Dependency Parsing and Constituency Parsing

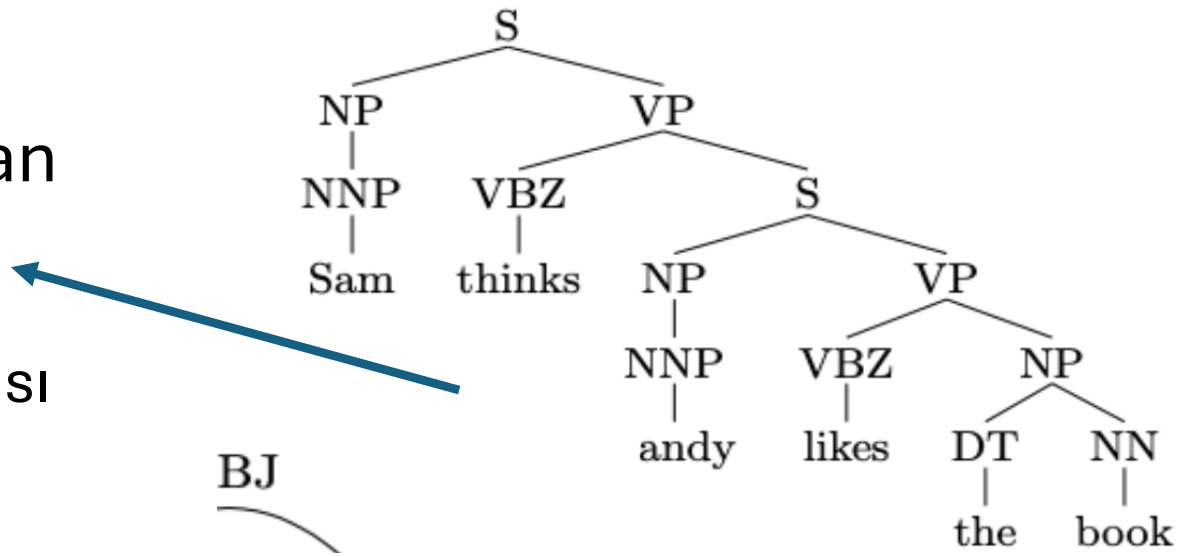
Bağımlı Çözümleme ve Dilbilimsel Yapının Oluşturucuları ile (POS) Çözümleme

Sözdizimsel Yapı (Syntactic Structure) /Constituency and Bağımlı Dilbilgisi Yapısı / Dependency Structure

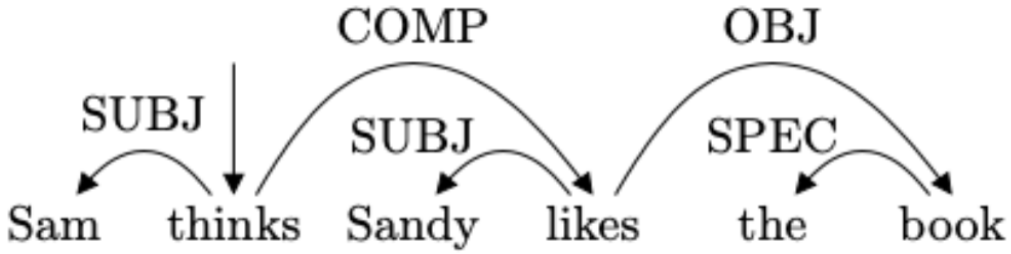
Dilbilgisi yapısına ilişkin farklı iki bakış açısı:

Constituency (Dilbilgisi yapısı oluşturan unsurlara / cümlenin öğelerine göre incelenmesi

- ❖ phrase structure grammar / cümle yapısı dil bilgisi
- ❖ context-free grammars / bağlamdan bağımsız dilbilgisi (CFG)

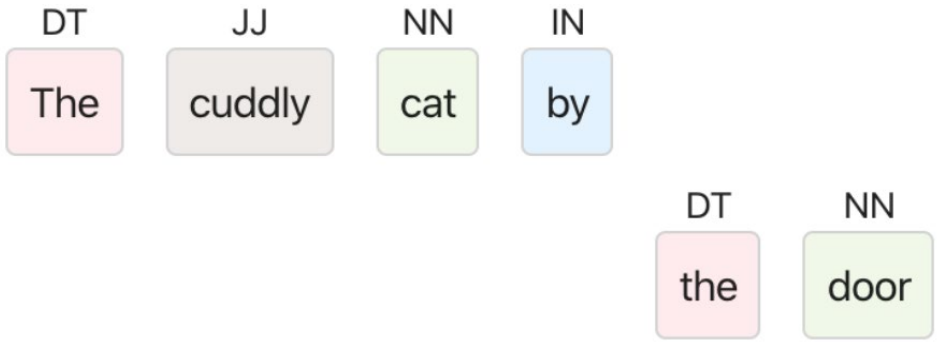


❑ Dependency Parsing / Bağımlı Çözümleme



Constituency (Syntactic) Structure / Sözdizimsel Yapı

- Cümle yapısı sözcükleri iç içe geçmiş olgulara (öğelere) göre düzenler.
  - İşlemlere sözcüklerle başlanır. Bunlar POS (Part of Speech) etiketleri ile belirlenir
    - ✓ the, cuddly, cat, by, the, door
    - ✓ DT, JJ, NN, IN, DT, NN
- Kelimeler birleşir ve ifadeler oluşturulur.
  - The cuddly cat, by, the door
  - NP→DT JJ NN IN NP →DT NN



- İfadeler (phrases) tekrarlı bir şekilde daha büyük ifadeler (phrases) oluşturur.

the cuddly cat, by the door

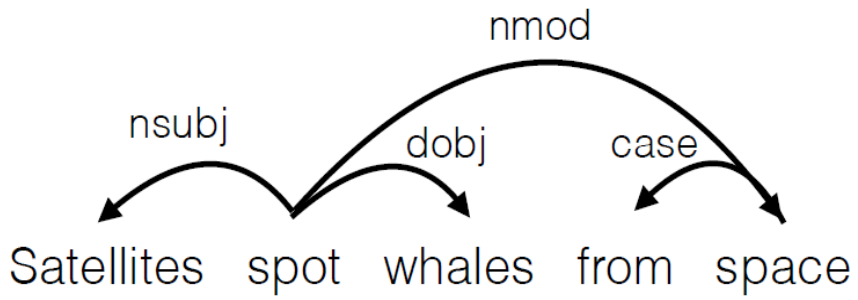
NP PP→IN NP

the cuddly cat by the door

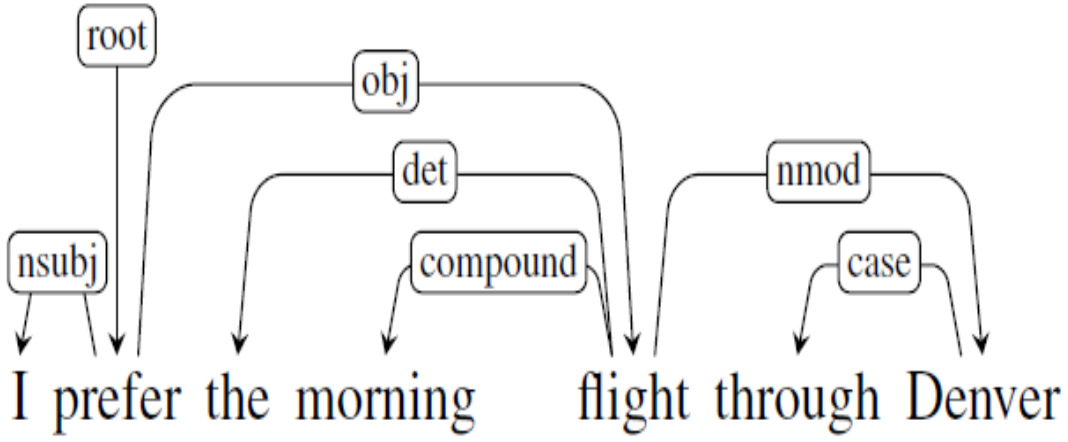
NP→ NP PP

Dependency Structure / Bağımlılık Yapısı

- Hangi kelimelerin hangi diğer kelimelere bağlı olduğunu, değiştirdiğini veya argümanı olduğunu gösterir.

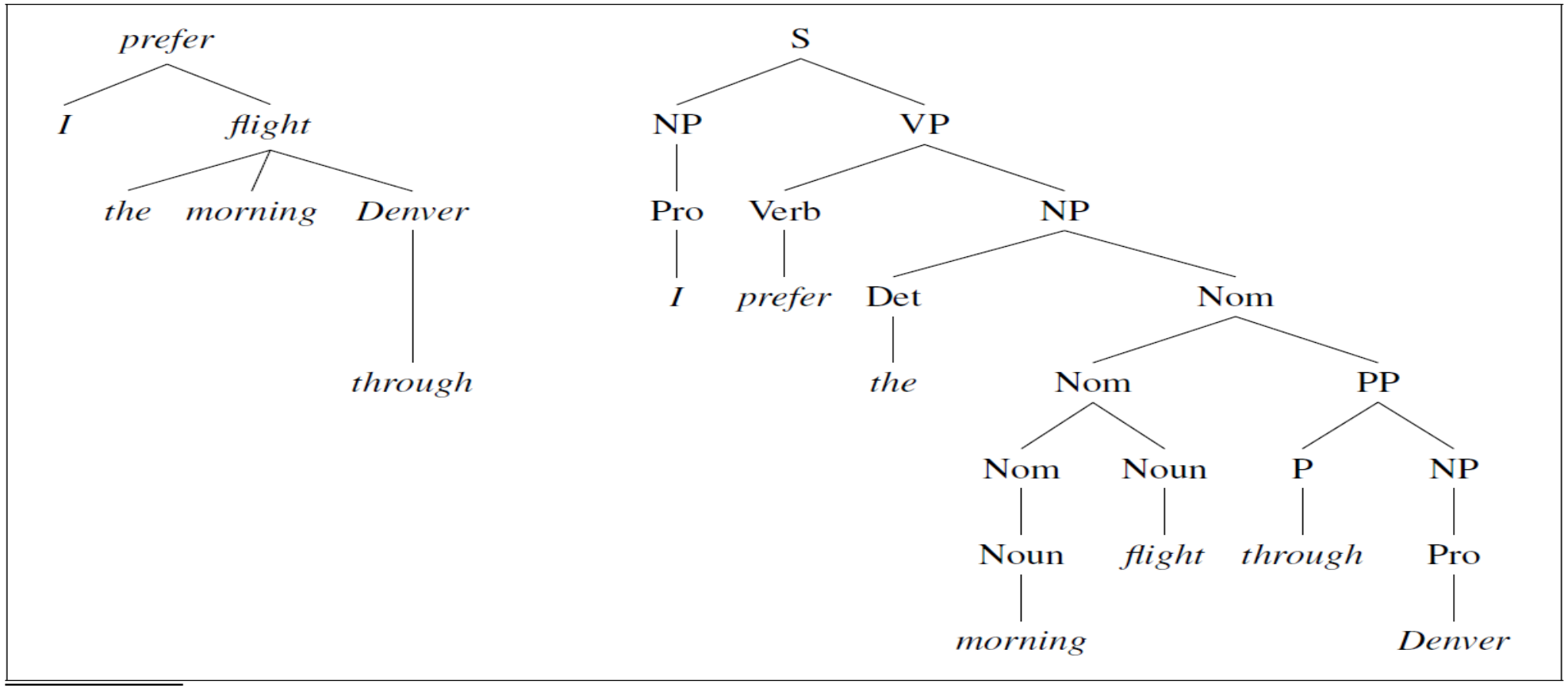


Dependency Parsing (Bağlılık Ayrıştırması / Çözümlemesi)



- Bir cümlenin sözdizimsel yapısı, kelimeler arasındaki yönlendirilmiş ikili dilbilgisi ilişkilerine göre tanımlanır.
  - Sözcükler arasındaki ilişkiler cümlenin üstünde, cümlenin üst (head/ baş) kısımlarından bağımlılara doğru yönlendirilmiş, etiketli tipli yaylarla gösterilir.
  - Bir kök düğümü ağacın kökünü, tüm yapının başını açıkça işaretler.
- Typed dependency structure / tiplenmiş bağlılık yapısı olarak adlandırılır.
  - Zira etiketler sabit bir dilbilgisi ilişkileri birikiminden (envanter) çıkarılıyor.

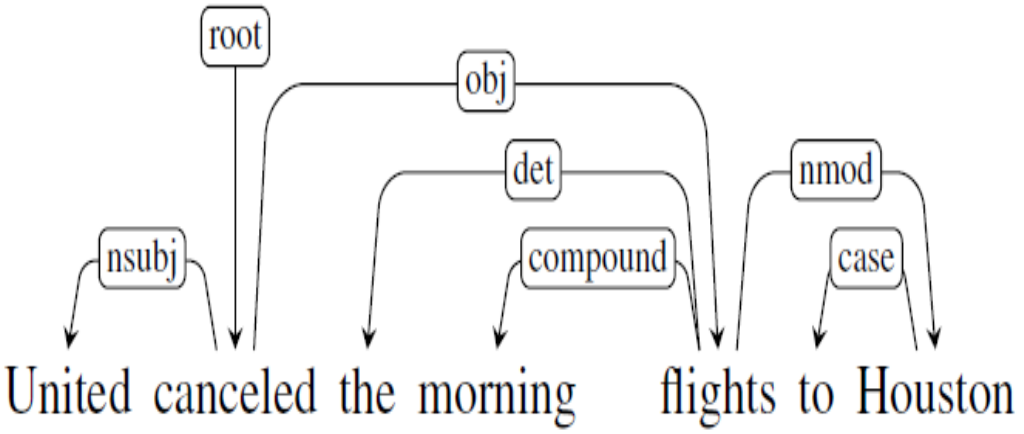
Dependency ve Constituent Parsing (Ayrıştırma)



Dependency Relations

Clausal Argument Relations	Description
NSUBJ	Nominal subject
OBJ	Direct object
IOBJ	Indirect object
CCOMP	Clausal complement
Nominal Modifier Relations	**Description**
NMOD	Nominal modifier
AMOD	Adjectival modifier
APPOS	Appositional modifier
DET	Determiner
CASE	Prepositions, postpositions and other case markers
Other Notable Relations	**Description**
CONJ	Conjunct
CC	Coordinating conjunction

Universal Dependency /Evrensel Bağımlılık projesinden bir örnek (2021)



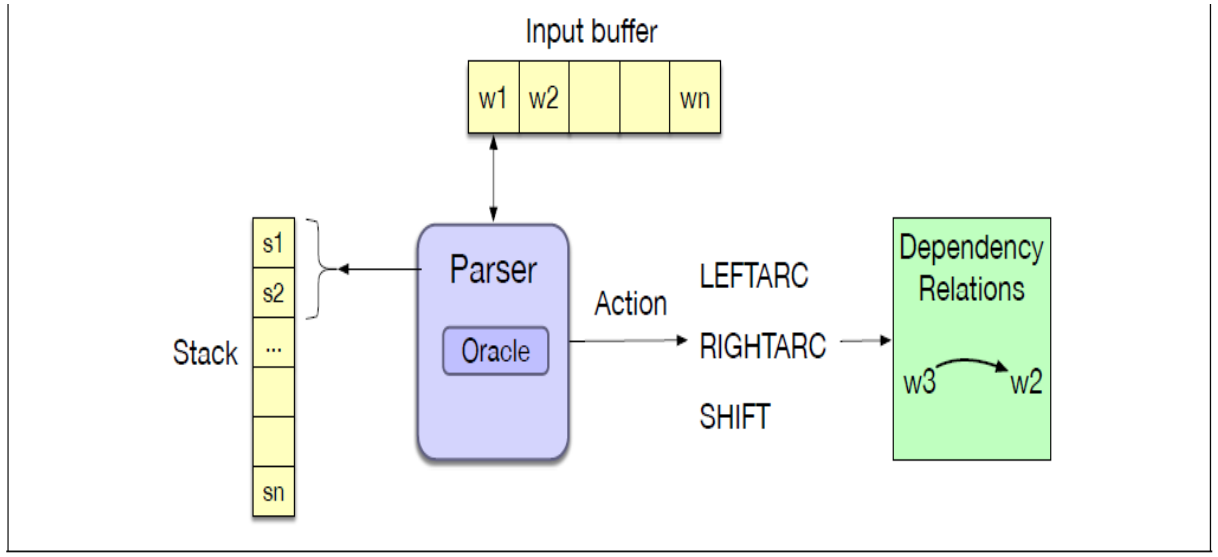
- Sık kullanılan ilişkilerin temel kümesi iki kümeye ayrılır.
  - Genellikle bir fiile göre sözdizimsel rolleri tanımlayan cümle ilişkileri
  - Sözcüklerin başlarını (head) değiştirebilmesini kategorilere ayıran değiştirici (modifier) ilişkiler

Clausal relations NSUBJ and DOBJ identify the subject and direct object of the predicate **cancel**, while the NMOD, DET, and CASE relations denote modifiers of the nouns *flights* and *Houston*.

Evrensel Bağlılık (Universal Grammar) İlişki Örnekleri

Relation	Examples with *head* and **dependent**
NSUBJ	**United** *canceled* the flight.
OBJ	United *diverted* the **flight** to Reno.
	We *booked* her the first **flight** to Miami.
IOBJ	We *booked* **her** the flight to Miami.
COMPOUND	We took the **morning** *flight*.
NMOD	*flight* to **Houston**.
AMOD	Book the **cheapest** *flight*.
APPOS	*United*, a **unit** of UAL, matched the fares.
DET	**The** *flight* was canceled.
	Which *flight* was delayed?
CONJ	We *flew* to Denver and **drove** to Steamboat.
CC	We flew to Denver **and** *drove* to Steamboat.
CASE	Book the flight **through** *Houston*.

Transition-Based Dependency Parsing



function DEPENDENCYPARSE(*words*) **returns** dependency tree

state ← {[root], [*words*], [] } ; initial configuration
while *state* **not final**
t ← ORACLE(*state*) ; choose a transition operator to apply
state ← APPLY(*t*, *state*) ; apply it, creating a new state
return *state*

- Çözümleyici (parser) cümleyi soldan sağa doğru işleme sokar.
- Cümlenin öğelerini arabellekten (buffer) yığına atanır.
- Her bir işleme adımında yığının en üstündeki **iki öğe** incelenir.
- Çözümleyici (oracle) ayrıştırmayı oluşturmak için hangi dilbilgisi kuralığının uygulanacağına karar verir.
- Olası geçişler, giriş cümlesi üzerinde soldan sağa her bir geçiş için kelimeleri inceler ve bir bağımlılık ağacı oluşturur.
- Bunlar gerçekleştirilebilecek **sezgisel** eylemlerdir.

Algoritmanın «Transition» Operatörleri

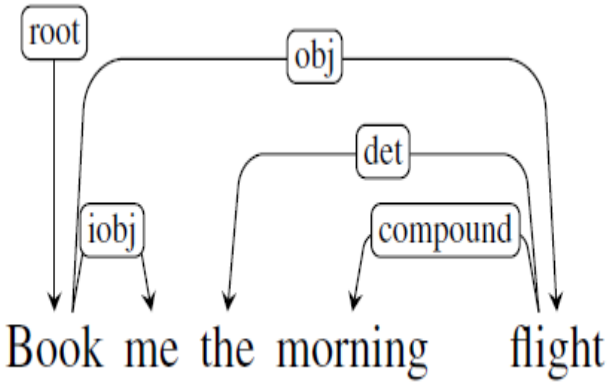
Algoritma üç farklı operatör kullanır:

- **LEFTARC**: Yığının en üstündeki kelime ile ikinci kelime arasında (head-dependent) başa-bağlı bir ilişki olduğu doğrulanır ve yığındaki ikinci kelime yığından çıkarılır.
- **RIGHTARC:** Yığındaki ikinci kelime ile en üstteki kelime arasında başa-bağlı (head-dependent) bir ilişki olduğu doğrulanır ve yığının **en üstündeki** kelime yığından çıkarılır.
- **SHIFT:** Kelime giriş belleğinden kaldırılır ve yığına atılır (push).

Not: LEFTARC ve RIGHTARC operatörleri «reduce» işlemi gerçekleştirir.

Algoritmanın işleyişi «shift –reduce parsing» ile eşdeğerdir.

Transition Dependency Çözümleme Örneği




```
function DEPENDENCYPARSE(words) returns dependency tree

  state ← {[root], [words], [] } ; initial configuration
  while state not final
    t ← ORACLE(state)       ; choose a transition operator to apply
    state ← APPLY(t, state) ; apply it, creating a new state
  return state
```

Stack	Word List	Relations
[root, book, me]	[the, morning, flight]	

«me» sözcüğü dahil yığına eklenenler

Stack	Word List	Relations
[root, book]	[the, morning, flight]	(book → me)

RIGHTARC için doğru operatör belirlenmelidir. «me» kelimesinin başı /head olarak «book» atanır ve «me» yığından çekilir (pop).

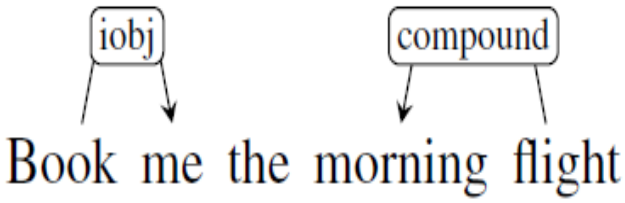
Stack	Word List	Relations
[root, book, the, morning, flight]	[]	(book → me)

SHIFT operatörü birkaç kere uygulanır ve adım adım her sözcük sıra ile yığına atanır.

Çözümleme Örneği / Transition Dependency (devam)

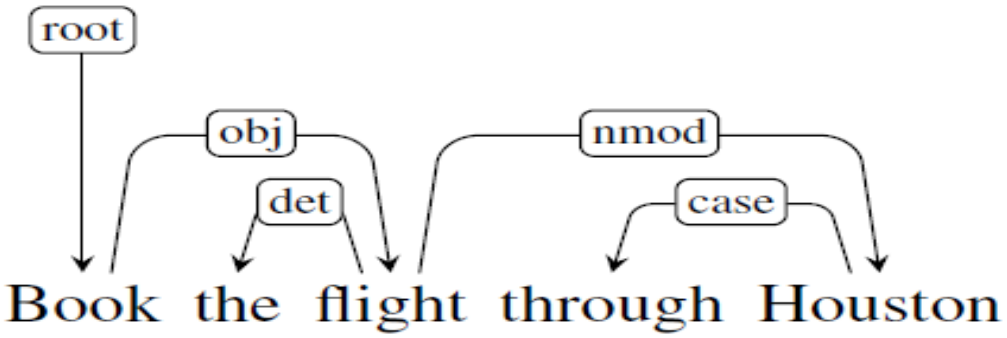
Stack	Word List	Relations
[root, book, the, flight]	[]	(book → me) (morning ← flight)

Tüm sözcükler yığına geçtiği için artık tüm işlemler sol tarafta yapılacaktır.
Bu işlemler uygun reduce operatörleri ile gerçekleşir.
LEFTARC operatörü kullanılarak yandaki duruma erişilir.



Sonuç olarak çözümleme yandaki gibi elde edilecektir.

- Etiketlenmiş ağaçların (labeled tree) oluşturulması için parametrik olarak verilmeleri gerekir.
- LEFTARC ve RIGHTARC operatörlerinin bağlılık etiketleri (dependency labels) LEFTARC (NSUBJ) ve RIGHTARC(OBJ) olarak betimlenecektir.



- LEFTARC: Yığının en üstündeki kelime ile ikinci kelime arasında (head-dependent) başa-bağlı bir ilişki olduğu doğrulanır ve yığındaki ikinci kelime yığından çıkarılır.
- RIGHTARC: Yığındaki ikinci kelime ile en üstteki kelime arasında başa-bağlı (head-dependent) bir ilişki olduğu doğrulanır ve yığının **en üstündeki** kelime yığından çıkarılır.

1. ne LEFTARC ne de RIGHTARC yazılabilir. Çünkü bir bağlılık etiketi içermez.

3. Başa bağlılık oluşturmak için LEFTARC seçilir

4.

Stack	Word buffer	Relations
[root, book, flight]	[through, Houston]	(the ← flight)

Step	Stack	Word List	Predicted Action
0	[root]	[book, the, flight, through, houston]	SHIFT
1	[root, book]	[the, flight, through, houston]	SHIFT
2	[root, book, the]	[flight, through, houston]	SHIFT
3	[root, book, the, flight]	[through, houston]	LEFTARC
4	[root, book, flight]	[through, houston]	SHIFT
5	[root, book, flight, through]	[houston]	SHIFT
6	[root, book, flight, through, houston]	[]	LEFTARC
7	[root, book, flight, houston]	[]	RIGHTARC
8	[root, book, flight]	[]	RIGHTARC
9	[root, book]	[]	RIGHTARC
10	[root]	[]	Done

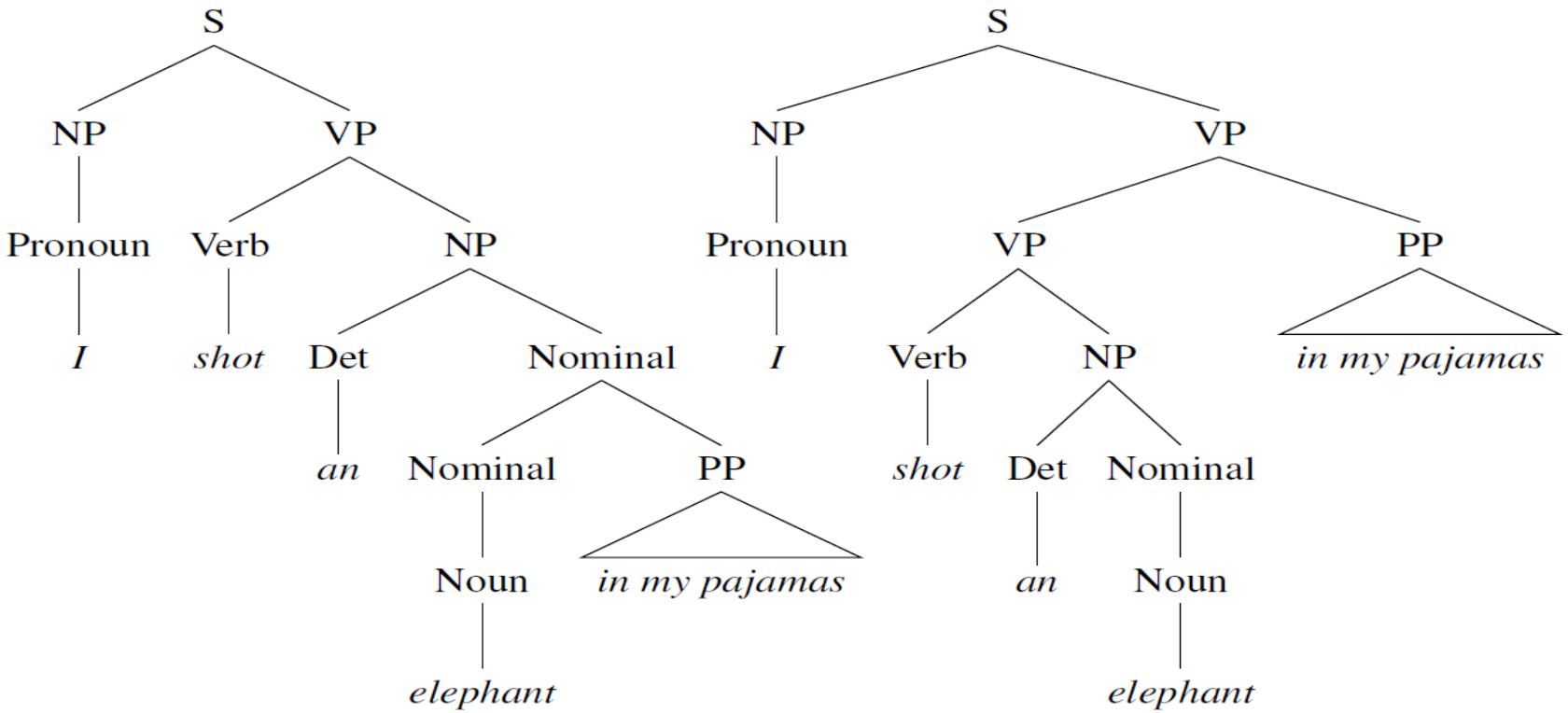
«Dependency Parsing» Algoritmasının Adım Adım İşleyişi

Step	Stack	Word List	Action	Relation Added
0	[root]	[book, me, the, morning, flight]	SHIFT	
1	[root, book]	[me, the, morning, flight]	SHIFT	
2	[root, book, me]	[the, morning, flight]	RIGHTARC	(book → me)
3	[root, book]	[the, morning, flight]	SHIFT	
4	[root, book, the]	[morning, flight]	SHIFT	
5	[root, book, the, morning]	[flight]	SHIFT	
6	[root, book, the, morning, flight]	[]	LEFTARC	(morning ← flight)
7	[root, book, the, flight]	[]	LEFTARC	(the ← flight)
8	[root, book, flight]	[]	RIGHTARC	(book → flight)
9	[root, book]	[]	RIGHTARC	(root → book)
10	[root]	[]	Done	

Constituency Parsing /Syntactic Parsing

Grammar	Lexicon
$S \rightarrow NP\ VP$	$Det \rightarrow that \mid this \mid the \mid a$
$S \rightarrow Aux\ NP\ VP$	$Noun \rightarrow book \mid flight \mid meal \mid money$
$S \rightarrow VP$	$Verb \rightarrow book \mid include \mid prefer$
$NP \rightarrow Pronoun$	$Pronoun \rightarrow I \mid she \mid me$
$NP \rightarrow Proper\text{-}Noun$	$Proper\text{-}Noun \rightarrow Houston \mid NWA$
$NP \rightarrow Det\ Nominal$	$Aux \rightarrow does$
$Nominal \rightarrow Noun$	$Preposition \rightarrow from \mid to \mid on \mid near \mid through$
$Nominal \rightarrow Nominal\ Noun$	
$Nominal \rightarrow Nominal\ PP$	
$VP \rightarrow Verb$	
$VP \rightarrow Verb\ NP$	
$VP \rightarrow Verb\ NP\ PP$	
$VP \rightarrow Verb\ PP$	
$VP \rightarrow VP\ PP$	
$PP \rightarrow Preposition\ NP$	

Belirsizlik (Ambiguity)



Cümle dilbilgisi kurallarına göre iki farklı şekilde çözümlenebildiği için **belirsizdir** ve istenilen durum değildir.

Bağlamdan Bağımsız Dilbilgisi (CFG) Tanımı (Tekrar)

- G dilbilgisi 4 parametreli olarak (4-tuple) tanımlanır. Bunlar:

  N, terminal olmayan sembollerin sonlu kümesi

  $\sum$ terminal sembollerinin sonlu kümesi (N ile ortak elemanı yoktur)

  P kuralların sonlu dizilişidir ve $A \rightarrow \beta$ formundadır.

  A bir nonterminal semboldür,

  β tüm sembollerin herhangi bir dizilişi olan dizgidir.

  $(\sum \cup N)^*$ olarak gösterilir.

  S başlangıç sembolüdür ve bir nonterminaldir.

Bağlamdan Bağımsız Dilbilgisi (CFG) Tanımı (Tekrar)

- A, B ve S gibi büyük harfler terminal olmayan sembolleri gösterir.
- S başlangıç sembolüdür ve nonterminaldir.
- α, β, γ gibi küçük küçük harfleri ($\sum \cup N)^*$ dan türetilmiş dizgileri gösterir.
- u, v, w gibi küçük Latin harfleri terminal sembollerinin bir dizilişi olan dizgileri gösterir.
- Bir G dilbilgisi tarafından oluşturulan L(G) dili, S başlangıç non terminalinden başlayıp adım adım türetmelerle verilen giriş cümlesinin elde edilmesi dir.

$$L(G) = \{ w \mid w \in \Sigma^* \quad ve \quad S \Rightarrow w^* \}$$

Olasılıksal Bağlamdan Bağımsız Dilbilgisi Probabilistic Context Free Grammar (PCFG)

- CFG, bir cümlenin tanımlanan dilbilgisi kurallarına ait olup olmadığını belirler.
- PCFG aynı cümle için olasılıklara bağlı olarak farklı çözümleme mekanizmaları sunar.

S	→	NP	VP	1.0
VP	→	Vi		0.3
VP	→	Vt	NP	0.5
VP	→	VP	PP	0.2
NP	→	DT	NN	0.8
NP	→	NP	PP	0.2
PP	→	IN	NP	1.0

Vi	→	sleeps	1.0
Vt	→	saw	1.0
NN	→	man	0.1
NN	→	woman	0.1
NN	→	telescope	0.3
NN	→	dog	0.5
DT	→	the	1.0
IN	→	with	0.6
IN	→	in	0.4

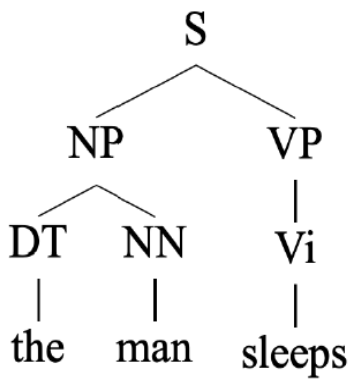
Bir G dilbilgisi G= (N,Σ, P ,S) ile verildiğinde

P: $\alpha \rightarrow \beta$ verildiğinde $q(\alpha \rightarrow \beta) \geq 0$ olmak üzere
Herhangi bir $X \in N$ *için*

$$\sum_{\alpha \rightarrow \beta : \alpha = X} q(\alpha \rightarrow \beta) = 1$$

Probabilistic context-free grammars (PCFGs)

For any derivation (parse tree) containing rules:
$\alpha_1 \rightarrow \beta_1, \alpha_2 \rightarrow \beta_2, \ldots, \alpha_l \rightarrow \beta_l$, the probability of the parse is: $\prod_{i=1}^{l} q(\alpha_i \rightarrow \beta_i)$



R				q
S	→	NP	VP	1.0
VP	→	Vi		0.3
VP	→	Vt	NP	0.5
VP	→	VP	PP	0.2
NP	→	DT	NN	0.8
NP	→	NP	PP	0.2
PP	→	IN	NP	1.0

Vi	→	sleeps	1.0
Vt	→	saw	1.0
NN	→	man	0.1
NN	→	woman	0.1
NN	→	telescope	0.3
NN	→	dog	0.5
DT	→	the	1.0
IN	→	with	0.6
IN	→	in	0.4

$$P(t) = q(\text{S} \rightarrow \text{NP VP}) \times q(\text{NP} \rightarrow \text{DT NN}) \times q(\text{DT} \rightarrow \text{the})$$
$$\times q(\text{NN} \rightarrow \text{man}) \times q(\text{VP} \rightarrow \text{Vi}) \times q(\text{Vi} \rightarrow \text{sleeps})$$
$$= 1.0 \times 0.8 \times 1.0 \times 0.1 \times 0.3 \times 1.0 = 0.024$$

Why do we want $\sum_{\alpha \rightarrow \beta : \alpha = X} q(\alpha \rightarrow \beta) = 1$?

Penn Treebank

Part-of-speech tagset

CC	Coordinating conj.	TO	infinitival *to*
CD	Cardinal number	UH	Interjection
DT	Determiner	VB	Verb, base form
EX	Existential there	VBD	Verb, past tense
FW	Foreign word	VBG	Verb, gerund/present pple
IN	Preposition	VBN	Verb, past participle
JJ	Adjective	VBP	Verb, non-3rd ps. sg. present
JJR	Adjective, comparative	VBZ	Verb, 3rd ps. sg. present
JJS	Adjective, superlative	WDT	Wh-determiner
LS	List item marker	WP	Wh-pronoun
MD	Modal	WP$	Possessive *wh*-pronoun
NN	Noun, singular or mass	WRB	Wh-adverb
NNS	Noun, plural	#	Pound sign
NNP	Proper noun, singular	$	Dollar sign
NNPS	Proper noun, plural	.	Sentence-final punctuation
PDT	Predeterminer	,	Comma
POS	Possessive ending	:	Colon, semi-colon
PRP	Personal pronoun	(	Left bracket character
PP$	Possessive pronoun	)	Right bracket character
RB	Adverb	"	Straight double quote
RBR	Adverb, comparative	‘	Left open single quote
RBS	Adverb, superlative	“	Left open double quote
RP	Particle	’	Right close single quote
SYM	Symbol	”	Right close double quote

https://surdeanu.cs.arizona.edu/mihai/teaching/ista555-fall13/readings/PennTreebankConstituents.html

Penn Treebank

Phrasal categories

ADJP	Adjective phrase
ADVP	Adverb phrase
NP	Noun phrase
PP	Prepositional phrase
S	Simple declarative clause
SBAR	Subordinate clause
SBARQ	Direct question introduced by *wh*-element
SINV	Declarative sentence with subject-aux inversion
SQ	Yes/no questions and subconstituent of SBARQ excluding *wh*-element
VP	Verb phrase
WHADVP	Wh-adverb phrase
WHNP	Wh-noun phrase
WHPP	Wh-prepositional phrase
X	Constituent of unknown or uncertain category
*	"Understood" subject of infinitive or imperative
0	Zero variant of *that* in subordinate clauses
T	Trace of wh-Constituent