Chapter 8:Linking Syntax and Semantics

8.1 **Semantic Interpretation and Compositionality**

example: *Jill loves every dog*

Syntactic structure: ((Jill) (loves (every dog)))

the unambiguous logical form:

(EVERY d : (DOG1 d) (LOVES1 l1 (NAME j1 “Jill”) d))

There seems to be no simple one to one correspondence between parts of the logical form and the constituents syntactic analysis.

For instance, VP is loves every dog, so every dog is a part of VP, but in the logical form:

(EVERY d : (DOG1 d) (LOVE ... We see VP is a part of quantitative phrase (EVERY d : (DOG1 d) ...).

Linking Syntax and Semantics

8.1 Semantic Interpretation and Compositionality

The unscoped version sentence would be:

(i) (EVERY d : (DOG1 d) (LOVES1 l1 (NAME j1 “Jill”) d))

(ii) (LOVES1 l1 (NAME j1 “Jill”) < EVERY d DOG1 >)
which is much closer in structure to the syntactic form.

- Note: the idiom problem

Example: *Jack kicked the bucket* has meaning as *Jack died.*

Example: (i) *Jack laughed.*

One suggestion is that VP Jack laughed is unary predicate

Linking Syntax and Semantics

8.1 Semantic Interpretation and Compositionality

Jack kissed Sue □(KISS1 k1 (NAME j1 “Jack”) (NAME s1 “Sue”))

What is the meaning of the VP kissed Sue ?. Again, it could be a unary predicate.

But so far we have no way to express such complex unary predicate.

The lambda calculus provides a formalism for this. In particular, the expression:

(KISS1 k1 (NAME j1 “Jack”) (NAME s1 “Sue”) □

λ x (kiss1 k1 x (NAME s1 “Sue”))) O

This form is a unary predicate

Linking Syntax and Semantics

8.1 Semantic Interpretation and Compositionality

Proposition: (λ x (kiss1 k1 x (NAME s1 “Sue”))) (NAME j1 “Jack”) is true if and only if (NAME j1 “Jack”) satisfies the predicate (λ x (kiss1 k1 x (NAME s1 “Sue”))), which by definition is true if only if

(KISS1 k1 (NAME j1 “Jack”) (NAME s1 “Sue”)) is true.

The expression:

(KISS1 k1 (NAME j1 “Jack”) (NAME s1 “Sue”)) was obtained by applying the Lambda expression;

(λ x (kiss1 k1 x (NAME s1 “Sue”))) to the argument (NAME j1 “Jack”).

This operation is called Lambda reduction

Linking Syntax and Semantics

8.1 Semantic Interpretation and Compositionality

Example: **Sue laughs and opens the door** □

(λ a (LAUGH s1 l2 a)   /* *VP : laughs* */

and (λ a (OPENS O1 a < THE d1 DOOR1 >))) /* opens the door */

These Lambda expression can be combined to form a complex unary predicate
(λ a (& (LAUGH s1 l2 a) (OPEN s1 O1 a < THE d1 DOOR1 >)))

For instance, it can be applied to a subject NP with logical form (NAME s1 "Sue") to form the meaning pf the original sentence.

(& (LAUGH s1 l2 (NAME s1 "Sue"))

(OPEN s1 o1 (NAME s1 "Sue") < THE d1 DOOR1 >)))

Linking Syntax and Semantics

8.1 Semantic Interpretation and Compositionality

The prepositional phrase modifies in noun phrase

Example: *The man in the store*

The noun phrase *The man in the store* contains the preposition *in the store.* The preposition might have <u>not in dependent meaning in the noun phrase.</u>

If prepositional phrase has an independent meaning, in this case the unary predicate:

(λo (IN – LOC1 o < THE s1 STORE1 >))

Then, the logical form of noun phrase *the man in the store* would be:

< THE m1 (MAN1 m1) (IN – LOC1 m1 < THE s1 STORE1 >) >

Linking Syntax and Semantics

8.1 **Semantic Interpretation and Compositionality**

While the logical form of the sentence *The man is in the store would be:*

(IN – LOC1 < THE m1 MAN1 > < THE s1 STORE1 >)

There is difficult to build a unary predicate of noun phrase in the case that a noun phrase contains a prepositional phrase.

Linking Syntax and Semantics

8.2 A simple Grammar and Lexicon with Semantic Interpretation

The main extension needed is to add a SEM feature to each lexical entry and grammatical rule.

Example :

(S SEM (? semvp ? semnp)) □ (NP SEM ? semnp) (VP SEM ? semvp)

The rule NP with SEM (NAME m1 “Mary”) and VP subconstituent with SEM

(λ a (SEES1 e8 a (NAME j1 “Jack”)))

Thus SEM of new S constituent is the expression

((λ a (SEES1 e8 a (NAME j1 “Jack”))) (NAME m1 “Mary”))

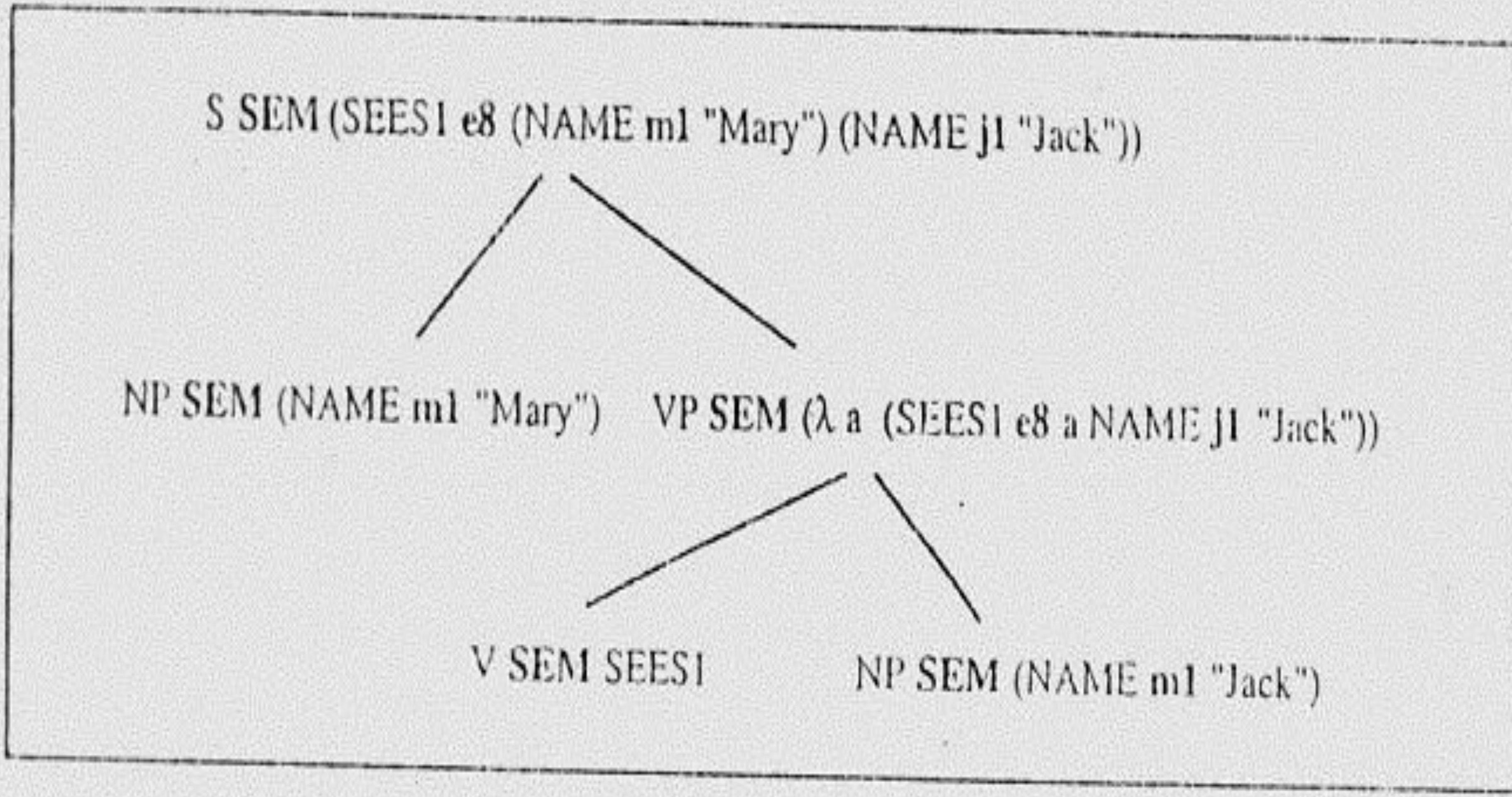

Figure 8.1 A parse tree showing the SEM features

Linking Syntax and Semantics

8.2 A simple Grammar and Lexicon with Semantic Interpretation

After Lambda reduction this expression would be the formula:

(SEES1 e8 (NAME m1 "Mary") (NAME j1 "Jack"))

In the lexicon, a word will have different word sense of every possible subcategorization it has, since there will be different arity predicates.

Example: The verb decide that has two entries:

(i) SUBCAT [– none]

(ii) SUBCAT [– pp: on] , in this case the verb has one more an object

- -Word fish has two entries in lexicon, because its SEM depends on whether it is singular or plural

a (art AGR 3s SEM INDEF1)
can (aux SUBCAT base SEM CAN1)
car (n SEM CAR1 AGR 3s)
cry (v SEM CRY1 VFORM base SUBCAT _none)
decide (v SEM DECIDES1 VFORM base SUBCAT _none)
decide (v SEM DECIDES-ON1 VFORM base SUBCAT _pp:on)
dog (n SEM DOG1 AGR 3s)
fish (n SEM FISH1 AGR 3s)
fish (n SEM (PLUR FISH1) AGR 3p)
house (n SEM HOUSE1 AGR 3s)
has (aux VFORM pres AGR 3s SUBCAT pastprt SEM PERF)
he (pro SEM HE1 AGR 3s)
in (p PFORM {LOC MOT} SEM IN-LOC1)
Jill (name AGR 3s SEM "Jill")
man (n SEM MAN1 AGR 3s)
men (n SEM (PLUR MAN1) AGR 3p)
on (p PFORM LOC SEM ON-LOC1)
saw (v SEM SEES1 VFORM past SUBCAT _np AGR ?a)
see (v SEM SEES1 VFORM base SUBCAT _np IRREG-PAST + EN-PASTPRT +)
she (pro AGR 3s SEM SHE1)
the (art SEM THE AGR {3s 3p})
to (to AGR – VFORM inf)

Figure 8.2 A small lexicon showing the SEM features

1. (S SEM (?semvp ?semnp) → (NP SEM ?semnp) (*VP SEM ?semvp*)
2. (VP VAR ?v SEM (λ a2 (?semv ?v a2))) → (*V[_none] SEM ?semv*)
3. (VP VAR ?v SEM (λ a3 (?semv ?v a3 ?semnp))) →
   (*V[_np] SEM ?semv*) (NP SEM ?semnp)
4. (NP WH – VAR ?v SEM (PRO ?v ?sempro)) → (*PRO SEM ?sempro*)
5. (NP VAR ?v SEM (NAME ?v ?semname)) → (*NAME SEM ?semname*)
6. (NP VAR ?v SEM <?semart ?v (?semcnp ?v)>) →
   (ART SEM ?semart) (*CNP SEM ?semcnp*)
7. (CNP SEM ?semn) → (*N SEM ?semn*)

Head features for S, VP, NP, CNP: VAR

Grammar 8.3 A simple grammar with SEM features

Linking Syntax and Semantics

- **8.3 Prepositional phrases and Verb phrase**

The rules for handling auxiliary verbs

(VP SEM (λ a1 (?semaux (?semvp a1))))

(AUX SUBCAT ? v SEM ? semaux) (VP VFORM ? v SEM ? semvp)

This rule inserts a modal operator in the appropriate place for the new VP.

For instance, if ?semaux is a modal operator such as CAN1 and ?semvp is lambda expression such as:

(λ a1 (LAUGHTS1 e3 x)), then according to the auxiliary rule, the SEM of the VP *can laugh* will be:

(λ a1 (CAN1 ((λ x (LAUGHS1 l3 x)) a1)) *this can be simplified to:*

(λ a1 (CAN1 (LAUGHS1 l3 a1))

Linking Syntax and Semantics

- **8.3 Prepositional phrases and Verb phrase**

Prepositional phrases (PP)

The prepositional phrase has two different semantic roles in sentence.

(i) PP is a modifier to a noun phrase or verb phrase.

(ii) PP is subcategorized for by a head word and preposition acts more as a flag for an argument position the as an independent predicate.

For ***(i) case:*** PP has SEM as a unary predicate:

(PP SEM (λ y(?semp y ?semnp)))□(P SEM ?semp)(NP SEM ?semnp)

Linking Syntax and Semantics

- **8.3 Prepositional phrases and Verb phrase**

Example: PP is *in the corner*

If SEM of P is IN – LOC and SEM of NP is < THE c1 (CORNER c1) >, then SEM of PP is a unary predicate:
(λ y (IN – LOC 1 y < THE c1 CORNER 1 >))

PP modifies the noun phrase:

Example: *the man in the corner:*
(CNP SEM (λ n1 (& (? semcnp n1) (? sempp n1)))) □
(CNP SEM ? semcnp) (PP SEM ? sempp)

+if *in the corner has* SEM of P is IN - LOC1 and SEM of *the corner* –NP is < THE c1 (CORNER1 c1) >, then PP would be a unary predicate:**(λ y (IN – LOC1 y < THE c1 CORNER1 >))**

Linking Syntax and Semantics

- **8.3 Prepositional phrases and Verb phrase**

Given that the SEM of the CNP *man* is unary predicate MAN1 and SEM of PP *in the corner* is (λ y (IN – LOC1 y < THE c1 CORNER1 >)), new SEM of CNP:

$$(\lambda\ n1\ (\&(MAN1\ n1)\ (\lambda\ y\ (\underbrace{IN\text{-}LOC1}_{IN1}\ y < THE\ c1\ CORNER1 >))\ n1)))$$

This can be simplified to:

$$(\ \lambda\ n1\ (\ \&\ (\ MAN1\ \ n1\ )\ (\ IN1\ \ n1 < THE\ \ c1\ \ CORNER1 >\ )))$$

- Combining this unary predicate with quantify such as *the* using rule 6 would be form a SEM of CNP *the man in the corner* such as

Linking Syntax and Semantics

8.3 Prepositional phrases and Verb phrase

< THE m2 (λ n1 (& (MAN1 n1) (IN1 n1 < THE c1 CORNER1 >)) m2 >. Which can be simplified to:

<THE m2 (&(MAN1 m2)(IN1 m2 <THE c1 CORNER1 >)) >

<u>PP modifies verb phrase</u>

Example: *VP cry in the corner,*

The that introduces the PP modifier would be: VP □ VP PP

V *cry* has the logical form: (λ x (CRIES1 e1 x))

VP *cries in the corner* has the logical form:

(λ x (CRIES1 e1 x) (IN1 e1 < THE c1 CORNER1 >)))

Linking Syntax and Semantics

8.3 Prepositional phrases and Verb phrase

Sine PP modifies VP, therefore we add the rule:

(VP VAR ? v SEM (λ x (& (? semvp x) (? sempp ? v))) □
(VP VAR ? V SEM ? semvp) (PP SEM ? sempp)

PP also appear as subcategorized constituents in verb phrases, in this case must be treated differently.

Example: PP *on a couch*

This PP indicates a location of the some object or event.

With verb decide, it can indicate the object that is being decided about.

Example: sentence *Jill decided on a couch*.

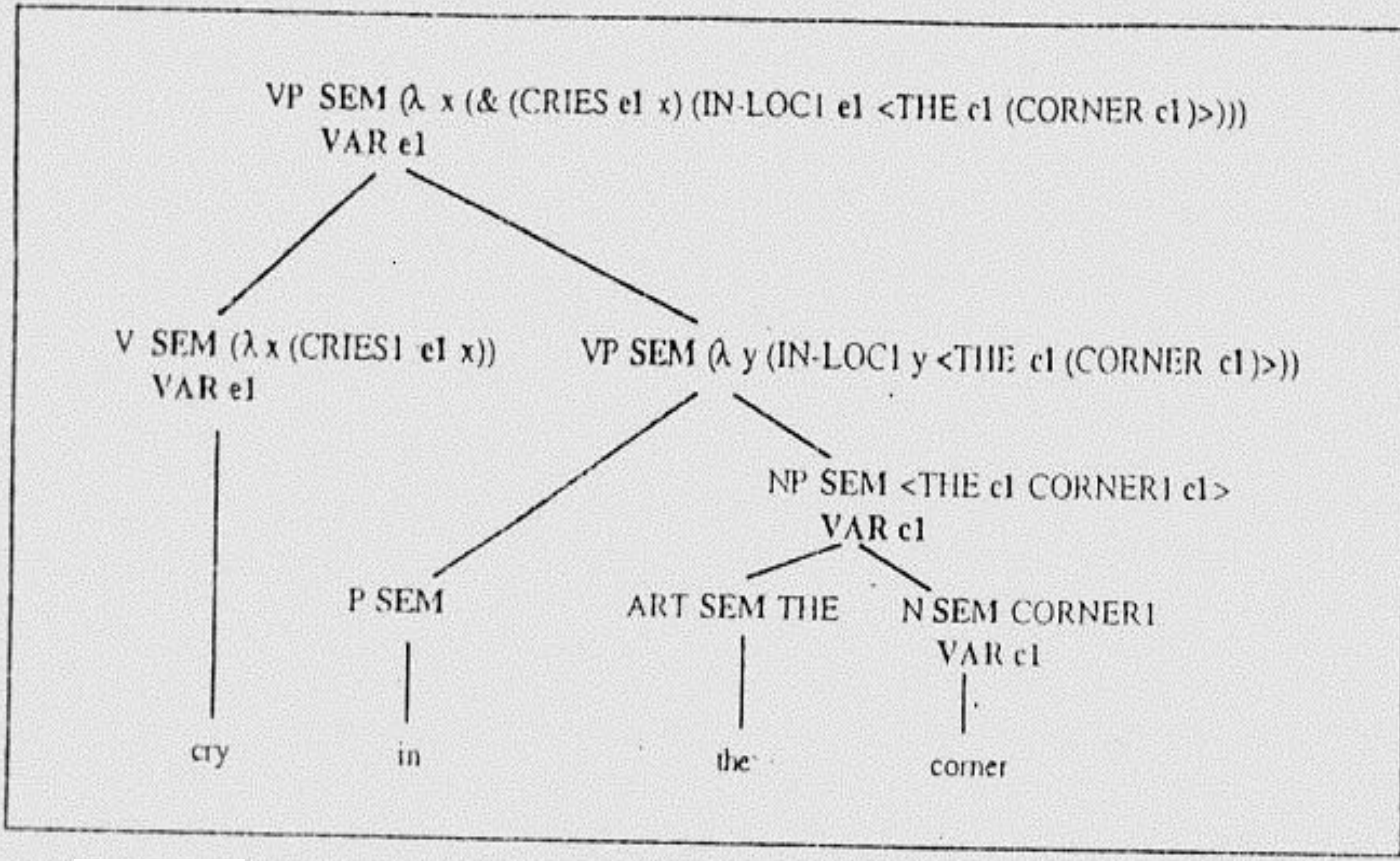

Figure 8.4 Using the VAR feature for PP modifiers of VPs

Linking Syntax and Semantics

8.3 Prepositional phrases and Verb phrase

Jill decided on a couch is ambiguous between two readings:

(i) Jill made a decision while she was on a couch.

(ii) Jill made a decision about a couch.

For (i) *on a couch* as adverbial PP.

For (ii), the appropriate rule is: VP □ V [- pp : on] NP PP [on]

and the logical form of the final VP is:

$(\lambda\ s\ (\mathrm{DECIDES-ON1}\ \ d1\ \ s\ < A\ \ c1\ \ (\mathrm{COUCH}\ \ c1) >))$

Linking Syntax and Semantics

8.3 Prepositional phrases and Verb phrase

The grammar in figure 8.5, we add into rule 8 a value +PRED, and PP acts as a modifier. We add into rule 9 –PERD, in that case the SEM of the PP is simply the SEM of the object.

Figure 8.6 shows the two reading of the VP *decide on a couch*.

The case: the decision is about a couch (with –PRED value) is shown in upper half of the figure, and its SEM is < A c1 COUCH1).

The case in which a decision is made on a couch is shown in the lower haft of the figure. The PP has + PRED, and its SEM is $(\lambda x (\text{ON-LOC1}\ \ x\ < A\ \ c1\ \ \text{COUCH1} >))$

8. (PP PRED + SEM (λ x (?semp x ?semnp))) →
   (*P SEM ?semp*) (NP SEM ?semnp)

9. (PP PRED – PFORM ?pf SEM ?semnp) →
   (*P ROOT ?pf*) (NP SEM ?semnp)

10. (VP VAR ?v SEM (λ ag1 (& (?semvp ag1) (?sempp ?v)))) →
   (*VP SEM ?semvp*) (PP PRED + SEM ?sempp)

11. (VP VAR ?v SEM (λ ag2 (?semv ?v ag2 ?sempp))) →
   (*V[_np_pp:on SEM ?semv*) (PP PRED – PFORM on SEM ?sempp)

12. (VP SEM (λ a1 (?semaux (?semvp a1)))) →
   (AUX SUBCAT ?v SEM ?semaux) (VP VFORM ?v SEM ?semvp)

13. (CNP SEM (λ n1 (& (?semcnp n1) (?sempp n1)))) →
   (*CNP SEM ?semcnp*) (PP PRED + SEM ?sempp)

Head features for PP: PFORM
Head features for VP, CNP: VAR

Grammar 8.5: Rules to handle PPs in verb phrases

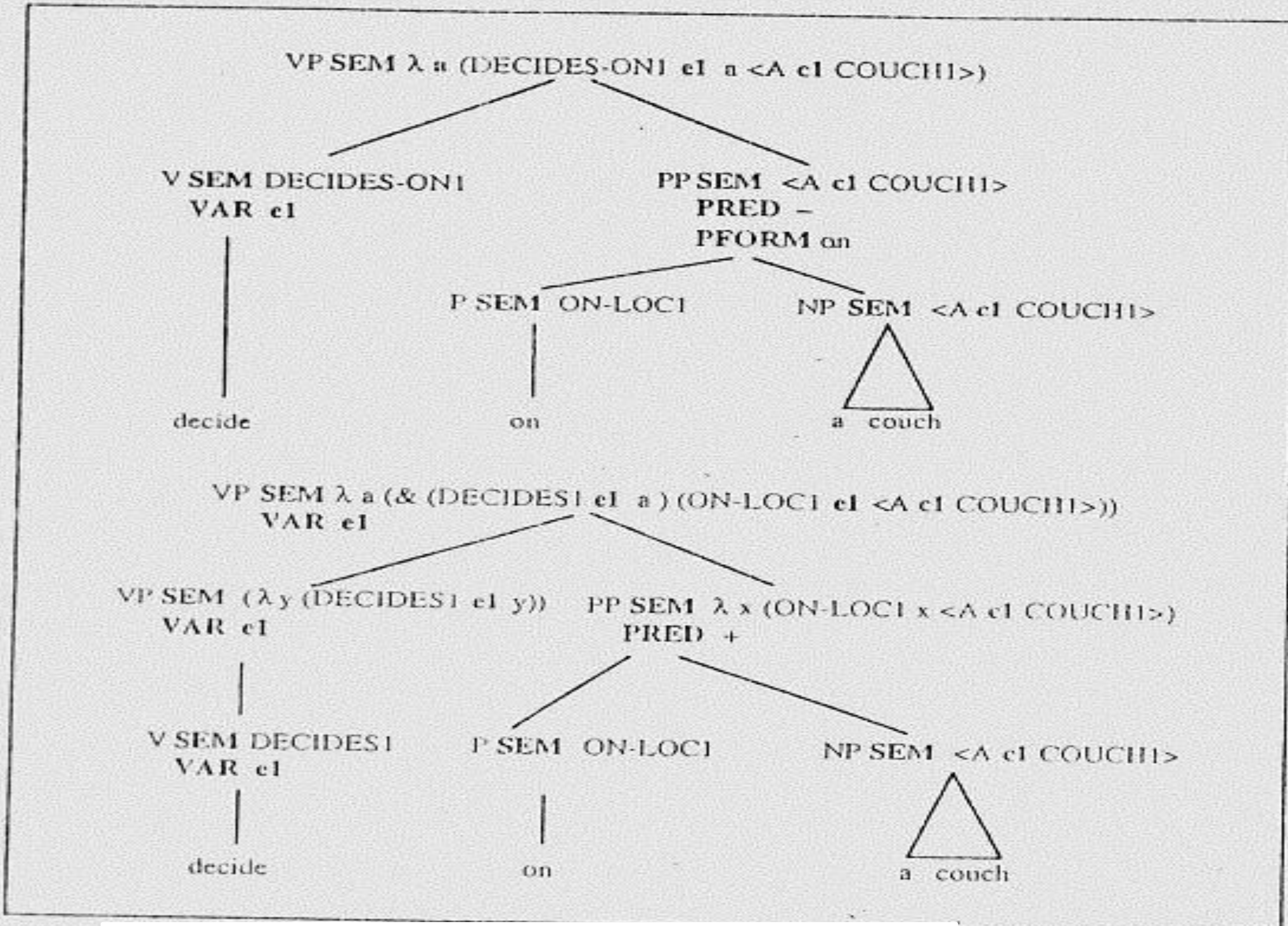

Figure 8.6: Two possible parse trees for the VP decide on a coach

Linking Syntax and Semantics

8.4 Lexicalized Semantic Interpretation and Semantic Roles (study oneself)

Linking Syntax and Semantics

8.5 Handling Simple Questions

Adding rules to interpret wh-terms, inverted sentence, and gap propagation required to handle wh-question.

Example: *some wh- questions*

Who did Jill see?
Who did Jill want to see?
Who did Jill want to win the prize?

The rule was introduced in the Chapter 5 to account for these questions was:

(S INV -) □ (NP WHQ AGR ? a)(S INV +GAP (NP AGR ? a))

To make the semantic interpretation work, we add SEM of the gap:

Linking Syntax and Semantics

8.5 Handling Simple Questions

(S INV – SEM (WH – query ? sems)) □

(NP WH Q AGR ? a SEM ? semnp) (S INV + SEM GAP (NP AGR ? a SEM ? semnp))

Figure 8.7 shows some rules, that are added to handle questions.

The lexical entries for the Wh-words would have to be extended with a SEM, for instance:

Who: (PRO WH {QR} SEM WHO1 AGR { 3s, 3p })

Figure 8.8 is a example , that is the syntactic tree of the question *Who did Jill see?*

14. (S INV – SEM (WH-query ?sems)) →

    (NP WH Q AGR ?a SEM ?semnp)

    (S INV + SEM ?sems GAP (NP AGR ?a SEM ?semnp))

15. (S INV + GAP ?g SEM (?semaux (?semvp ?semnp))) →

    (*AUX AGR ?a SUBCAT ?s SEM ?semaux*)

    (NP AGR ?a GAP – SEM ?sempp)

    (VP VFORM ?s GAP ?g SEM ?semvp)

16. (NP WH Q VAR ?v SEM <WH ?v (?sempro ?v)>) →

    (*PRO WH Q SEM ?sempro*)

Grammar 8.7: Rules to handle simple wh questions

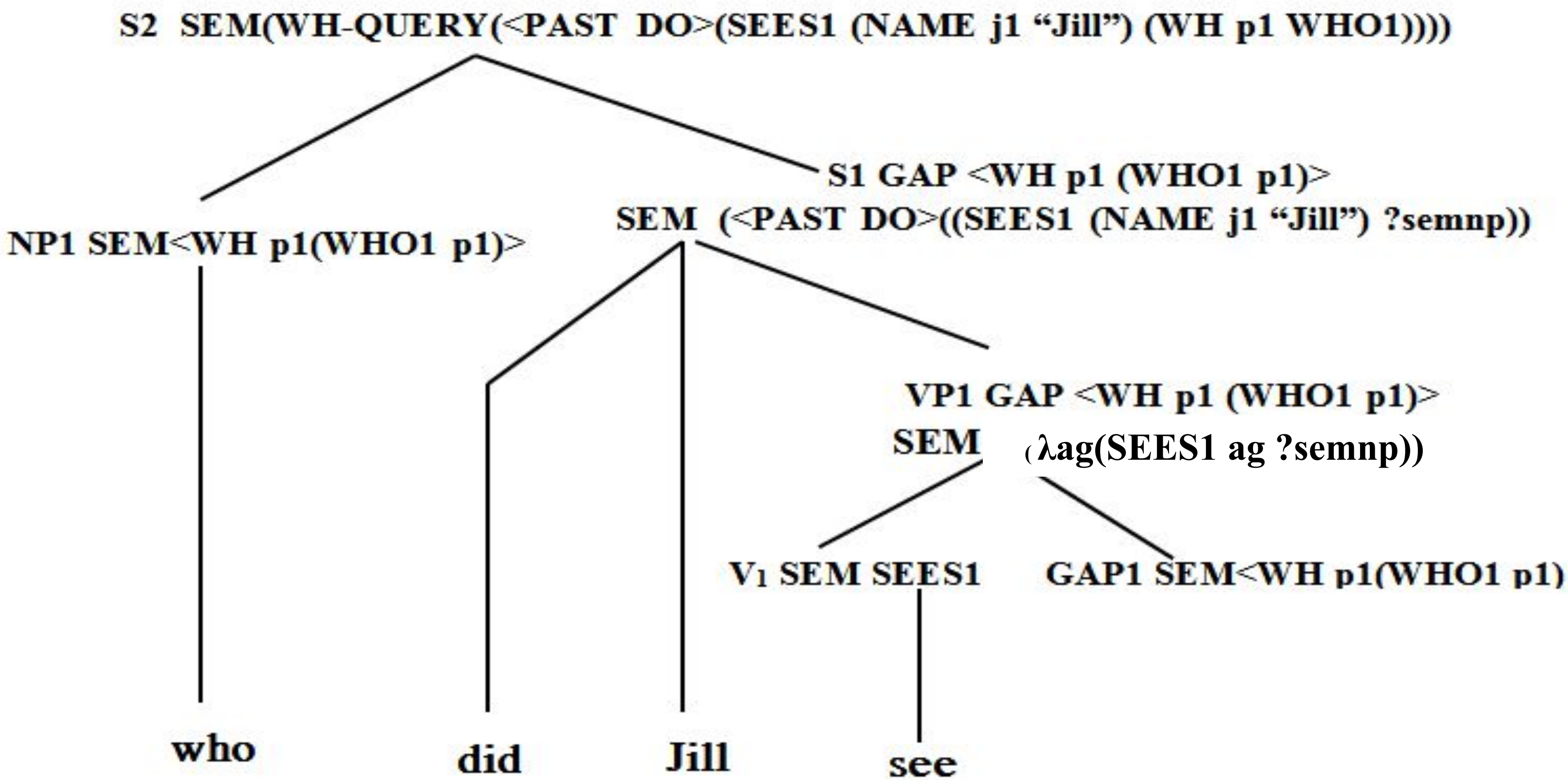

Figure 8.8: The syntactic tree of the question "Who did Jill see?"

Linking Syntax and Semantics

8.5 Handling Simple Questions

Using rule 16 , the word who would be parse as noun phrase

(NP WH Q VAR ? v SEM (WH ? v (? sempro ? v))) □
(PRO WH Q SEM ? sempro)

WHO1 AGR { 3s, 3p}

Applying and combining properties from node Who , we have new node NP1 SEM < WH p1 (WHO1 p1) >.

and rule S1 has the constituents AUX, NP and VP on the right hand side, such as:

Linking Syntax and Semantics

8.5 Handling Simple Questions

(S1 INV + GAP ? g SEM (? semaux (? smvp, ? semnp)) □

(i) (AUX AGR ? a SUBCAT ? s SEM ? semaux)

(ii) (NP AGR ? A GAP – SEM semnp)

(iii) (VP VFORM ? s GAP ? g SEM ? semvp)

We determine SEM of these constituents, such as:

(i)has the SEM < PAST DO >

(ii)has the SEM (NAME j1 “Jill”)

(iii) the rule (VP1 VFORM ? s GAP ? g SEM ? semvp) □

(V SEM ? v GAP ? g SEM ? semvp)

where SEM ? v is SEES1

and GAP ? g = SEM < WH p1 (WHO1 p1) >

SEM ? Semvp = λ ag (? semv ag ? semnp)

Linking Syntax and Semantics

8.5 Handling Simple Questions

where SEM ? v is SEES1

and GAP ? g = SEM < WH p1 (WHO1 p1) >

SEM ? Semvp = λ ag (? semv ag ? semnp)

Reducing to the node S1:

S1 GAP< WH p1(WHO1 p1) > SEM (< PAST DO

(SEES1(NAME j1 “Jill”) ?semnp))

And reducing to the node S2:

SEM (WH – QUERY (< PAST DO > (SEES1 (NAME j1 “Jill”)

(WH p1 WHO1))))

Linking Syntax and Semantics

8.6 Semantic Interpretation Using Feature Unification

The basic idea is to introduce new feature for argument position that earlier would have been filled using lambda reduction.

Example: *The rule 1* in the grammar 8.3 has the form, such as:

(S SEM (? semvp ? semnp)) □ (NP SEM ? semnp) (VP SEM ? semvp)

A new feature SUBJ is added into rule 1, then this rule becomes:
(S SEM ? semvp) □ (NP SEM ? semnp) (VP SUBJ ? semnp SEM ? semvp)

The SEM of the subject is passed into the VP constituent as the SUBJ feature and the SEM equations for the VP insert the subject in the correct position.

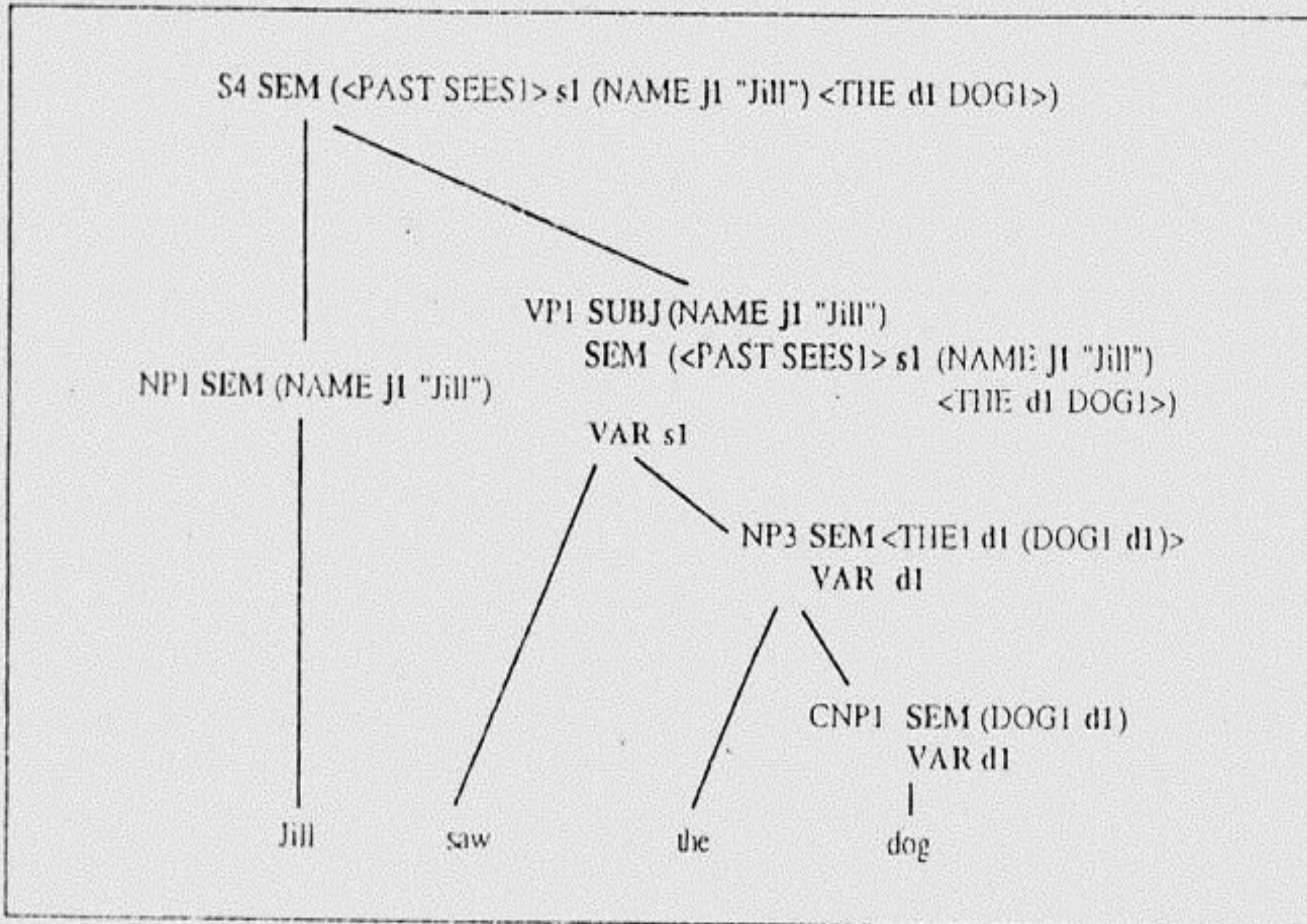

Figure 8.9: The parse tree for *Jill saw the dog* using the SUBJ feature

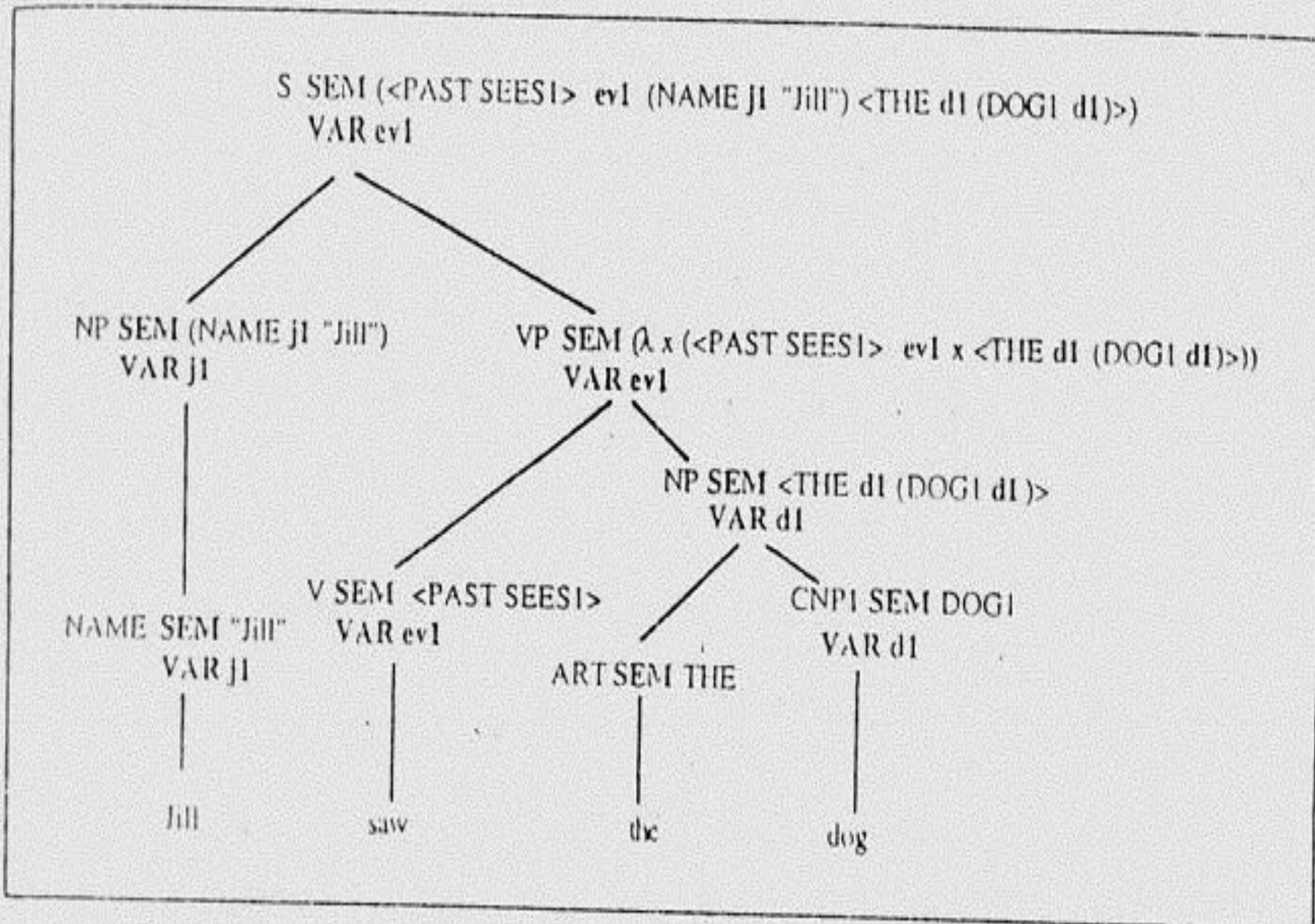

Figure 8.10: The parse of *Jill saw the dog* showing the SEM and VAR features

Linking Syntax and Semantics

8.6 Semantic Interpretation Using Feature Unification

The new version of rule 3 in Grammar 8.3 that does this is

(VP VAR ? v SUBJ ? semsubj SEM (? semv ? v ? semsubj ? semnp)) □ (V [- none] SEM ? semv) (NP SEM ? semnp)

Figure shows how this rule builds the SEM of the sentence

Jill saw the dog. Compare this to the analysis built using Grammar 8.3 shown in Figure 8.6. The differences appear in the treatment of the VP.

Grammar 9.14 is a version of Grammar 8.3 reformulated using this technique

Linking Syntax and Semantics

An advantage of the feature SUBJ technique

- No special mechanism need be introduced to handle semantic interpretation. In particular there is no need to have a lambda reduction step.
- A grammar specified in this form is reversible and hence can be used to generate sentences as well as parse them

But not all lambda expressions can be eliminated using these techniques.

Example: *Sue and Sam saw Jack*

The meaning of the VP must still be a lambda expression

1. (S SEM ?semvp) →
   (NP SEM ?semsubj) (*VP SUBJ ?semsubj SEM ?semvp*)
2. (VP VAR ?v SUBJ ?semsubj SEM (?semv ?v ?semsubj)) →
   (*V[_none] SEM ?semv*)
3. (VP VAR ?v SUBJ ?semsubj SEM (?semv ?v ?semsubj ?semnp)) →
   (*V[_np] SEM ?semv*) (NP SEM ?semnp)
4. (NP VAR ?v SEM (PRO ?v ?sempro)) → (*PRO SEM ?sempro*)
5. (NP VAR ?v SEM (NAME ?v ?semname)) → (*NAME SEM ?semname*)
6. (NP VAR ?v SEM <?semart ?v ?semcnp>) →
   (ART SEM ?semart) (*CNP SEM ?semcnp*)
7. (CNP VAR ?v SEM (?semn ?v)) → (*N SEM ?semn*)

Head features for S, VP, NP, CNP: VAR

Grammar 8.11: A simple grammar with SEM features

Linking Syntax and Semantics

8.7 Generating Sentence from Logical Form

Example: generate a sentence from a logical form by grammar 9.3:

(< PAST SEES1> s1 (NAME j1 “Jill”) <THE d1 (DOG1 d1) >)

The grammar 8.3 has only one rule S. If we try to unify the SEM value in rule 1 with this logical form it will fail.

Rule 1 in the grammar 8.3:

(S SEM (? semvp ? semnp) □ (NP SEM ? semnp) (VP SEM ? semvp)

Linking Syntax and Semantics

8.7 Generating Sentence from Logical Form

The lambda reduction was used to convert the logical form, which was.

((λa (< PAST SEES1 > s1 a < THE d1 (DOG1 d1) >)) (NAME j1 “Jill”))

There is an inverse operation to lambda reduction, called lambda abstraction, that could be to find a match.

There are three possible lambda abstractions of the logical form, namely:

(λ e (<PAST SEES1> e (NAME j1 “Jill”) <THE d1(DOG1 d1) >)
(λ a (< PAST SEES1 > s1 a < THE d1 (DOG1 d1) >))
(λ o (< PAST SEES1 > s1 (NAME j1 “Jill”) o))

Linking Syntax and Semantics

8.7 Generating Sentence from Logical Form

Only second abstraction will yield an appropriate sentence.

- In many ways parsing and realization are very similar processes.
- A realizer starts with the logical form and tries to find a tree to account for it and hence determine to find the words to realize it.

Example: Realizing an S with SEM

(<PAST SEES1> s1(NAME j1 “Jill”) <THE d1(DOG1 d1) >)

Linking Syntax and Semantics

8.7 Generating Sentence from Logical Form

Rule 1 in the grammar 9.14, such as:

(S SEM ? semvp) □

(NP SEM ? semsubj) (VP SUBJ ? semsubj SEM ? semvp)

While SEM semvp is (< PAST SEES1 > s1 (NAME j1 “Jill”) < THE d1 (DOG1 d1) >)

The SEM of NP is unconstrained. So the technique may fail into an infinite loop.

One method for avoiding this problem is to expand the constituents in a different order.

Linking Syntax and Semantics

8.7 Generating Sentence from Logical Form

The algorithm in figure 8.12 gives process to perform realization.

The realization algorithm operates on a list of constituents much like the basic top-down parser described in chapter 3. It continues to rewrite constituents in this list until the list consists only of lexical constituents, at which point the words can be generated.

Consider this algorithm operating with grammar 8.11 and initial input:

(S SEM (< PAST SEES1 > s1 (NAME j1 "Jill") < THE d1 ((DOG1 d1 >))

Initialization: Set L to a list containing the constituent that you wish to generate.

Do until L contains no nonlexical constituents:

1. If L contains a constituent C that is marked as a nonlexical head,
2. Then use a rule in the grammar to rewrite C. Any variables in C that are bound in the rewrite should be instantiated throughout the entire list.
3. Else choose a nonlexical constituent C, giving preference to one whose SEM feature is bound, if one exists. Use a rule in the grammar to rewrite C. Any variables in C that are bound in the rewrite should be instantiated throughout the entire list.

Figure 8.12: A head driven realization algorithm

Linking Syntax and Semantics

8.7 Generating Sentence from Logical Form

- The S constituent is rewritten based on rule1 to produce the following constituent list:

(NP SEM ? semsubj)

(VP SUBJ ? semsubj

SEM (<PAST SEES1> s1(NAME j1 “Jill”) <THE d1(DOG1 d1)>))

Expanding VP based on rule 3. The SEM of VP:

SEM (?semv ?v ?semsubj ?semnp)

As a result of the match, the following variables are bound:

? semv □ < PAST SEES1 >

? v □ s1

? semsubj □ (NAME j1 “Jill”)

? semnp □ < THE d1 (DOG1 d1) >

Linking Syntax and Semantics

8.7 Generating Sentence from Logical Form

Thus we obtain the following list after rewriting the VP and instantiating the variables throughout the list:

(NP SEM (NAME j1 “Jill”)

(V [- np] SEM < PAST SEES1 >

(NP SEM < THE d1 (DOG1 d1) >)

Since there is no non-lexical head, the algorithm now picks any non-lexical constituent with a bound SEM, say first NP. Only rule 5 will match:

(NAME SEM “Jill”)
(V [- np] SEM < PAST SEES1 >)
(NP SEM < THE d1 (DOG1 d1) >)

Linking Syntax and Semantics

8.7 Generating Sentence from Logical Form

The remaining NP is selected next, the rule 6 is matched.

(NAME SEM "Jill")

(V [- np] SEM < PAST SEES1 >)

(ART SEM THE)

(CNP SEM DOG1)

The CNP is selected next, and rewritten as a common noun with SEM DOG1, and the algorithm is completed. The constituent list is now a sequence of the lexical categories:

Linking Syntax and Semantics

8.7 Generating Sentence from Logical Form

The constituent list is now a sequence of the lexical categories:

(NAME SEM "Jill")

(V [- np] SEM < PAST SEES1 >)

(ART SEM THE)

(N SEM DOG1)

It is simple to produce the sentence *Jill saw the dog* from lexical constituents (after the SEMs)

EXERCISE OF CHAPTER 8

1) Simplify following formulas using lambda reduction:

$(\lambda x\ (P\ x))\ A)$

$(\lambda\ x\ (x\ A))\ (\lambda\ y\ (Q\ y)))$

$((\lambda\ x\ ((\lambda\ y\ (P\ y))\ x\ ))\ A)$

2) Using the interpretation rules defined in this chapter and defining any rules that you need, give a detailed trace of the interpretation of the sentence ***The man gave the apple to Bill.*** Give the analysis of each constituent and show its SEM feature.

EXERCISE OF CHAPTER 8

3) Draw the parse trees showing the semantic interpretation for the constituents for following questions. Give the lexical entries showing the SEM feature for each word used that is not defined in this chapter, and define any additional rules needed that are not specified in this chapter..

Who saw the dog ?.

Who did John give the book to ?.