12. Natural language understanding

12.1 Understanding

- “Aoccdrnig to a rscheearch at Cmabrigde Uinervtisy, it deosnt mttaer in waht oredr the ltteers in a wrod are in; the olny iprmoetnt tihng is taht frist and lsat ltteer be at the rghit pclae. The rset can be a toatl mses and you can sitll raed it wouthit porbelm. Tihs is bcusae the huamn mnid deos not raed ervey lteter by istlef, but the wrod as a wlohe.”

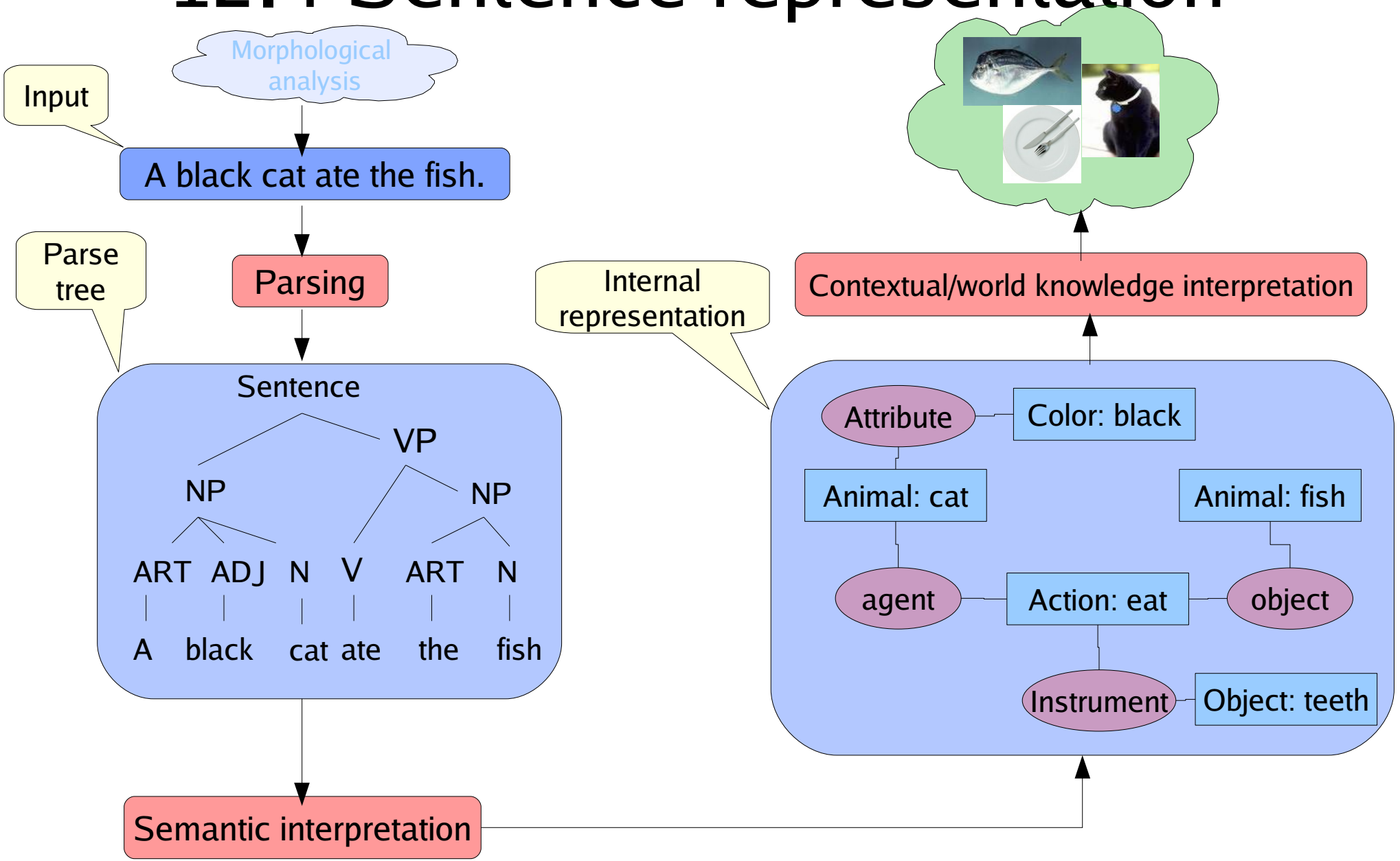
12.2 Parts of speech

- Words belong to lexical categories (syntactic classes), which in most languages are:
  - **Nouns** label objects and events (dog, car, party)
  - **Verbs** label actions, states or relations (cry, run)
  - **Adjectives** describe nouns (large, red, accurate)
  - **Adverbs** describe actions (quickly, indeterminately)
  - **Pronouns** substitute for nouns (she, it, they)
  - **Connectives** join phrases and sentences (and, but)
  - **Quantifiers** (some, many, all)
- Many languages also have prepositions and articles, etc.

12.3 Process of understanding

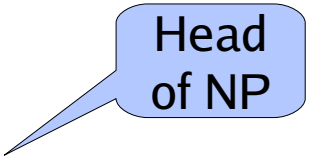
- *Morphological analysis* to
  - identify parts of speech
  - determine how words break down into components, and how this affects their grammatical status.
- *Syntactic analysis*
  - uses a *lexicon* (dictionary of word meanings) and *grammar* (set of rules)
  - reveals the syntactic structure.
  - determines if the sentence is valid.
- *Semantic analysis*
  - builds a representation of objects and actions that the sentence is describing.
  - solves ambiguities.
- Perform an action as a result

12.4 Sentence representation

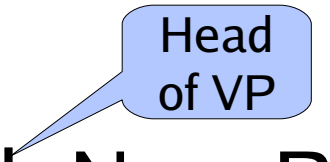


12.5 Language structure

- NounPhrase NP → Noun
  | Article Noun
  | Adjective Noun
  | Article Adjective Noun



- VerbPhrase VP → Verb
  | Verb NounPhrase
  | Adverb Verb NounPhrase



- Sentence S → NounPhrase VerbPhrase

12.6 Parsing and generation

- *Lexicon*
  - Dictionary of words and their meanings.
  - Stores possible categories of words (if they can appear as nouns, adjectives, etc.)
- *Parsing* algorithm
  - Procedure that combines grammatical rules to generate the tree that reflects the structure of the sentence.
  - Uses grammar and lexicon: knowledge about the structure and the contents of linguistic objects.
- Generation
  - A process of producing a legal sentence from logical form of the sentence.

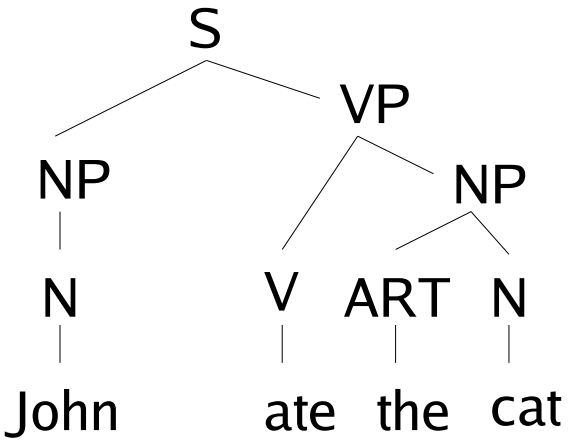
12.7 Grammar

- The same kind of grammars that are used to parse artificial languages (e.g., programming languages) are used to parse natural languages.
- Simple context free grammar:
  - S → NP VP
  - VP → V NP
  - NP → N | ART N
  - V → ate
  - ART → the
  - N → John | cat

Start symbol

Non-terminal symbol

Terminal symbols are words in the language



"John ate the cat."

- Legal sentence = any sequence of terminals that can be derived from the rules of grammar.

12.8 Top-down and bottom-up parsers

"John ate the cat."

- Top-down, driven by the grammar:

  S
  ⇒ NP VP
  ⇒ N VP
  ⇒ John VP
  ⇒ John V NP
  ⇒ John ate NP
  ⇒ John ate ART N
  ⇒ John ate the N
  ⇒ John ate the cat.

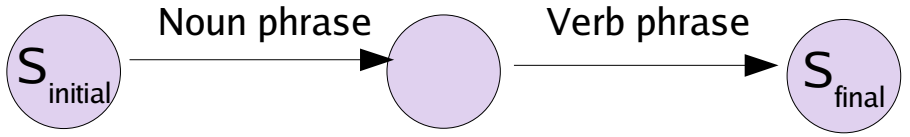
- Bottom-up, driven by the lexicon:

  John ate the cat
  ⇒ N ate the cat
  ⇒ N V the cat
  ⇒ N V ART cat
  ⇒ N V ART N
  ⇒ NP V ART N
  ⇒ NP V NP
  ⇒ NP VP
  ⇒ S

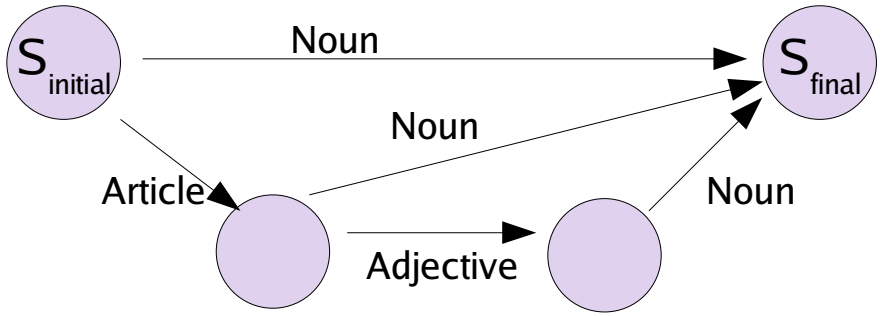
12.9 Recursive transition network

- Represents the grammar as a set of finite-state automata.
- A network for each non-terminal in the grammar.
- Arc labels refer to other networks or word categories.

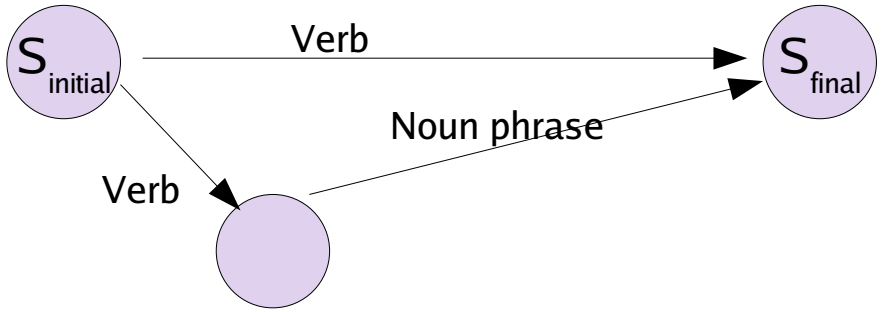
Sentence:
$S \rightarrow NP\ VP$



Noun phrase:
$NP \rightarrow N$
$NP \rightarrow ART\ N$
$NP \rightarrow ART\ A\ N$

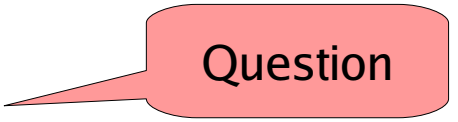


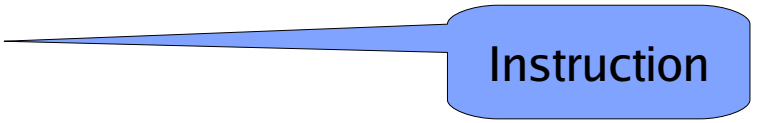
Verb phrase:
$VP \rightarrow V$
$VP \rightarrow V\ NP$



12.10 Chart parsing

- With large grammars top-down and bottom-up methods repeat the same matches over and over again.
- Transition networks cannot handle the following sentences:
  - Have all the fish been fed?
  - Have all the fish.





- Solution: Keep record of rules that have been partially matched or have been completed.

12.11 Chart parsing example

0 The 1 dog 2 can 3 jump 4 the 5 fence 6

Starting point:
1. S → • NP VP

Add rules for NP:
2. NP → • ART ADJ N
3. NP → • ART N
4. NP → • ADJ N

Extend rules after 'The':
1. S_0 → • NP VP
2. NP_0 → ART • ADJ N
3. NP_0 → ART • N
4. NP_0 → • ADJ N

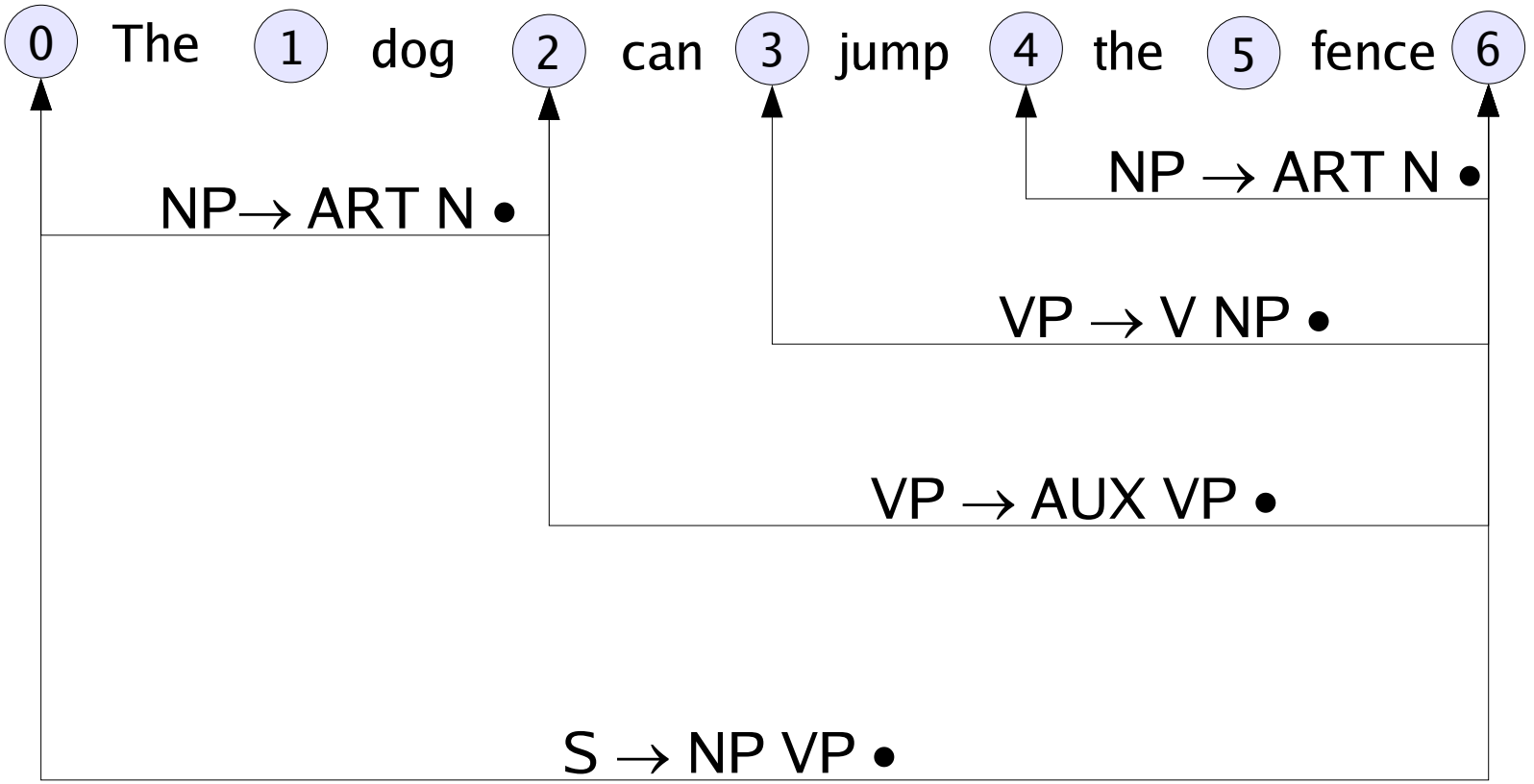
Extend rules after 'dog':
1. S_0 → • NP VP
2. NP_0 → ART • ADJ N
3. NP_0 → ART N •
4. NP_0 → • ADJ N

Extend rule 1:
1. S_0 → NP • VP

Introduce VP rules:
1. S_0 → • NP VP
2. NP_0 → ART • ADJ N
3. NP_0 → ART N •
4. NP_0 → • ADJ N
5. VP_2 → • AUX VP
6. VP_2 → • V NP

Grammar:
S → NP VP
NP → ART ADJ N | ART N | ADJ N
VP → AUX VP | V NP

12.12 Completed chart



12.13 Problems with CFG

- Agreement; a CFG will accept all of these:
  - The dog barks & *The dogs barks.
  - A dog barks & *A dogs bark.
- Long distance dependencies:
  - “What did you say you thought Bill wanted to wear?”
- Solution (attempts):
  - Feature systems
  - Augmented transition networks
  - Context sensitive grammar

12.14 Syntactic features

- Define constraints as feature structures, for instance:

  ART1: (**CAT** ART
  ROOT a
  NUMBER s)

  NP1: (NP **NUMBER** s
  1.(ART **ROOT** a
  NUMBER s)
  2.(N **ROOT** dog
  NUMBER s))

- And rules:

  (NP **NUMBER** ?n)

  →

  (ART **NUMBER** ?n)

  (N **NUMBER** ?n)

12.15 Feature Systems in English

- Person and number (NUMBER feature)
  - First person: speaker included
  - Second person: listener included
  - Third person: neither speaker or listener included
- Verb form and subcategories
  - VFORM feature for base, pres, past, inf and ing.
  - SUBCAT feature handles the interaction between words and their complements.
  - Examples:

VP → (V **SUBCAT** _np_vp:inf)
(NP)
(VP **VFORM** inf)

Value	**Verb**	**Example**
_none	laugh	Jack laughed.
_np	find	Jack found a key
_np_np	give	Jack gave Sue a key.
_vp:inf	fly	Jack wants to fly.
_np_vp:ing	catch	Jack caught Sam looking at his desk

12.16 What is meaning?

- Non-linguistic
  - Sound of a fire alarm means something is burning.
  - Vervet monkeys' alarm call means there is a predator of a certain kind nearby.
- Meaning of a word
  - In terms of other words (lexical relation)
  - In term of decomposition (semantic primitives)
  - In terms of perception and action
  - Knowledge of words vs. knowledge of world:

    "Do you know what gate you are going to?"
- Why?
  - To make inferences
  - To answer questions or to respond to requests

12.17 Semantic interpretation

- Mapping of sentence to logical form.
- *Logical form* = context-independent meaning of the sentence. For example:

  (OR1 (LOVES1 JACK1 SUE1)(LOVES1 JACK1 MARY1))
- Mapping from logical form to final representation = *contextual interpretation*.
- *Compositional* process
  - Meaning of a constituent is reconstructed from the meanings of the sub-constituents.
  - Interpretation can be built incrementally.
  - The assumption makes grammars easier to extend and maintain, but for semantics it poses problems.

12.18 How to represent logical form?

- Several natural-language constructs (kind of) correspond to FOPL operators:
  - Connectives
  - Quantifiers
- However, several of them are context dependent
  - Indexical terms — for instance the meaning of 'I' and 'you' depend on who is talking.
  - Tense –dis tinction of time, or duration of an action in time.

12.19 Logical operators

- Connectives
  - "Tom went home and had a drink"vs. " Tom had a drink and went home."
  - "If you are thirsty, (then) there is some juice in the fridge."
  - "If I am sick, (then) I'll go to see a doctor."(implicit equivalence)
- Modal operators
  - "John believes George is not happy."
  - "John believes the president is not happy."
- Generalized quantifiers: a, the, some, many, most, several, a few, etc.

  (MOST1 **d1**: (DOG1 **d1**)(BARKS1 **d1**)) ≠

  (MOST1 **d2**: (BARKS1 **d2**) (DOG **d2**))

12.20 Ambiguity

- Role vs. value interpretation
  - "I always liked the people living next door."
  - "Bring me the largest tomato in the garden."
  - "The president changes often."
- Definite vs. indefinite referent
  - Is the hearer expected to identify the referent NP?
    - "Bring me the largest tomato in the garden."
    - "I'm looking for a book by Hemingway."

Role interpretation

Role vs. value interpretation

Definite referent

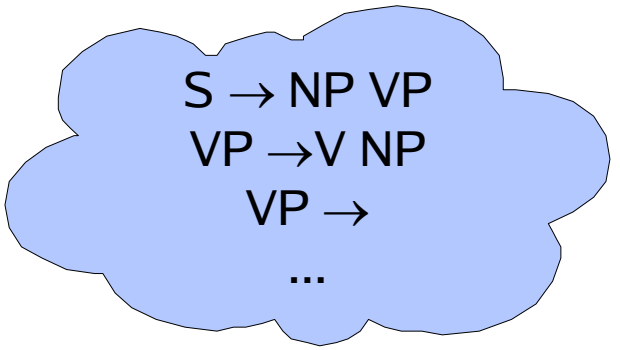
Indefinite referent

12.21 Roles

- Despite similar surface structure, the meaning structure may differ:
  - “Phyllis likes (detests, trusts) Harold.”
  - “Phyllis sickens (bores, inspires) Harold.”
- Differs in different languages:
  - I like you.
  - Du gefällst mir.

12.22 Challenges to compositionality

- Semantic interpretation is a compositional process.
- No correspondence between syntactic structure and logical form.
  - "Jill loves every dog."
  - ((**NP** Jill) (**VP** loves (**NP** every dog)))
  - (EVERY *d*: (DOG *d*) (LOVES *l1* (NAME *j1* "Jill") *d*)); this asserts that for each dog *d* there is an event *l1* consisting of Jill loving *d*.
- Idioms
  - "Jack kicked the bucket."
    - ⇒ (DIE1 *d1* (NAME *j1* "Jack"))
    - ⇒ (KICK1 *k1* (NAME *j1* "Jack") <THE *b* BUCKET>
  - "The bucket was kicked by Jack."



12.23 How to represent compositionality in semantics?

- Introduce new sense for words appearing in idioms:
  - Sense of kick meaning DIE1, and subcategory for an object type BUCKET1.
- Need a uniform way of associating semantic structure to any syntactic constituent.
- For instance, present the meaning of each VP as a unary predicate:
  - "Jack laughed."

    $\Rightarrow$ *laughed* is a predicate that is true for any entity that laughed in the past.
- Can this be generalized to any VP?

12.24 Lambda calculus

- Let's have an example: “Jack kissed Sue.”
- Logical form:

  (KISS1 *k1* (NAME *j1* “Jack”) (NAME *s1* “Sue”))
- We could have a unary predicate *kissed Sue*, which is true for any entity that kissed Sue.
- Lambda calculus provides formalism for this:
  - λ *x* (KISS1 *k1 x* (NAME *s1* “Sue”))
  - This lambda expression is a predicate with one argument.
  - Applying the λ expression to argument is called lambda reduction:
    - ((λ *x* (KISS1 *k1 x* (NAME *s1* “Sue”)))(NAME *j1* “Jack”))
    - Output: (KISS1 *k1* (NAME *j1* “Jack”) (NAME *s1* “Sue”))

12.25 Semantic features

- Add a SEM feature to each lexical entry and grammatical rule:
  (S **SEM** (?semvp ?semnp))→(NP **SEM** ?semnp)(VP **SEM** ?semvp)
- **SEM** feature indicates the possible senses of the word.
- Sample lexicon:

  a (art **AGR** 3s **SEM** INDEF1)
  the (art **AGR** {3s 3p} **SEM** DEF1)
  she (pro **AGR** 3s **SEM** SHE1)
  Jill (name **AGR** 3s **SEM** "Jill")
  dog (n **SEM** DOG1 **AGR** 3s)
  dogs (n **SEM** (PLUR DOG1) **AGR** 3p)
  can (aux **SUBCAT** base **SEM** CAN1)
  see (v **SEM** SEES1 **VFORM** base **SUBCAT** _np)
  saw (v **SEM** SEES1 **VFORM** past **SUBCAT** _np)

- Sample grammar:

(S **SEM** (?semvp ?semnp)) →
(NP **SEM** ?semnp)(VP **SEM** ?semvp)

(VP **SEM** (λ a2 (?semv ?v a2))) → (V[_none] **SEM** ?semv)

(VP **SEM** (λ a3 (?semv ?v a3 ?semnp)))
→ (V[_np] **SEM** ?semv)(NP **SEM** ?semnp)

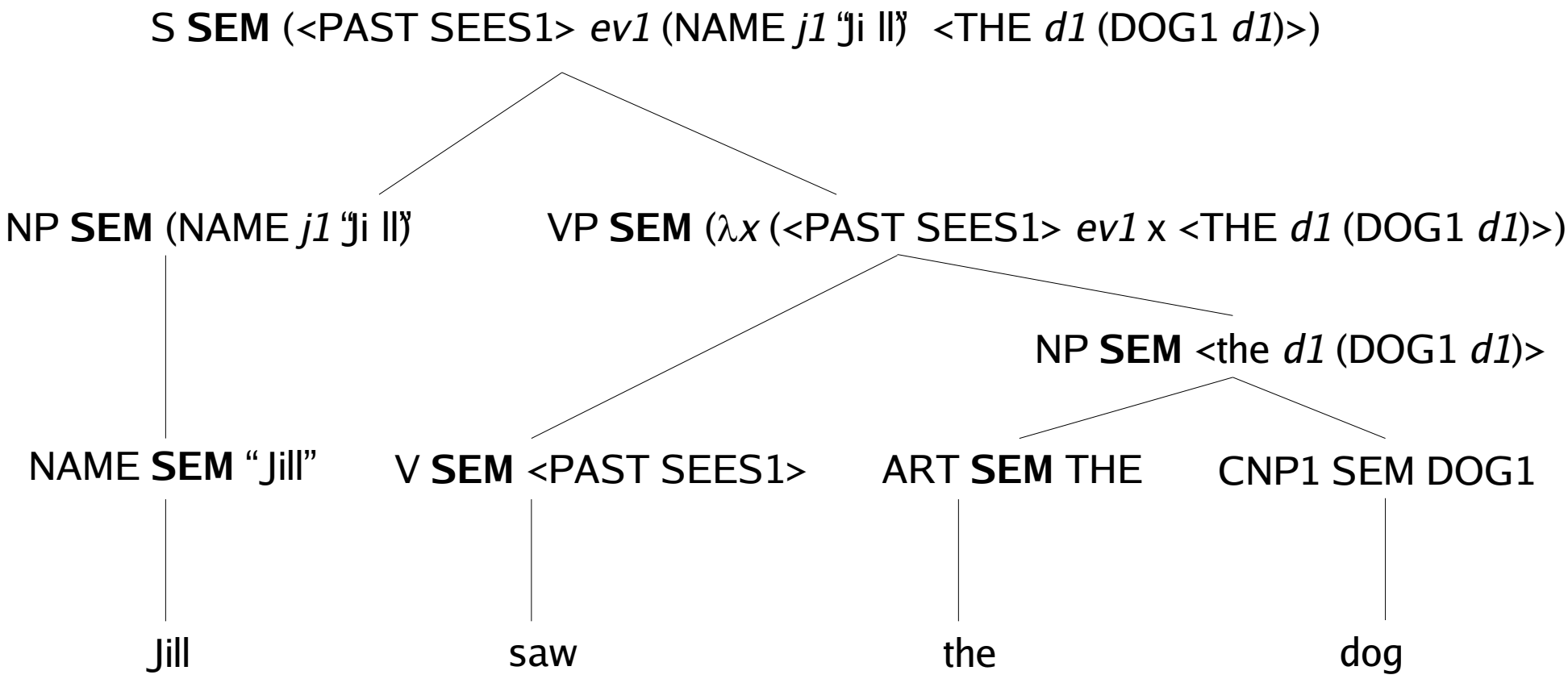
(NP **SEM** (NAME ?v ?semname)) →
(NAME **SEM** ?semname)

(NP **SEM** (?semart ?v (semcnp ?v))) →
(ART **SEM** ?semart)(CNP **SEM** ?semcnp)

(CNP **SEM** ?semn) → (N **SEM** ?semn)

12.26 Parse tree with SEM features

- "Jill saw the dog."



12.27 World knowledge

- What does it mean for a sentence to make sense in a context?
- Logical consistency
  - Many logically consistent readings, only some of which make sense.
- Coherence
  - Easy to determine how each sentence relates to the others.

1a. Jack took out a match.
1b. He lit a candle.

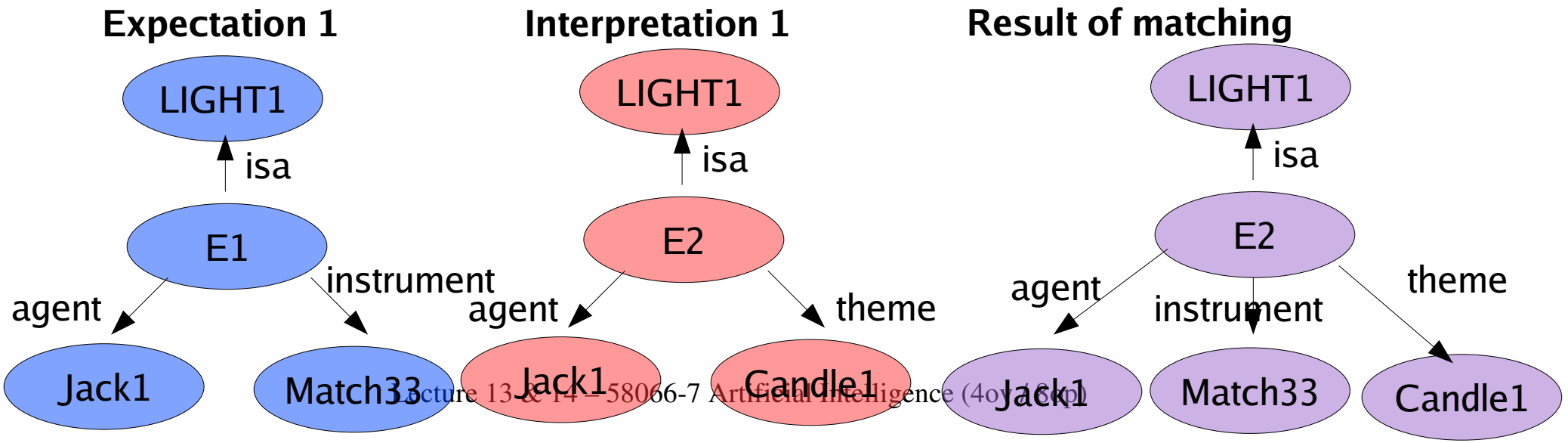
2a. Jack took out a match.
2b. The sun set.

12.28 Expectations

- Plausible eventualities.
- Formed by the content of previous sentences + inferences made.
- The problem more formally: Given a set of expectations $E_1, \ldots, E_n$, and set of possible interpretations for the sentence $I_1, \ldots, I_k$, determine the set of pairs E_i, I_j such that they match.
- Continuing the example 1:
  - Expectation 1 after reading 1a:
    - Jack is going to use the match to light something.
    - LIGHT1(E1) & Agent(E1)=Jack1 & Instrument(E1)=match33
  - Interpretation 1 of 1b:
    - LIGHT1(E2) & Agent(E2)=Jack1 & Theme(E2)=Candle1
  - Interpretation 2 of 1b:
    - ILLUMINATE1(E3) & Agent(E3)=Jack1 & Theme(E3)=Candle1

12.29 Matching expectations and interpretations

- Potential problem if both expectation and interpretation contain information not contained in the other.
- Equality-based techniques
  - Two formulas match if there is a consistent set of equality assumptions that make E_i entail I_j.
  - Formulas are encoded as semantic networks:



- Abduction-based techniques
  - Find an explanation why the current sentence is true: use the situation and the expectations to construct an argument that proves the current sentence.
  - More formally, given a specific situation S (includes the expectations) and possible interpretations I_1,..., I_k, consider assumptions A_j to be added to S such that for each I_j
    - $S \& A_j$ entails I_j.
    - $S \& A_j$ are consistent
  - Choose I_j that requires the minimum set of assumptions.

12.30 Centering theory

- Center is the focus of the discourse
  - tends to remain the same for a few sentences.
  - pronominalized.
- Examples:
  - 3a. Jack left for the party late.
  - 3b. When he arrived, Sam met him at the door.
  - 3c. He decided to leave early.
- Consider these two examples:

4a. Jack poisoned Sam.
4b. He died within a week.

5a. Jack poisoned Sam.
5b. He was arrested in a week.

- Centering theory would either predict the same antecedent, or not choose between them.
- Expectation matching method identifies the correct referent.
  - Recency and semantic constraints
    - "Jack drank the wine on the table. It was brown and round."
  - Preferences from local context:
    - "Jack walked over to the table and drank the wine. It was brown and round."
  - In both cases, the latter sentence is found anomalous or humorous.

12.31 Understanding actions

- Performed by intentional agents (as opposed to force and inertia in physics)
- In order to understand causal relationships $\rightarrow$ generate expectations.
- Actions relate to states by
  - Effect causality
    - Hold after or while the action is executed.
    - Intended effects
    - Side effects
    - Failed action may have side effects.
  - Precondition causality
    - Hold just before or during the activity.

- Example:

  The Action Class ***BUY(e)***:

  Roles: Buyer, Seller, Object, Money

  Constraints: Human(Buyer), SalesAgent(Seller), IsObject(Object),
  Value(Money, Price(Object))

  Preconditions: AT(Buyer, Loc(Seller))
  OWNS(Buyer, Money)
  OWNS(Seller, Object)

  Effects: ¬OWNS(Buyer, Money)
  ¬OWNS(Seller, Object)
  OWNS(Buyer, Object)
  OWNS(Seller, Money)

  Decomposition: Give(Buyer,Seller,Money)
  Give(Seller, Buyer, Object)

6a. Jack bought a stereo at the mall.
6b. He now owns it.

7a. Jack bought a stereo at the mall.
7b. Now he can disturb his neighbors late at night.

12.32 Understanding stereotypical situations

- *Scripts*
  - can be used to control the generation of expectations
  - identify common situations and scenarios
  - Involve many different action, potentially by different agents, and specify a common causality links between them.
- Select a script, and then find the place in the script (*now point*)
- Different kinds of sentences are matched to different parts of the script:
  - Sentence describing an action as a goal → the action names the script.
  - Sentence describing a state as a goal → match contents against the effects of the script.
  - Sentence describing an action in execution → match against the decomposition.

- Example script:

TRAVEL-BY-TRAIN(e):

Roles: Actor, Clerk, SourceCity, DestCity, Train, Station, Booth, Ticket, Money

Constraints: Person(Actor), Person(Clerk), City(SourceCity), City(DestCity), TrainSation(Station), ..., In(Station, SourceCity), DepartCity(Train, SourceCity), ... TicketFor(Ticket, SourceCity, DestCity), Value(Money, Price(Ticket)), ...

Preconditions: Owns(Actor, Money)
At(Actor, SourceCity)

Effects: ¬Owns(Actor, Money)
¬At(Actor, SourceCity)
At(Actor, DestCity)

8a. Sam wanted to take a train to Rochester.
8b. He purchased a ticket at the station.

Decomposition: GoTo(Actor, Station)
Purchase-Ticket(Actor, Clerk, Ticket, Station)
GoTo(Actor, Loc(Train))
GetOn(Actor, Train)
Travel(Train, SourceCity, DestCity)
Arrive(Train, DestCity)
GetOff(Actor, Train)