Natural Language Processing II (SC674)

Prof. Feng Zhiwei

Ch8. Semantics in NLP

8.1 Representing Meaning

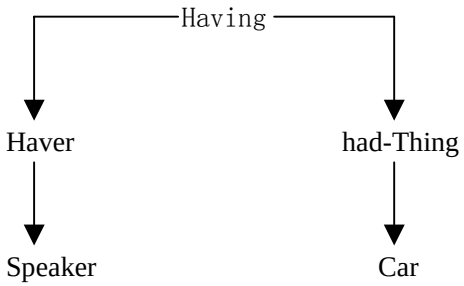
Representing meaning can bridge the gap from linguistic inputs to the kind of non-linguistic knowledge needed to perform a variety of tasks involving the meaning of linguistic inputs.

In NLP, there are four frequently used meaning representations (“I have a car”)

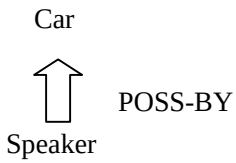
- First Order Predicate Calculus (FOPC)

$$\exists \text{ x, y Having (x)} \wedge \text{Haver (Speaker, x)} \wedge \text{HadThing (y, x)} \wedge \text{Car (y)}$$

- Semantic Network



- Conceptual Dependency diagram



- Frame-based Representation

Having
Haver: Speaker
HadThing: Car

These representations can be used to link linguistic inputs to the world and to our knowledge about it.

8.1.1 Predicate-Argument Structure

All human languages have a form of predicate-argument arrangement at the core of their semantic structure. One of the most important jobs of a grammar is to help organize this predicate-argument structure.

■ Verb as predicate:

E.g. 1. I want Chinese food.

2. I want to spend less than five dollars.

3. I want it to be close by here

There examples can be classified as having one of the following three syntactic argument frame:

NP **want** NP

NP **want** inf-VP

NP **want** NP inf-VP

These syntactic frames specify the number, position and syntactic category of the arguments that are expected to accompany a verb.

For example, (1) specifies the following facts:

① There are two arguments (I, Chinese food) to this predicate (want);

② Both arguments must be NPs;

③ The first argument (I) is pre-verbal and plays the role of the subject;

④ The second argument (Chinese food) is post-verbal and plays the role of the direct object.

This kind of information is quite valuable in capturing a variety of important facts about syntax. By analyzing easily observable semantic information associated with these frame, we can also gain considerable insight into our meaning representation:

① Semantic role (thematic role, or case role): e.g. in sentences 1, 2, 3, the pre-argument always plays the role of the entity doing the wanting (the *wanter*), the post-verbal argument plays the role of the concept that is *wanted*.

② Semantic restriction on these roles (selectional restriction): In sentences 1,2,3, each initial argument must belong to the concepts that can play the role of wanter in any straightforward manner

■ Preposition as predicate:

E,g, "A Chinese restaurant **under** fifteen dollars"

"under is a preposition. It can be characterized as two-arguments predicate where the first argument is an object that is being placed in some relation to the second argument.

Under (*ChineseRestaurant*, $15)

■ Noun as predicate:

E.g. "Make a **reservation** for this evening for a table for two persons at 8:00."

In this sentence, the predicate-argument structure is based on the concept understanding the noun "reservation", rather than "make", the main verb of this sentence:

Reservation (Hearer, Today, 8PM, 2)

8.1.2 First Order Predicate Calculus (FOPC)

8.1.2.1 Elements of FOPC:

We can use CFG to specify the syntax of FOPC as follows:

Formula → AtomicFormula

| Formula Connective Formula

| Quatifier Variable … Formula
| ¬ Formula
| (Formula)

AtomicFormula → Predicate (Term …)

Term → Function (Term …)
| Constant
| Variable

Connective → ∧ | ∨ | =>

Quantifier → ∀ (for all) | ∃ (there exists)

Constant → *A* | *VegetarianFood* | *Sanchon* | …
Variable → x | y | ...
Function → *LocationOf* | *CuisineOf* | ...

FOPC provides three types of Term: constant, function, and variable.

Constants in FOPC refer to specific objects in the world being described.. E,g. single capitalized letter "A", or concrete words as "*VegetarianFood* ", "*Sanchon*".

Functions in FOPC correspond to concepts that are often expressed in English as genitive such as "the location of Sanchon" or "Sanchon's location". It is written as:

LocationOf (Sanchon)

FOPC functions are syntactically the same as single argument predicates. Functions provide a convenient way to refer to specific objects without having to associate a named object.

Variable in FOPC is a mechanism for referring to objects. Variables give us the ability to make assertions and draw inferences about objects without having to make reference to any particular named object.

E.g. 1. "Sanchon serves vegetarian food" can be described in FOPC as:

Server (Sanchon, VegetarianFood)

This FOPC sentence asserts that "Server", a two-place predicate, holds between the objects denoted by the constants "Sanchon" and "VegetarianFood".

E.g. 2. "Sanchon is a restaurant" can be described in FOPC as:

Restaurant (Sanchon)

This is a one-place predicate, not to relate multiple objects, but rather to assert a property of a single object "Sanchon".

E.g. 3. "I only have five dollars and I don't have a lot of time." Can be described by complex representation with logical connectives in FOPC as follows:

Have (Speaker, FiveDollars) ∧ ¬ Have (Speaker, LotOfTime)

The recursive nature of the grammar CFG allow an infinite number of logical formulas to be created through the use of these logical connectives. Thus we have the ability to create an infinite number of representations using a finite device FOPC.

8.1.2.2 The semantic of FOPC

The objects, properties, and relations represented in a FOPC knowledge base acquire their meanings by virtue of their correspondence to objects, properties, and relations out in the external world modeled by the knowledge base. Therefore, FOPC sentences can be assigned a value of True or False based on whether the propositions they encode are in accord with the world or not.
E.g. “Log-house is near KAIST.”
Its FOPC representation is:

Near (LocationOf (Log-house), LocationOf (KAIST))

This logical formula can be assigned a value of True of False based on whether or not the real Log-house is actually close to KAIST or not.
We can use the “database semantics” to determine the truth of our logical formulas.
The atomic formulas are taken to be true if they are literally present in the knowledge base or if they can be inferred from other formula that are in the knowledge base.
Following Truth Table can give the semantics of various logical connectives.

P	O	ㄱ P	P∧Q	P∨Q	P=>Q
-	-	+	-	-	+
-	+	+	-	+	+
+	-	-	-	+	-
+	+	-	+	+	+

Fig. Truth Table

“+” means “true”, “-“ means “false”, “∧” means “and”, “ㄱ ” means “not”, “∨” means “or”, “=>” means “implies”.

8.1.2.3 Variable and Quantifiers

The variables are used in two ways in FOPC:

- To refer to particular anonymous objects;
- To refer generally to all objects in a collection.

The two quantifiers in FOPC are:

- ∀ : “for all”
- ∃ : “there exists”

E. g.1. “a restaurant that serves Japanese food near KAIST”

Here reference is being made to an anonymous object of a specified category with particular properties. Its FOPC formula is:

∃ x Restaurant (x)
∧ Serves (x, JapaneseFood)
∧ Near ((LocationOf (x), LocationOf (KAIST))

It says that for this sentence to be true there must be at least one object such that if we were to substitute it for the variable x, the resulting sentence would be true.
For example, we substitute x with “Maru”

∃ x Restaurant (Maru)

∧ Serves (Maru, JapaneseFood)
∧ Near ((LocationOf (Maru), LocationOf (KAIST))

E.g. 2. “All vegetarian restaurants serve vegetarian food.”

Its FOPC formula is:

$\forall$ x VegetarianRestaurant (x) => Serves (x, VegetarianFood)

For this sentence to be true, it must be the case that every substitution of a known object for x must result in a sentence that is true.

We can divide up the set of all possible substitutions into the set of objects consisting of vegetarian restaurant and the set consisting of everything else.

- The substituted object actually is a vegetarian restaurant:

  VegetarianRestaurant (Sanchon) => Serves (Sachon, VegetarianFood)

If we assume that we know that the consequent clause:

Serves (Sanchon, VegetarianFood)

is true then this sentence as a whole must be true. Both the antecedent and the consequent have the value True and, therefore the sentence itself can have the value True. ((In according to the Truth table :“if P is true and Q is true, then P=>Q must be true”)

- The substituted object is non-vegetarian restaurant:

  VegetarianRestaurant (Maru) => Serves (Maru, VegetarianFood)

  If the antecedent P “VegetarianRestaurant (Maru)” is false, then regardless of the consequent Q “Serves (Maru, VegetarianFood)” is true or false, the implication “P=>Q” shall be always true, so the implication “VegetarianRestaurant (Maru) => Serves (Maru, VegetarianFood)” must be true.

Variables in logical formulas must be either existentially ($\exists$) or universally ($\forall$) quantified. To satisfy an existentially quantified variable, there must be at least one substitution that results in a true sentence. Sentence with universally quantified variables must be true under all possible substitution.

8.1.2.4 Inference

The inference is the ability to add valid new propositions to a knowledge base, or to determine the truth of proposition not explicitly contained within knowledge base.

The modusponens is the most important inference method provided by FOPC.

Modus Ponens is informally known as “if-then reasoning”. We can define modus ponens as follows:

$$\frac{\alpha \quad \alpha \Rightarrow \beta}{\beta}$$

Where, α and β are FOPC formulas.

E. g. VegetarianRestaurant (Sanchon)

VegetarianRestaurant (x) => Serves (x, VegetarianFood)

Serves (Sanchon, VegetarianFood)

Here, the formula “VegetarianRestaurant (Sanchon)” matches the antecedent of the rule, thus

allowing us to use modus ponens to conclude: “Serves (Sanchon, VegetarianFood)”.
Modus Ponens is put to practical use in one of two ways:

- **Forward chaining**: As soon as a new fact is added to the knowledge base, all applicable implication rules are found and applied, each resulting in the addition new facts to the knowledge base. These new propositions in turn can be used to fire implication rules applicable to them. The process continues until no further facts can be deduced.
- **Backward chaining**: modus ponens is run in reverse to prove specific propositions, called “queries”. The process is as follows:

  ① First step: To see if the query formula is true by determining if it is present in the knowledge base. If “yes”, the query is confirmed. If “no”, then go to second step.

  ② Second step: To search for applicable implication rules present in the knowledge base. An applicable rule is one where the consequent of the rule matches the query formula. If there are any such rules, then query can be proved if the antecedent of any one them can be shown to be true. This can be performed recursively by backward chaining on the antecedent as a new query.

  E.g. If we ask to verify the truth of the proposition:

  “Serves (Sanchon, VegetarianFood)”,

  Since it is not present in the knowledge base, a search for an applicable rule is initiated. After substituting, the constant “Sanchon” for the variable x, our next task is to prove the antecedent of the rule “VegetarianRestaurant (Sanchon)”. This fact presents in our knowledge base, so the proposition “Serves (Sanchon, VegetarianFood)” is true.

The reasoning backward is from known consequents to unknown antecedents. To be specific, by reasoning backward we mean that if the consequent of a rule is known to be true, we assume that the antecedent will be as well. For example, let’s assume that we know the “Serves (Maru, VegetarianFood)” is true. Since this fact matches the consequent of our rule, we might reason backwards to the conclusion that “VegetarianRestaurant (Maru)”. But we know, there is possibility for Restaurant Maru to serve meat, because it is a Japanese restaurant (of course, it may also serve vegetarian food.)

Obviously the reasoning backwards is an invalid form of plausible (seeming to be true) reasoning. The plausible reasoning from consequent to antecedent is known as “**abduction**”.

8.1.3 Some Linguistically Relevant Concepts

8.1.3.1 Categories

There are two representations for category:

- To represent the category as predicate:

The most common way to represent categories is to create a unary predicate for each category of interest. E.g.

VegetarianRestaurant (Sanchon)

But if we want to designate the “most popular” member of a given category as in the following expression:

MostPopular (Sanchon, VegetarianRestaurant)

But it is not legal FOPC formula since the arguments to predicates in FOPC must be terms, not other predicates.

One way to solve this problem is to use technique called "**reification**".

- To represent category by reification technique:

By reification technique, we can represent the category of VegetarianRestaurant as an object just as "Sanchon" is. The notion of membership in such a category is denoted via a membership relation **ISA**:

ISA (Sanchon, VegetarianRestaurant)

The relation denoted by ISA (is a) holds between objects and the categories in which they are members.

The reification can be extended to create hierarchies of categories.

AKO (VegetarianRestaurant, Restaurant)

Here the relation **AKO** (a kind of) holds between categories and it denotes a category inclusion relationship.

8.1.3.2 Events

The representation for events that we have used until now has consisted of single predicate with as many arguments. E.g. "Make a reservation for this evening for a table for two persons at 8 in Log-house Restaurant." Can be represented as:

Reservation (Hearer, Log-house, Today, 8PM, 2)

Now we discuss how to represent a series of events>

E.g. ① I ate

②. I ate a sandwich.

③ I ate a sandwich at my desk.

④ I ate at my desk.

⑤ I ate lunch.

⑥ I ate a sandwich for lunch.

⑦ I ate a sandwich for lunch at my desk.

The representation approaches are as follows:

- To create as many different eating predicates to handle all of the ways of the action.

  Eating1 (Speaker)
  Eating2 (Speaker, Sandwich)
  Eating3 (Speaker, Sandwich, Desk)
  Eating4 (Speaker, Desk)
  Eating5 (Speaker, Lunch)
  Eating6 (Speaker, Sandwich, Lunch)
  Eating7 (Speaker, Sandwich, Lunch, Desk)

But this approach comes at a rather high cost.

- To use meaning postulate

E.g $\forall$ w, x, y, z Eating7 (w, x, y, z) => Eating6 (w, x, y)

This approach ties together the semantics of two of our predicates. Other postulates could be created to handle the rest of the logical relations among the various Eatings and the connections from them to other related concept.

But this approach might be made to work in small domains, and it has a scalability problem.

- To adapt the structure with same predicate with different arguments, and some of the arguments possible miss from surface forms.

  ∃ w, x, y Eating (Speaker, w, x, y)

  ∃ w, x Eating (Speaker, Sandwich, w, x)

  ∃ w Eating (Speaker, Sandwich, w, Desk)

  ∃ w, x Eating (Speaker, w, x, Desk)

  ∃ w, x Eating (Speaker, w, Lunch, x)

  ∃ w Eating (Speaker, Sandwich, Lunch, w)

  Eating (Speaker, Sandwich, Lunch, Desk)

This approach directly yields the obvious logical connections among these formulas without use of meaning postulate Specifically, all of the sentences with ground terms as arguments logically imply the truth of the formulas with existentially bound variables as arguments.

But, this approach has two deficiencies:

Firstly, It makes too many commitments.

Secondly, it does not let us individuate events.

E.g If we have following formulas:

∃ w, x Eating (Speaker, w, x, Desk)

∃ w, x Eating (Speaker, w, Lunch, x)

∃ w Eating (Speaker, w, Lunch, Desk)

If we knew that the first two formulas referring to the same event, they could be combined to create the third representation. However, with the current representation we have no way of telling if this possible. The independent facts that “I ate at my desk” and “I ate lunch” do not permit us to conclude that “I ate lunch at my desk”.

- To employ reification to elevate events to objects that can be quantified and related to an other object via sets of defined relation.

  ∃ w ISA (w, Eating)

  ∧Eater (w, Speaker) ∧Eaten (w, Sandwich)

  This representation states that there is an eating event where the Speaker is doing the eating and a sandwich is being eaten.

  The meaning representation of example ① and ⑥ can be constructed

similarly:

$\exists$ w ISA (w, Eating)

∧Eater (w, Speaker)

$\exists$ w ISA (w, Eating)

∧Eater (w, Speaker) ∧Eaten (w, Sandwich) ∧MealEaten (w, Lunch)

With this reified-event approach:

- There is no need to specify a fixed number of arguments for a given surface predicate, rather as many roles and filters can be glued on as appear in the input.
- No more roles (arguments) are postulated than are mentioned in the input.
- The logical connections among closely related examples is satisfied without the need for meaning postulates.

8.1.3.3 Temporal Logic

The temporal logic (tense logic) holds that the time flows inexorably forward, and that events are associated with either points or intervals in time, as on a timeline. An ordering can be imposed on distinct events by situating them on the timeline. Specifically, we can say that one event precedes another, if the flow of time leads from the first event to the second.. By those notions, we can have the notions of past, present and future.

E.g. I arrived in Seoul

I am arriving in Seoul.

I will arrive in Seoul.

All three events would share the following representation, which lacks any temporal information.

$\exists$ w ISA (w, Arriving)

∧ Arriver (w, Speaker) ∧ Destination (w, Seoul)

The temporal information provided by the tense of the verb can be exploited by predicating additional information about the event variable w. We can add temporal variables representing the interval corresponding to the event (i), the end of point of the event (e), Such we can get following different representations:

$\exists$ i, e, w ISA (w, Arriving)

∧ Arriver (w, Speaker) ∧ Destination (w, Seoul)
∧ IntervalOf (w, i) ∧ EndPoint (i, e) ∧ Precedes (e, Now)

$\exists$ i, e, w ISA (w, Arriving)

∧ Arriver (w, Speaker) ∧ Destination (w, Seoul)
∧ IntervalOf (w, i) ∧ MemberOf (i, Now)

$\exists$ i, e, w ISA (w, Arriving)

∧ Arriver (w, Speaker) ∧ Destination (w, Seoul)
∧ IntervalOf (w, i) ∧ EndPoint (i, e) ∧ Precedes (Now, e)

The variable “i” stands for the interval of time associated with the event. The variable “e” stands for the end of that interval. The two-place predicate “Precedes” represents the notion that the first time point argument precedes the second time; the constant “Now” refers to the current time. For past event, the end point of the interval must precede the “Now”. For future event, the current time must precede the end of the event. For the event in the present, the current time is contained within the event interval.

In order to represent the perfect tense, Reichenbach introduced the notion of a **reference point.** The notion of reference point (R) is separated out from the utterance time (U) and the event time.(E)

E.g. When John’s flight departed, I ate lunch.

When John’s flight departed, I had eaten lunch.

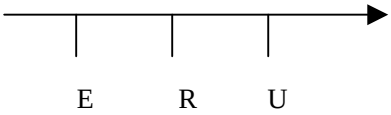
The “departure” event is the reference point. The “eating” is the event. In the first sentence, the reference point (departure) and the eating point occurred in same time; in second sentence, the eating point precedes the reference point.

Simple past

When John’s flight departed, I ate.

Past perfect

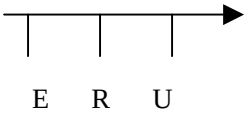
When John’s flight departed, I had eaten.



If we apply Reichenbach approach to the various English tense, we can illustrate these tense as following diagram:

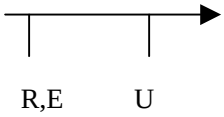
Past perfect

I had eaten



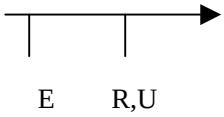
Simple past

I ate



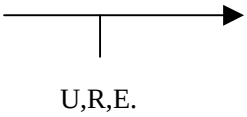
Present perfect

I have eaten



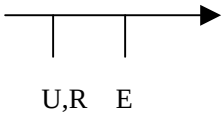
Present

I eat



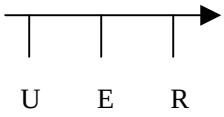
Simple future

I will eat



Future perfect

I will have eaten



8.1.3.4 Modal Logic:

There are some words have a world creating ability. E.g. “believe, want, know, imagine” etc. their meaning representation contain logic formulas that are not intended to taken as true in the real world, but rather as part of some kind of hypothetical world..

These world-creating words generally take various sentence-like constituents as arguments.

E.g. “I believe that Mary ate Japanese food.”

Applying our event-oriented approach, there are two events underlying this sentence: a believing event relating the speaker to some specific belief, and an eating event that plays the role of the believing thing. Using our reified event approach, we can produce following kind of representation:

$\exists$ u, v ISA (u, Believing) $\wedge$ ISA (v, Eating)

$\wedge$ Believer (u, Speaker) $\wedge$ BelievedProp (u, v)

$\wedge$ Eater (v, Mary) $\wedge$ Eaten (v, JapaneseFood)

However, in conjunctive representations like this all of the individual conjuncts must be taken to be true. In this case, this results in a statement that there actually was an eating of Japanese food by Mary. However, It is just a "believe", it is not certainly a fact.

In order to solve this problem, we can introduce an operator called "Believes" that takes two FOPC formulas as its arguments: a formula designating a believer, and a formula designating the believed proposition. Applying this operator would result in the following representation:

Believes (Speaker, $\exists$ v ISA (v, Eating) $\wedge$ Eater (v, Mary) $\wedge$ Eaten (v, JapaneseFood)

Under this approach, the contribution of the word "believes" to this meaning representation is not a FOPC proposition at all, but it is rather an operator that is applied to the believed proposition.

The operator like "believes" that apply to logical formulas are known as "modal operator". A logic augmented with such operators is known as modal logic.

The modal logic can be applied in commonsense knowledge representation.

8.1.4 Alternative Approaches to Meaning

8.1.4.1 Meaning as Action

In this approach, the utterances are viewed as actions, and the meaning of these utterances resides in procedures that are activated in the hearer as a result of hearing the utterance. Terry Winograd said (1972):

"One of the basic viewpoints underlying the model is that all language use can be thought of as a way of activating procedures within the hearer. We can think of an uttrance as a program – one that indirectly causes a set of operations to be carried out within the hearer's cognitive system." (<Understanding Natural Language>, Academic Press)

E.g. "John is walking to the store" activates the process state the walking event.

"John walked to the store' activates the result state.

"John walked to the store every week" activates the iterative process.

8.1.4.2 Meaning as Truth

The role of formal meaning representations in computational linguistics, NLP, AI, and cognitive modeling, is quite different from the role in more philosophical circles.

To philosophers, the translation of a sentence from its original natural form to another artificial form does not get us any closer to its meaning. Under this view, the important work is to determine the mapping from meaning representation to the world being modeled. The function that determine the truth conditions of sentences is the particular interesting work in this approach.

8.1 Semantic Analysis

8.1.1 Syntax-Driven Semantic Analysis

This approach is based on the principle of compositioality proposed by Frege (called as Frege principle of compositionality). The key idea of compositionality is that the meaning of a sentence can be composed from the meaning of the parts.

The meaning of a sentence is not based solely on the words that make it up, it is based on the ordering, grouping, and relations among the words in the sentence. The meaning of a sentence is partially based on based on its syntactic structure. Therefore, in syntax-driven semanticanalysis, the composition of meaning representations is guides by the syntactic components and relations provided by the grammar.

We assume that the syntactic analysis of a input sentence will form the input to a semantic analyzer. It is a pipeline-oriented approach:

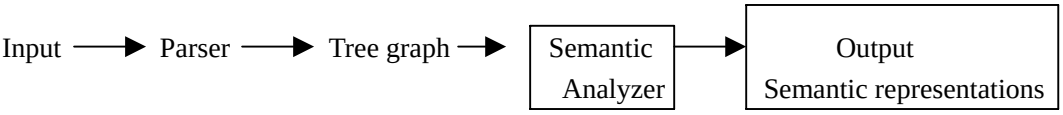


Fig. 1 Pipeline approach

In Fig. 1, the syntactic representation is tree graph. But other syntactic representation can also be used: e.g. feature structure, lexical dependency diagram, etc.

Generally speaking, the process of semantic analysis can be divided to following steps:

1. To associate the normal FOPC expression with lexical units.
2. To copy the semantic values from children to parents.
3. To associate the function-like λ -expressions with lexical items, then to apply the function-like λ -expressions to the semantics of one or more children of a constituent.
4. To use the complex-terms to allow quantified expressions to be temporarily treated as terms.

Example: semantic analysis of sentence “Maru serves meat.”.

1. To associate the normal FOPC expression with lexical units.

The tree graph of the sentence is:

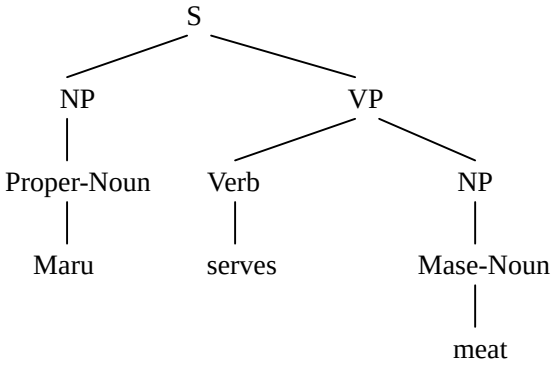


Fig. 2 tree graph

We use the augmenting CFG rules with semantic attachments to do the syntactic persing.

Abstractly, the augmenting CFG rules have following structure:

$$A \rightarrow \alpha_1 \ldots \alpha_n \quad \{f(\alpha_j.sem, \ldots, \alpha_k.sem)\}$$

The semantic attachment to the basic CFG rule is showed in the {..} to the right of the rule’s syntactic constituents. This notation states that the meaning representation assigned to the constituent A, which we will denote as **A.sem**, can be computed by running the function f on some

subset of the semantic attachments of A's constituents.

For our example, we will begin with the more concrete entities in the sentence, as specified by the noun phrase, and work our way up to the more complex expressions representing the meaning of the entire sentence. The concrete entities in our sentence are represented by the FOPC constants *Maru* and *Meat*. Our first task is to associate these constants with the constituents of the tree that introduced them.

ProperNoun → Maru {*Maru*}

MassNoun → meat {*Meat*}

These two rules specify that the meanings associated with the subtrees generated by these rules consists of the constants *Maru* and *Meat*

2. To copy the semantic values from children to parents.

In the tree, the upper NP's obtain their meaning representations from the meanings of their children. We will assume the meaning representation \s of the children are simply copied to the parents.

NP → ProperNoun {*ProperNoun.sem*}

NP → MassNoun {*MassNoun.sem*}

These rules state that the meaning representations of the noun phrases are the same as the meaning representations of their individual components, denoted by *ProperNoun.sem* and *MassNoun.sem*. In general, it will be the case that for non-branching grammar rules, the semantic expression associated with the child will be copied unchanged to the parent.

3. To associate the function-like λ-expressions with lexical items and then to apply the function-like λ-expressions to the semantics of one or more children of a constituent.

.

After copying the semantic representation to parent, we can move on to the event underlying this utterance as specified by verb serves. A general **Serving** event involves a **Server** and something **Served**, it can be shown as following logical formula:

$$\exists \text{ e, x, y ISA (e, Serving)} \wedge \text{Server (e, x)} \wedge \text{Served (e, y)}$$

For the semantic attachment of verb "serves", we can simply take this logical formula as its semantic attachment:

Verb → serves

$$\{\exists \text{ e, x, y ISA (e, Serving)} \wedge \text{Server (e, x)} \wedge \text{Served (e, y)}\}$$

Then we move up the parse tree, the next constituent is the VP that is not the non-branching grammar rule, YP dominates both "serves" and "neat". Unlike the NPs, we cannot simply copy the meaning of these children (nodes "Verb" and "NP") up to the parent VP. We need to incorporate the meaning of the NP into the meaning of the Verb and assign the resulting representation to the **VP.sem**.

However, the FOPC formula we attached to the v.sem does not provide any advice about when and how each of its three quantified variables should be replaced.

In this case, we can use the **lambda notation** to resolve this problem.

The lambda notation is a notational extension to FOPC. It provides exactly the kind of formal parameter functionality that we need.

The lambda notation extends the syntax of FOPC to include expressions of the following form:

λ x P(x)

Such expressions consist of the Greek symbol λ, followed by one or more variables, followed by a FOPC expression that makes use of those variables.

The usefulness of these λ-expressions is based on the ability to apply them to logical terms to yield new FOPC expressions, where the formal parameter variables are bound to the specified terms. This process is known as λ-reduction. The following expressions illustrate the application of a expression to the constant A, followed by the result of performing a reduction on this expression:

λ x P(x) (A)

P(A)

The λ-notation provides us two capabilities: 1. the formal parameter list makes a set of variables within the body available; 2. the λ-reduction process implements the desired replacement of variables with terms.

This technique can use the expression as the body of another as on the following expression:

λ x λ y Near (x, y)

The following expressions illustrate a single λ-application and subsequent reduction with this kind of embedded λ-expression:

λ x λ y Near (x, y) (KAIST)

λ y Near (KAIST, y)

The resulting expression is still a λ-expression. The first reduction bound the variable x and removed the outer λ, thus revealing the inner λ-expression. This resulting λ-expression can be applied to another term to arrive at a fully specified logical formula as in the following:

λ y Near (KAIST, y) (Log-House)

Near (KAIST, Log-House)

This technique is called "**currying**". The currying is a way of converting a predicate with multiple arguments into a sequence of single argument predicate.

With the λ-notation and the process of λ-reduction, we have the tools needed to return to the semantic attachment for our VP constituent.

We have:

Verb → serves

{ ∃ e, x, y ISA (e, Serving) ∧ Server (e, x) ∧ Served (e, y)}

Firstly, change the semantic attachment of the verb to λ-expression:

Verb → serves

λ x λ y{ ∃ e, x, y ISA (e, Serving) ∧ Server (e, x) ∧ Served (e, y)}

The body of this verb attachment consists of a λ-expression inside a λ-expression. The outer expression provides the variable that is replaced by the first λ-reduction, while the inner expression can be used to bind the second variable. This ordering of the variables in the multiple layers λ-expression in semantic attachment of the verb explicitly encodes facts about the expected location of a Verb's argument in the syntax.

Secondly, apply this expression to the semantic attachment of VP to combine it with NP semantics:

The attachment for our transitive VP rule specifies a λ−application where the λ−expression is provided by Verb.sem and the argument is provided by NP.sem (the child of VP):

VP → Verb NP {Verb.sem (NP.sem)}

This λ−application results in the replacement of y with the value contained in NP.sem. A λ−reduction removes the λ revealing the inner expression with the parameter y replaced by the constant "Meat". This expression, the meaning of the VP "serves meat", is then the value of VP.sem:

λ x{ ∃ e, x ISA (e, Serving) ∧ Server (e, x) ∧ Served (e, Meat)}

To complete the semantic analysis of this example, we must create the semantic attachment for the S rule. The S rule must incorporate an NP argument into the appropriate role in the event representation now residing in the VP.sem. It should consist of another λ−application where the value of VP.sem provides the λ−expression and the sentence-initial NP.sem provides the final argument to be incorporated:

S → NP VP {VP.sem (NP.sem)}

The result of this λ−application is as follows:

∃ e ISA (e, Serving) ∧ Server (e, Maru) ∧ Served (e, Meat)

The parse tree with semantic attachment of this sentence is:

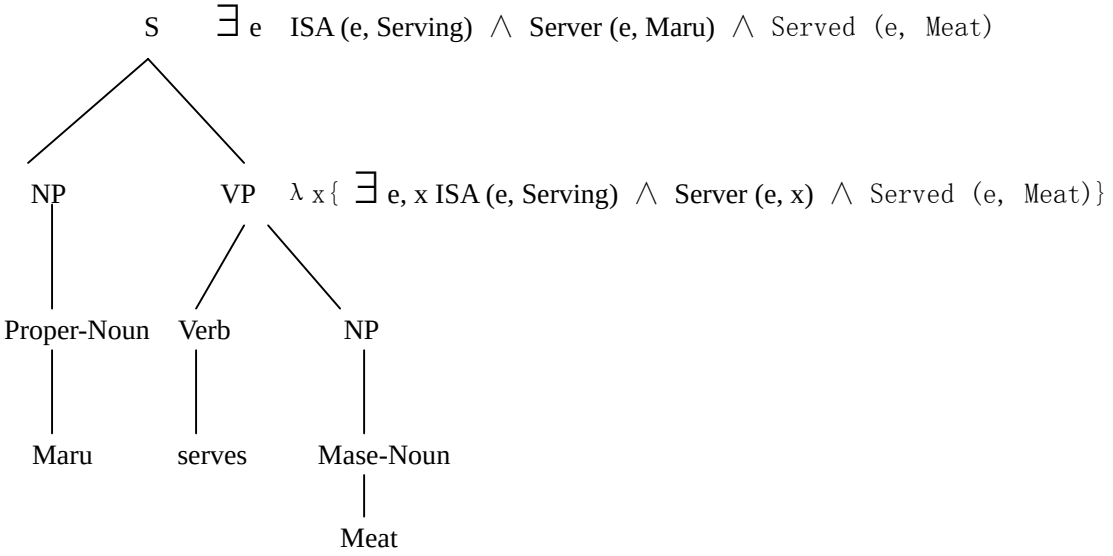


Fig.3 tree graph with semantic attachment

By this approach, we can transform a syntactic tree of the sentence to its semantic representation. This approach is driven by syntax, so it is an approach of syntax-driven semantic analysis

4. To use the complex-terms to allow quantified expressions to be temporarily treated as terms.

Let us consider the sentence "A restaurant serves meat".

Since this sentence is unchanged from above example, we can restrict our attention to the derivation of the semantics of the subject noun phrase and its subsequent integration with the verb phrase in the S rule.

We may assume that following formula is a plausible representation for the meaning of the subject:

∃ x ISA (x, Restaurant)

Then we can embedded this representation of subject in the “Server” predicate:

$\exists$ e ISA (e, Serving) $\wedge$ Server (e, $\exists$ x ISA (x, Restaurant)) $\wedge$ Served (e, Meat)

Although this expression has a certain intuitive appear, it is not a valid FOPC formula. “$\exists$ x ISA (x, Restaurant)" cannot appear as argument in predicate, such arguments are limited to FOPC terms.

We can solve this problem with the notion **complex-term**. The complex-term can allow FOPC expression like “$\exists$ x ISA (x, Restaurant)” to appear in places where normally only ordinary FOPC terms would appear.

Formally, a complex-term is an expression with following three-part structure:

<Quantifier variable body>

Applying this notation to our sentence, we arrive at the following representation

$\exists$ e ISA (e, Serving) $\wedge$ Server (e, <$\exists$ x ISA (x, Restaurant)>) $\wedge$ Served (e, Meat)

We can rewrite any predicate using a complex-term according to the following schema:

P (<Quantifier variable body>)
=>
Quantifier variable body Connective P (variable)

In other words, the complex-term:

- is extracted from the predicate in which it appears;
- is replaced by the variable that represents the object in the question;
- and has its variable, quantifier, and body reprocessed to the new expression through the use of an appropriate connective.

According to this schema, we can have:

Server (e, <$\exists$x ISA (x, Restaurant)>)

=>

$\exists$ x ISA (x, Restaurant) $\wedge$ Server (e, x)

The connective that is used attach the extracted formula to the front of the new expression depends on the type of the quantifier being used:

- $\wedge$ is used with quantifier $\exists$
- => is used with quantifier $\forall$

The semantic representation of our sentence will be as follows:

$\exists$ e ISA (e, Serving) $\wedge$ $\exists$ x ISA (x, Restaurant) $\wedge$ Server (e, x) $\wedge$ Served (e, Meat)

The complex-term

Haver (e, <$\forall$ x ISA (x, Restaurant)>)

Is extended to

$\forall$ x ISA (x, Restaurant) => Haver (e, x)

In our sentence “a restaurant serves meat“, the NP semantic attachment is fairly straightforward.

NP → Det Nominal {<Det.sem x Nominal.sem(x)>}

This attachment creates a complex-term consisting of a quantifier retrieved from the “Det”, following by an arbitrary variable, and then an application of the λ -expression associate with the “Nominal” to that variable. This λ -application ensure that the correct variable appears within the predicate by the “Nominal”.

The attachment for “Det” simply specifies the quantifier to be used:

Det → a {$\exists$ }

The job of the nominal category is to create the ISA formula and λ -expression needed for use in the NP:

Nominal → Noun { λ x ISA (x, Noun.sem)}

Finally, the noun attachment simply provides the name of the category being discussed:

Noun → restaurant {Restaurant}

That is just the meaning in our semantic representation ” $\exists$ x ISA (x, Restaurant)”.

The schema given above to translation expression containing complex-terms into FOPC expression is not unique. Consider following example:

“Every restaurant has a menu”.

We can express it as:

$\exists$ e ISA (e, Having)

$\wedge$ Haver (e, < $\forall$ x ISA (x, Restaurat)>)

$\wedge$ Had (e, < $\exists$ y ISA (y, Menu)>)

If the complex-terms filling the ”Haver“ and the “Had“ roles are rewritten so that the quantifier for the haver role has the outer scope, then the result is the following meaning representation, which corresponds to the common-sense interpretation of this sentence:

$\forall$ x ISA (x, Restaurant) =>

Haver (e, x) $\wedge$ $\exists$ e ISA (e, Having) $\wedge$ $\exists$ y ISA (y, Menu) $\wedge$ Had (e, y)

Replace ISA with predicate ‘Having’ in ‘$\exists$ e ISA (e, Having)’:

$\forall$ x ISA (x, Restaurant) =>

Haver (e, x) $\wedge$ $\exists$ e Having (e) $\wedge$ $\exists$ y ISA (y, Menu) $\wedge$ Had (e, y)

Combine two quantifiers $\exists$ as one:

$\forall$ x ISA (x, Restaurant) =>

Haver (e, x) $\wedge$ $\exists$ e, y Having (e) $\wedge$ ISA (y, Menu) $\wedge$ Had (e, y)

Then we have:

$\forall$ x ISA (x, Restaurant) =>

$\exists$ e, y Having (e) $\wedge$ Haver (e, x) $\wedge$ ISA (y, Menu) $\wedge$ Had (e, y)

This means, for all restaurants, every restaurant has a menu.

On the other hand, if the terms are rewritten in the reverse order, then the following FOPC representation results, which states that there is one menu that all restaurant share:.

$\exists$ y ISA (y, Menu) $\wedge$ Had (e, y) $\wedge$ $\exists$ e ISA (e, Having)

$\wedge$ Haver (e, < $\forall$ x ISA (x, Restaurant)>)

Extend complex-term "Haver (e, < $\forall$ x ISA (x, Restaurant)>)" to "$\forall$ x ISA (x, Restaurant=> Haver (e,x)", we have:

$\exists$ y ISA (y, Menu) $\wedge$ $\forall$ x ISA (x, Restaurant) => Haver (e, x) $\wedge$ Had (e, y) $\wedge$ $\exists$ e ISA (e, Having)

Replace ISA with predicate "Having" in "$\exists$ e ISA (e, Having)":

$\exists$ y ISA (y, Menu) $\wedge$ $\exists$ x ISA (x, Restaurant) => Haver (e, x) $\wedge$ Had (e, y) $\wedge$ $\exists$ e Having (e)

Then we have:

$\exists$ y ISA (y, Menu) $\wedge$ $\forall$ x ISA (x, Restaurant) => $\exists$ e Having (e) $\wedge$ Haver (e, x) $\wedge$ Had (e, y)

This means; there exits a menu and all restaurant has this menu. This meaning is a little strange!

This example illustrates the problem of ambiguous quantifier scooping – a single formula with two complex-terms gives rise to two distinct and incompatible FOPC representations.

8.1.2 Robust Semantic Analysis

There are two primary ways of instantiating a syntax-driven approach in practical systems: Semantic grammar and Information Extraction.

8.1.2.1 Semantic grammar

The syntactic structures provided by traditional CFG grammar are often not particularly well-suited for the task of compositional semantic analysis. There is often mismatch. This

mismatch between the structures provided by traditional grammars and those needed for compositional semantic analysis typically manifests in the three ways:

① Key semantic elements are often widely distributed across parse trees, such complicating the composition of the required meaning representation.

② Parse tree often contain many syntactically motivated constituents that play essentially no role in semantic processing.

③ The general nature of many syntactic constituents results in semantic attachment that creates nearly vacuous meaning representation.

E.g. “I want to go to eat some Japanese food today.”

Its parse tree is as follows:

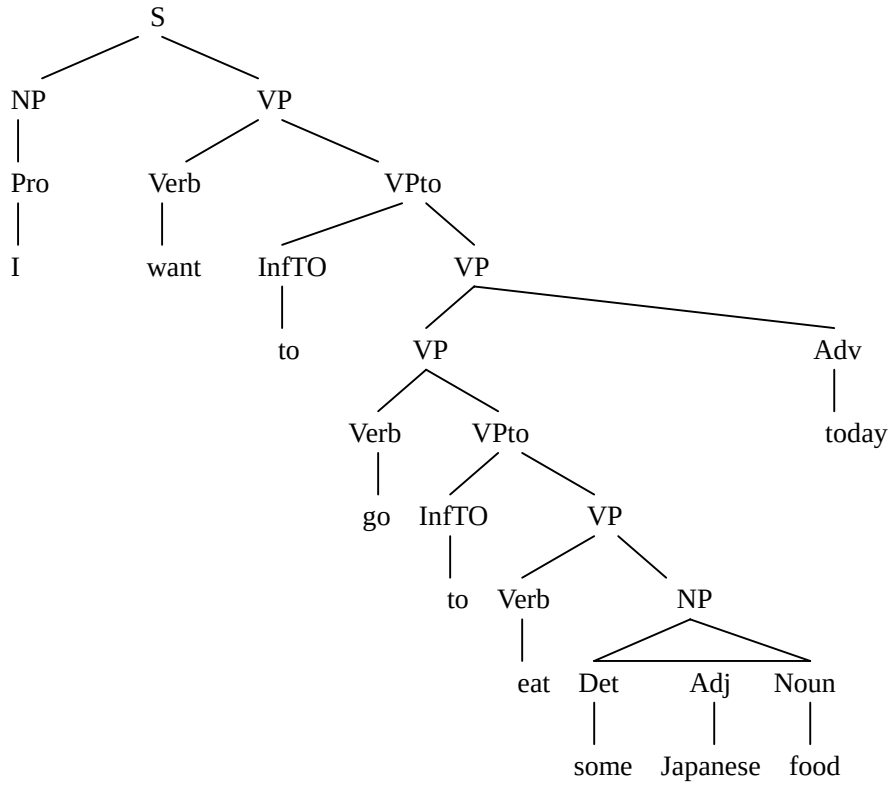


Fig. 4.

In this parse tree, the key components of the meaning representation widely are distributed throughout the tree. At the same time, most of the nodes in the tree contribute almost nothing to the meaning of this sentence. This structure requires three λ –expressions and a complex-term to bring the contentful elements together at the top of the tree. In this tree, the semantic attachment of the compound nominals and adjective phrases are so general as to nearly meaningless.

Nominal → Adj Nominal

{ λ x Nominal.sem (x) ∧ AM (x, Adj.sem)}

AM means “Adjective Modifier”. Applying this attachment results in the following meaning representation:

$$\exists x \, ISA(x, Food) \wedge AM(x, Japanese)$$

All nominal that fit this pattern receive the same vague interpretation that roughly indicates that the nominal is modified by the adjective. This is very far from what know that expressions like “Japanese food” and “Japanese restaurant” mean; they denote food prepared in a particular way,

and restaurants that serve food prepared that way. However, There is no way to get this very general rule to produce such an interpretation.

These problems can be overcome through Semantic Grammar (Brown, Burton, 1975).

- Semantic grammas is more directly oriented toward serving the needs of a compositional analysis.
- In Semantic grammar, the rules and constituents of the grammar are designed to correspond directly to entities and relations from the domain being discussed.
- In semantic grammar, the key semantic components occur together within single rules, and rules are made no more general than is needed to achieve sensible semantic analysis.

E.g. For analysis of sentence "I want to go to eat some Japanese food today", we can write the rule of semantic grammar like as follows:

InfoRequest → User want to go to eat FoodType TimeExpr

In the right hand side of this rule, the non-terminals and terminals are mixed freely. In this case, "User, FoodType, TimeExpr" represent semantically motivated non-terminal categories for this domain, There is no need for λ –expression, since this flat rule elevate all the relevant argument to the top of tree.

In the rule

FoodType → Nationality FoodType

It means that the FoodType is to prepare in the style associated with the Nationality constituent.

Semantic grammar can deal with the anaphor problem in NLP. Semantic grammar allow parsers to make highly specific predictions about upcoming input, based on the categories being actively predicated by this parser. Given this ability, anaphoric reference can be associated with specific semantic categories.

E.g. If we have two sentences as follows:

When does flight KE852 arrive in Seoul?

When does it arrive in Beijing?

These sentences can be analyzed with a rule of semantic grammar with non-terminals Flight and City:

InfoRequest → when does Flight arrive in City

The system can search back in the dialogue for places where the Flight constituent has been recently used to find candidate reference for the pronoun "it".

The disadvantages of semantic grammar are:

- Almost complete lack of reuse in the approach. The grammar is too specific to the particular domain.
- Unavoidable growth in the size of the grammar for a single domain.:

8.1.2.2 Information Extraction

In the application fields like as extracting information about joint ventures from business news, understanding weather reports, or summarizing simple information on the stock market from radio report, do not necessarily require the detailed understanding the full sentence. In these cases, we can take the technique of information extraction.

The properties of information extraction:

- The desired knowledge can be described by a relatively simple and fixed template, or frame, with slot that need to be filled in with material form text.

- Only a part of the information in the text is relevant for filling in this template or frame, the rest can be ignored.

For example, Following is the information extracted about the international joint ventures from business news>

TIE-UP-1

Relationship:	TIE-UP
Entities:	“Bridgestone Sports Co.”
	“a local concern”
	“a Japanese trade house”
Joint Venture Company	“Bridgestone Sports Taiwan Co.”
Activity	ACTIVITY-1
Amount	NT$20000000

ACTIVITY-!:

Company	“Bridgestone Sports Taiwan Co.”
Product	“iron and “metal wood” clubs”
Star Date	DURING: January 1990

The first two sentences of a sample article:

“Bridgestone Sports Co. said Friday it has set up a joint venture in Taiwan with a local concern and a Japanese house to produce golf clubs to be shipped to Japan..”

“The joint venture, Bridgestone Sports Taiwan Co., capitalized at 20 million new Taiwan dollars, will start production in January 1990 with production of 20,000 iron and “metal wood” clubs a month.”

Many Information extraction systems based on cascades of finite-state automate can extract pertinent information while ignoring irrelevant parts of the input.

8.2 Lexical Semantics

The lexicon has a highly systematic structure that governs what words can mean, and how they can be used. This structure consists of relations among words and their meanings, as well as the internal structure of individual words. The linguistic study of this systematic, meaning related, structure is called lexical semantics.

The lexicon should not be thought of as a finite listing, but rather as a creative generator of infinite meaning.

Due the difference on the definition of the term “word”, in this paragraph, we prefer to use the term “lexeme”.

8.2.1 Relations among lexemes and their senses

8.2.1.1 Homonymy

Homonymy is a relation that holds between lexemes that have the same form with inrelated meanings. The items taking part in such a relation are called homonyms.

E.g. bank: ① financial institution: “A bank can hold the investments in an account in the client’s name.” – bank1

② sloping mound: “As the agriculture development on the east bank, the river will shrink even more.” – bank2

There are some different types similar as homonym.

- Homophones: lexemes with same pronunciation but different spelling. E.g. wood – would; be – bee.
- Homographs: lexemes with identical orthographic forms and different pronunciations. E.g. bass [bæs]: fish that have prickly skins and can be eaten. bass [beis]: deep or low in sound.

Homonymy in NLP:

- In spelling correction:: homophones can lead to real-word spelling errors, as when lexemes such as “weather” and “whether” are interchanged.
- In speech recognition: homophones such as “to”, “two” and “too” cause obvious problems
- In Text-To-Speech (TTS) system: Homographs with distinct pronunciation will cause problems. E.g. bass[bæs] and bass [beis].

8.2.1.2 Polysemy

The phenomenon of multiple related meanings within a single lexeme is known as polysemy. In polysemy, the meanings within a single lexeme are related. But in homolymy, the meanings of the homonym are not related.

E.g. “head”: ① the part of the body that contains brain, eyes, ears, noise and mouth.
② the end where this part rest. E.g. “the head of the bed”
③ brain. E.g. “Can’t you get these facts into your head?”

There are two criteria that are typically invoked to determine whether or not the meanings of two lexemes are related or not,

- Etymology criteria: bank1 has an Italian origin, bank2 is of Scandinavian origin. Bank1 and bank2 are homonym.
- Zeugma criteria: to combine two distinct senses of the checked lexeme into a single sentence using a conjunction, then check whether the sentence is acceptable. If no, we can judge that the lexeme is homonym.

E.g. “Which of those flights serve breakfast?” (“serve” means “to offer food to eating”)
“Does ASIANA serve Philadelphia? (”serve” means “to work for”)
If we combine them to a new sentence using ‘and”::
:* ”Does ASIANA serve breakfast and Philadelphia?”

This sentence is ill-formed in some way. The oddness of this sentence indicates there is no sensible way to make a single sense of “serve” work for both breakfast and Philadelphia. The underlying concepts invoked by serve in the first sentence cannot be applied in any meaningful way to Philadelphia.

The Zeugma criteria are not always valid to differentiate homonymy and polysemy. Zeugma criteria can test the meaning distance between different polysemes, but it can not decide whether these different meanings are not related.

E.g “They play the soccer.” Here “play” means “to do sport that passed the time pleasantly”
“They play the basketball”.
“They play the piano”. Here “play” means “to perform a music instrument”.
“They play doctors and nurses. Here “play” means “Children amuse oneself by pretending to be some roles in the games”.

Using Zeugma approach, we have following new sentences:

"They play the soccer and the basketball." – it is acceptable.

*"They play the soccer and the piano." – it is odd!

**"They play the soccer and doctors" – it is very odd!!

In the zeugma approach, the more oddness of new sentence, the less acceptable of this new sentence.

Therefore zeugma is a good approach to decide the meaning distance of the words.

The difference between homonymy and polysemy is important in linguistics. However, in most NLP system do not distinguish homonymy from polysemy in any meaningful way. Both homonymy and polysemy are processed as the problem Word Sense Disambiguation (WSD).

8.2.1.3 Synonymy

The traditional definition of synonymy in linguistics is a deceptively simple definition: different lexemes with the same meaning.

We prefer following definition based on the substitutability: two lexemes will be considered synonyms if they can substituted for one another in a sentence without changing either the meaning or the acceptability of the sentence.

E.g. "How big is that plane?"

"Would I be flying on a large or small plane?"

Exchanging "big" and 'large" in two sentences has no noticeable effect on either the meaning or acceptability of these sentences.

However, if we take the notion of substitutable in all possible environments, then true synonyms in English are very few.

In this case, we will fall back on a weaker notion that allows us to call two lexemes synonyms if they are substitutable on some environment.

The substitutability depend on four influences:

- Polysemy

e.g. "Miss Kim became a kind of big sister to Mrs. Park's son.".

*"Miss Kim became a kind of large sister to Mrs. Park's son."

In first sentence, "big" has one of its polyseous senses the notion being "older". Since "large" lacks this sense among its many meanings. It is not substitutable for "big" in those environments where this sense is required.

- Subtle shade of meaning

E.g. "What is the cheapest first class fare?"

"What is the cheapest first class price?"

"fare" is better suited to the cost for various service (e.g. coach, business first class fares), while "price" is better applied to the tickets that present this services. By this reason, the exchanging "price" for 'fare" in the second sentence leads to a certain amount of oddity.

- Collocational constraints

E.g. "They make a big mistake."

"They make a large mistake."

There is a preference for using "big" rather that "large" when referring to mistakes of a critical or important nature.

- Register: the registers are the factors such as politeness, group status, and other similar social pressures.

8.2.1.4 Hyponymy

The hyponymy is the pairings where one lexeme denotes a subclass of the pther. The hyponymy relation is not symmetric. In the pairings, the more specific lexeme is hyponym, and the more general lexeme is hypernym.

E.g. the relationship between “car” and ”vehicle” is one hyponymy. “car” is a hyponym and the “vehicle” is a hypernym.

In the schema

That is a x. => That is a y.

If x is a hyponym of y, then in any situation where the sentence on the left is true, the newly created sentence on the right must be also true.

E.g. That is a car => That is a vehicle.

The set of hyponymy relations are useful in ontology, taxonomy, and object hierarchy.

- Ontology: A set of distinct objects resulting from an analysis of a domain, or microworld.
- Taxonomy: A particular arrangement of the elements of an ontology into a tree-like class inclusion structure.
- Object hierarchy: It is a notion in computer science. An object hierarchy is based the notion that objects from an ontology arranged in a taxonomy, can receive, or inherit, the features from their ancestors in a taxonomy.

The set of hyponymy relations are useful as approximation to such structures..

8.2.1.5 WordNet:

WordNet is a database of lexical relations for English (Fellbaum, 1998).

WordNet consists of Three databases: Noun, Verb, Adjective & Adverb.

In their most complete form, WordNet’s sense entries consist of a set of synonyms, a dictionary-style definition, or gloss, and some example uses

The scope of WordNet 1.6:

Category	Unique Forms	Number of Senses
Noun	94474	116317
Verb	10319	22066
Adjective	20170	29881
Adverb	4546	5677

Noun relations in WordNet:

- Hypernym: from concepts to superordinates. E.g. breakfast → meal
- Hyponym: from concepts to subtypes. E.g. meal → lunch
- Has-Member: from group to their members. E.g. faculty → professor
- Member-Of: from members to their groups. E.g. professor → faculty.
- Has-Part: from whole to part. E.g. table → leg
- Part-Of: from parts to whole. E.g. leg → table.
- Antonym: opposites. E.g. leader ⇔ follower

Verb relations in WordNet:

- Hypernym: from events to superordinate events. E.g. fly →→ travel
- Troponym: from events to their subtypes. E.g. walk → stroll.

- Entails: from events to the events they entail. E.g. snore → sleep.
- Antonym: Opposites. E.g. increase ⇔ decrease.

Adjective and Adverb relations in WordNet:

- Antonym: opposites. E.g. heavy ⇔ light (Adjective)
  quickly ⇔ slowly

Two WordNet entries are considered synonyms if they can be successfully substituted in some context. A set of synonyms can be organized to a synset.
Each synset is related to its immediately more general and more specific synsets via direct hypernym and hyponym relations. To find chains of more general or more specific synsets, we can simply follow a transitive chain of hypernym and hyponym relations
Following are the hyponymy chains for two separate senses of the lexeme "bass". Note that the chains are completely distinct only converging at "entity".

Sense 3
bass, basso –
(an adult male singer with the lowest voice)
→ musician, instrumentalist, player
→ performer, performing artist
→ entertainer
→ person, individual, someone …
→ life form, organism, being …
→ entity, something
→ causal agent, cause, causal agency
→ entity, something

Sense 7
bass –
(the member with the lower range of a family of musical instruments) e.g. bass drum
→ musical instrument
→ instrument
→ device
→ instrumentality, instrumentation
→ artifact, artefact
→ object, physical object
→ entity, something

In this depiction of hyponymy, successively more general synsets are shown on successive indented lines. The first chain starts from the concept of a human bass singer. Its immediately super-ordinate is a synset corresponding to the general concept of a singer. Following this chain leads to concepts such as entertainer and person. The second chain, which starts from musical instrument, has a completely different chain leading to such concepts as musical instrument, device and physical object. Both paths do eventually join at the synset "entity" which basically serves as a placeholder at the top of the hierarchy..

8.2.2 The internal Structure of Words

8.2.2.1 Thematic Roles

Thematic roles are a set categories which provide a shallow semantic language for characterizing certain arguments of verbs.

E.g. John broke a bat

John opened a door.

The FOPC representations are as follows:

$\exists$ e, x, y ISA (e, Breaking) $\wedge$ Breaker (e, John) $\wedge$ BrokenThing (e, y) $\wedge$ ISA (y, BaseballBat)

$\exists$ e, x, y ISA (e, Opening) $\wedge$ Openner (e, John) $\wedge$ OpenedThing (e, y) $\wedge$ ISA (y, Door)

The “Breaker” and ‘Opener” are both volitional actors, often animate, and they have direct causal responsibility for their events. We can use a thematic role to express this commonality. We say that the subjects of both these verbs are agents. AGENT is the semantic role which represents an abstract idea such as volitional causation. .The direct objects both these verbs, the BreakenThing and OpenedThing, are both prototypically inanimate objects which are affected in some way by the action. The thematic role for these participants is THEME.

In the sentence “John broke his collarbone”, “John” is EXPERIENCER..

In the sentence “The quake broke glass in several downtown skyscrapers”, “quake” is FORCE.

In the sentence “It broke his jaw”, “It” is the breaking event as the INSTRUMENT of some other AGENT and FORCE.

One common use of thematic roles in computational systems is as a shallow semantic language. Followings are some commonly-used thematic roles:

- AGENT: The volitional causer of an event. E.g. “The waiter spilled the soup.”
- EXPERIENCER: The experiencer of an event. E.g. “John has a headache.
- FORCE: The non-volitional causer of the event. E.g. “The quake broke the glass.”
- THEME: The participant most directly affected by an event. E.g. ”He broke the ice.”
- RESULT: The end of product of an event. E.g. The Korean government has built the World-Cup Stadium.
- CONTENT: The proposition or content of a propositional event. E.g. John asked: “What is your name?”
- INSTRUMENT: An instrument used in an event. E.g. John writes with a pencil.
- BENEFICIARY: The beneficiary of an event. E.g. John reserved a room for his boss.
- SOURCE: The origin of the object of a transfer event. John flew in from Beijing.
- GOAL: The destination of an object of a transfer event. E.g. John drove to Seoul.

Fillmore noted that there are prototypical patterns governing which argument of a verb will become the subject of an active sentence. He proposed thematic hierarchy for assigning the subject role:

AGENT => INSTRUMENT => THEME

- If the thematic description of a verb includes an AGENT, an INSTRUMENT, and a THEME,

it is the AGENT which will be realized as the subject.

- If the thematic description only includes an INSTRUMENT and a THEME, it is the INSTRUMENT which will become the subject.
- The THEME is the subject of passive sentence.

E.g. John opened the door.
AGENT THEME
John opened the door with the key.
AGENT THEME INSTRUMENT
The key opened the door.
INSTRUMENT THEME
The door was opened by John.
THEME AGENT

8.2.2.2 FrameNet

The Berkeley FrameNet (Fillmore, 2001, Baker, 1998, Lowe, 1997) is a project to study the relation between conceptual structure and grammatical function in English.

This project is creating an online lexical resource for English, based on **frame semantics** and supported by corpus evidence. The `starter lexicon' is available to the public by May, 2000, and will contain at least 2000 items - verbs, nouns, and adjectives - representative of a wide range of semantic domains. The aim is to document the range of semantic and syntactic combinatory possibilities (valences) of each word in each of its senses, through manual annotation of example sentences and automatic capture and organization of the annotation results. The FrameNet database is in a platform-independent format, and can be displayed and queried via the web and other interfaces.

A **semantic frame**, henceforth **frame** is a script-like structure of inferences, linked by linguistic convention to the meanings of linguistic units - in our case, lexical items. Each frame identifies a set of **frame elements (FEs)** - participants and props in the frame. A frame semantic description of a lexical item identifies the frames which underlie a given meaning and specifies the ways in which FEs, and constellations of FEs, are realized in structures headed by the word.

Valence descriptions provide, for each word sense, information about the sets of combinations of FEs, **grammatical functions** and **phrase types** attested in the corpus.

The annotated sentences are the building blocks of the database. These are marked up in XML and form the basis of the lexical entries. This format supports searching by lemma, frame, frame element, and combinations of these.

The FrameNet database acts **both as a dictionary and a thesaurus**. The dictionary features include definitions (from the Concise Oxford Dictionary, 10th Edition, courtesy of Oxford University Press), tables showing how frame elements are syntactically expressed in sentences containing each word, annotated examples from the corpus, and an alphabetical index. Like a thesaurus, words are linked to the semantic frames in which they participate, and frames, in turn, are linked to wordlists and to related frames.

The FrameNet corpus is the 100-million-word British National Corpus (BNC), used through the courtesy of Oxford University Press (OUP). The semantic annotation is carried out using the Alembic Workbench (MITRE Corporation). The syntactic annotation, which adds grammatical function and phrase type to each annotated phrase, is handled by an in-house tagging program. Each FrameNet entry will provide links to other lexical resources, including WordNet synsets and the COMLEX subcategorization frames.

A FrameNet entry for a word lists every set of arguments it can take, including the possible sets of semantic roles, syntactic phrases, and their grammatical function.

Each FrameNet thematic role is defined as part of a frame, and each frame as part of a domain.

E.g. Domain COGNITION has:

- Frames STATIC COGNITION (believe, think, understand);
- Frame COGITATION (brood, ruminate)
- Frame JUDGMENT (accuse, admire, rebuke)

All of COGNITION frames define the thematic role COGNIZER. In the JUDGMENT frame, the COGNIZER is referred to as the JUDGE. The frame also includes an EVALUEE, a REASON, and a ROLE.

E.g. JUDGE: John respects Kim for being so brave.
EVALUEE: John respects Kim for being so brave.
REASON: John respects Kim for being so brave
ROLE: John respects Kim as a scholar

In FrameNet, each entry is also labeled with phrase types.

E.g. for the judgment verb “appreciate” (including active sense and static cognition):

-- In active sense: (in the sense “to be thankful or grateful for”)

a. JUDGE REASON EVALUEE
NP/Subj NP/Obj PP(in)/Comp
I still appreciate good manners in men.

b. JUDGE EVALUEE REASON
NP/Subj NP/Obj PP(for)/Comp
I could appreciate it for the music alone.

c. JUDGE REASON
NP/Subj NP/Obj
I appreciate your kindness.

d. JUDGE EVALUEE ROLE
NP/Subj NP/Obj PP(for)/Comp

I did not appreciate the artist as a dissending voice.

-- static cognition (in the sense “understand”):

a. COGNIZER CONTENT

NP/Subj Sfin/Comp

They appreciate that communication is a two-way process.

b. COGNIZER CONTENT

NP/Subj Swh/Comp

She appreciated how far she had fallen from grace.

From these examples, some generalization can be drawn about the realization of different thematic roles: JUDGE, COGNIZER and AGENT in general are realized as subjects of active sentence; and ROLES are often realized as PPs with the preposition “as”; CONTENT is often realized as some kind of S.

How you can discovery of Frames

First, collect lists of words with similar meanings, where you think the similarity is because they are all built on the same semantic frame.

Speaking	**Judging**	**Classifying**
speak, say, tell, talk, inform, discuss, complain, report, assert, affirm	admire, appreciate, belittle, scorn, blame, commend, denigrate, deplore, disapprove, condemn, respect, evaluate, judge	categorize, classify, define, interpret, depict, describe, regard, construe

Definition of Communication/Statement Frame

Characterize the frames and identify (and name) the actors and props in situations understood in terms of the frame: Communication/Statement (i.e., monologic, informing)

Frame Description:

A person (Speaker) produces some linguistic object (Message) while addressing some other person (Addressee) on some topic (Topic).

Frame Elements:

- Speaker: [The salesman] **told** me it was guaranteed for life.
- Addressee: Are you **speaking** [to me]?
- Topic: She **said** something quite interesting [about you].
- Message: I **informed** them [that I was planning to quit].
- Medium: He **said** it [in last night's broadcast].

Refinements on "Message":

- Message-Content (I **said** [that I was planning to quit].)
- Message-Form (She **said** ["I can't stand this any longer!"])
- Message-Category (They **told** me [your age].)
- Message-Type (We're going to **say** [a few words].)

Definition of Cognition/Categorization Frame

Frame Description:

A person (the Cognizer) categorizes something (the Item). The Category into which the Item is placed may be expressed, as may the Criterion used as the basis for categorization.

Frame Elements:

- Cognizer: [The botanist] **categorized** the specimens.
- Item: The botanist **categorized** [the specimens].
- Category: The city **classified** the building [as a historic monument].
- Criterion: The city **classified** the buildings [according to their age].

Definition of Cognition/Judgment Frame

Frame Description:

A person (the Cognizer) makes a judgment about something or someone (the Evaluee). The judgment may be positive or negative. The target word may entail that the judgment is expressed verbally (e.g. *scold*) or it may not (e.g. *blame*). There may be a Reason for the judgment or a Role in which the Evaluee is evaluated.

Frame Elements:

- Cognizer: [John] **respects** Kim for being so brave.
- Evaluee: John **respects** [Kim] for being so brave.
- Reason: John **respects** Kim [for being so brave].
- Role: John **respects** Kim [as a scholar].

Preliminary Exploration of Corpus Examples

Collect examples of each word on the list.

Recognize the possibility of polysemy: choose those examples in which the word has the sense being examined.

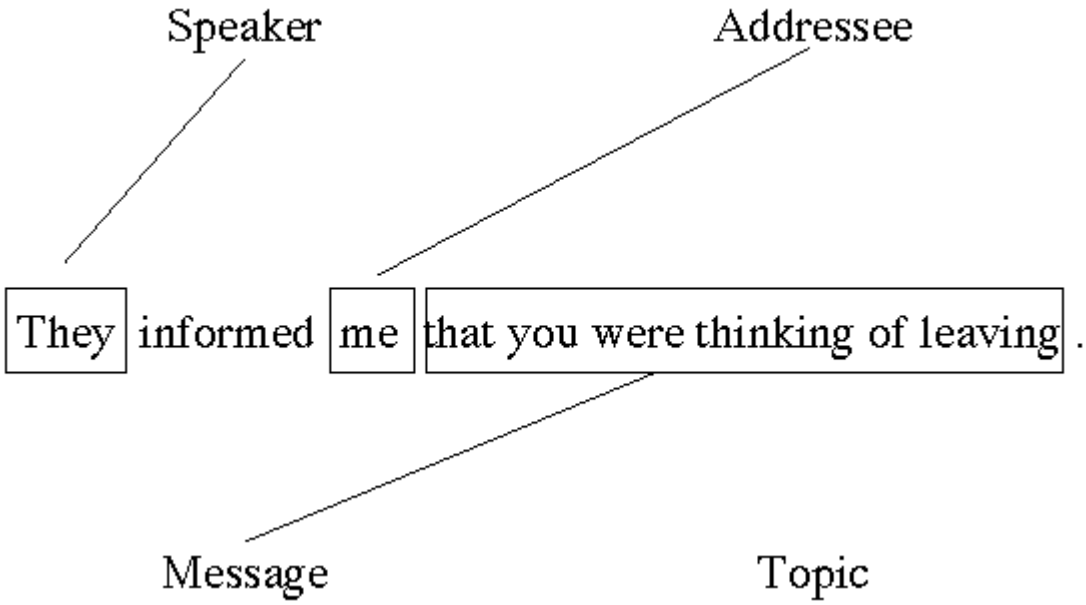
Speak, *talk* and *discuss* have both monologic and dialogic uses; for this frame we want only the monologic ones.

- The teacher **discussed** the next homework assignment. (monologic)

- My neighbors and I **discussed** your membership application. (dialologic)

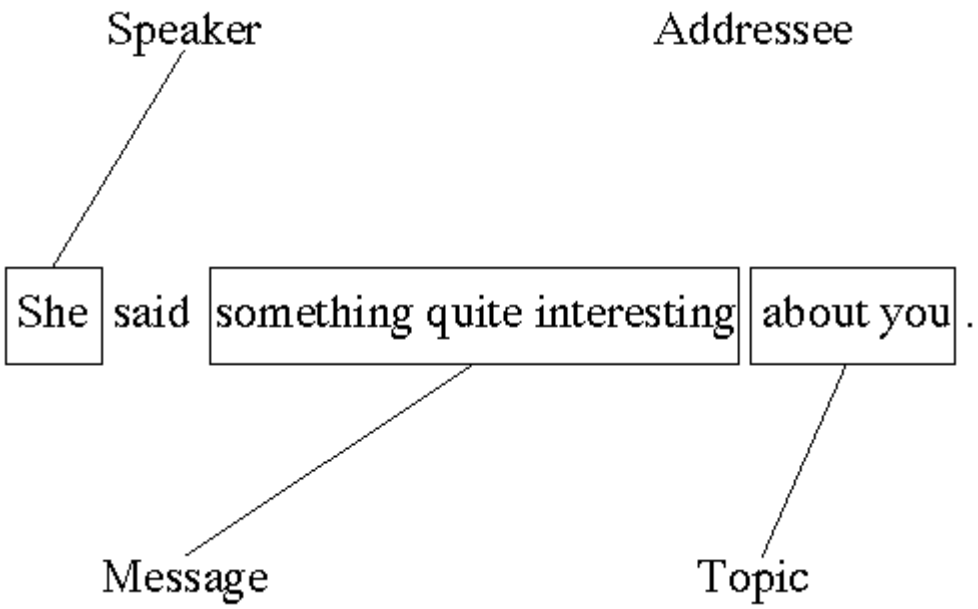
Hand Marking: *Inform*

Identify constituents in the example sentences and label them by the frame elements they realize.



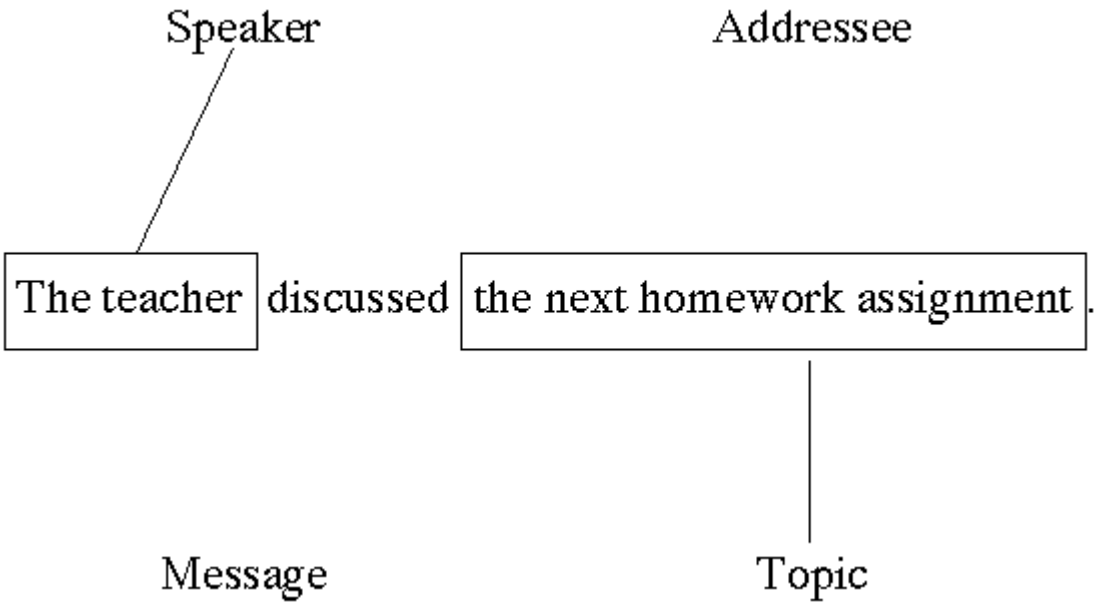
Hand annotation of: They **informed** me that you were thinking of leaving.

Hand Marking: *Say*



Hand annotation of: She **said** something quite interesting about you.

Hand Marking: *Discuss*



Hand annotation of: The teacher **discussed** the next homework assignment.

Making Generalizations

Sample generalizations for Communication/Statement verbs

- Of the basic verbs, only SAY freely **precedes** quotations.
  *She informed me "It's getting late".
  - elaborations of SAY: whisper, shout,...
  - hundreds of verbs can **follow** a quotation: she sighed, he bellowed, they admired, ...
- Some of the basic verbs do not have a place for the Message
  - *She spoke that she didn't want to go.
  - *The professor discussed that it was a hard problem.
- TELL and INFORM take Addressee as direct object.
- DISCUSS takes Topic as direct object.

Facing reality:

Actually it's hugely more complicated than what we've seen so far; we'll discuss null instantiation, frame inheritance, blending, etc. We shall face the reality.

Resources used in FrameNet

The Corpus

The British National Corpus (BNC) is a large sample of modern (British) English taken from a number of genres. It was created in the UK by a consortium of publishing houses, universities, and government agencies, was completed in 1994, and was made available to European researchers in 1995.

The Corpus comprises 90% written language and 10% transcribed speech, totalling over 100,000,000 running words.

The Corpus has been processed in certain ways by the Consortium. It has been **tokenized**, which means that the boundaries of sentences are indicated, contractions are separated, individual word tokens are assigned numbered locations, and punctuation marks are indexed. Also, it has been **pos-tagged** (each word is tagged for part of speech) in a refined system of 65 word classes.

The version of the Corpus which we use has further been **lemmatized**, which means that inflectional (and dialect) variants are identified as instances of the same lemma. The lemmatizing was done at the University of Stuttgart. The lemmatized version was made available to us through the courtesy of the Institut fuer Maschinelle Sprachverarbeitung at the University of Stuttgart.

FrameNet has the use of this corpus by agreement with Oxford University Press, leader of the BNC Consortium.

If you have interest for FrameNet, please to access its Web site:
http;// www.icsi.berkeley.edu/~framenet

8.2.2.3 Selectional Restriction

A selectional restriction is a semantic constraint imposed by a lexeme on the concepts that can fill the various argument roles associated with it.

The selectional restrictions imposed by differenty lexemes, and different senses of the same lexeme, may occur at widely varying levels of specofity, with some lexemes expressing very general conceptual categories, and other expressing very specific ones.

Consider the verbs “imagine, lift, diagonalize”:

e.g. I cannot imagine what this lady does all day.

In rehearsal I often ask the musicians to imagine a tennis game.

He lifted the fish from the water.

To diagonalize a matrix is to find its eigenvalues.

For “imagine”. Its AGENT role is restricted to humans and other animate entities, and only a few semantic restrictions that can fill its THEME role. For “lift”, its Theme must be something liftable. For “diagomalize”, it has a very specific constraint on the filter of its THEME role: it has to be a matrix.

Approaches for representing Selectional restrictions:

- FOPC

If we have FOPC representation:

$$\exists \text{ e, x, y Eating (e)} \wedge \text{Agent (e,x)} \wedge \text{Theme (e,y)}$$

To stipulate the selectional restriction that y must be something edible, we simply add a new term to this representation:

$$\exists \text{ e, x, y Eating (e)} \wedge \text{Agent (e,x)} \wedge \text{Theme (e,y)} \wedge \text{ISA (y, EdibleThing)}$$

When a phrase like „ate a hamburger“ is encountered, we can add new selectional restriction in the FOPC representation:

$$\exists \text{ e, x, y Eating (e)} \wedge \text{Agent (e,x)} \wedge \text{Theme (e,y)} \wedge \text{ISA (y, EdibleThing)} \wedge \text{ISA (y, Hamburger)}$$

This representation is reasonable since the membership of y in the category “Hamburger” is consistent with its membership in the category “EdibleThing”, assuming a reasonable set of facts in the knowledge base.

However, using FOPC to perform the simple task of enforcing selectional restriction is overkill. We may use more practical approach.

- WordNet Synset: In example “ate a humburger“, Among 60,000 synset of WordNet, we can find the gloss of synset {food, nutrient} as “any substance that can be metabolized by an organism to give energy and build tissue”
- Given this synset, we cvan specify it as the selectional restriction on the THEME role of the verb “eat”, thus limiting filters of this role to lexemes in this synset and its hyponyms. The chain of hypernyms for “hambueger” reveals that the hamburgers are indeed food.

  Sense 1
  Hambueger, beefburger –
  (a fried cake of minced beef served on a bun)
  ➔ sandwich
  ➔ snack food
  ➔ dish
  ➔ nutriment, nourishment, sustenance …
  ➔ food, nutrient
  ➔ substance, matter
  ➔ object, physical object
  ➔ entity, something

This is a evidence form WordNet that hamburgers are edible.

This approach also allow individual lexemes to satisfy restrictions of varying levels of specificity. E.g. Consider what happens when we apply this approach to the THEME roles of “imagine, lift, diagnolize”. We restrict imagine’s THEME to the synset {entity, something}, lift’s THEME to {object, physical object}, and diagonilize to {matrix}. This arrangement correctly permits “imagine a hamburger” and “lift a hamburger”, while also correctly ruling out “diagonalize hamburger”.

8.2.2.4 Primitive Decomposition

Through the componential analysis of meaning, we can do primitive decomposition.

Based on CD theory, the sentence “The waiter brought Mary a check” can be described as follows:

$$\exists \text{ x, y Atrans (x)} \wedge \text{Actor (x, Waiter)} \wedge \text{Object (x, Check)} \wedge \text{To (x, Mary)}$$
$$\wedge \text{ Ptrans (x)} \wedge \text{Actor (x, Waiter)} \wedge \text{Object (x, Check)} \wedge \text{To (x, Mary)}$$

The verb “brought” is translated into the two primitive ATRANS and PTRANS to indicate the fact that the waiter both physically conveyed the check to Mary and passed control of it to Mary.
The lexemes “kitten, puppy, child” can be decomposed into more primitive notions:

$\exists$ x ISA (x, Feline) ISA (x, Youth)

$\exists$ x ISA (x, Canine) ISA (x, Youth)

$\exists$ x ISA (x, Human) ISA (x, Youth)

The lexemes “cat, dog, person” can be decomposed into more primitive notions:

$\exists$ x ISA (x, Feline) ISA (x, Adult)

$\exists$ x ISA (x, Canine) ISA (x, Adult)

$\exists$ x ISA (x, Human) ISA (x, Adult)

In machine translation, the primitives can be used as language independent meaning representations (interlinguas).

8.3 Meaning-text Theory (MTT)

The Meaning-Text theory (MTT) was put forward in Moscow by Zolkovski and Melcuk (1965, 1987) in “On semantic synthesis (of text)” (Problems of Cybernetics, 19, 177-238).
MTT considers that a natural language is a correspondence between meanings and texts. Although this conception of language is a more or less accepted by everybody, it appears that most contemporary linguistic theories do not model natural language in the same way as MTT.

8.3.1 What is a natural language?

What is a natural language? The answer of MTT is based on the three following postulates:

- Postulate 1: Natural language is considered as) a many-to-many correspondence between meanings and texts.
- Postulate 2: The meaning-text correspondence is described by a formal device which simulates the linguistic activity of a native speaker.
- Postulate 3: Given the complexity of the meaning-text correspondence, intermediate levels of (utterance) representation have to be distinguished; more precisely, a syntactic and a morphological level.

The first MTT postulate means that the description of a natural language $\mathbb{L}$ consists of the description of the correspondence between the set of meanings of $\mathbb{L}$ and the set of texts of $\mathbb{L}$.

It is similar as Saussurian theory, that is, a linguistic sign with a signifie (meaning) and a significant (text).
The second postulate stresses on the fact that a natural language must be described as a correspondence. A speaker speaks. A meaning-text model must model the speaker activity, that is ,

model how a speaker transforms what he wants to say (a meaning) into what he says (a text). A natural language must be described as a Meaning-Text correspondence and moreover that the direction from meaning to text must be privileged.

The third postulate asks for several comments. Most linguistic theories consider a morphological and a syntactic level of representations. What is important here is that these levels are intermediate between the semantic and phonological level (= meaning and text). This means that the correspondence form meaning to text will be completely modular: a correspondence between the semantic and the syntactic level, a correspondence between the syntactic and the morphologic level and a correspondence between the morphological and the phonological level.

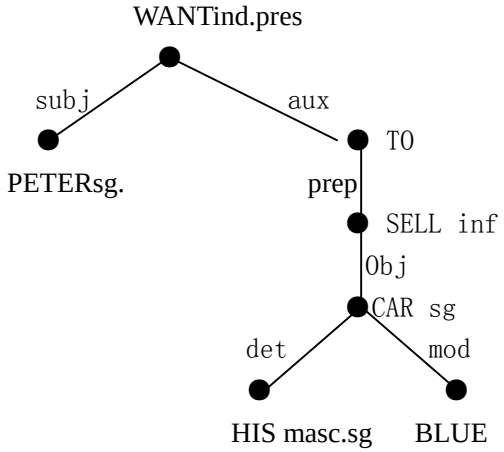
Representations of different levels have different structural organizations: semantic representations are graphs (of predicate-argument relations), syntactic representations are (non ordered) dependency trees and morphological representations are strings. MTT carefully pays attention to the geometry of the representation: a morphological representation is one-dimensional (a string), a syntactic representation is two-dimensional a tree) and a semantic representation is multi-dimensional (a graph).

E.g. “Peter wants to sell his blue car” can be represented as following:

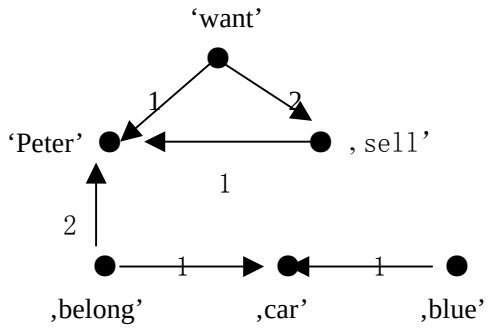
Morphological representation:

PETERsg WANTind.pres.sg TO SELLinf HISmasc.sg BLUE CARsg

Syntactic representation:



Semantic representation:



There is a fundamental distinction between dependency structure and phrase structural: a phrase structure contains the linear order of the words of the sentence, but dependency structure does not contain the linear order of the words in sentence. In other words, a phrase structure tree does not separate the syntactic structure from the morphological structure. Contemporary theories, such as

HPSG even mix the semantic representation with the phrase structure representation and use a single formalism-feature structure to represent all these objects. It seems that MTT postulates, even if they are given different formulations, are more or less accepted by the whole linguistics community.

Brody said:

“It is a truism that grammar relates sound and meaning. Theories that account for this relationship with reasonable success postulate representational levels corresponding to sound and meaning and assume that the relationship is mediated through complex representations that are composed of small units. ” (Brody Michael, Lexico-Logical Form: A Radically Minimalist Theory, Cambridge: MIT Press)

A natural language must be described as a correspondence. This pint will be emphasized now.

8.3.2 How to describe a language?

Melcuk said:”Un MST (Modele Sens-Texte) doit faire la meme chose : traduire un sens donne en un texte qui l’exprime (voila pourquoi ce modele est ‘traductif’

Sylvain Kahane (Universite Paris 7) proposes a very general formal definition of what a grammar is in the spirit of MTT. Such grammar will be called a transductive grammar （转构语法） by analogy with transducer. (The tranducer theory is limited to the correspondence between strings)

Let S and S’ be two sets of structures (graphs, trees, orders…). A transductive grammar G between S and S’ is a formal grammar which associates elements of S with elements of S’. As a formal grammar, G contains a finite set of rules, which are called the correspondence rules, A correspondence rule associates a piece of structure from elements of S with a piece of structure from elements of S’. Consequently, a transductive grammar G defines more than a correspondence between the sets of structures S and S’. Indeed, for each couple (S, S’) that are associated by G, G also defines partitions of the structures S and S’ and a one-to-one mapping $\Phi_{(S, S')}$ between the pieces of these two partitions.