CS 533
Natural Language Processing
Lecture 3 – February 10, 2003

Matthew Stone

Department of Computer Science
Center for Cognitive Science
Rutgers University

Language and the World Outline

Introducing knowledge representation

Language and context

Computing semantics
- semantic knowledge
- semantic representations
- implementing semantics

Language and ontology

Logic and computational semantics

Introducing knowledge representation

Language is about the world
- Dialogue systems need to ***understand*** what the user says in terms of some ***domain*** – the objects, actions, and tasks the system knows about.
- Dialogue systems need to be able to ***report*** information they have – conclusions about domain objects, actions and tasks – back to the user in natural language.

We'll see that *knowledge of the world* is required to talk about it.

Introducing knowledge representation

Knowledge representation (KR) is the general practice of specifying information about the world for computer systems.

Connecting language and the world requires knowledge representation.
- Domain elements come from prior KR that enables system's reasoning.
- Domain elements have to be described in semantic terms using KR methods.

Language and context

Key point:
- Utterances use general information (***semantics***) to construct ***interpretations*** that fit a specific ongoing context.

Language and context

Consider:

[At a restaurant]

Q: Would you like a drink with dessert?

A: I'd like coffee.

Language and context

Consider:

[At an ice-cream shop]

Q: Which of these flavors do you want?

A: I'd like coffee.

Language and context

Consider:

[At your host's English country estate]

Q: What will you have to drink tomorrow morning?

A: I'd like coffee.

Language and context

Consider:

[At hacker showdown]

Q: Will you program for Team Coke or Team Coffee?

A: I'd like coffee.

Coffee

A beverage

A kind of ice cream

A team of programmers

I'd like

Give it to me now

Arrange for me to get it tomorrow

Assign me to it

Definitions

Interpretation:

what somebody means by an utterance in context

Go get me a hot cup of coffee now.

Meaning:

what rules of language say about interpretation

The speaker has a preference for a kind of event that generally results in some kind of possession for him of something associated with coffee.

In dialogue, you want the interpretation

For example, this tells you how to respond:

> When someone tells you they want you to do something, do it.

However, you start with the meaning

You need to draw ***inferences*** from the meaning, together with your ***knowledge of the situation***, to ***derive*** the interpretation.

Aside:

> Your inference should be constrained and predictable, if you are to do it efficiently and your users are to understand it.

KR for NLP

Formalize the ***meanings*** of utterances.
Formalize ***context*** in same terms.
Draw inferences to derive ***interpretation***.

KR Methodology

Knowledge level analysis
> what information does the system need?

Representations and algorithms
> what formal methods realize this information?

Implementation
> what physical operations realize those methods?

KR Methodology

Attention to real data!
> You can't make stuff up: if it's ***false*** it won't work.

Respect for system functionality!
> It's not enough to be ***true***, it has to be ***useful***.

Standards and consistency!
> Inference works by ***matching***: *A, A implies B, so B.*
> ***Consistency*** or else: *A, AA implies B, so nothing.*

KR for NLP: The Knowledge Level

Characterize the information about language and the world the system must have to connect language to context as people do.

KR for NLP: Knowledge Level Interpretation

Documenting interpretation means saying what an utterance *describes*.

Speakers use utterances to portray the things we are interested in – things as we conceive of them, individuated abstractly through our ideas about real-world causation, function and intention.

Description: examples

The word *coffee* may describe:

coffee (a kind of beverage, made from certain roasted and ground seeds and known for its stimulant qualities).

coffee-ice-cream (a kind of frozen confection, flavored with the beverage coffee)

team-coffee (a group that represents its collective identity in the beverage coffee and its stimulant effects)

Description and ontology

The things *described* in an interpretation should already be part of the system's *ontology*, or model of its domain.

This means the system can reason and act on the basis of the interpretation, and can use its existing reasoning to help construct the interpretation.

Description and ontology

The things *described* in an interpretation should already be part of the system's *ontology*, or model of its domain.

Therefore you need to understand what the issues are in building such domain models, and how language fits in.

Ontology-building and language

Start by developing a list of the kinds of things that the system will have to reason about and talk about.

Sample utterances provide a supplement and check for this brainstorming.

Ontology-building and language

Then organize what you have to come up with a single consistent perspective on the domain.

What ***granularity*** of objects (e.g., book vs. copy, edition, text, content)?

What ***organization*** of concepts (e.g., ***window*** lego piece or ***window-of*** room fixture)

Utterances can show what perspectives users take.

Ontology-building and language

Then formalize the generalizations that do useful work for the application.

By developing standards for annotating utterance interpretation and by assessing generalizations in the same terms, you can help see whether further information is needed.

KR for NLP: Knowledge level Meaning

Write generalizations that characterize not just individual utterance interpretations, but the complete range of utterances that the system will handle.

Apply KR methodology:

Brainstorm semantic categories.

Organize semantic categories.

Formalize semantic generalizations.

Meaning and description

If we represent interpretations by what individuals words have described, we can represent ***meaning*** as ***constraints*** on what words ***can*** describe.

Meaning and description Examples

Coffee can describe any kind of thing connected with the beverage **coffee,** and always does so.

Meaning and description Examples

I can and does describe the speaker of the current utterance.

Meaning and description Examples

Would like can describe any kind of event that brings its grammatical subject into a certain kind of possession relationship with its grammatical object.

Would like expresses the subject's preference for an event of this kind, and so describes a particular **state.**

Meaning and description: KR tasks

Identifying and organizing abstract concepts

being-connected-with, speaker-of, bring-about, possession, preference

Stating general principles to characterize interpretations in these terms.

Constraints, meaning and interpretation

Meaning:

Coffee can describe any kind of thing connected with the beverage **coffee**, and always does so.

Context:

The beverage **coffee** is connected with the beverage **coffee** (in virtue of being it)

Interpretation:

Coffee describes **coffee**.

Constraints, meaning and interpretation

Meaning:

Coffee can describe any kind of thing connected with the beverage **coffee**, and always does so.

Context:

The confection **coffee-ice-cream** is connected with the beverage **coffee** (in virtue of its flavor)

Interpretation:

Coffee describes **coffee-ice-cream**.

Constraints, meaning and interpretation

Meaning:

Coffee can describe any kind of thing connected with the beverage **coffee**, and always does so.

Context:

The group **team-coffee** is connected with the beverage **coffee** (in virtue of its name)

Interpretation:

Coffee describes **team-coffee**.

Constraints, meaning and interpretation

Meaning:

Would like can describe any kind of event that brings its subject into a certain kind of possession relationship with its object.

Context:

The relation **physical-control** is a kind of possession and **bringing-a-mugful-of-coffee-to-the-user** is a kind of event that can generally result in the user's **physical-control** of the beverage **coffee**.

Interpretation:

Would like describes **bringing-a-mugful-of-coffee-to-the-user**.

Constraints, meaning and interpretation

Meaning:

Would like can describe any kind of event that brings its subject into a certain kind of possession relationship with its object.

Context:

The relation **physical-control** is a kind of possession and **dishing-a-scoop-of-coffee-ice-cream-to-the-user** is a kind of event that can generally result in the user's **physical-control** of the confection **coffee-ice-cream**.

Interpretation:

Would like describes **dishing-a-scoopful-of-coffee-ice-cream-to-the-user**.

Constraints, meaning and interpretation

Meaning:

Would like can describe any kind of event that brings its subject into a certain kind of possession relationship with its object.

Context:

The relation **team-affiliation** is a kind of possession and **assigning-the-user-to-team-coffee** is a kind of event that can generally result in the user's **team-affiliation** to the group **team-coffee**.

Interpretation:

Would like describes **assigning-the-user-to-team-coffee**.

Constraint-satisfaction models of interpretation

The interpretation is the ***best match*** for semantic constraints in the current context.

Efficient algorithms.

Theoretical motivation.

Clear requirements for representations.

KR for NLP: Representations and Algorithms

Representations

- Formal structures
- Used to define computational operations
- Correspond to information about the world

Algorithms

- Explicit, mechanical, abstract descriptions of operations an implementation will carry out.
- Function that produces meaningful results across a meaningful range of circumstances.

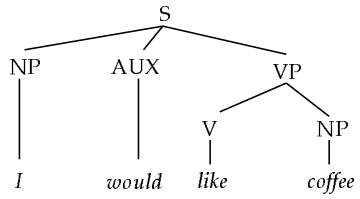
KR for NLP: Representing Interpretation

Step 1: Find representations of domain elts.

- Symbol `u` → **user**
- Symbol `k_c` → **coffee**
- Symbol `r_p_c` → **physical-control**
- Symbol `k_b_m` → **bringing-a-mugful-of-coffee-to-the-user**
- Symbol `s_p` → **preference-for-mugful**

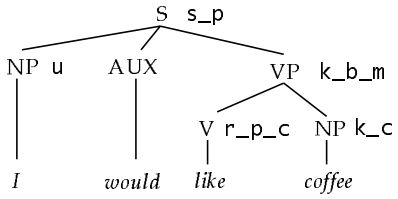
KR for NLP: Representing Interpretation

Step 2. Integrate domain elements to linguistic representations



KR for NLP: Representing Interpretation

Step 2. Integrate domain elements to linguistic representations



When to index…

When the interpretation of the utterance is different in different contexts because we take the utterance to describe a different salient domain element.

When to index…

When contexts following the utterance allow something implicit in the utterance to be evoked with a pronoun or other elliptical expression.

Where to index…

Anywhere that syntactic modifiers could attach and describe the same individual.

KR for NLP: Representing Meaning

Representing categories of interpretation and generalizations over interpretation

Ingredients:

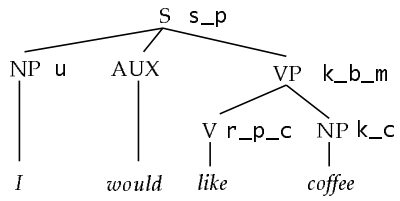
predicates: symbols for categories + maybe

variables: symbols that range over individuals

operators: symbols that describe information

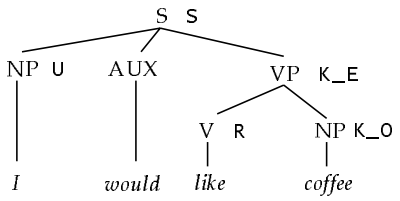
Abstracting over interpretation

Integrate domain elements to linguistic representations



Abstracting over interpretation

Abstract away using variables



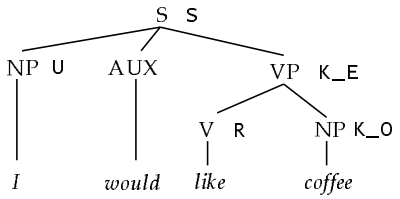
Aside: Instantiation

Replace a variable (or set of variables) with a corresponding specific representation everywhere the variables occur.

Sample instantiation:

[U ← u ; R ← r_p_c ; K_O ← k_c;
K_E ← k_b_m ; S ← s_p]

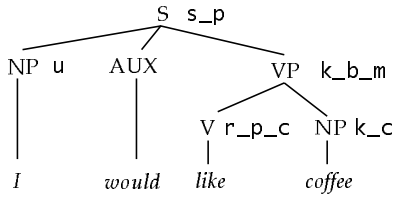
Apply instantiation



Sample instantiation:

[U ← u ; R ← r_p_c ; K_O ← k_c;
K_E ← k_b_m ; S ← s_p]

Apply instantiation, ctd.



Where this leaves us

Constructing the interpretation from the meaning is finding a suitable instantiation.

Use *predicates* to *constrain* the instantiation.

Sample semantic constraints

For this structure:

- `connected(K_O, k_c)`
  - `K_O` is connected with the beverage coffee
- `speaker(U)`
  - `U` is the speaker of the utterance
- `possession(R)`
  - `R` is a kind of possession
- `result(K_E, holds(R, U, K_O))`
  - restrict attention to effects of `K_E`, and check that one of these effects is to get `U` in `R` with `K_O`
- `preference(S, U, K_E)`
  - `S` represents `U`'s interest in seeing an event of type `K_E`

Describing domain elements in semantic terms

For this structure:

- `connected(k_c, k_c)`
  - `k_c` is connected with the beverage coffee
- `speaker(u)`
  - `u` is the speaker of the utterance
- `possession(r_p_c)`
  - `r_p_c` is a kind of possession
- `result(k_b_m, holds(r_p_c, u, k_c))`
  - restrict attention to effects of `k_b_m`, and check that one of these effects is to get `u` in `r_p_c` with `k_c`
- `preference(s_p, u, k_b_m)`
  - `s_p` represents `u`'s interest in seeing an event of type `k_b_m`

Computing instantiations by matching constraints

Match `connected(K_O, k_c)` with `connected(k_c, k_c)`

get `[K_O ← k_c]`

Match `speaker(U)` with `speaker(u)`

get `[K_O ← k_c; U ← u]`

Match `possession(R)` with `possession(r_p_c)`

get `[K_O ← k_c; U ← u; R ← r_p_c]`

Match `result(K_E,holds(R, U, K_O))` with `result(k_b_m,holds(r_p_c, u, k_c))`

get `[K_O ←k_c;U ← u;R ← r_p_c;K_E ← k_b_m]`

Computing instantiations by matching constraints

Match `preference(S, U, K_E)` with `preference(s_p, u, k_b_m)`

get final instantiation:

`[ K_O ← k_c ; U ← u ; R ← r_p_c ; K_E ← k_b_m ; S ← s_p ]`

Constraint-satisfaction: Formal Problem Statement

Given:

- A set of variables
- A set of constraints involving those variables
- A database of facts

Find:

- An instantiation for variables such that each constraint instance is listed in the database.

Constraint-satisfaction: An Algorithm

Primitive operations:

1. Match a constraint in the database

Set ← match(*Instantiation, Constraint, DB*)

Augment the variables in *Instantiation* so that *Constraint* has a matching instance in *DB,* and return the resulting instantiations in *Set.*

Constraint-satisfaction: An Algorithm

Primitive operations:

2. Search over the elements of a set

search *Set* for *Element* where *Expr* returns *Result*

search through *Set* to find some *Result*. *Element* provides a placeholder for each element of *Set* as the search considers it. *Expr* defines the operation considered in the search.

Constraint Satisfaction: An Algorithm

Data structure: List of constraints

[] – empty list

[Head|Tail] – list whose first element is Head and whose remainder is the list Tail

Constraint Satisfaction: An Algorithm

solve(*instantiation-so-far*, *remaining-constraints*)

Recursion – base case:

solve(*S*, [])
    return *S*

Constraint Satisfaction: An Algorithm

solve(*instantiation-so-far*, *remaining-constraints*)

Recursion – complex case:

solve(*S*, [*C*|*Cs*])
    Set ← match(*S*, *C*, db)
    search *Set* for *E* where solve(*E*, *Cs*) returns *R*
    return *R*

KR for NLP: Implementation

Prolog – syntax.

Variables – begin with upper case.
constants – begin with lower case.
terms – function(Arguments)
atomic clause (fact) – predicate(Arguments)
rule – Fact :- Body1, Body2,…BodyK.

KR for NLP: Implementation

Prolog contains primitives for matching and search that are equivalent to what we just used.

- You get Prolog to compute something by giving it a ***goal*** – something to prove.
- You try to prove the goal by searching over all possibilities.
- You can prove a goal by matching a fact.
- You can prove a goal by matching the head of a rule, and by proving each atom in the body as a goal.

Prolog constraint satisfaction

At each point, Prolog maintains an instantiation of variables to values.

Each time a goal is matched, it extends the input instantiation to an output instantiation.

Prolog has a metapredicate clause(…) that checks what's in Prolog's database.

Prolog constraint satisfaction

solve(*remaining-constraints*)
extend the implicit instantiation to an output instantiation under which *remaining-constraints* are satisfied.

Base case:

```
solve([]).
```

Prolog constraint satisfaction

solve(*remaining-constraints*)
extend the implicit instantiation to an output instantiation under which *remaining-constraints* are satisfied.

Recursive case:

```
solve([C|Cs]) :- clause(C),
solve(Cs).
```

Prolog constraint satisfaction

Sample knowledge base – Prolog facts.

```
connected(k_c, k_c).
speaker(u).
possession(r_p_c).
result(k_b_m, holds(r_p_c, u, k_c)).
preference(s_p, u, k_b_m).
```

Prolog constraint satisfaction

Sample goal:

```
solve([connected(K_O, k_c),
        speaker(U),
        possession(R),
        result(K_E, holds(R, U, K_O)),
        preference(S, U, K_E)]).
```

Prolog constraint satisfaction

Computes an *answer*

```
K_O = k_c
U = u
R = r_p_c
K_E = k_b_m
S = s_p
```

Language and ontology: Elements of interpretation

Natural language describes ***generalized individuals*** not just ordinary objects.

E.g., we've already seen reference to ***kinds, relations, events and states***.

Language and ontology

In each of these examples a ***generalized individual*** provides the topic of the question and the content of the answer:

What did you buy? A fish.
Where is my coat? In the bathtub.
Where did they go? Into the bathtub.
What did you do? Go to the store.
What happened next? Billy fell out the window.
How did you cook the eggs? Slowly.
How long was the fish? Twelve feet.

Language and ontology

KR for NLP demands a precise understanding of generalized individuals.

- You have to know which ones are described in any utterance interpretation.
- You have to categorize generalized individuals, and express true relationships among them.

Elements of interpretation: Objects and their associates

My pushpin: a round bead of blue plastic affixed to a short spike of steel wire ending in a point.

What *generalized individuals* can be involved in a description like this?

Objects

An object is something that can move coherently as a unit, and maintains its internal physical relationships while in motion.

The pushpin itself is the only object here.

Parts

A part identifies some but not all of an object because it has its own shape, its own history, and/or its own action.

Parts of the pushpin:
the head, the spike, the point.

Sets

Distributive – characterize a set in terms of properties its members share.

"The package contains some yellow pushpins, but I never use them."

Sets

Collective – characterize a set in terms of what it does as a whole.

"Two blue pushpins attach the announcement securely to the board."

Sets

Intermediate readings

"I briefly secured the annoucements with pushpins..."

Discourse connectivity

"But they eventually tore holes in them."

Stuff

the material that makes things up

plastic, steel…

occurs in objects in specific quantities.

Elements of interpretation: Categories

In natural language, kinds are entities with their own distinctive properties.

Dodos are extinct.
Paper clips were invented in the early 1900s.

Reference to kinds

Cover the edge with some tape.

Picks out the kind of tape that's required (maybe Scotch tape, maybe masking tape, maybe duct tape).

Reference to kinds

Isn't it the most versatile fastener ever invented?

Discourse connectivity reminds us that the kind must have been described earlier to be picked up by a pronoun here.

What kinds are there?

Kinds are useful elements of a domain.
They categories things meaningfully not based on surface features but based on the underlying causal processes that make those things what they are.

What kinds are there?

Natural kinds – created by the same general causal regularities in the natural world and when they derive common characteristics from this.

water, tigers, elms, jade,…

What kinds are there?

Artifact categories – recognizably support a specific use in virtue of their effective structure, composition or design.

chairs, tables, pens, ink, …

What kinds are there?

Social categories – classify individuals in terms of the roles, powers, or responsibilities people give them in social practices.

students, teachers, money, laws, signals…

Real-world distinctions (not in languge merely)

Vinegar

*This wine has turned into **vinegar**.*
- natural kind

*Add some **vinegar** to the salad dressing.*
- artifact

*The E.U. taxes soy sauce as **vinegar**.*
- social category

Elements of interpretation: Events and their associates

We can talk directly about intervals of time

I lived in Philadelphia.
I lived in Philadelphia in 1994.
When I lived in Philadelphia, I had no car.

But linear time is less central than causality.

Causality in *when*-clauses

When I took your pawn, you took my queen.

When I took your pawn, I did not know it was protected by your knight.

When I won my only game against Bobby Fischer, I used the Ruy Lopez opening.

Events as individuals

Isolate a pattern of change, initiated by a coherent causal process and effected within a definite time.

Actions are a special subclass of events: those that agents choose for a reason.

Events and description

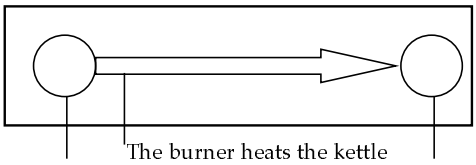
One object, many descriptions:
a rock, a seat, a throne.

One event, many descriptions:
John makes a noise, John says "I'd like coffee"; John answers.

Causal relations individuate events.

Parts of events

I boil water.



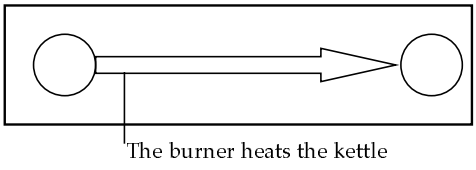
Parts of events

Complex events that aggregate actions with their effects over time are called *accomplishments*.

Instantaneous events are called *achievements*.

Parts of events

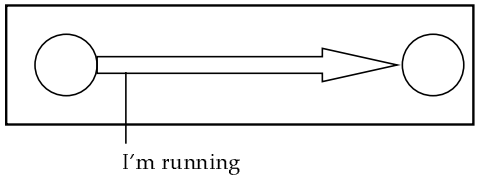
I boil water.



Processes are recurring dynamics that can continue indefinitely – analogous to the stuff that makes up objects.

Parts of events

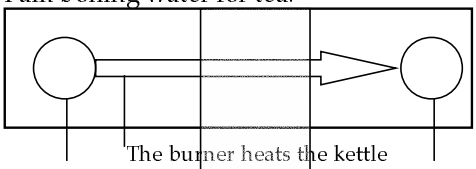
I run a mile.



Activities are processes that are chosen and maintained by agents.

Aspect – linguistic descriptions that focus on parts of events

I am boiling water for tea.



The burner heats the kettle

I ignite the stove (action) The water starts boiling

focuses on this slice of time

Measuring out events

I am boiling water.

Measured by temperature of water (not quantity of water).

I am drinking the water.

Measured by quantity of water (not other properties of water).

Aspectual composition

I boil water:

Event – the indefinite quantity of water undergoes a definite change of temperature.

I drink water:

Activity – the indefinite quantity of water leads to an activity of indefinite duration.

Elements of interpretation: Space and quantity

Key points:

Causality in how objects are individuated and how they are categorized.

Places:

Under the bed is a good place to hide.

Paths:

From NY to Phila is 150 km.

Measurements:

The Empire State Bldg is 443 meters tall.

Places and causality

Chris stood at the window.

Chris stood at the door.

Chris stood at the mirror.

Places and causality

Chris stood near the window.

Chris stood near the door.

Chris stood near the mirror.

Paths and causality

Chris went to the window.
Chris went to the door.
Chris went to the mirror.

Measurements and causality

The Empire State Bldg is tall enough to see from NJ.

Choose a tall glass to hold plenty of ice.

Give me one of the tall glasses.

Elements of interpretation: Abstractions

Language also allows us to talk about *states, possibilities, facts* and *propositions*.

States in interpretation

A *state* is a fine-grained aspect of the enduring condition of the world over an interval of time.

Winning the lottery caused Terry's happiness.
Terry's happiness lasted three years.
Chris saw Terry's happiness.

Possibilities in interpretation

Possibilities are hypothetical scenarios involving a view of the general (maybe underdetermined) consequences of specific assumptions.

Terry might win the lottery.
Chris would be jealous.

Facts in interpretation

Facts reify the immutable data that constitutes the world, in all its coherent regularity and contingent detail.

The fact that Terry departed early meant that most of the party was fun.
Terry's early departure meant that most of the party was fun.

Facts vs. events and states

Terry's departure disturbed Chris.

Event:

Terry packs noisily, stomps out the door and slams it. Chris wakes up but doesn't know what has happened.

Fact:

Terry slinks out unnoticed. But when Chris discovers this, Chris is wildly upset at the implications.

Propositions in interpretations

Propositions reify the content of representations as something that can be true or false.

Chris suggested that Terry had departed.

Language and Logic

Prolog is an implementation of deduction for a fragment of first-order logic.

Our last topic is to connect KR for NLP to logic, and to see what alternatives there are to Prolog for NLP implementations.

Logic and representation

Logic is the characterization of algorithms for inference.

- ***Logical language*** describes representations in a formal way.
- ***Models*** describe the intended interpretation of the logical language.
- ***Proofs*** are traces of algorithms for drawing correct conclusions from assumed representations.
- ***Soundness*** says you can't prove anything false.
- ***Completeness*** says can prove everything true.

Prolog and first-order logic

Prolog implements a ***fragment*** of a first-order language
(Horn clauses with Skolem terms)

Prolog search implements a form of the ***resolution*** proof procedure for the logic.

Prolog computes all and only the consequences that hold in a specific ***canonical model*** of the program specification.

Other logics for KR for NLP

Description logic
Hybrid logic
Modal logic
Higher-order logic
Nonmonotonic logic

Description logic

logic of concepts – all logical expressions abstract away from particular individuals.

Concepts and inference – examples:

speaker

speaker ∨ *hearer*

participant ::= *speaker* ∨ *hearer*

participant ∧¬*hearer* ⊃ *speaker*

Description logic, ctd.

Roles describe relationships among individuals at the concept level

Roles and concepts – examples:

state ∧

(some type possession) ∧

(the logical_subject speaker) ∧

(the logical_object (some associate_of coffee))

Description logic, ctd.

Inferences include

subsumption – must anything that satisfies one concept also satisfy the other?

consistency – is it conceivable that any object could satisfy this concept?

Problems are decidable

(though complexity depends on particular logic)

Hybrid logic

Extends description logic with special concepts that are true uniquely of particular individuals.

"The story repeats itself."

state ∧

(some type repeating) ∧

(the logical_subject (i ∧ *(some type story)))* ∧

(the logical_object i)

Modal logic

Adds ***modal operators*** that instruct you to take only specific information into account.

Example: formalizing causality in resultative constructions.

Resultatives

These sentences all describe the same event:

The rain made the grass flat and wet.

The rain hit the grass flat.

The rain hit the grass wet.

The rain wet the grass.

The rain wet the grass flat.

The rain caused the grass to be flat and wet.

But they each say something different about what happened.

Resultatives

The rain ***hit*** the grass flat.
It was the impact that flattened the grass; just the exchange of force explains it.

The rain ***wet*** the grass flat.
The weight of the water flattened the grass; just the properties of wet grass explains it.

The rain ***made*** the grass flat and wet.
The rain did it directly, through a simple interaction.

Modal formalism

The rain ***hit*** the grass flat.
[BALLISTICS] $(result(c, s) \wedge flat(s,g))$

The rain ***wet*** the grass flat.
[INUNDATION] $(result(c, s) \wedge wet(s,g) \wedge flat(s,g))$

The rain ***made*** the grass flat and wet.
[DIRECT] $(result(c, s) \wedge flat(s,g) \wedge wet(s,g))$

Higher-order logic

Allows variables for ***functions*** and ***properties***.

Example: VP ellipsis.
John saw his mother. Bill did too.
Semantic representation:
$see(j, m(j)).\ P(b).$
Parallelism constraint:
$R(b) = P(b) \wedge R(j) = see(j, m(j))$
Solutions include:
$P = \lambda x.see(x, m(x))$ and $P = \lambda x.see(x, m(j))$

Nonmonotonic logic

Allows you to make assumptions and reconcile competing explanations.

Example:
If the referent of the subject of the previous sentence matches a pronoun's agreement information then normally the pronoun picks up the same referent.
But there are exceptions…

HW

Implementation:
Code up the constraint-satisfaction solver described today in Prolog.
Create a database to describe each of the contexts for (1) described in the text.
Check that the solver computes the intended resolution for each context.

HW

Representation:
Pick a small household tool – an appliance, gadget, toy, etc..
Spend a little time brainstorming: what objects, parts, categories, events, actions, places, do you need to talk about this thing?
You should easily be able to come up with a page or two.
How much variability will there be in how we talk about these things?