented as a reasoning system. Some preliminary results are introduced and discussed.

1 Two Chomskyan Assumptions

1.1 The modularity of language processing

Since the very beginning of the study, natural language processing (NLP) has been recognized as a major aspect of artificial intelligence (AI). For example, by proposing his test for machine intelligence [Turing, 1950], Alan Turing implicitly assumes that the existence of other cognitive functions needed by intelligence can be recognized in a system's language performance.

Though the ability of using natural languages is widely acknowledged as necessary for (human-like) intelligence, its relation with other cognitive functions (such as perception, reasoning, learning, and action) is a controversial topic. Two well-known positions are represented by Noam Chomsky and Jean Piaget, respectively, and their opinions are contrasted in a debate in 1975 participated by them and some other researchers [Piattelli-Palmarini, 1980]. One key topic of the debate is the relation between language and intelligence. According to Chomsky, human language competence is basically innate and modular, as a

language instinct in the a form of a "universal grammar" hard-wired into the human brain. On the contrary, Piaget believes that languages are learned from experience as other human knowledge, using the general-purpose learning mechanism of the human mind.

In cognitive science, this debates continues in various forms, though in AI, the Chomskyan approach has been dominating the field of natural language processing (NLP). The existing NLP research projects usually aim at the building of NLP systems or modules that do not depend on the other cognitive functions, though the resulting system can be integrated with other functions, serving as an "NLP interface" of the overall system to the outside [Allen, 1994, Jurafsky and Martin, 2009, Russell and Norvig, 2010]. Even when functions like learning and reasoning become necessary, they are customized to work within the NLP module. Consequently, NLP is handled as independent to the other fields of AI. Only in a very small number of projects is NLP treated uniformly with other functions, such as knowledge representation and reasoning [Shapiro and Neal, 1982].

This modular treatment of NLP has several reasons. Outside AI, it is from the Chomskyan tradition in linguistics, and the philosophical opinion that the mind has a modular architecture [Fodor, 1983]. As shown in the Chomsky-Piaget debate [Piattelli-Palmarini, 1980], though the Piagetian approach is philosophically appealing and has supporting evidence from developmental psychology, it has not provided a detailed *constructive* explanation on how natural languages are processed by a general-purpose cognitive process. Inside AI, the mainstream has turned away from the study of versatile and multifunctional systems, and focused on domain-specific and problem-specific functions, which leads to serious fragmentation of the field [Brachman, 2006]. Though many researchers may agree that NLP should depend on other cognitive functions, they usually consider the isolated research on NLP a necessary simplification at the current stage.

1.2 The autonomy of syntax, semantics, and pragmatics

Another fundamental assumption in linguistics established by Chomsky is the opinion that syntax, semantics, and pragmatics are *autonomous* to each other, and should be studied in that order. In particular, the syntactic structure of a sentence in a natural language can be analyzed without considering its meaning and usage. According to this opinion, the language competence of a human being is partly represented by a *grammar*, similar to how a formal language is specified. Equipped with such a grammar, the comprehension process of a sentence includes the *parsing* of the sentence that maps the linear sequence of input words into a tree structure for internal representation. After that, semantic analysis will determine the *meaning* of the words and the sentence. Eventually, pragmatic analysis will determine how to use the sentence, by considering the goals of the speaker and the listener, the current context, and other factors [Chomsky, 1957, Dik, 1978].

Though this opinion became popular and influential in linguistics, it has

strong competitors there. For instance, Dik's "functional grammar" is based on the opposite assumption that "pragmatics is the all-encompassing framework within which semantics and syntax must be studied; semantics is subservient to pragmatics and syntax to semantics; the priorities run from pragmatics via semantics to syntax" [Dik, 1978]. Similarly, cognitive linguistics assumes "grammar is conceptualization" and "knowledge of language emerges from language use" [Croft and Cruse, 2004]. The relation between semantics and pragmatics has become a hot topic in recent years, too, where the key topic is whether, or to what extent, semantics is autonomous to pragmatics [Szabó, 2005].

Once again, though the Chomskyan assumption has been challenged in linguistics, it has been dominating the NLP study in AI. Most of the projects carries out language understanding in the order of syntax first, then semantics, and finally pragmatics, where the first step depends on the existence of a grammar. Even the statistical approach of NLP, which is usually perceived as anti-Chomsky, makes the same assumption, though it takes a grammar to be probabilistic and empirically acquired [Manning and Schütze, 1999, Jurafsky and Martin, 2009]. There have been some attempts to use "meaning-driven rules" in syntactic analysis [Schank, 1980], but such attempts have never become mainstream.

1.3 A uniform approach

As mentioned above, the two Chomskyan assumptions — the modularity of language processing and the autonomy of syntax, semantics, and pragmatics — are under debate in cognitive science, but are often taken as self-evident in AI. A major reason for this is the lack of alternative solutions.

The main objective of this article is not to continue the theoretical discussion of the two assumptions, but to introduce a new approach toward NLP where the two assumptions are rejected. This approach does not target precisely on NLP, but treats it as part of a larger goal, that is, to build a computer system with human-like intelligence, which has been called "artificial general intelligence" (AGI) in recent years to distinguish such an objective from the conventional AI, which is domain and problem specific [Goertzel and Pennachin, 2007, Wang and Goertzel, 2007].

In the following, I will explain the NLP-related capability in NARS, an AGI project developed in the framework of a reasoning system [Wang, 2006, Wang, 2013]. NARS is based on the assumption that *intelligence is the ability for a system to adapt to its environment while working with insufficient knowledge and resources* [Wang, 2011]. It means that the system depends on finite capacity in information processing, works in real time, and is open to novel tasks and knowledge. Because of this assumption, NARS is very different from the traditional reasoning systems in several fundamental aspects.

The basic ideas of NLP in NARS can be summarized as the following:

- To represent linguistic knowledge in the same form as other types of knowledge, with various levels of abstraction, so that morphological, lexical, and

grammatical knowledge are unified, and are directly associated with semantic and pragmatic knowledge.

- To obtain linguistic knowledge from multiple forms of inference, including deduction, induction, abduction, and revision, so it can be derived from the system's experience in language usage.

- To use linguistic knowledge selectively in NLP tasks to achieve scalability and context-sensitivity, according to their records in usefulness and relevance to the current context, so as to capture the fluid nature of meaning.

- To carry out NLP tasks as other cognitive tasks, using the same domain-independent reasoning-learning mechanism, so as to unify NLP into a model of general intelligence.

NARS is an open source project with demonstrations and working examples. Given the length restriction of this article, it is impossible to describe all aspects of NARS in detail here. In the following, only the aspects of the system that are directly related to NLP are described. For the other aspects of the project, see [Wang, 2006, Wang, 2013] and other publications, many of which are available at the author's website[1].

2 Knowledge Representation in NARS

2.1 Terms and statements in Narsese

NARS uses a formal language, dubbed *Narsese*, for internal representation and external communication. It is a *term-oriented* language and belongs to the *term logic* school (which uses subject-copula-predicate sentences and syllogistic rules), rather than the *predicate logic* school (which uses predicate-arguments sentences and truth-functional rules) [Wang, 2006].

The basic component of Narsese is a *term*, an identifier that names a concept. A term is a recognizable entity in the system's (internal and external) experience, that is, it is whatever the system can recognize or name. In its simplest form, an *atomic term* is represented by a sequence of characters from an alphabet, such as that of English.

A Narsese *statement* relates a few terms to each other, and its basic form is an *inheritance statement* "$S \rightarrow P$", where 'S' is the subject term, 'P' the predicate term, and '$\rightarrow$' the *inheritance* copula, which is a reflexive and transitive relation between two terms. Intuitively, the statement says that S is a specialization of P, and P is a generalization of S. For example, knowledge "A robin is a bird" can be represented as "$robin \rightarrow bird$" in Narsese. A variant of *inheritance* is the *similarity* copula, '$\leftrightarrow$', which is reflexive, transitive, and symmetric. For example, knowledge "A boat is just like a ship" can be represented as "$boat \leftrightarrow ship$".

[1] `http://www.cis.temple.edu/~pwang/`

To express complicated content in a term-oriented language, Narsese utilizes *compound term* of various types, each of which is formed from some component terms by a *term connector* that serves a logical function. They include

Sets. A term can (though does not have to) be a set with its instances or properties enumerated. Term $\{Pacific, Atlantic, Indian, Antarctic, Arctic\}$ is an *extensional set*, and it can be used in similarity statement "$\{Pacific, Atlantic, Indian, Antarctic, Arctic\} \leftrightarrow ocean$" to list the oceans; term $[red, round]$ is an *intensional set* that can be used in "$apple \rightarrow [red, round]$" to express "Apples are red and round".

Intersections and differences. A compound term can be specified using the instances (or properties) of existing terms. Term $(bird \cap [black])$ is an *extensional intersection* representing "black bird", while term $(bird - [black])$ is an *extensional difference* representing "non-black bird". The intuitive meaning of the intersection and difference connectors are similar to their counterparts in set theory, though not identical to them.

Products and images. Non-copula relations can be expressed by inheritance statements. Knowledge "Cats eat fish" can be equivalently represented as inheritance statements "$(cat \times fish) \rightarrow food$", "$cat \rightarrow (food /_{-} fish)$", and "$fish \rightarrow (food /cat\ _{-})$". Here $(cat \times fish)$ is a *product* that represents the relation between "cat" and "fish", $(food /_{-} fish)$ is an *extensional image* that represents "things that take fish as food", and $(food /cat\ _{-})$ is an *extensional image* that represents "things that cats take as food".[2]

Statements. Narsese allows a statement to be used as a compound term to form a "higher-order" statement. For instance, knowledge "John knows that the Earth is round" can be represented as "$\{John \times (\{Earth\} \rightarrow [round])\} \rightarrow know$", where $know$ is a relation between a cognitive system $John$ and a statement "$\{Earth\} \rightarrow [round]$", and this knowledge itself can be represented in Narsese as "$know \rightarrow (([cognitive] \cap system) \times statement)$".

Compound statements. Similar to the treatment in propositional logic, compound statements can be composed from simpler statements, using statement connectors *conjunction* ($\wedge$), *disjunction* ($\vee$), and *negation* ($\neg$). Furthermore, two "higher-order" copulas are defined between statements: the *implication* copula, '$\Rightarrow$', intuitively stands for "if-then", and the *equivalence* copula, '$\Leftrightarrow$', for "if-and-only-if", though they are not defined exactly as in propositional logic.

Beside atomic and compound terms, Narsese also has several special types of terms:

[2]Here the *image connectors* are used in the *infix* form, while in the previous publications on NARS they are often used in the *prefix* form. There are no difference between the two forms except in display format.

Variable. A variable term stands for another term. An *independent variable* is identified with the prefix $ and can be substituted by any term (under certain condition); an *dependent variable* is identified with the prefix # and can substitute any term.[3]

Event. An event is a statement whose truth-value has a specified temporal duration, such as "The door is open".

Operation. An operation is an event that can be realized by the system itself, such as "To open the door".

Events and operations are described in detail in [Wang, 2013] and omitted here. They are mentioned merely to show that Narsese can consistently represent declarative, episodic, and procedural knowledge, in a way that is similar to logic programming [Kowalski, 1979].

2.2 Semantics of Narsese

While the traditional "mathematical logic" was invented to formalize theorem proving in mathematics, NARS is built to capture the "laws of thought" in everyday thinking. A fundamental difference between these two types of reasoning is that in the former, the reasoning systems derive implied truths (the theorems) from the assumed truths (the axioms), while in the latter, all knowledge of the system is summarized experience, so may be revised by new experience. Accurately speaking, no empirical knowledge can be considered as "axioms" that cannot be challenged by new evidence. However, it does not mean that every statement is equally justifiable and acceptable. Instead, it means that the basis of justification cannot be a constant set of axioms, but the ever-expanding experience of the system. This is why the system is named "NARS", for "Non-Axiomatic Reasoning System", and why its major semantic notions "meaning" and "truth-value" are specified according to an "experience-grounded semantics" (EGS) [Wang, 2005, Wang, 2006].

In this article, EGS is only informally introduced. Roughly speaking, the "experience" of NARS consists of a stream of Narsese sentences coming into the system during its life cycle. At a given moment, for a given statement the past experience may provide *positive* (agreeing) or *negative* (opposing) evidence. EGS defines a measurement of "amount of evidence" for an idealized (binary) version of Narsese, so, in principle, the amount of positive and negative evidence for the statement is represented by two non-negative numbers w^+ and w^-, respectively, and their sum is the total amount of evidence w the system has for the statement at that moment.

The *truth-value* of the statement is $\langle f, c \rangle$, a pair of real numbers in [0, 1], and it is formally defined according to available evidence. Here f is "frequency"

[3]The *independent/dependent* variables are intuitively similar to the *universal/existential* variables in predicate logic, respectively, though the accurate definitions are different. The notations for variable terms in this paper are displayed differently from those in [Wang, 2013] and some other publications, though there is no difference in their processing.

and defined as $f = w^+/w$, and c is "confidence" and defined as $c = w/(w+1)$. Therefore, *frequency* is the proportion of positive evidence among all available evidence, so is intuitively similar to probability, though it is fully defined on past experience, and does not necessarily converge to a limit value. On the other hand, *confidence* is the ratio of the amount of evidence at the current moment compared to that after the coming of a new piece of evidence with a unit amount, and therefore indicates the *sensitivity* of the belief to new evidence, or its *stability*. At the beginning, when the system knows nothing about the statement, its c is 0; with the coming of evidence, c increases monotonically, though in theory it never reaches its upperbound 1. Therefore NARS will never be "absolutely sure" about any of its empirical knowledge, and all of its knowledge is revisable by new evidence, with different easiness.

In NARS, the *meaning* of a term is neither determined by "the object it refers to", or by a given "definition" or fixed frame with slots and values, but is defined by its role in the system's experience. If we see the knowledge of NARS as a graph, with the terms as vertices and statement as edges, then the meaning of a term is specified by all the edges connecting the term to other terms. This meaning is "experience-grounded" in the sense that all the connections are built according to the experience of the system, so in principle may change over time.

2.3 Linguistic knowledge in Narsese

Since a "term" is just a name of an internal data structure of the system, and does not require a predetermined denotation or definition, it can be used to represent any type of entities or patterns. In the previous examples, terms like *bird* and $fish$ are amodal categories, and English words are used as their names to associate them to the corresponding human concepts, though they can be equivalently named $T3721$ or *niao*, since their meaning have little to do with their names, but is determined by their relations with other terms. A term can even name a sensation or operation of the system that does not exactly correspond to any human concept. Therefore, all linguistic knowledge is represented by terms and statements, too. In the following, bold font will be used to indicate words in English, such as **bird** and **fish**.

The *semantic relation* between a word (like **bird**) and the category it represents (like *bird*, $T3721$, or *niao*) can be handled in the same way as other conceptual relations, i.e., by *product* and *image* introduced before, such as "$\{\mathbf{bird} \times bird\} \rightarrow represent$". In general, *represent* is a many-to-many relation between a word and a category. So the same word can represent different categories, and therefore the above statement can coexist with "$\{\mathbf{bird} \times chicken\} \rightarrow represent$" and "$\{\mathbf{fowl} \times bird\} \rightarrow represent$". Furthermore, each instance of the relation is "true to a degree", as indicated by its truth-value defined previously. In this way, the system can keep multiple possible representation relations, while still know what is the usual choice of word (for a given category) or category (for a given word).

Similarly, the syntactic relations among words, phrases, and sentences can be represented in Narsese as other relations. For example, " 'Cats eat fish' is an

English sentence" in Narsese is statement "$\{\mathbf{Cats} \times \mathbf{eat} \times \mathbf{fish}\} \rightarrow ([English] \cap sentence)$". The meaning of larger linguistic larger units (such as phrases, sentences, texts) can be expressed using the *represent* relation, too, though the related terms are usually compounds with internal structures, as in statement "$\{(\mathbf{Cats} \times \mathbf{eat} \times \mathbf{fish}) \times ((cat \times fish) \rightarrow food)\} \rightarrow represent$", where $food$ is a relational category where the first argument takes the second argument as food. From these relations, other relations, such as co-occurrence of the words in sentences, can be derived.

Syntactic categories can be expressed as terms in Narsese, too, like in statement "$\{\mathbf{fish}\} \rightarrow noun$". Again, this statement is true to a degree, and can coexist with "$\{\mathbf{fish}\} \rightarrow verb$". Similarly, "The past tense form of 'want' is 'wanted"' can be represented as "$\{\mathbf{want} \times \mathbf{wanted}\} \rightarrow past\text{-}tense$".

Morphological knowledge about words and morphemes is expressed like syntactic knowledge. For example, $affix$ can indicate a 3-argument morphological relation as exemplified by "$\{\mathbf{want} \times \mathbf{ed} \times \mathbf{wanted}\} \rightarrow affix$". Using independent variable $\$x$ and dependent variable $\#y$, the system can represent knowledge "A verb with 'ed' added becomes its past tense form" as

$$(\{\$x\} \rightarrow verb) \Rightarrow ((\{\$x \times \mathbf{ed} \times \#y\} \rightarrow affix) \wedge (\{\$x \times \#y\} \rightarrow past\text{-}tense))$$

Obviously, this statement will have many counterexamples, which will be counted as its negative evidence, and are stored separately, such as "$\{\mathbf{go} \times \mathbf{went}\} \rightarrow past\text{-}tense$".

In this way, the system's linguistic knowledge exists in various levels of generality, and includes both general patterns and special cases. The difference between linguistic and non-linguistic knowledge are not in their *format*, but in their *content*, since the latter includes terms specific to a natural language (such as **fish**) or natural languages (such as *verb*).

In summary, like other terms in NARS, the meaning of a word like **fish** or **want** is determined by its experienced relations with other terms (i.e., what the system has learned about this word), and these relations can be either semantic or syntactic (even pragmatic, like **Hello**), according to the nature of the relation involved.

3 Inference Rules in NARS

3.1 Syllogistic rules

Using a term logic, NARS utilizes *syllogistic* rules. Such a rule takes two premises (which contain a common term) to derive a conclusion (between the other two terms). When the copula involved is *inheritance*, there are three basic syllogistic inference rules:

	deduction	**abduction**	**induction**
first premise	$M \rightarrow P$	$P \rightarrow M$	$M \rightarrow P$
second premise	$S \rightarrow M$	$S \rightarrow M$	$M \rightarrow S$
conclusion	$S \rightarrow P$	$S \rightarrow P$	$S \rightarrow P$

These rules are named using the terminology introduced by C. S. Peirce [Peirce, 1931], though the exact definitions of the rules in NARS are not the same as his. In NARS, the *deduction* rule extends the transitivity of the inheritance copula from binary to many-valued, while the *induction* rule and the *abduction* rule can be seen as "reversed deduction", obtained from the *deduction* rule by exchanging the conclusion with a premise, then renaming the terms.

Each inference rule has an associated truth-function to calculate the truth-value of the conclusion from the truth-values of the premises. The truth-functions are designed to decide the truth-value of the conclusion according to the evidence provided by the premises alone. In this way, the inference rules in NARS are still "truth-preserving", though according to EGS, truth-value is evaluated with respect to the system's *past* experience, neither to the state of affairs in the world, nor to the future experience of the system.

The derivations of the truth-functions can be found in [Wang, 2013] and some other publications about NARS. For the current discussion, it is enough to know that the *deduction* rule is *strong*, as the confidence value of its conclusion is relatively high (it can approach 1), while the other two rules are *weak*, as they only produce conclusions with low confidence values (less than 0.5). If the truth-values are omitted and the rule is applied among binary statements, the strong rules are still valid, but the weak rules are not.

When the *inheritance* copula '$\rightarrow$' in the above rules is replaced by the *implication* copula '$\Rightarrow$', the inference rules remain valid in NARS:

	deduction	**abduction**	**induction**
first premise	$M \Rightarrow P$	$P \Rightarrow M$	$M \Rightarrow P$
second premise	$S \Rightarrow M$	$S \Rightarrow M$	$M \Rightarrow S$
conclusion	$S \Rightarrow P$	$S \Rightarrow P$	$S \Rightarrow P$

This group of rules is isomorphic to the previous group, in the sense that the truth-functions are the same, though the meaning of the sentences is different, due to the use of different copulas. When one term is taken to mean "something that the system knows" and is implicitly represented, a third group of rules can be obtained:

	deduction	**abduction**	**induction**
first premise	$S \Rightarrow P$	$P \Rightarrow S$	P
second premise	S	S	S
conclusion	P	P	$S \Rightarrow P$

This last group is closer to how these three types of inference are specified in the current AI research, in the framework of propositional logic [Hobbs et al., 1993, Flach and Kakas, 2000]. The above syllogistic rules show a common principle of NARS: in its conclusion, each rule summarizes the evidence provided by the premises, and the conclusion usually contains compound terms that are not in the premises.

Through substitutions, variable terms may be eliminated or introduced in reasoning. For example, from "$\{Tweety\} \rightarrow [yellow]$" and "$\{Tweety\} \rightarrow$

[*yellow*]" the system derives "($\$x \rightarrow bird$) $\Rightarrow$ ($\$x \rightarrow [yellow]$)" by induction, with a relatively low confidence value.[4] Similarly, if the system is initially given "($\$x \rightarrow bird$) $\Rightarrow$ ($\$x \rightarrow [yellow]$)" and "$\{Tweety\} \rightarrow [yellow]$", it will derives "$\{Tweety\} \rightarrow bird$" by abduction, also with a relatively low confidence value. Finally, from "($\$x \rightarrow bird$) $\Rightarrow$ ($\$x \rightarrow [yellow]$)" and "$\{Tweety\} \rightarrow bird$", it will derives "$\{Tweety\} \rightarrow [yellow]$" by deduction, with a relatively high confidence value. This relationship among deduction, induction, and abduction was the same as suggested by C. S. Peirce [Peirce, 1931], though he didn't have a numerical truth-value involved in the inference, and the details of formalization in NARS is also different from Peirce's approach.

3.2 Compositional and structural rules

The terms S, P, and M in the syllogistic rules can be either atomic or compound, since their internal structures are ignored in these inference steps. NARS has other inference rules sensitive to the structure of compound terms, so can compose or decompose compound terms. This is one way of the system to "create new concepts".

For example, from "$\{Tweety\} \rightarrow [yellow]$" and "$\{Tweety\} \rightarrow bird$", a *compositional* rule derives "$\{Tweety\} \rightarrow ([yellow] \cap bird)$". Different from the syllogistic rules, this rule produces conclusions that have a term $([yellow] \cap bird)$ that is not in the premises, which may or may not be among the terms already known to the system — if not, then it is a new concept, otherwise it is a new relation of the known concept.

As explained before, in NARS the meaning of a term is determined by its (available) relations with other terms. Consequently, the inference activity more or less changes the meaning of the terms involved, though all the inference are defined formally. As in the above example, assume initially the system knows nothing about the term $\{Tweety\}$ except "$\{Tweety\} \rightarrow [yellow]$" and "$\{Tweety\} \rightarrow bird$", then the above inference makes the meaning of $\{Tweety\}$ richer by adding a third relation into it, though it is derived from the other two.

For a compound term like $([yellow] \cap bird)$, the situation is more complicated. The system not only know its *semantic* relation with $\{Tweety\}$, but also its *syntactic* relations with its components, $[yellow]$ and $bird$. Different from semantic relations (which are learned and multi-valued), syntactic relations in Narsese are defined and binary. Since there is only a constant number of term connectors like '$\cap$', the system instinctively knows how to handle them (i.e., their meaning is embedded in the related inference rules). Consequently, the meaning of a compound term is determined both by its semantic relations and its syntactic relations.

If the system is initially given "$\{Tweety\} \rightarrow ([yellow] \cap bird)$" and the syntactic relation between $bird$ and $([yellow] \cap bird)$ is taken into account, a decompositional rule will be invoked to derive "$\{Tweety\} \rightarrow bird$". In this

[4] From the same premises, induction also derives "($\$x \rightarrow [yellow]$) $\Rightarrow$ ($\$x \rightarrow bird$)", "$bird \rightarrow [yellow]$", and "$[yellow] \rightarrow bird$". The difference among them is discussed in [Wang, 2013] and some other publications on NARS. The situation for abduction is similar.

case, $([yellow] \cap bird)$ is understood literally, as birds that are yellow. For every term connector, there are rules that form the corresponding compound terms, as well as rules that decompose them into their components.

However, compound terms are not always handled in this way. According to EGS, the meaning of compound terms in NARS is "semi-compositional", that is, the meaning of a compound term is *partially*, though not *completely*, determined by the meaning of its components plus the meaning of the term connector. In the above inference step, $([yellow] \cap bird)$ is used "literally", i.e., according to its syntactic relation with its components. However, the compound term can be used by other inference rules (like the syllogistic rules) *as a whole*, without considering its internal structure. Consequently, it may end up with some meaning that cannot be reduced to the meaning of its components. For example, under certain circumstance the system may form the belief that yellow birds have certain unique properties that are shared neither by other birds nor by other yellow things. Compound terms with such unique properties or instances will gradually become independent of their components, and are more and more often used like atomic terms.

When a natural language is processed in NARS, this semi-compositionality will be extended from Narsese to that language. It means that a phrase or sentence in English will be initially understood as a structure consisting of words, but when some of them gradually obtain enough uniqueness from the experience of the system, they will be more and more often treated as irreducible. Whether a compound term should be treated as a structure becomes a matter of degree. In language understanding, the decomposition of sentence stops when a meaningful and plausible semantic representation is obtained, which may happen before the word level is reached. This treatment can explain the *productivity* and *systematicity* of languages, as well as the forming and using of idioms and proverbs.

3.3 Revision and choice rules

The previous description shows some similarity between NARS and statistical NLP, since both use statistical information of linguistic materials. However, the two approaches also have important differences. One major difference is that statistical (or probabilistic) approaches usually assume the system gets all the relevant information at the beginning, then do statistical inference on them. In this way, all the probabilistic evaluations in the model are based on the same chunk of evidence, which is implicitly expressed. On the contrary, NARS works in *real time*, which means new evidence arrives from time to time, and the system must make judgments under time pressure, which forces it to omit relevant information when a conclusion is reached. Therefore, in NARS each judgment is made with its own evidential base — this is what the *confidence* value measures.

One direct consequence is that the truth-values in NARS are not necessarily *consistent*, in the sense that at the same time in the system the same statement can be given multiple truth-values, according to different evidential bases.

Whenever such a pair of inconsistent judgments are identified, the *revision rule* derives a judgment based on the pooled evidence [Wang, 2009]. This rule serves the functions of evidence accumulation and inconsistency resolution, and allows the system to tolerant the conflicts among its beliefs.

Narsese consists of not only *declarative sentences* that represent the system's *beliefs* (also called *judgments* or *knowledge*), but also *interrogative sentences* and *imperative sentences* that represent the system's *questions* and *goals*, respectively. The representation and processing of goals are explained in [Wang, 2012, Wang, 2013] and will not be discussed in this article. Here we only describe the processing of questions.

Formally, in Narsese a "question" is a statement without a truth-value. One type of question contains *query variables* to be instantiated to make the statement true, while another type does not contain such variables and only asks a truth-value to be decided. A question can be answered by a matching belief the system has, but since in NARS every belief is true to a degree, no answer can be perfect. What the system does is to find the "best possible" answer.

The major tasks in NLP, *language understanding* and *language generation*, become question-answering tasks in NARS that contain Narsese sentence like "$\{sentence \times ?x\} \rightarrow represent$" and "$\{?x \times term\} \rightarrow represent$", respectively. In the former, the task is to find a term that the given sentence represents; in the latter, the task is to find a sentence that represents the given term. In Narsese, a term with a '?' prefix is a *query variable* to be instantiated. On the other hand, "$\{sentence \times term\} \rightarrow represent$" is a question for the system to evaluate the truth-value for the proposed relation between the given sentence and term.

If for a given question there are more than one answer, the *choice rule* is invoked to decide which one is better. For the variable-instantiation type of question, the answer with a higher *expectation* value (which is a function of *frequency* and *confidence*) is usually chosen[5]; for the For the truth-evaluation type of question, the answer with a higher *confidence*) value is chosen [Wang, 2009].

For a given question, beside directly searching for an answer, NARS also uses the inference rules *backward* to produce derived questions, whose answers will lead to the answers of the original question.

One major topic in NLP is *ambiguity*. While most approaches attempt to interpret every sentence in a unique way, in NARS it is possible for the same sentence to be related to multiple terms as "its meaning". The choice rule will select the best candidate whenever a unique answer must be chosen, though in the meanwhile, the other inferior candidates may still more or less contribute to the system's understanding of the sentence.

In summary, the inference rules of NARS form a general-purpose reasoning and learning mechanism, by which the system can summarize its past experience into beliefs, which provide generalization and abstraction of the experience to various levels, and can be use to guide the system's behavior when a new problem is solved. This reasoning and learning mechanism handles syntactic, semantic,

[5]There are other factors involved, such the simplicity and relevance of the answer.

and pragmatic knowledge about a natural language in a unified manner, just like how non-linguistic knowledge is handled.

4 Memory and Control in NARS

4.1 NARS overview

As a reasoning system, NARS can accept three types of *tasks*: (1) a piece of new knowledge to be absorbed, (2) a question to be answered, and (3) a goal to be achieved. A task is carried out by interacting with the system's *beliefs*, which are the knowledge already absorbed by the system. Both tasks and beliefs are represented as sentences in Narsese, though they are treated differently in the system. At any moment, the system usually have multiple tasks, and a much larger number of beliefs.

As a finite system working in real-time and open to novel problems, NARS can accept a new task at any moment, and there is no restriction on the *content* of the task, as far as it is in a *form* that is recognizable by the system. The "native language" of the system is Narsese, with its grammar built into the system. NARS can learn an arbitrary natural language by treating its words and sentences as special terms, and its linguistic knowledge as special statements, as described previously.[6] In NARS, the primary function of a natural language is communication, and other functions, like knowledge representation, are secondary, since most of the system's knowledge are not represented using natural languages, but in Narsese.

Since there is no restriction in content, NARS accepts a task even when the system does not have the relevant knowledge to provide a satisfactory solution for it. Instead, the system always attempts to provide the best solution it can find using available beliefs.

Ideally, to get "the best solution according to available knowledge" means to let the task interact with every relevant belief. However it is impossible in NARS, given its assumption on the insufficiency of resources — there are new tasks coming from the outside environment, as well as derived from existing tasks, and each of them requires to be processed as soon as possible. Consequently, the system has to dynamically allocates its computational resources (mainly time and space) among the tasks, so each of them will only interact with some of the beliefs. The resource allocation mechanism is similar to how an operating system manages the resources of a computer system, though in NARS the processing of tasks are not isolated from each other, and each of them does not follow a predetermined algorithm, nor ends according to certain predetermined criterion.

Since each interaction between a task and a belief corresponds to an inference step with the two as premises, the life time of NARS consists of a sequence of "working cycles", each of which takes a roughly constant time, and carries

[6] NARS can be further extended to interact with the environment through a sensorimotor mechanism [Wang, 2006, Wang, 2013], though that topic will not be discussed in this paper.

out an inference step plus some input/output activities. The processing of a specific task is a sub-sequence that may interweave with the other tasks, and may have shared steps with the others. Given the ever-changing nature of the environment, the accurate processing path and result for a given task may be practically neither predictable nor repeatable, except in very special situations.

Furthermore, in NARS a task does not have a predetermined "computational complexity", since its time and space expense not only depends on the content of the task (i.e., the corresponding Narsese sentence), but also on the time pressure attached to it, as well as the current situation within the system. This issue will be further explained in the following.

4.2 System architecture

NARS consists of the following major components:

- a *memory* containing all beliefs and tasks,
- an *inference engine* deriving new tasks from a given task–belief pair,
- a *task buffer* keeping new (input or derived) tasks,
- one or more *input/output channels* connecting the task buffer and the outside environment,
- a *control center* managing the working process.

As mentioned previously, NARS uses a *term logic*, which has the nice property that two sentences can be used as premises in an inference step only when they share a common term. To take advantage of this feature, in NARS the tasks and beliefs are accessed via *concepts*, according to the terms in them. For example, a task or belief with the statement "$robin \rightarrow bird$" as content can be accessed directly from the concepts named by atomic terms *robin* and *bird*, as well as compound term $(robin \rightarrow bird)$. Consequently, every inference step occurs within some concept named by the shared term, and sentences in different concepts will not directly interact with each other. Furthermore, this property guarantees the semantic relatedness of the premises and the conclusion.

In parallel to the situation of *term*, the meaning of a *concept* is defined by its experienced relationship with other concepts. For example, the meaning of concept *bird* at a certain moment is determined by what the system knows and is thinking about the term *bird*. Restricted by insufficient resources, NARS cannot take all existing beliefs into account when processing every task, but has to use them *selectively*. Therefore, the *current meaning* of a concept is determined by a *subset* of the tasks and beliefs existing in the system that form the *full meaning* of the concept.

An important feature that distinguishes NARS from many other categorical models is that the meaning of a concept is not fixed, but *fluid* and changes over time [Wang and Hofstadter, 2006]. This change takes two major forms: the *long-term* changes in the *full meaning*, and the *short-term* changes in the

current meaning. These changes are not arbitrary, but are determined by the system's experience and the current context.

Now we can say that NARS gets "new concepts" in several ways: it can accept a novel term from its environment, create one using its inference rules, or give new meaning to an existing term. The initial meaning of a concept is usually simple, though it may become complicated as the system gets more related experience. In particular, the meaning of a composed concept may gradually become different from its literal meaning, which is directly formed from the meaning of its components.

As a special type of term (and concept), linguistic terms (words, phrases, sentences, etc.) have the same property. In NARS, the "meaning" of a word depends on what the system knows about it, as well as what associations of the word are brought into consideration at the moment.

4.3 Working cycle

As a reasoning system, the running of NARS consists of repeated working cycles. In each cycle, the following steps are taken:

1. Select a concept from the memory,
2. Select a task from the concept,
3. Select a belief from the concept,
4. Derive new tasks from the selected task and belief and put them into the buffer,
5. Return the belief, task, and concept with feedback information,
6. Preprocess the tasks in the buffer.

All the selections in the first three steps are *probabilistic* and *priority-based*. In NARS every data item (belief, task, and concept) has a priority value attached, which is proportional to the probability for it to be selected at the moment. Initially, every input task has the option to include a priority value to indicate its relative urgency, specified by the user or other system providing the task. In this way, the user can influence (though not decide) how soon this task will be processed. Beside this optional value, the system will also evaluate its quality so as to assign an initial priority to the task. If the preprocessing of the task triggers the creation of related belief and concept, their initial priority will be decided accordingly. Then, each time an item is selected, its priority will be adjusted according to the immediate result of the inference step. In the long run, the priority value of an item is decided by the following major factors:

its intrinsic quality, such as the clarity and simplicity of a concept, or the confidence of a judgment in a task or belief;

its performance history, that is, whether the item has been useful to the system in the past;

its immediate relevance, that is, whether it is directly related to the current situation, as reflected in the existing tasks.

Without describing all the details, the above introduction explains that NARS always tries to use its available resources in the most efficient manner, judged according to its own experience. Due to the assumption of insufficient knowledge and resources, there is no way to promise any absolute correctness or optimality, but the system is not treating every possibility as equal. While the inference rules decide what can be derived in the system, the memory structure and control strategy selectively turns some *possible* derivations into *actual* derivations.

At the end of a working cycle, the buffer may have some tasks accumulated, either from the input channels, or from the inference activity. Each of them will be processed briefly:

- Duplicated tasks are merged,
- Conflict judgments are handled by the revision rule,
- Matching question and judgment are checked by the choice rule to see if the judgment provides the best-so-far answer to the question,
- New tasks satisfying certain conditions will become added into the corresponding concepts,
- New tasks satisfying certain conditions will trigger the creation of corresponding concepts or beliefs,
- New tasks satisfying certain conditions will become output, sending to the user or another system,

  ... etc.

The outgoing tasks of NARS are of the same types as the incoming tasks, that is, a task can be a *judgment* (conclusion, answer), a *question*, or a *goal*. Consequently, multiple NARS-like systems can cooperate via communication when solving problems, and the language they use can either be Narsese or an acquired (natural or artificial) language.

Since NARS is designed to depend on a constant amount of storage space, the number of concepts and the number of tasks and beliefs within each concept have upper bounds. When the memory (or a concept) is "full", the concept (or task/belief) with the lowest priority value is removed. This policy and the decay that happens to all priority values form a *forgetting* mechanism. Consequently, the system not only constantly gets new tasks, beliefs, and concepts from the environment and the reasoning process, but also loses some old ones from time to time. In a relatively stable environment, the system should gradually get some useful concepts that each has an *essential meaning* from which the most of the other relations can be derived.

At the beginning of a life cycle of the system, by default the system's memory starts empty, though for practical applications it may start with preloaded

content. During a life cycle, the content of the memory changes constantly, as the result of new experience and reasoning activity, and the internal states of the system never repeat. Therefore, if a task is given to the system at different moments of its life cycle, it may be treated more or less differently. That is why at the problem-solving activities do not follow an algorithm, and nor have predetermined ends [Wang, 2006].

Applied to NLP, it means that the processing of each language understanding or language generation task is determined at the run time by the available knowledge and resources at the moment, rather than following a fixed routine and with a constant expense.

4.4 Meaning and context

Traditionally, it was assumed that a linguistic manifestation (such as a word, a phrase, or a sentence) has a *meaning* that independent of the *context* in which it is used. However, such an opinion has been challenged in linguistics [Szabó, 2005]. In logic and AI there have been attempts of formalizing context, usually by using an additional argument c to indicate the *context* in which a proposition p is true [Barwise and Perry, 1983, McCarthy, 1993]. In NLP, *context* usually means the other texts proceeding and following the text being analyzed, as exemplified by the definition of "context-free" and "context-sensitive" grammars [Jurafsky and Martin, 2009]. Once again, this article will not survey the related literature, but focuses on how "context" is represented in NARS, as well as its relation with meaning.

Though "context" is defined differently in different fields and approaches, it is introduced, as far as this discussion goes, to capture the *change in meaning and truth-value*, that is, the same word may mean differently in different situations, and the same sentence may be true in certain situations, but false in certain other situations. For this reason, the representation and processing of context depends on the definition of truth and meaning, which are the main content of a semantic theory.

As explained previously, the semantic theory of NARS is EGS, in which both truth-value and meaning are defined as functions of the system's experience. Since "experience" is the input stream that expands in time, EGS provides a natural treatment for context.

The meaning of a term in NARS is defined by its experienced relations with other terms. At any moment, the set of beliefs directly concerning the term consist of its *general meaning*, and the active beliefs (i.e., the ones with relatively high priority values) consist of its *current meaning*. Though both types of meaning change over time, it is the latter that is "context sensitive". Even when new beliefs are produced in a period of time, the priority among them still changes, because certain beliefs may be activated by the related new tasks. For example, the changes in the *external environment* cause adjustment of the priority distributions, and so do the the changes in the *internal environment*, such as pursuing or forgetting of goals.

The truth-value of a statement in the system is also context dependent,

because when evaluating it, the system usually cannot take all existing evidence into consideration, but only consider the active evidence, which is determined by the priority distribution among the beliefs, which changes from situation to situation.

Consequently, even when no new knowledge is obtained, the meaning of terms and the truth-value of statements may change from time to time. Such changes are not arbitrary, but are directly related to what the system is experiencing at the moment, both in its external environment and internal environment, which can be referred to as the "context", in which meanings of terms and truth-values of statements are determined and used in the processing of tasks.

On the other hand, it does not means that these is nothing "context free" or "context independent". As mentioned previously, in a given environment the system may develop some concepts that have "essence" or stable meanings. However, the distinction between "rigid" concepts and "fluid" concepts is relative and a matter of degree.

The above description about the relation between meaning and context is applicable to all terms in NARS, so it is also the case for all the terms that are specific to a natural language. That is, what a word means may more or less be different when the environment of the system changes. Compared to the other approaches, this treatment of context is more general, in that

- The context of a word includes, but not limited to, the other words proceeding or following it in the text it belongs. For example, what a word means often depends on the system's expectations, which are not completely determined by the proceeding text.

- A context may or may not have an identifier, and transitions between contexts may be either sudden or gradual. When a context is identified by a term, it helps to divide a concept into sub-concepts and to specify their boundary, though without such an identifier, the context dependency of meaning can still be reflected in the system.

5 NLP as Reasoning and Learning

All task processing activities in NARS, including NLP as special cases, are carried out as *reasoning* following inference rules. On the other hand, when the focus is on the long-term effects of the activities, they can all be considered as different forms of *learning*, since what the system does is to use the experience to adapt to the environment.

5.1 A simple example

NARS has been formally specified and partially implemented [Wang, 2013]. Recently, some preliminary experiments on NLP have been carried out in the system. Since NARS has no built-in competence specific to for any natural

language, the system's NLP ability comes from learning. In principle, it is like how a human child learns a language [Bloom, 2000, Tomasello, 2003], that is, the learning starts from simple words, then proceeds to phrases and sentences, with syntactic, semantic, and pragmatic knowledge learned side-by-side.

In the following, a working example processed by NARS is explained, in a simplified form (compared with the actual form produced by NARS). More details of these examples can be found at the author's website.

Initially, NARS is given the following beliefs:

$$\{\mathbf{cat} \times cat\} \rightarrow represent \ \langle 1, 0.9 \rangle \tag{1}$$

$$\{\mathbf{fish} \times fish\} \rightarrow represent \ \langle 1, 0.9 \rangle \tag{2}$$

$$\{\{\mathbf{cat} \times \mathbf{eat} \times \mathbf{fish}\} \times ((cat \times fish) \rightarrow food)\} \rightarrow represent \ \langle 1, 0.9 \rangle \tag{3}$$

The meaning of these sentences has been explained before: each of the first two associate a word to an atomic term, and the last one a sentence to a statement, using the *represent* relation, for which the system has no other knowledge. At the current stage, such given samples of *represent* relations provide the starting point of NLP, though in the future some of these relations can be established by the sensorimotor mechanism of the system, or derived from other information.

This example is set at an early stage of language learning, where sentences are in the "telegraphic speech" form, i.e., as sequences of words. Features like tense, plurality, and the use of capital letter have not been taken into account yet, though in principle they can all be handled at later stages of learning.

To make the discussion simple, all the initial beliefs are given the default truth-value $\langle 1, 0.9 \rangle$. According to the previous description of EGS, we see that it corresponds to the situation where the statement has been tested nine times, and in all cases the inheritance relation is confirmed.

Since the objective of this test is to check the capability of the grammar rules and the inferential rules, the details of memory and control are left out. Instead, in each inference step two premises are selected manually, and only the relevant conclusions are mentioned in the following discussion.

From knowledge (1)-(3), the system can uses the *induction* rule to derive generalized knowledge, and in the process variable terms are introduced into the conclusions:

$$(\{\$1 \times \$2\} \rightarrow represent) \Rightarrow$$
$$(\{\{\$1 \times \mathbf{eat} \times \mathbf{fish}\} \times ((\$2 \times fish) \rightarrow food)\} \rightarrow represent) \ \langle 1, 0.45 \rangle \tag{4}$$

$$((\{\$1 \times \$2\} \rightarrow represent) \wedge (\{\$3 \times \$4\} \rightarrow represent)) \Rightarrow$$
$$(\{\{\$1 \times \mathbf{eat} \times \$3\} \times ((\$2 \times \$4) \rightarrow food)\} \rightarrow represent) \ \langle 1, 0.29 \rangle \tag{5}$$

The above conclusions will contribute to the meaning of the phrase "eat fish" and the word "eat", respectively. Here we can see that how far a sentence will be generalized depends on the selection of the other premise, and the further the generalization goes, the less confident the result will be.

At this stage, if the system gets other sample sentences with "eat fish" or "eat" and are also mapped into the same terms, similar inductive generalizations

will happen. The resulting beliefs will be combined by the revision rule to get more and more confident generalizations. On the other hand, if counter examples are found, the corresponding inductive conclusions will have low *frequency* values, which will also decrease the *frequency* values of the overall summary, though still increase its *confidence* value. With the accumulation of evidence, the repeated patterns in the language will produce linguistic knowledge with higher and higher confidence, while all the knowledge remains revisable by new experience.

From (4)-(5) and the following given beliefs

$$\{\mathbf{dog} \times dog\} \rightarrow represent\ \langle 1, 0.9\rangle \tag{6}$$

$$\{\mathbf{meat} \times meat\} \rightarrow represent\ \langle 1, 0.9\rangle \tag{7}$$

the system can derive the following conclusions using the *deduction* rule:

$$\{\{\mathbf{dog} \times \mathbf{eat} \times \mathbf{fish}\} \times ((dog \times fish) \rightarrow food)\} \rightarrow represent\ \langle 1, 0.41\rangle \tag{8}$$

$$\{\{\mathbf{dog} \times \mathbf{eat} \times \mathbf{meat}\} \times ((dog \times meat) \rightarrow food)\} \rightarrow represent\ \langle 1, 0.26\rangle \tag{9}$$

These conclusions have relatively low confidence, but can still let the system understand and produce novel sentences it never heard before, by answering questions like

$$\{\{\mathbf{dog} \times \mathbf{eat} \times \mathbf{fish}\} \times ?x\} \rightarrow represent\ ? \tag{10}$$

$$\{?y \times ((dog \times meat) \rightarrow food)\} \rightarrow represent\ ? \tag{11}$$

For a given question, if there are multiple candidate answers, the choice rule will decide the best answer, as explained previously.

In this way, NARS can understand and produce novel sentences in a natural language through reasoning, according to the linguistic knowledge it learned from its experience. This process does not require the existence of a complete grammar of the language, and nor is there a pure syntactic parsing process.

In this example, the only syntactic information involved is word order, though it does not mean that this approach can only process this type of information. Since every recognizable pattern in the system's experience can be named by a term and stored in a concept, other linguistic relations (which can be syntactic, grammatical, morphological, etc) are uniformly expressed as *conceptual* relations that can be used by the inference rules.

In general, the learning of a natural language is handled just like the learning of knowledge in other domains. The system will use the available knowledge to organize the experience into simple, stable, and applicable patterns to be applied to novel situations.

5.2 Comparisons and implications

The NARS approach toward NLP is fundamentally different from the other approaches explored in AI and linguistics, though still similar to them here or there.

As discussed at the beginning of the article, NARS rejects the two fundamental Chomskyan assumptions, i.e., the modularity of language processing and the autonomy of syntax, semantics, and pragmatics. NARS does have an innate competence, but it is a reasoning-learning capacity. Narsese has a built-in formal grammar, but it is for a *logical* language, which is very different from natural languages. For instance, Narsese sentences are already in tree structure and NARS has no innate notions of noun, verb, adjective, etc. Furthermore, in NARS the syntactic knowledge of a natural language does not take the form of a grammar, but associated to individual words or types of words with various level of abstraction. When processing an input English sentence, there is no parsing, but a reasoning process that uniformly handle pragmatics, semantics, and syntax.

Similar to the positions taken by function grammar [Dik, 1978] and cognitive linguistics [Croft and Cruse, 2004], in NARS language understanding starts at pragmatics, since NLP tasks are all carried out to satisfy certain needs of the system, and in certain context. Then comes semantic analysis where words are mapped into terms, and the ambiguity in the process are partly resolved by the pragmatic factors (motivation, context, etc). Syntactic analysis becomes necessary only when the meaning to be expressed is complicated enough for compound terms to be used, as well as there is additional information on tense, plurality, etc., to be expressed. Even in this situation, syntactic analysis still depends on the guidance of pragmatic and semantic factors.

There are other NLP models where semantic information is used to guide syntactic analysis, such as *case grammar* [Fillmore, 1968], *functional grammar* [Dik, 1978], *conceptual dependency theory* [Schank, 1980], etc., where the semantic information is typically represented by *frames* that each corresponds to a verb, with functional slots to be filled in for *agent*, *goal*, *recipient*, and so on. NARS shares some ideas with these approaches, as shown by knowledge like "$(\{\$1 \times \$2\} \rightarrow represent) \wedge (\{\$3 \times \$4\} \rightarrow represent)) \Rightarrow (\{\{\$1 \times \mathbf{eat} \times \$3\} \times ((\$2 \times \$4) \rightarrow food)\} \rightarrow represent)$", though the meaning of a word is not predetermined by a single frame, but learned as a set of judgments. On the other hand, the representation of NARS is also different from semantic network [Shapiro and Neal, 1982] in that Narsese contains a constant number of term types that are directly recognized and processed by the inference rules.

NARS agrees with the statistical approach of NLP on that the linguistic knowledge is statistical by nature, and should be learned from actual usage data. However, NARS does not make the usual assumption that the statistical data comes from a consistent probability distribution. Consequently, the calculations of truth-values do not follow probability theory (though have similarity with it in certain places). Furthermore, language learning in NARS is not "corpus-based", but happens in a context where NLP is associated with motivation, sensorimotor, and every sentence serves certain purpose and function to the system, rather than as an observation of linguistic materials. In this way, pragmatic and semantic information are related to syntactic information at the very beginning, rather than associated with it later. Finally, the statistical approach treats a sentence according to the *average* situation, while NARS pro-

cesses every sentence in a *case-by-case* manner, by taking the current context into account.

Compared with other inference-based NLP works [Shapiro and Neal, 1982, Hobbs et al., 1993, Krovetz, 2000], NARS differs from them in all the major components of a reasoning system:

- The system has a term-oriented language with two-factor truth-values,
- The language is used according to an experience-grounded semantics,
- The inference rules are syllogistic and compositional,
- The memory structure is concept-centered and priority-based,
- The control strategy manages dynamic resource allocation.

5.3 Summary

The NARS approach toward NLP has the following major properties:

- The processing of natural languages is unified with other cognitive processes, such as reasoning, learning, and categorizing. The only specialty is the linguistic knowledge, which is language-specific.
- All types of linguistic knowledge are learned from experienced language use, rather than built into the system. The learning process follows inference rules, including *deduction*, *induction*, *abduction*, *analogy*, *revision*, and so on. The system can directly accept syntactic, semantic, and pragmatic knowledge from the outside, so as to bypass some internal learning process. Such input knowledge will still be revised according to the system's experience. Compared to the existing approaches of NLP, this approach may be more similar to how a human being learns a natural language, though NARS is not designed as a descriptive model of the human mind.
- A natural language is treated as a conceptual system that changes over time. New words, phrases, and sentences are introduced from time to time, and the existing ones may change their meaning gradually. Even grammatical conventions may change over time. Some of the changes are long-term and irreversible (such as the learning and evolving of word meaning), while some other changes are short-term and temporary (such as the content-dependent interpretations of words). The system has the ability to adapt to such changes that come from the environment, as well as to initiate such changes by using a language creatively.
- Though the distinction of syntax, semantics, and pragmatics still exist, these aspects of a language are not processed separately in the system. Syntactic knowledge relates the words and phrases in a natural language in various ways; semantic knowledge associates the words and phrases to

the terms/concepts; pragmatic knowledge links knowledge to goals (not discussed in this paper). However, they are all conceptual relations in NARS.

- Syntactic knowledge has various generality, with concepts from specific (like "cat" and "eat") to general (like "noun" and "verb") at multiple levels. In the long run, the system will balance the demand on generality and specificity, so as to improve the efficiency of the usage of its knowledge.

NLP in NARS is a new research topic that has only produced some preliminary results, so it is too early to make strong conclusions about its strength and weakness. However, at least we can say that it has many novel and interesting properties, so deserves further exploration.

References

[Allen, 1994] Allen, J. F. (1994). *Natural Language Understanding*. Addison-Wesley, Reading, Massachusetts, 2nd edition.

[Barwise and Perry, 1983] Barwise, J. and Perry, J. (1983). *Situations and Attitudes*. MIT Press, Cambridge, Massachusetts.

[Bloom, 2000] Bloom, P. (2000). *How Children Learn the Meanings of Words*. MIT Press, Cambridge, MA.

[Brachman, 2006] Brachman, R. J. (2006). (AA)AI — more than the sum of its parts, 2005 AAAI Presidential Address. *AI Magazine*, 27(4):19–34.

[Chomsky, 1957] Chomsky, N. (1957). *Syntactic Structures*. Mouton, The Hague.

[Croft and Cruse, 2004] Croft, W. and Cruse, D. A. (2004). *Cognitive Linguistics*. Cambridge University Press, Cambridge.

[Dik, 1978] Dik, S. C. (1978). *Functional Grammar*. North-Holland, Amsterdam.

[Fillmore, 1968] Fillmore, C. J. (1968). The case for case. In Bach, E. and Harms, R. T., editors, *Universals in Linguistic Theory*, pages 1–88. Holt, Rinehart and Winston, New York.

[Flach and Kakas, 2000] Flach, P. A. and Kakas, A. C. (2000). Abductive and inductive reasoning: background and issues. In Flach, P. A. and Kakas, A. C., editors, *Abduction and Induction: Essays on their Relation and Integration*, pages 1–27. Kluwer Academic Publishers, Dordrecht.

[Fodor, 1983] Fodor, J. A. (1983). *The Modularity of Mind*. MIT Press, Cambridge, Massachusetts.

[Goertzel and Pennachin, 2007] Goertzel, B. and Pennachin, C., editors (2007). *Artificial General Intelligence*. Springer, New York.

[Hobbs et al., 1993] Hobbs, J. R., Stickel, M. E., Appelt, D. E., and Martin, P. (1993). Interpretation as abduction. *Artificial Intelligence*, 63(1-2):69–142.

[Jurafsky and Martin, 2009] Jurafsky, D. and Martin, J. H. (2009). *Speech and Language Processing*. Pearson Prentice Hall, Upper Saddle River, New Jersey, 2 edition.

[Kowalski, 1979] Kowalski, R. (1979). *Logic for Problem Solving*. North Holland, New York.

[Krovetz, 2000] Krovetz, R. (2000). Viewing morphology as an inference process. *Artificial Intelligence*, 118:277–294.

[Manning and Schütze, 1999] Manning, C. D. and Schütze, H. (1999). *Foundations of Statistical Natural Language Processing*. MIT Press, Cambridge, MA, USA.

[McCarthy, 1993] McCarthy, J. (1993). Notes on formalizing contexts. In *Proceedings of the thirteenth international joint conference on artificial intelligence*, pages 555–560.

[Peirce, 1931] Peirce, C. S. (1931). *Collected Papers of Charles Sanders Peirce*, volume 2. Harvard University Press, Cambridge, Massachusetts.

[Piattelli-Palmarini, 1980] Piattelli-Palmarini, M., editor (1980). *Language and Learning: The Debate between Jean Piaget and Noam Chomsky*. Harvard University Press, Cambridge, Massachusetts.

[Russell and Norvig, 2010] Russell, S. and Norvig, P. (2010). *Artificial Intelligence: A Modern Approach*. Prentice Hall, Upper Saddle River, New Jersey, 3rd edition.

[Schank, 1980] Schank, R. C. (1980). Language and memory. *Cognitive Science*, 4:243–284.

[Shapiro and Neal, 1982] Shapiro, S. C. and Neal, J. G. (1982). A knowledge engineering approach to natural language understanding. In *Proceedings of the 20th Annual Meeting of the Association for Computational Linguistics*, pages 136–144.

[Szabó, 2005] Szabó, Z. G., editor (2005). *Semantics versus Pragmatics*. Oxford University Press, Oxford.

[Tomasello, 2003] Tomasello, M. (2003). *Constructing a language: A usage-based theory of language acquisition*. Harvard University Press, Cambridge, Massachusetts.

[Turing, 1950] Turing, A. M. (1950). Computing machinery and intelligence. *Mind*, LIX:433–460.

[Wang, 2005] Wang, P. (2005). Experience-grounded semantics: a theory for intelligent systems. *Cognitive Systems Research*, 6(4):282–302.

[Wang, 2006] Wang, P. (2006). *Rigid Flexibility: The Logic of Intelligence*. Springer, Dordrecht.

[Wang, 2009] Wang, P. (2009). Formalization of evidence: A comparative study. *Journal of Artificial General Intelligence*, 1:25–53.

[Wang, 2011] Wang, P. (2011). The assumptions on knowledge and resources in models of rationality. *International Journal of Machine Consciousness*, 3(1):193–218.

[Wang, 2012] Wang, P. (2012). Solving a problem with or without a program. *Journal of Artificial General Intelligence*, 3(3):43–73.

[Wang, 2013] Wang, P. (2013). *Non-Axiomatic Logic: A Model of Intelligent Reasoning*. World Scientific, Singapore.

[Wang and Goertzel, 2007] Wang, P. and Goertzel, B. (2007). Introduction: Aspects of artificial general intelligence. In Goertzel, B. and Wang, P., editors, *Advance of Artificial General Intelligence*, pages 1–16. IOS Press, Amsterdam.

[Wang and Hofstadter, 2006] Wang, P. and Hofstadter, D. (2006). A logic of categorization. *Journal of Experimental & Theoretical Artificial Intelligence*, 18(2):193–213.