1st paper: used to be 5-7, now 4-6
2nd paper: used to be 8-12, now 7-10

What Is Semantics?

Lexical vs. compositional semantics

Compositional semantics tries to explain three things:

1. How the meaning of a complex expression depends on the meanings of its parts and their syntactic arrangement.

2. Why one sentence entails another sentence.

3. Why certain sentences are synonymous.

4. Why certain sentences are ambiguous.

Hardegree, "Basic Categorial Semantics"

**2. The Basic Picture of Semantic Processing**

Step 1: Syntactic decomposition

Syntax is the study of

We want to assign every expression a syntactic type (category), in a way that explains why some expressions are syntactically acceptable ("well-formed") and some are not ("ill-formed).

Syntactic types

Definition of primitive syntactic type:
N is a primitive syntactic type
S is a primitive syntactic type

Definition of syntactic type:
If A is a primitive syntactic type, then A is a syntactic type
If A and B are syntactic types, then A → B is a syntactic type
If A and B are syntactic types, then A X B is a syntactic type

Nothing else is a syntactic type

Interpretation of →

(Note: Hardegree uses D where I use N. It's a long story.)

N is for 'name', in particular what are called 'proper names' like my name (Michael), my wife's name (Jenny) or your name.

S is for 'sentence'.

So everything of type N or S is well-formed (grammatically acceptable). You never find names that are ungrammatical, like *Tim or *Betty. And although we popularly think there are sentences that are ungrammatical, if we set up our grammar right, nothing with type S will be.

(Think of this as our criterion of adequacy: a grammar is correct iff all the S's are grammatical sentences.)

Types of the form A → B are 'functor' types. A functor is a symbol that stands for a function. So '+' is a functor, and the function it stands for is addition.

What 'A → B' means is: if some expression E has this type, and you combine E with an expression E* that has type A, you get a new expression E(E*) that has type B.

According to our definition, N → S is a syntactic type. What sorts of expressions have this type? Well, intransitive verbs: 'smokes' 'smiles' 'runs' etc. If you combine 'smokes' with an N like 'Tim' you get an S: 'Tim smokes'

But if N is a syntactic type, and N → S is a syntactic type, so too is N → (N → S). And these will be the transitive verbs, like 'admires' or 'pushes'. If you combine 'pushes' with 'Tim', you get 'pushes Tim' (type N → S), which when you combine with another N like 'Jim', you get 'Jim pushes Tim', an S.

There will also be 'recursive' types, ones that have as their output something in their input. Consider the verb 'believes'. 'John believes the sky is falling' seems to be composed of 'John', an N, 'believes', and 'the sky is falling' an S. So 'believes should be of the form S → (N → S).

Interpretation of X

‘A X B’ is the type of pairs of syntactic objects. Suppose you have a word like ‘and’ that doesn’t just have one input S, but two: It’s raining and I’m all wet. ‘and’ is going to have type S X S → S: it takes a pair of sentences and returns a new sentence.

You can handle words that have more than two inputs by iterating the pair operator. You might think of ‘give’ this way: (N X N) X N → S, e.g. ‘Tim gave Bill chickenpox’.

‘X’ is not really needed, which is an insight due to Schönfinkel (though erroneously attributed to Curry). You can treat ‘give’ either as a functor with 3 inputs: (N X N) X N, or one that takes one N input and returns a functor that takes one N input and returns a functor that takes one input and returns an S: N → (N → (N → S)).

The Lexicon

The lexicon is a pairing of words with their syntactic types, and other features (semantic features, but we’re ignoring those right now). Example:

Tim, N
Fred, N
Bill, N
Swims, N → S
Smokes, N → S
Snores, N → S
Likes, N → (N → S)
Loves, N → (N → S)
Adores, N → (N → S)
Believes, S → (N → S)
Doubts, S → (N → S)
Knows, S → (N → S)

From the point of view of our theory now, determining whether a particular sentence is well-formed (grammatical) or not is a matter of consulting the lexicon, and seeing whether a particular combination of expressions result in an S.

[Talk about the handedness problem.]

[Show some derivations and how they can be diagrammed as trees]

Step 2: Lexical assignment of meaning

Isomorphism Thesis: Semantics is isomorphic (identical in form) to syntax

Definition of primitive semantic types:

E is a primitive semantic type
T is a primitive semantic type

Definition of semantic types:

If A is a primitive semantic type, then A is a semantic type
If A and B are semantics types, then A → B is a semantic type
If A and B are semantic types, then A X B is a semantic type
Nothing else is a semantic type

Interpretation of the primitive types

E is the type of entities (sometimes called 'individuals'). These are the sorts of things that are named by syntactic objects of type N. In fact, in our system every type N syntactic expression means a type E semantic object.

T is the type of truth-values (true and false, sometimes designated 1 and 0). The meaning of a true S is true and the meaning of a false S is false. (So every S has a meaning of type T.)

[Talk about truth-conditions and being true or false at worlds.]

Interpretation of →

Syntactic expressions of types like N → S we called 'functors' and I said that functors are expressions that have as their meaning functions.

So quite naturally, semantic objects of type E → T are functions, in particular functions from entities to truth values.

Remember our N → S examples: 'swims' 'smokes' and 'smells'. What is the function that 'swims' means? This one: the function that takes as input people and returns 'true' if they swim and 'false' if they don't swim.

Interpretation of X

[Describe Cartesian products]

The isomorphism thesis (again): any expression with syntactic type A has semantic type B, where A and B differ only in that all the occurrences of N is A are replaced by occurrences of E in B and all the occurrences of S in A are replaced with occurrences of T in B.

Step 3: Semantic recomposition

Isomorphism Thesis: Semantics is isomorphic (identical in form) to syntax