의존구문분석 (Dependency Parsing)

남궁영
한국해양대학교 컴퓨터공학과
자연언어처리실험실
young_ng@kmou.ac.kr

2019. 04. 17.

국립 한국해양대학교
KOREA MARITIME AND OCEAN UNIVERSITY

Content

Content

Syntactic Parsing

Basic Theory of Syntactic Parsing

Methodology of Syntactic Parsing

I Basic Theory of Syntactic Parsing

❖ The process of **analyzing the construction of a sentence** by recognizing a sentence and assigning a syntactic structure to it.

- Input: a **Sentence**(String) or Grammar
- Output: a **Parse trees**

❖ Two views of linguistic structure:

- **Phrase structure**
  - ordering and organizing words into nested **constituents**
- **Dependency structure**
  - analyzing the **relation** between head and modifier

I Methodologies of Syntactic Parsing (1/2)

Constituency Parsing

- **Phrase structure** organizes words into nested constituents
  - Rules
  - Terminals & Non-terminals

[Rules]

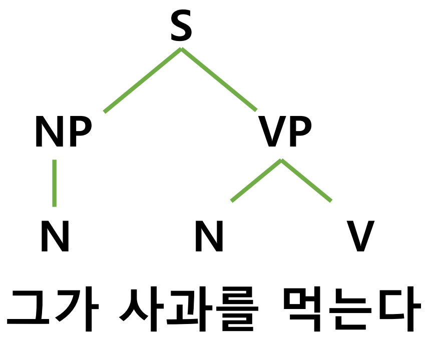
S → NP VP

VP → N V

NP → N

…

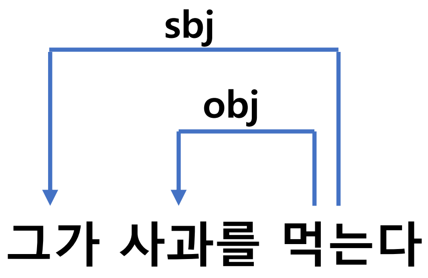
[Phrase Structure Trees]



Dependency Parsing

- **Dependency structure** shows which words depend on (modify or are arguments of) which other words.
  - nodes: word-tokens
  - links: **dependency relations** between words

[Dependency Trees]



| C. D. Manning. 2017. cs224n-2017-lecture6. pp. 3-5
https://web.stanford.edu/class/archive/cs/cs224n/cs224n.1174/lectures/cs224n-2017-lecture6.pdf

I Methodologies of Syntactic Parsing (2/2)

Constituency Parsing

- **Not flexible with word-orders**
- Language dependent
- No semantic information

Dependency Parsing

- Suitable for **free word order(FWO) languages**
  - constituent of the structure is quite fluid
  - frequent omission is occurred

| J. D. Choi. 2009. Dependency Parsing. p.4
https://www.slideshare.net/jchoi7s/dependency-parsing-1968241

| D. Payne. 2009. Human Language Technology-Sentence Grammar. p.56

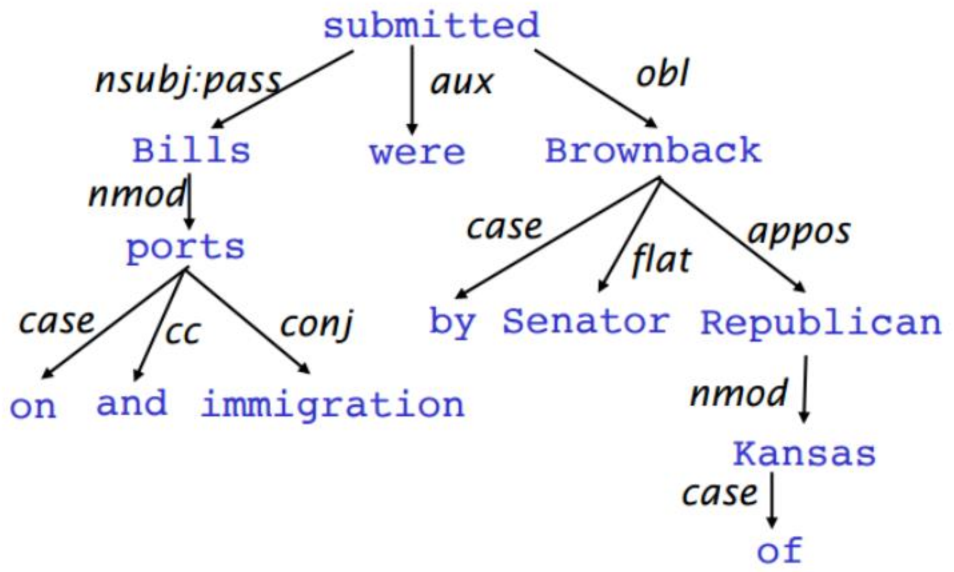
Dependency Parsing

Basic Theory of Dependency Parsing
Dependency Parsing Algorithm
Dependency Parsing Resources
Universal Dependency
Evaluation Metrics

II Basic Theory of Dependency Parsing

❖ Dependency syntax postulates that syntactic structure consists of relations between lexical items, normally binary asymmetric relations (“arrows”) called dependencies

{ Head }	**{ Dependent }**
governor	modifier
superior	inferior
regent	subordinate



❖ Conditions: **Connected / Acyclic / Single-head**

- Only one word is a dependent of ROOT

Ⅱ Basic Theory of Dependency Parsing

❖ Definitions

$\boldsymbol{L = \{l_1, l_2, \dots, l_m\}}$: Arc label set

$\boldsymbol{X = x_0, x_1, \dots, x_n}$: Input sentence

$\boldsymbol{Y}$: Dependency Graph/Tree

$$\boldsymbol{(i, j, k) \in Y \ indicates \ x_i \xrightarrow{l_k} x_j}$$

II Basic Theory of Dependency Parsing

❖ **Projectivity**: A main characteristic of dependency trees

Non-projective

❖ with crossing arcs

❖ good for free word order languages

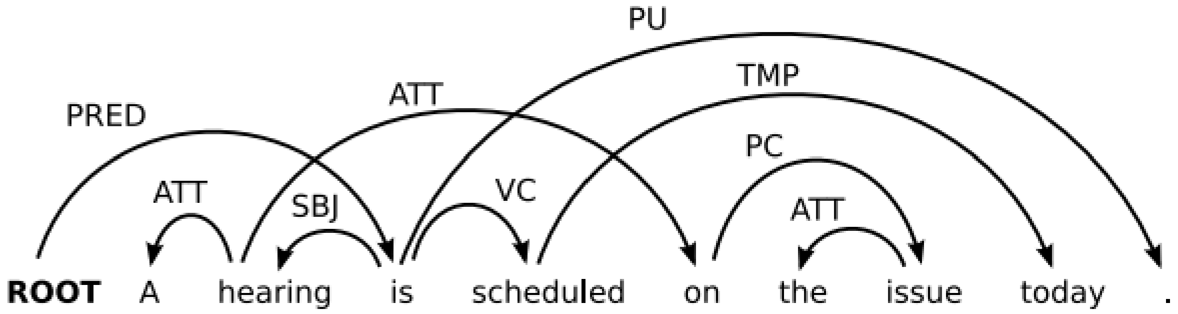
❖ good for long distance dependencies.

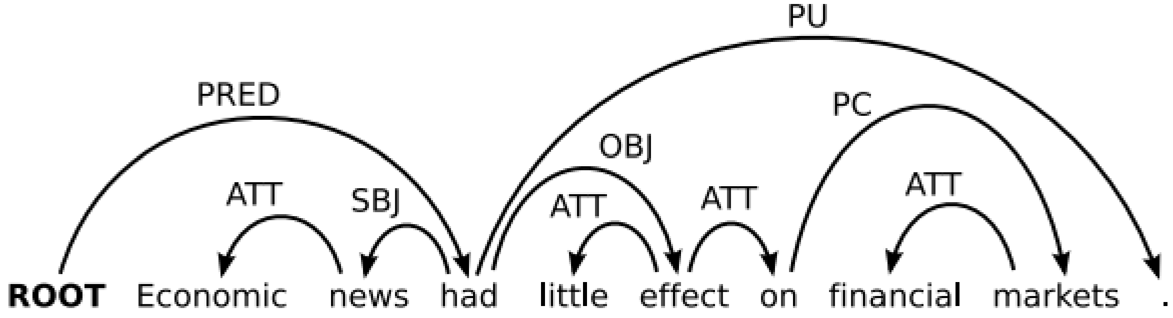
Projective

❖ does not contain any crossing arcs

if $i \rightarrow j$, then $i \rightarrow * \, i'$ for any i'
such that $i < i' < j$ or $j < i' < i$.

❖ many dependency parsing algorithms can only handle projective trees





| Kubler, Nivre, and McDonald. 2009. Dependency Parsing.

II Basic Theory of Dependency Parsing

❖ Supervised dependency parsing

- Learning
  - Given a training set ***D*** of sentences (annotated with dependency graphs), induce a parsing model ***M*** that can be used to parse new sentences.

- Parsing
  - Given a parsing model ***M*** and a sentence ***S***, derive the optimal dependency graph ***G*** for ***S*** according to ***M***

⟹ the algorithms used to learn the model from data and parse a new sentence with the model

| S. Kubler, et al. Dependency parsing, Morgan & Claypool, 2009.

II Dependency Parsing Algorithm

Graph-based Model

- ❖ Define a space of candidate dependency trees for a sentence
- ❖ Find the best parse for the entire sentence

- ❖ Eisner(1996)
- ❖ Collins et al. (1999)
- ❖ McDonald et al. (2005)

Transition-based Model

- ❖ Define a transition system for mapping a sentence to its dependency tree
- ❖ Learning: induce a model for predicting the next state transition, given the transition history
- ❖ Parsing: construct the optimal transition sequence, given the induced model

- ❖ Yamada and Matsumoto (2003)
- ❖ Nivre and Scholz (2004)

Ⅱ Dependency Parsing Algorithm

Graph-based Model

❖ Chart-based Parsing

- projective
- Eisner's Algorithm

❖ Maximum Spanning Tree(MST)

- non-projective
- Chu-Liu-Edmonds' Algorithm

Transition-based Model

❖ Shift-Reduce Parsing

- usually produce projective parses
- Arc-standard
- Arc-eager

GRAPH-BASED

Ⅱ Dependency Parsing Algorithm

❖ **Define a space of candidate dependency trees for a sentence**

- Learning: induce a model for **scoring** an entire tree
- Parsing: find a tree with the **highest score**, given the induced model

Factor the weight/score graphs by subgraphs

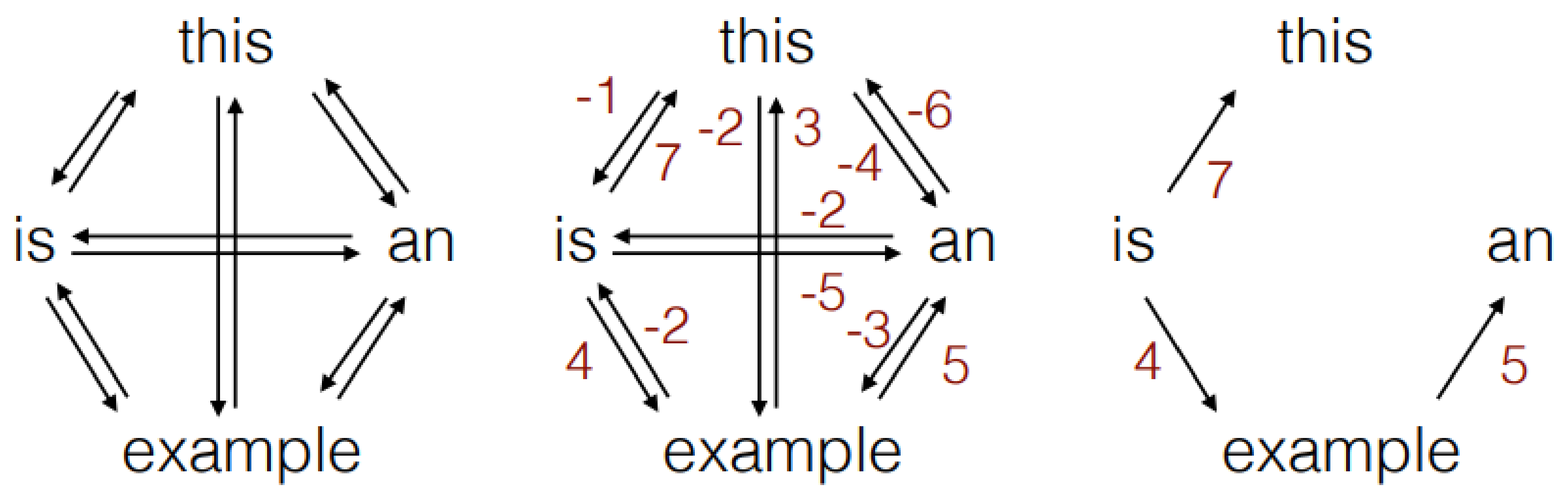
$$Y^* = \underset{Y \in \Phi(X)}{argmax} \sum_{(x_i \to x_j) \in Y} score(x_i \to x_j)$$

- Finding dependency tree with highest score = finding MST in directed graphs

II Dependency Parsing Algorithm

❖ Chu-Liu-Edmond's Algorithm

- Express sentence as fully connected directed graph
- Score each edge independently
- Find maximum spanning tree



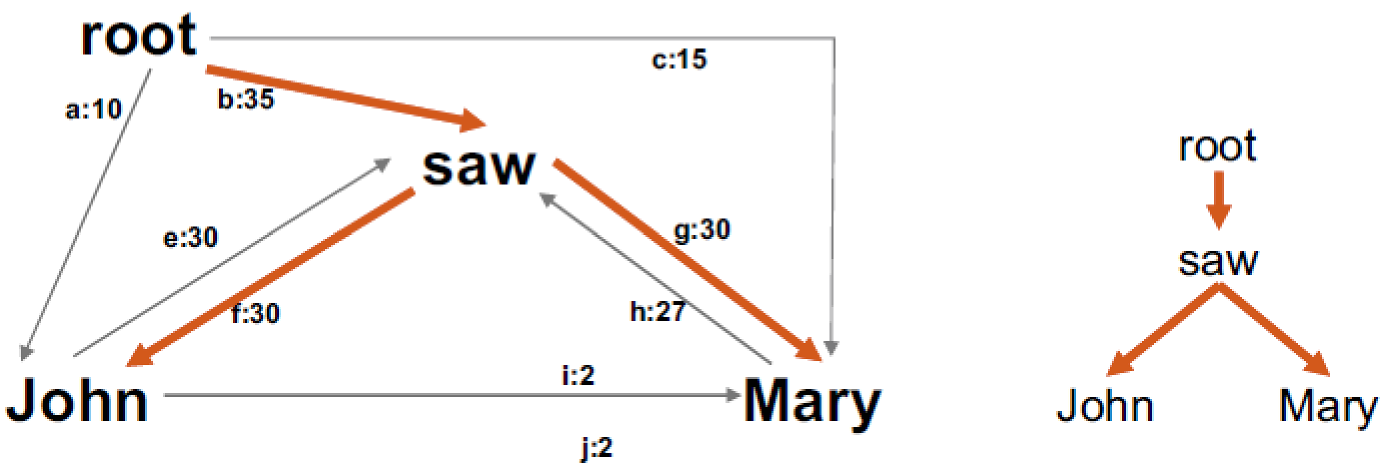
| Graham Neubig. 2019. Parsing with Dynamic Programming, CS11-747, Carnegie Mellon University
http://phontron.com/class/nn4nlp2019/assets/slides/nn4nlp-15-dpparsing.pdf

GRAPH BASED

II Dependency Parsing Algorithm

❖ Chu-Liu-Edmond's Algorithm

- Every node should have **one incoming edge**(=one head)
- Select the edge of the highest score for each node
- If there's a **cycle**, the edge to be removed depends on **an incoming edge**.
  - Contract / Expand
- Termination condition:



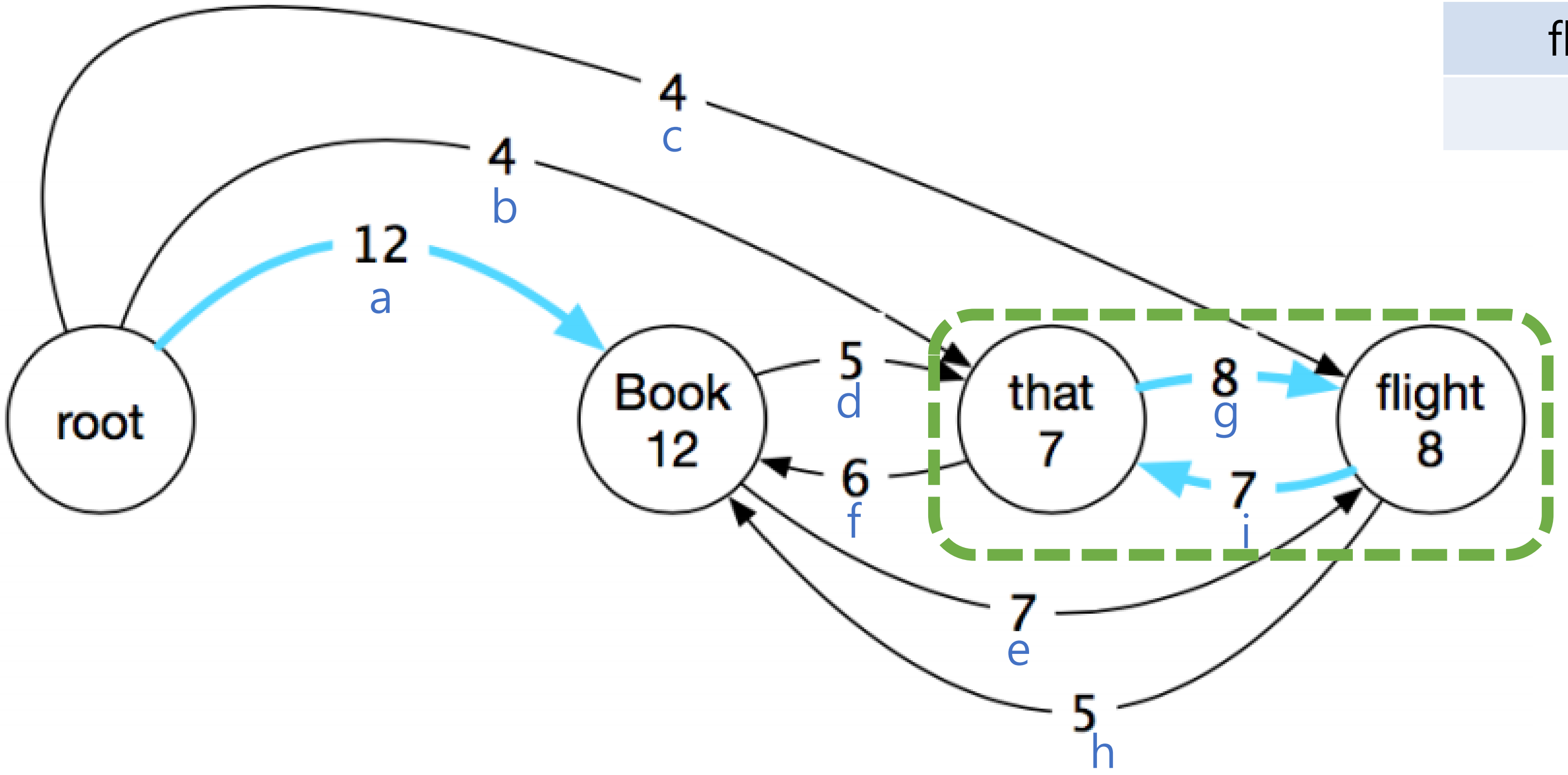
| K. Lee. Syntactic Parsing. 2018 Summer School on Natural Language Processing. p.26

Ⅱ Dependency Parsing Algorithm

GRAPH-BASED

❖ **Chu-Liu-Edmond's Algorithm**

- **Find the Best Incoming**



Best Incoming Edge	
Book	a
that	i
flight	g

Kicks Out

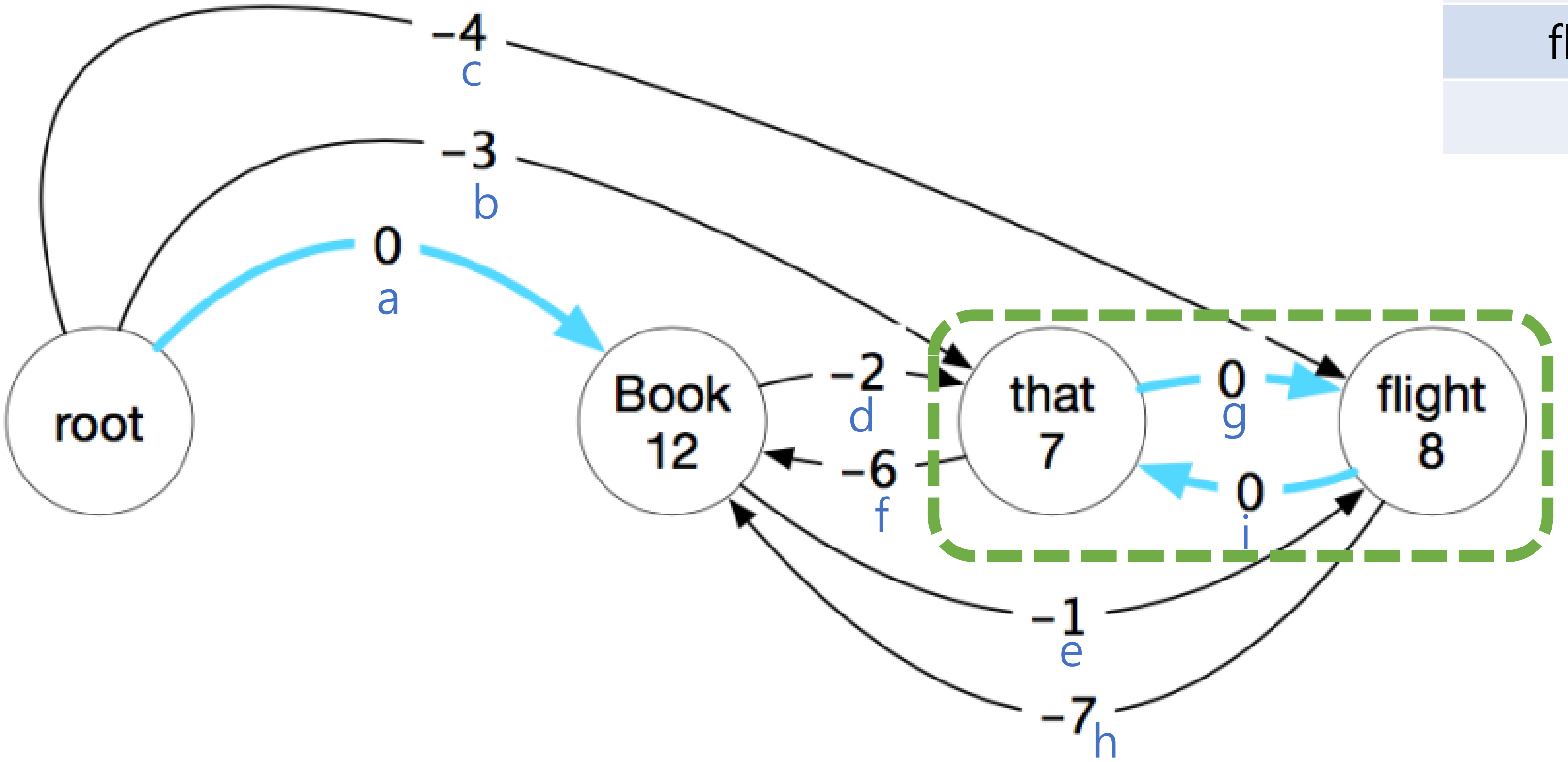
| Graham Neubig. 2019. Parsing with Dynamic Programming, CS11-747, Carnegie Mellon University
http://phontron.com/class/nn4nlp2019/assets/slides/nn4nlp-15-dpparsing.pdf

GRAPH BASED

Ⅱ Dependency Parsing Algorithm

❖ Chu-Liu-Edmond's Algorithm

▪ Subtract the Max for Each



Best Incoming Edge	
Book	a
that	i
flight	g

Kicks Out
b → i
d → i
c → g
e → g

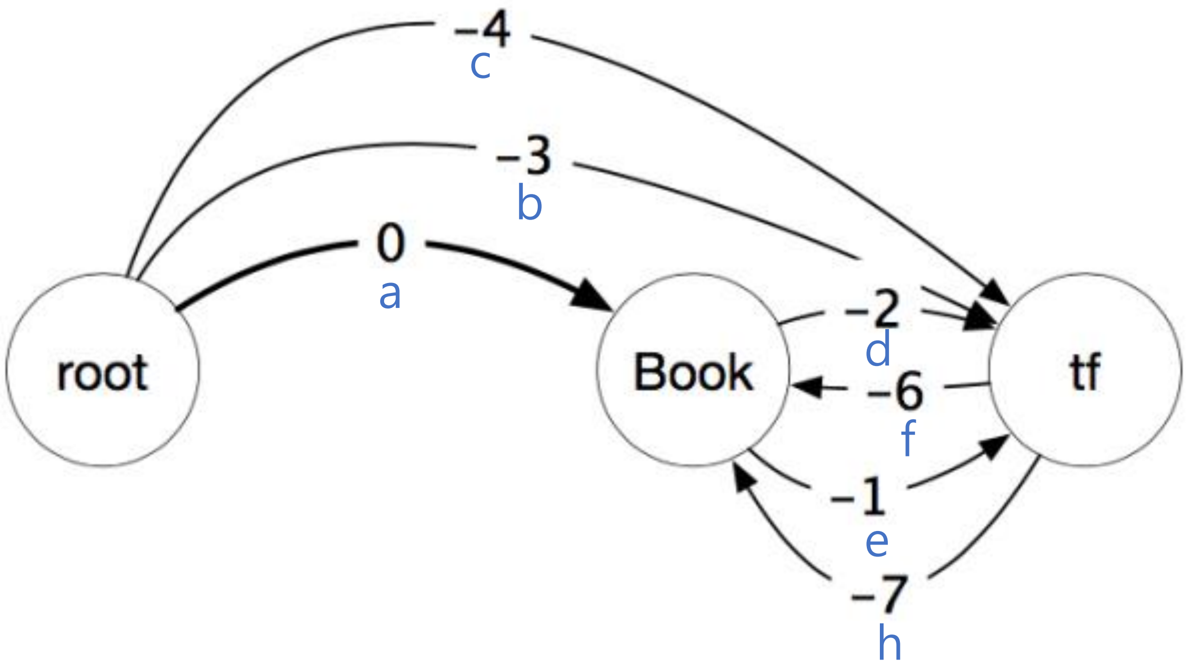
| Graham Neubig. 2019. Parsing with Dynamic Programming, CS11-747, Carnegie Mellon University
http://phontron.com/class/nn4nlp2019/assets/slides/nn4nlp-15-dpparsing.pdf

GRAPH BASED

Ⅱ Dependency Parsing Algorithm

❖ Chu-Liu-Edmond's Algorithm

- Contract a Node



Best Incoming Edge	
Book	a
that	i
flight	g
that-flight	

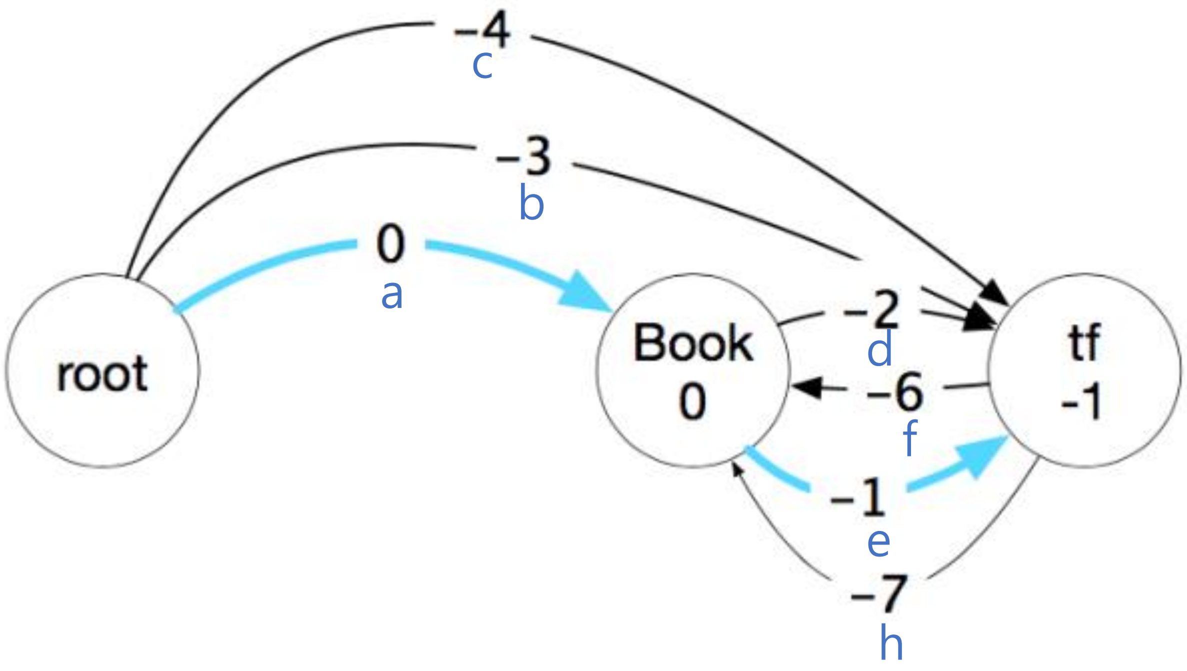
Kicks Out
b → i
d → i
c → g
e → g

| Graham Neubig. 2019. Parsing with Dynamic Programming, CS11-747, Carnegie Mellon University
http://phontron.com/class/nn4nlp2019/assets/slides/nn4nlp-15-dpparsing.pdf

GRAPH BASED

Ⅱ Dependency Parsing Algorithm

❖ **Chu-Liu-Edmond's Algorithm**
- **Recursively Call Algorithm**



Best Incoming Edge	
Book	a
that	i
flight	g
that-flight	e

Kicks Out
b → i
d → i
c → g
e → g

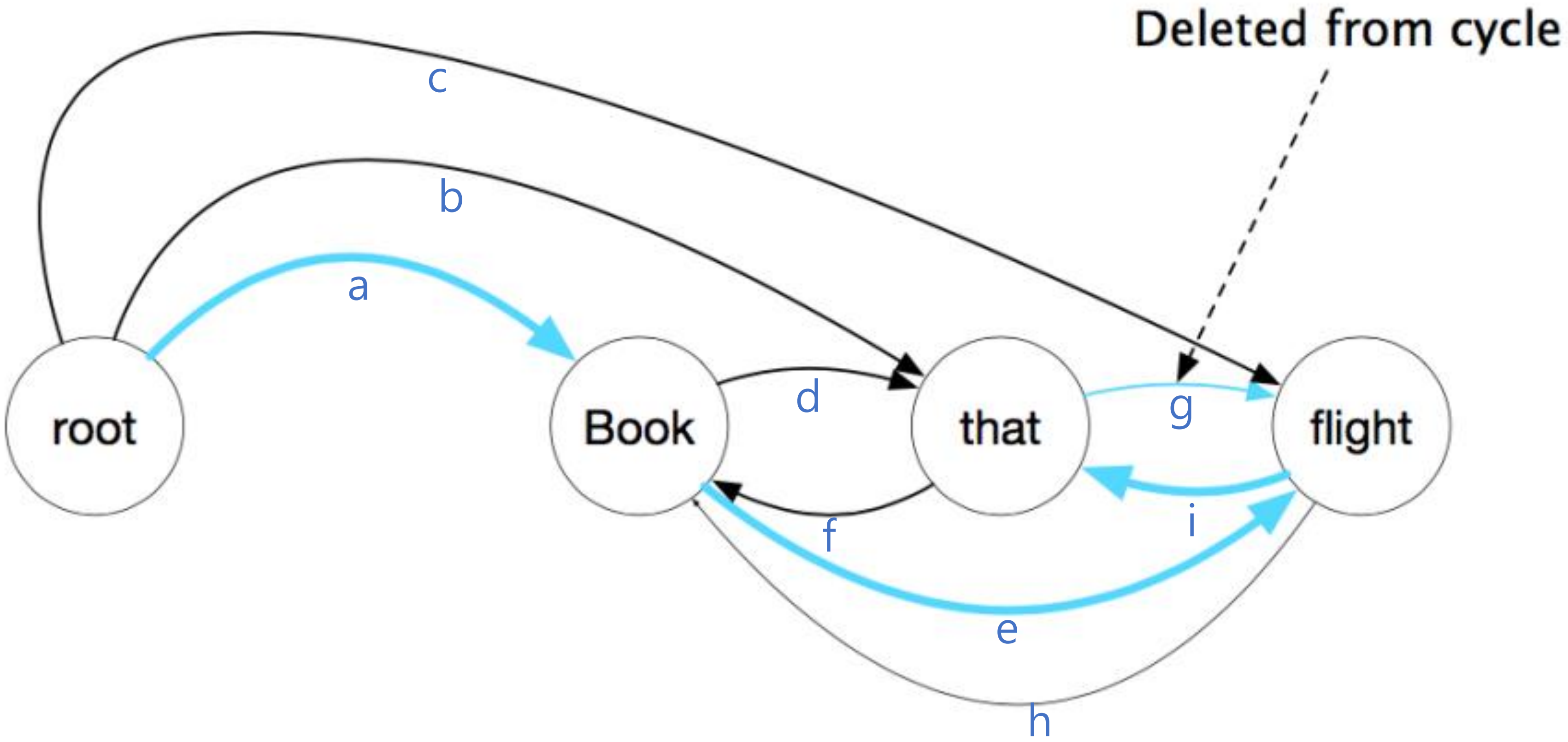
| Graham Neubig. 2019. Parsing with Dynamic Programming, CS11-747, Carnegie Mellon University
http://phontron.com/class/nn4nlp2019/assets/slides/nn4nlp-15-dpparsing.pdf

Ⅱ Dependency Parsing Algorithm

GRAPH-BASED

❖ Chu-Liu-Edmond's Algorithm

- Expand Nodes and Delete Edge



Best Incoming Edge	
Book	a
that	i
flight	g ✖
that-flight	e

Kicks Out
b → i
d → i
c → g
e → g

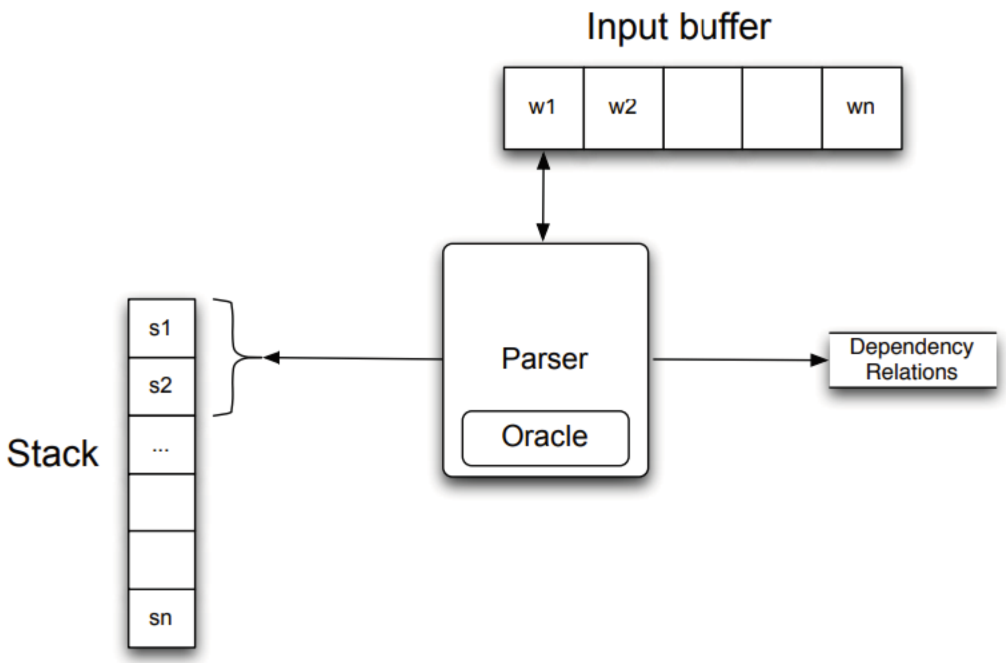
| Graham Neubig. 2019. Parsing with Dynamic Programming, CS11-747, Carnegie Mellon University http://phontron.com/class/nn4nlp2019/assets/slides/nn4nlp-15-dpparsing.pdf

TRANS.-BASED

Ⅱ Dependency Parsing Algorithm

❖ **Define a transition system for mapping a sentence to its dependency tree**

- Learning: induce a model for **predicting the next state transition**,
  given the transition history
- Parsing: construct the optimal transition sequence, given the induced model



Ⅱ Dependency Parsing Algorithm

TRANS. BASED

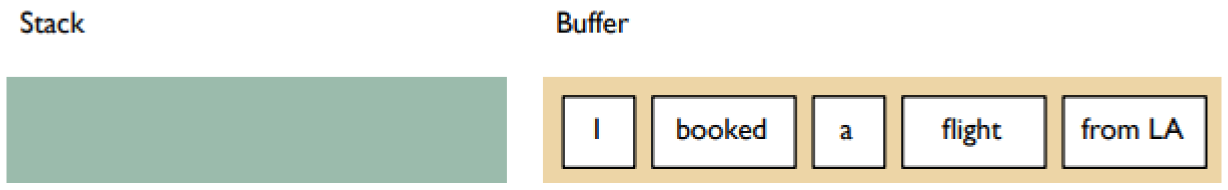
❖ **Arc-standard transition-based parser**

	Start: σ = [ROOT], $\beta = w_1, \ldots, w_n$, A = $\varnothing$
Shift	1. Shift $\quad \sigma, w_i\vert\beta, A \rightarrow \sigma\vert w_i, \beta, A$
Reduce : create dependencies	2. Left-Arc$_r$ $\quad \sigma\vert w_i\vert w_j, \beta, A \rightarrow \sigma\vert w_j, \beta, A \cup \{r(w_j, w_i)\}$
	3. Right-Arc$_r$ $\quad \sigma\vert w_i\vert w_j, \beta, A \rightarrow \sigma\vert w_i, \beta, A \cup \{r(w_i, w_j)\}$
	Finish: $\sigma = [w]$, $\beta = \varnothing$

| https://web.stanford.edu/class/archive/cs/cs224n/cs224n.1174/lectures/cs224n-2017-lecture6.pdf

Ⅱ Dependency Parsing Algorithm

❖ **Arc-standard transition-based parser**

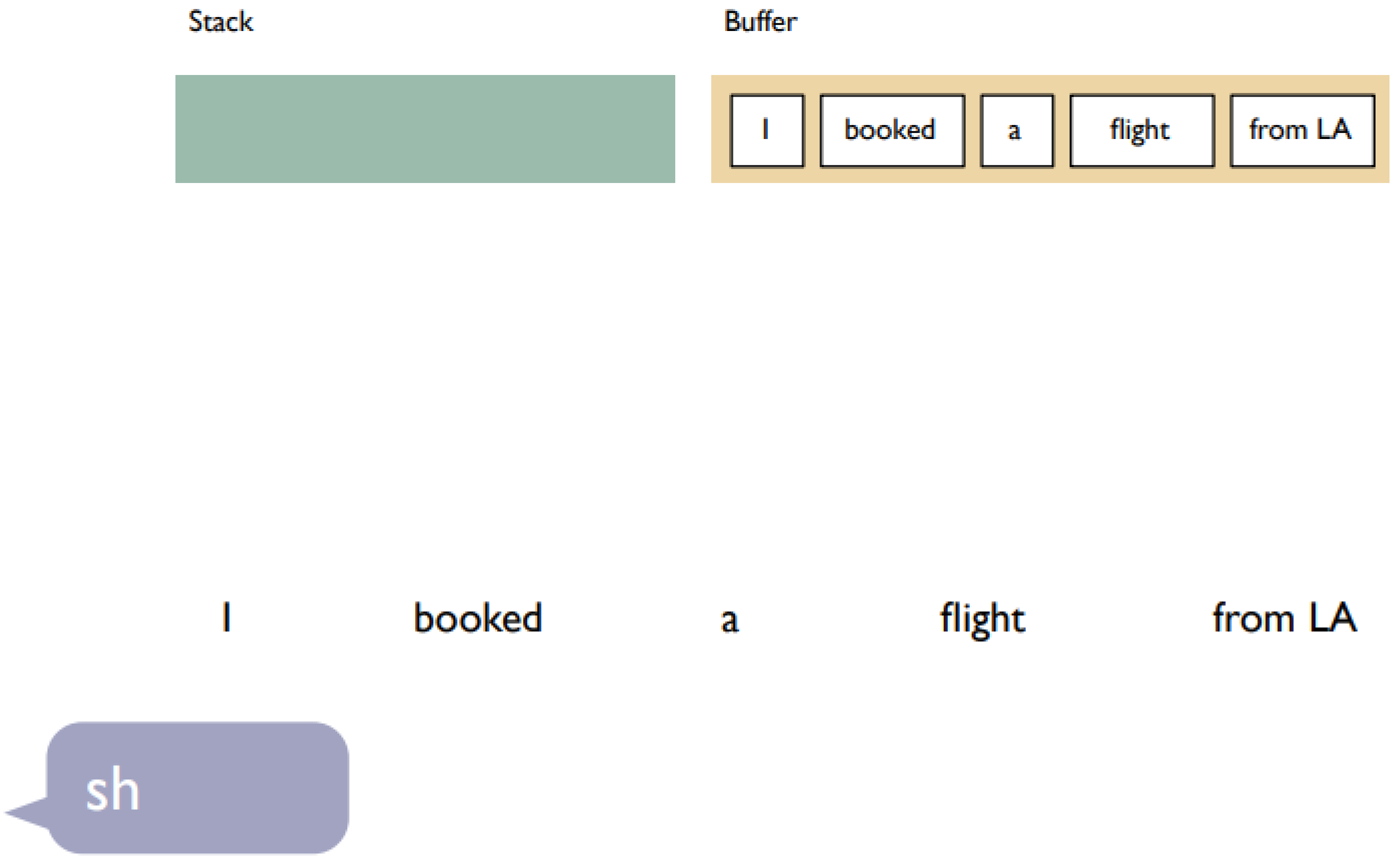


I booked a flight from LA

Ⅱ Dependency Parsing Algorithm

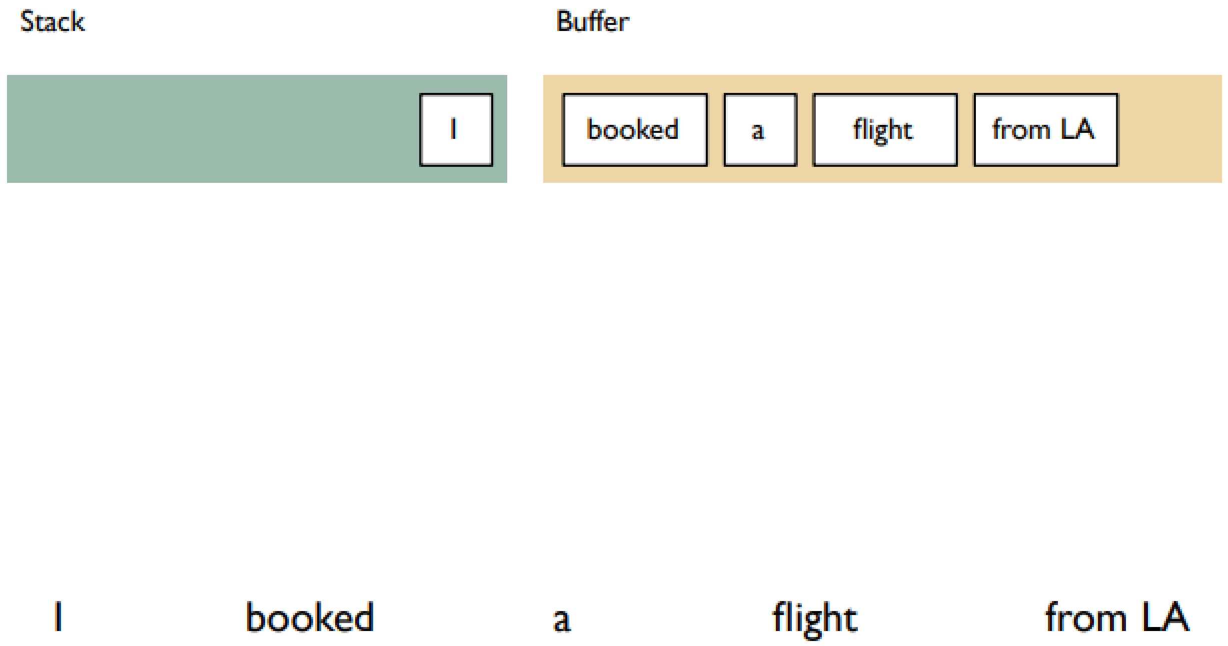
TRANS.-BASED

❖ **Arc-standard transition-based parser**



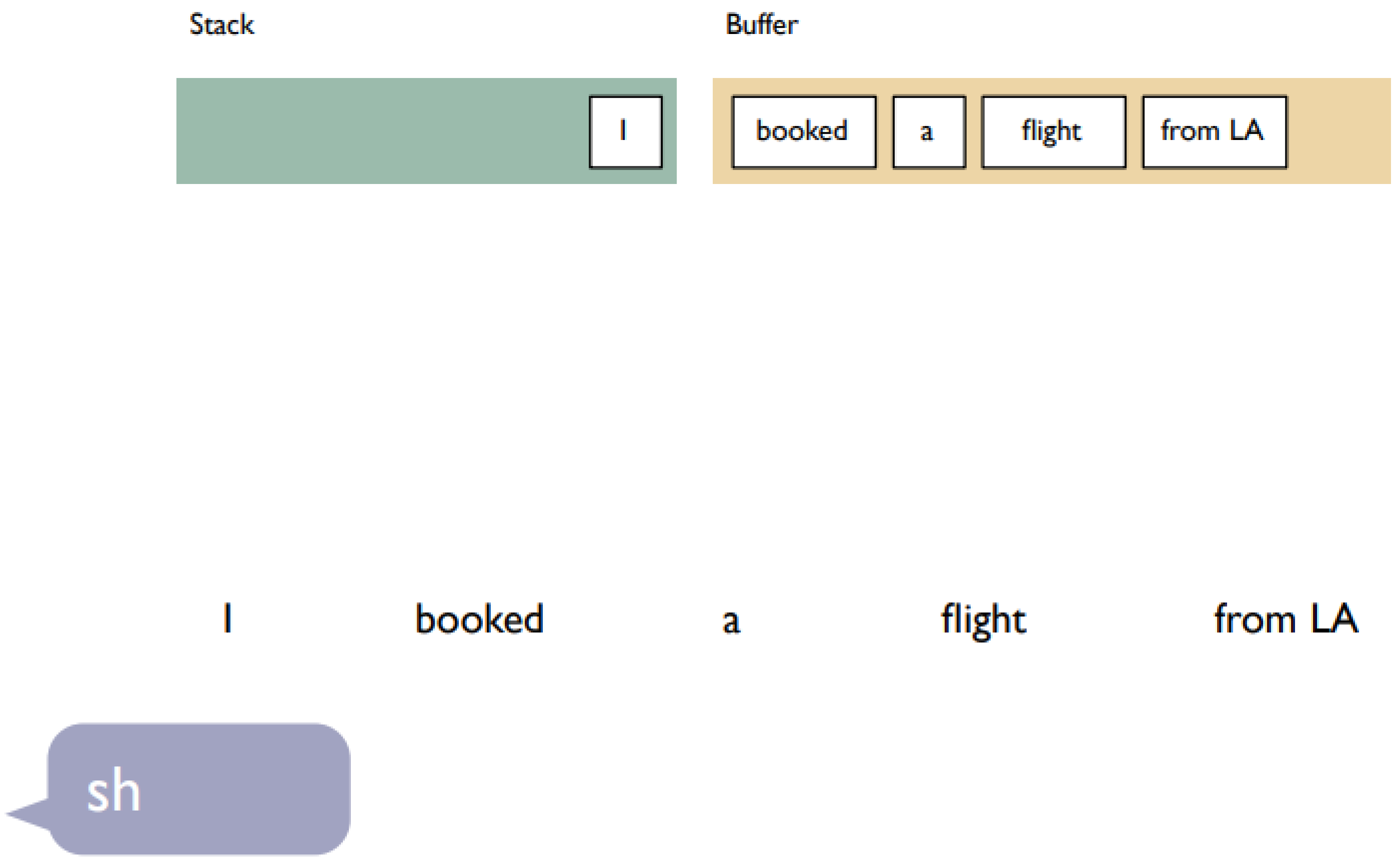
Ⅱ Dependency Parsing Algorithm

❖ Arc-standard transition-based parser



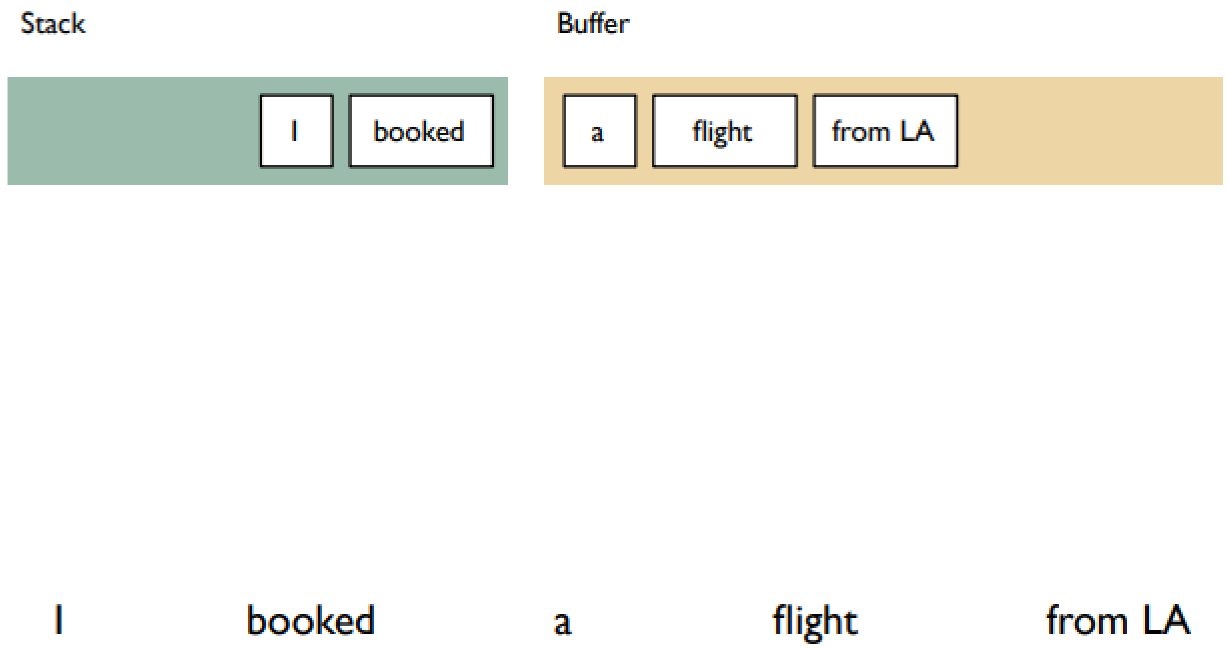
Ⅱ Dependency Parsing Algorithm

❖ **Arc-standard transition-based parser**



II Dependency Parsing Algorithm

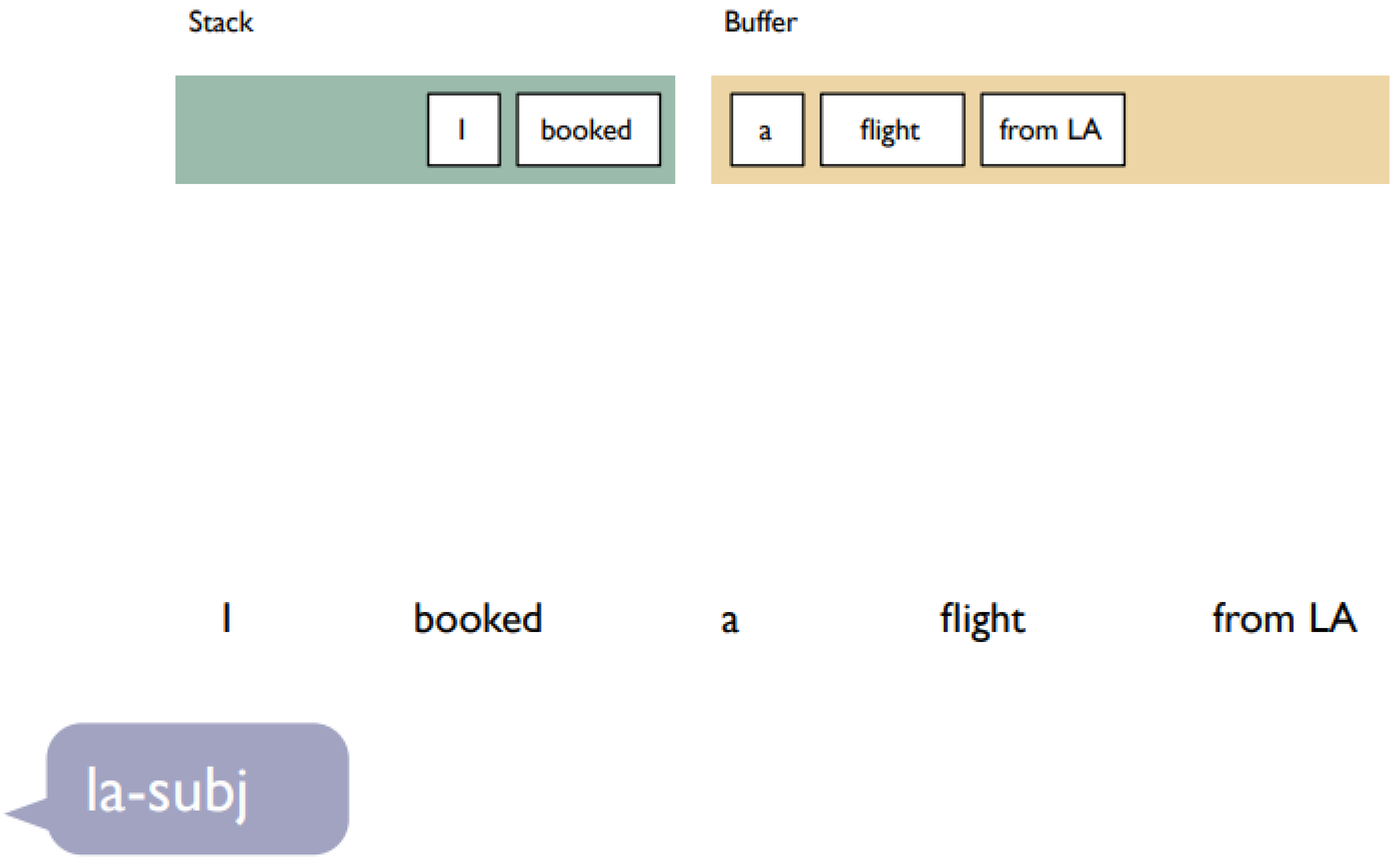
❖ Arc-standard transition-based parser



TRANS-BASED

Ⅱ Dependency Parsing Algorithm

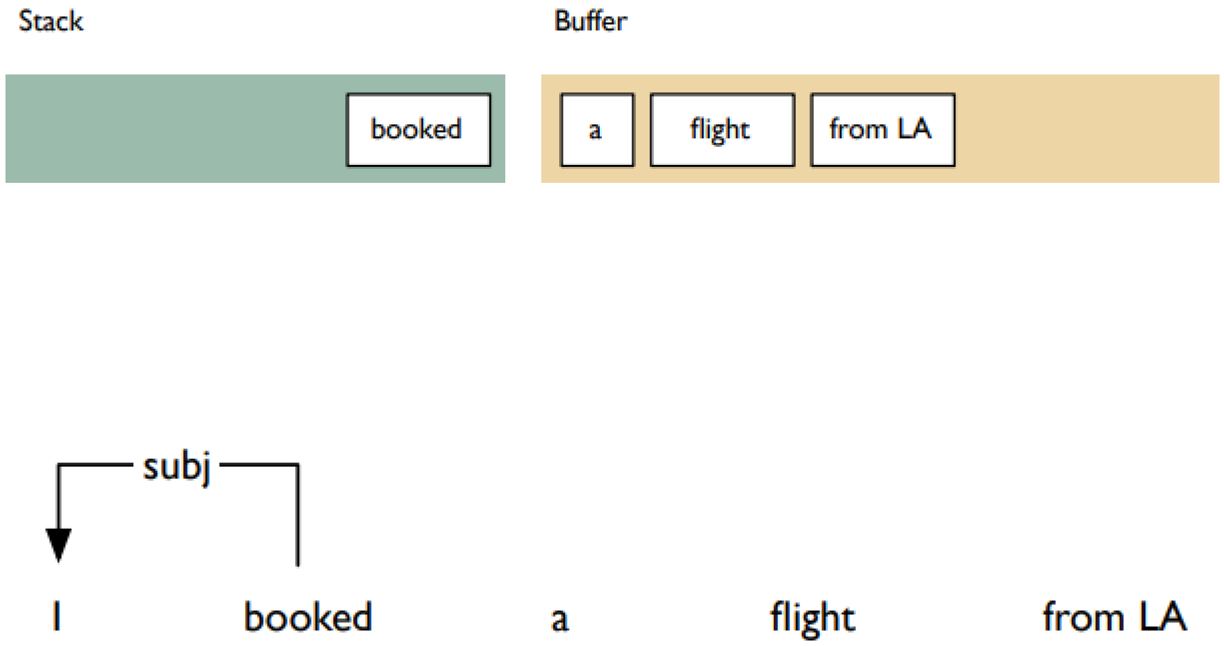
❖ **Arc-standard transition-based parser**



TRANS-BASED

Ⅱ Dependency Parsing Algorithm

❖ Arc-standard transition-based parser

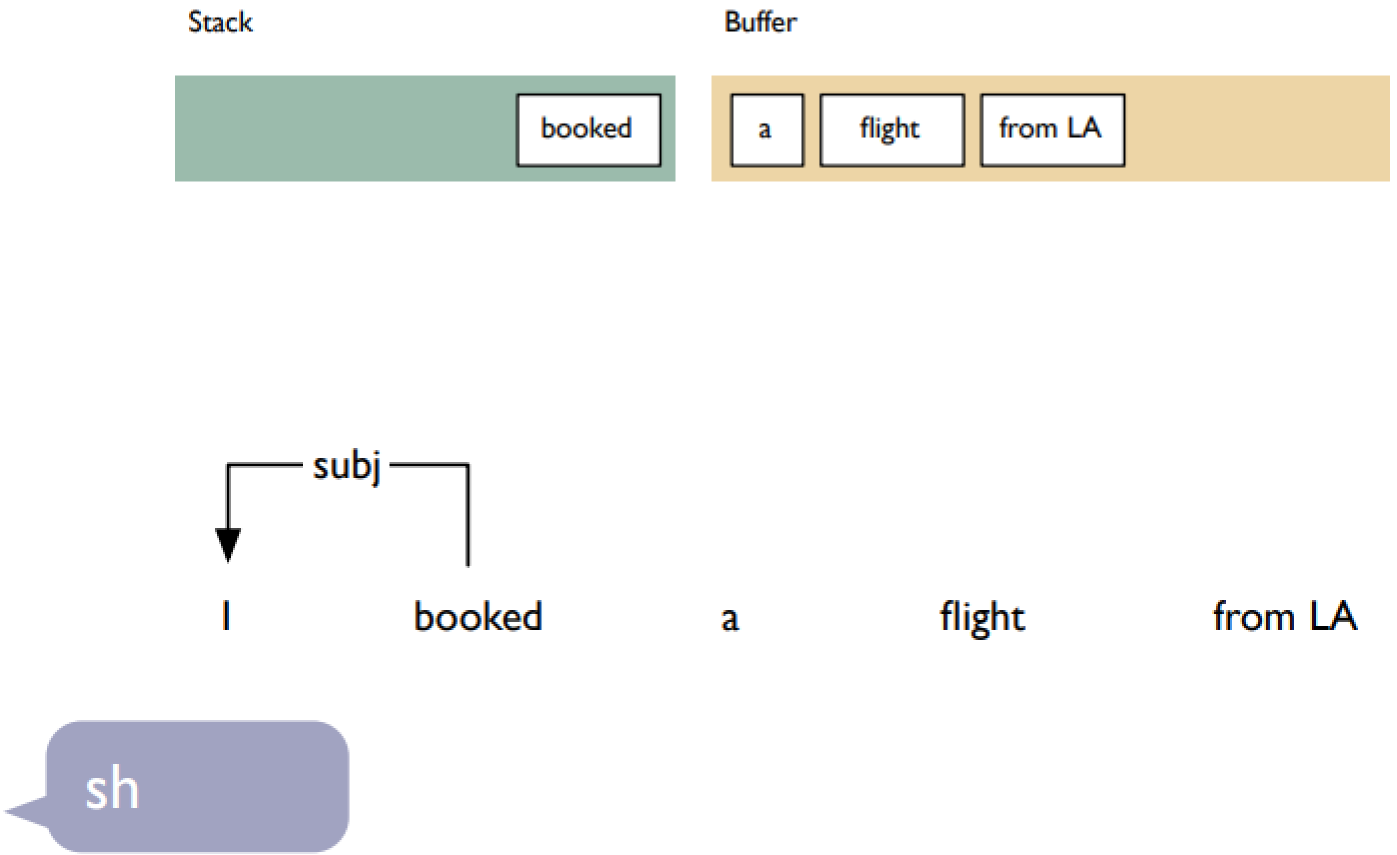


| https://cl.lingfil.uu.se/~sara/kurser/5LN455-2014/lectures/5LN455-F8.pdf. pp.14-33

Ⅱ Dependency Parsing Algorithm

TRANS.-BASED

❖ Arc-standard transition-based parser

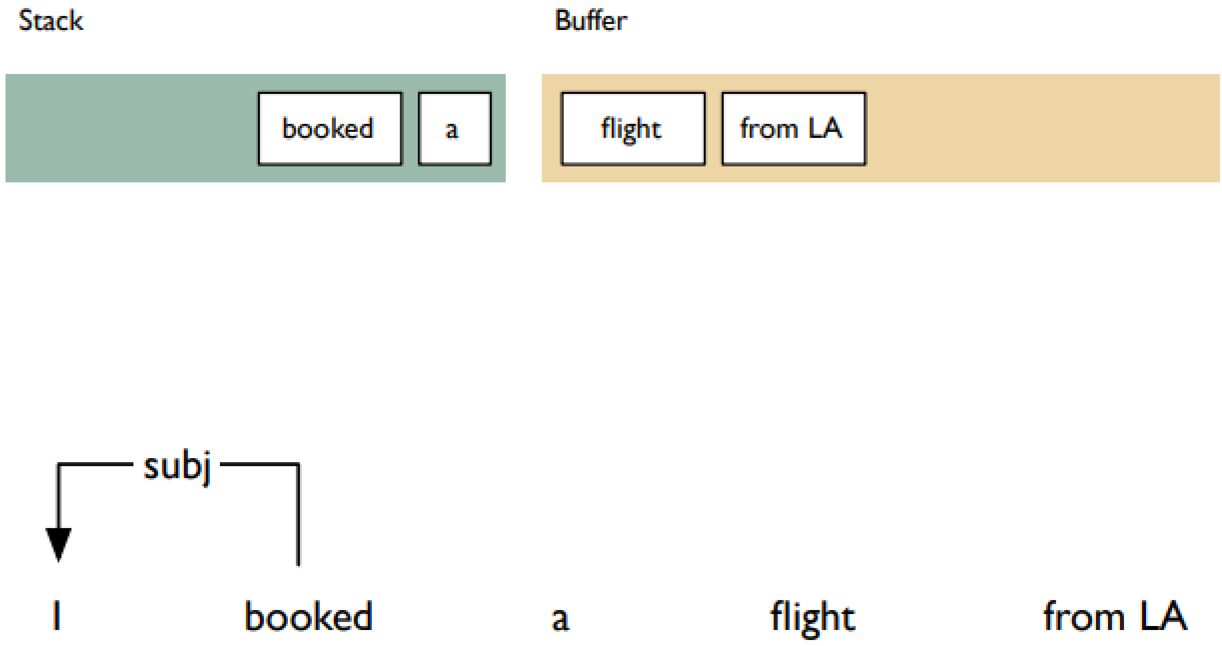


| https://cl.lingfil.uu.se/~sara/kurser/5LN455-2014/lectures/5LN455-F8.pdf. pp.14-33

TRANS.-BASED

Ⅱ Dependency Parsing Algorithm

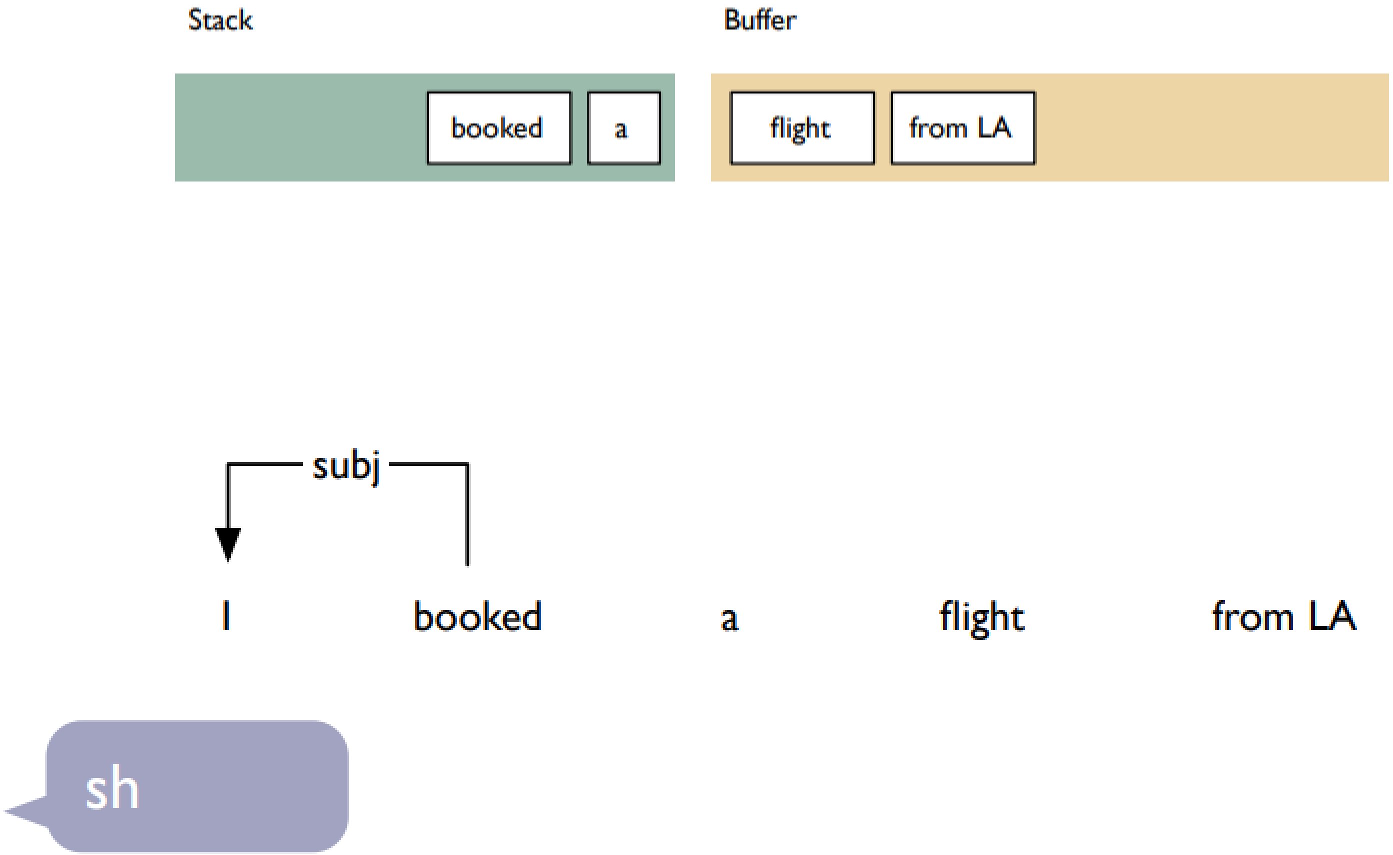
❖ Arc-standard transition-based parser



Ⅱ Dependency Parsing Algorithm

TRANS.-BASED

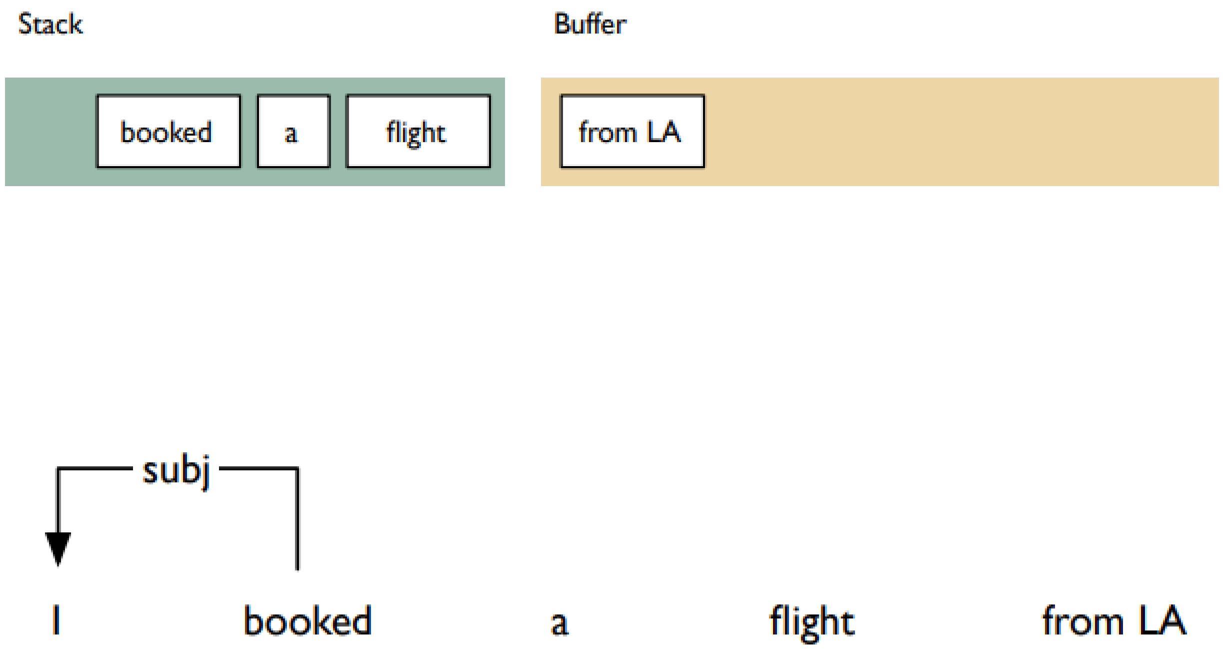
❖ Arc-standard transition-based parser



TRANS.-BASED

Ⅱ Dependency Parsing Algorithm

❖ **Arc-standard transition-based parser**

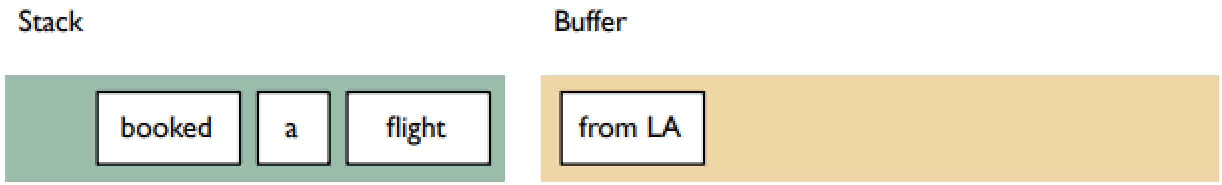


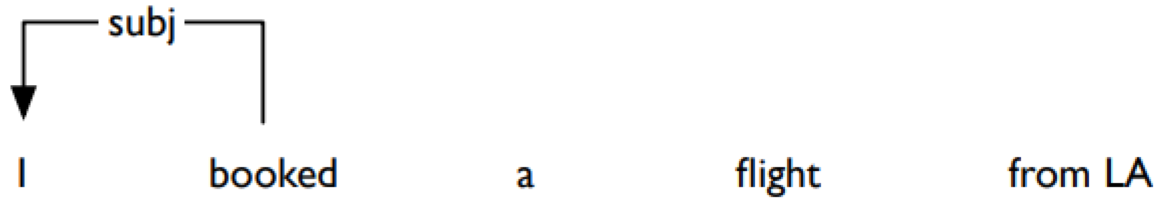
| https://cl.lingfil.uu.se/~sara/kurser/5LN455-2014/lectures/5LN455-F8.pdf. pp.14-33

TRANS.-BASED

Ⅱ Dependency Parsing Algorithm

❖ **Arc-standard transition-based parser**



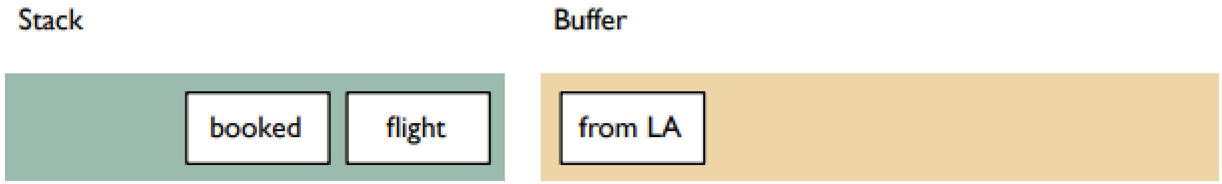


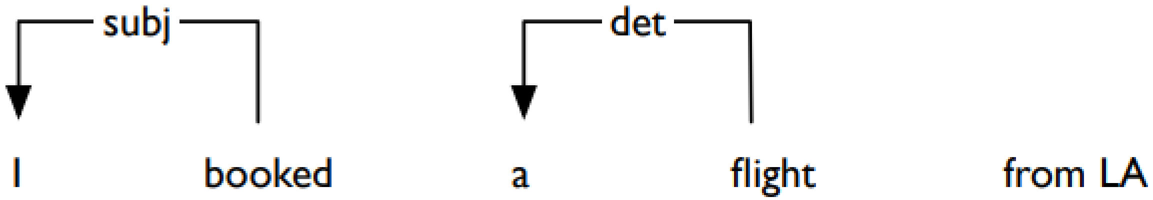
la-det

Ⅱ Dependency Parsing Algorithm

TRANS.-BASED

❖ **Arc-standard transition-based parser**



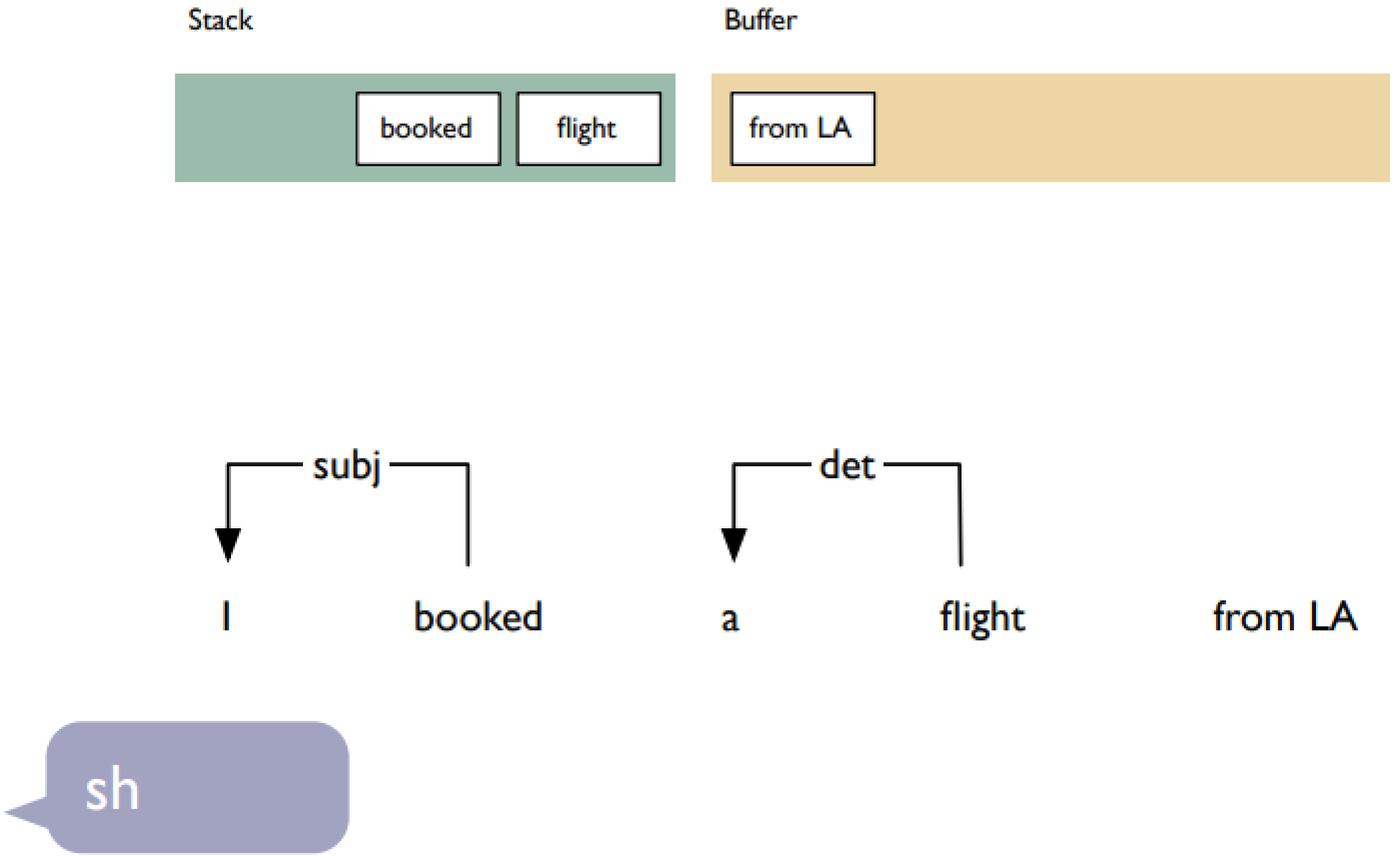


| https://cl.lingfil.uu.se/~sara/kurser/5LN455-2014/lectures/5LN455-F8.pdf. pp.14-33

Ⅱ Dependency Parsing Algorithm

TRANS.-BASED

❖ Arc-standard transition-based parser

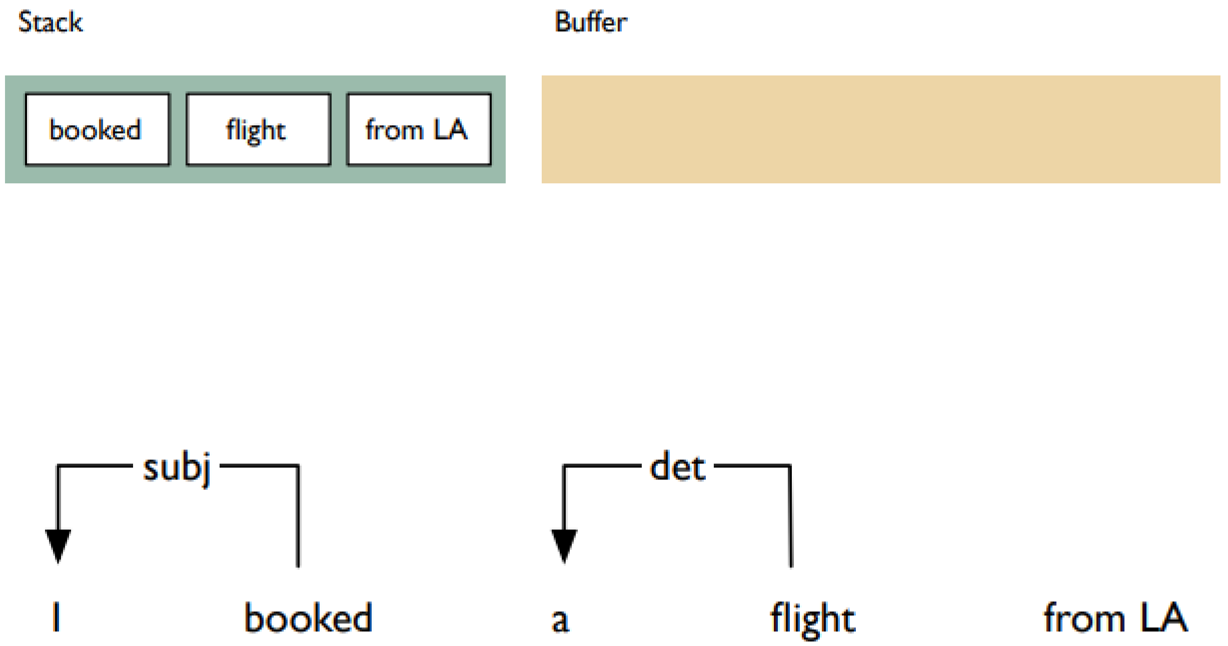


| https://cl.lingfil.uu.se/~sara/kurser/5LN455-2014/lectures/5LN455-F8.pdf. pp.14-33

Ⅱ Dependency Parsing Algorithm

TRANS.-BASED

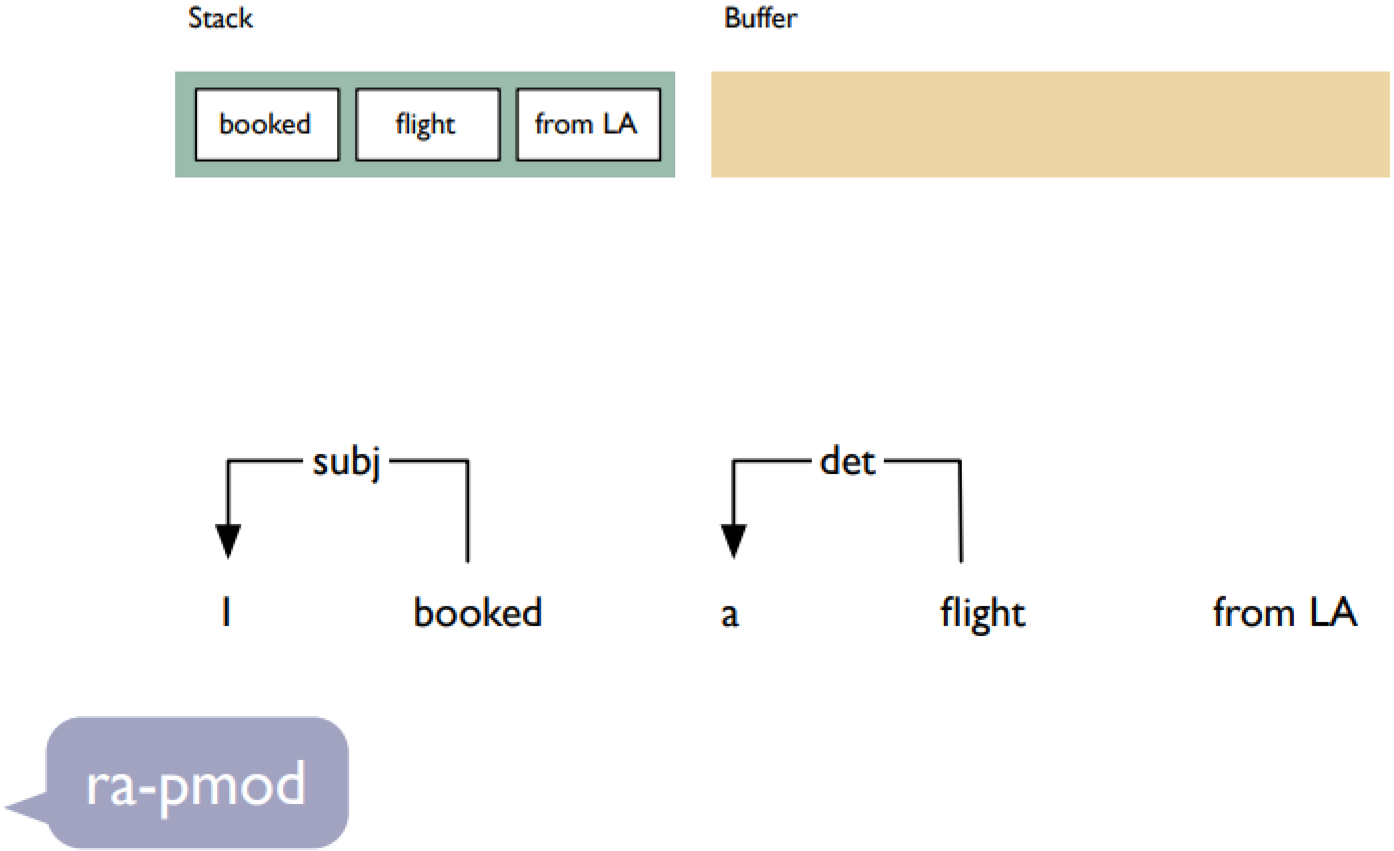
❖ **Arc-standard transition-based parser**



TRANS.-BASED

Ⅱ Dependency Parsing Algorithm

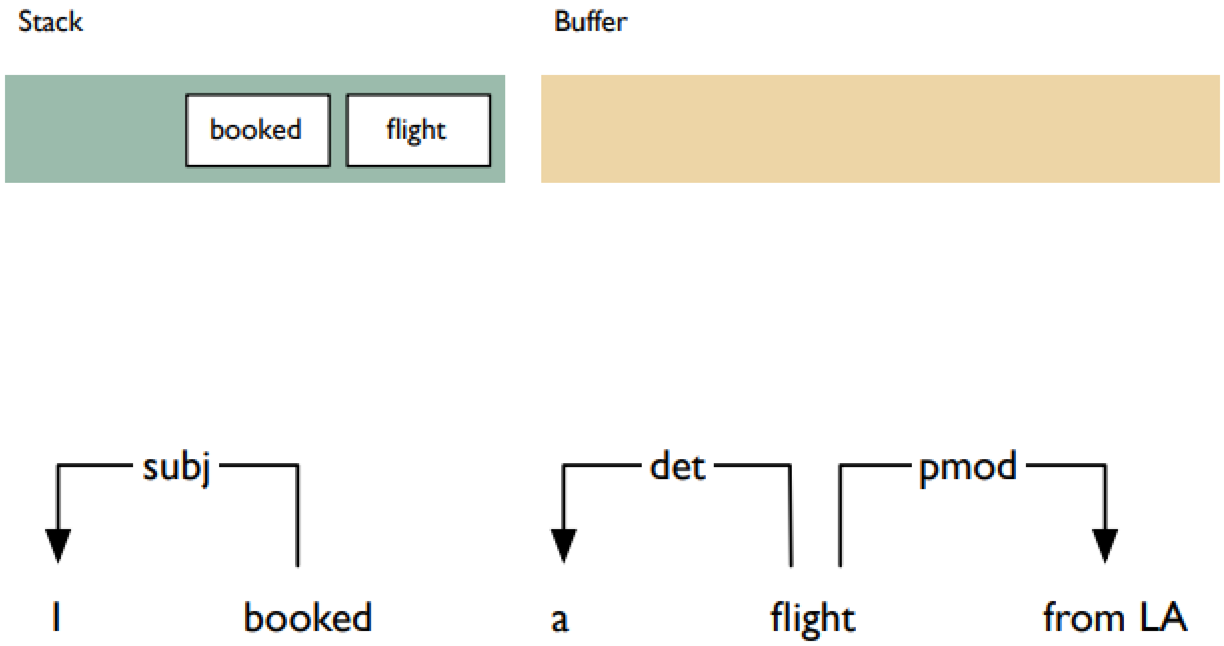
❖ **Arc-standard transition-based parser**



Ⅱ Dependency Parsing Algorithm

TRANS. BASED

❖ Arc-standard transition-based parser

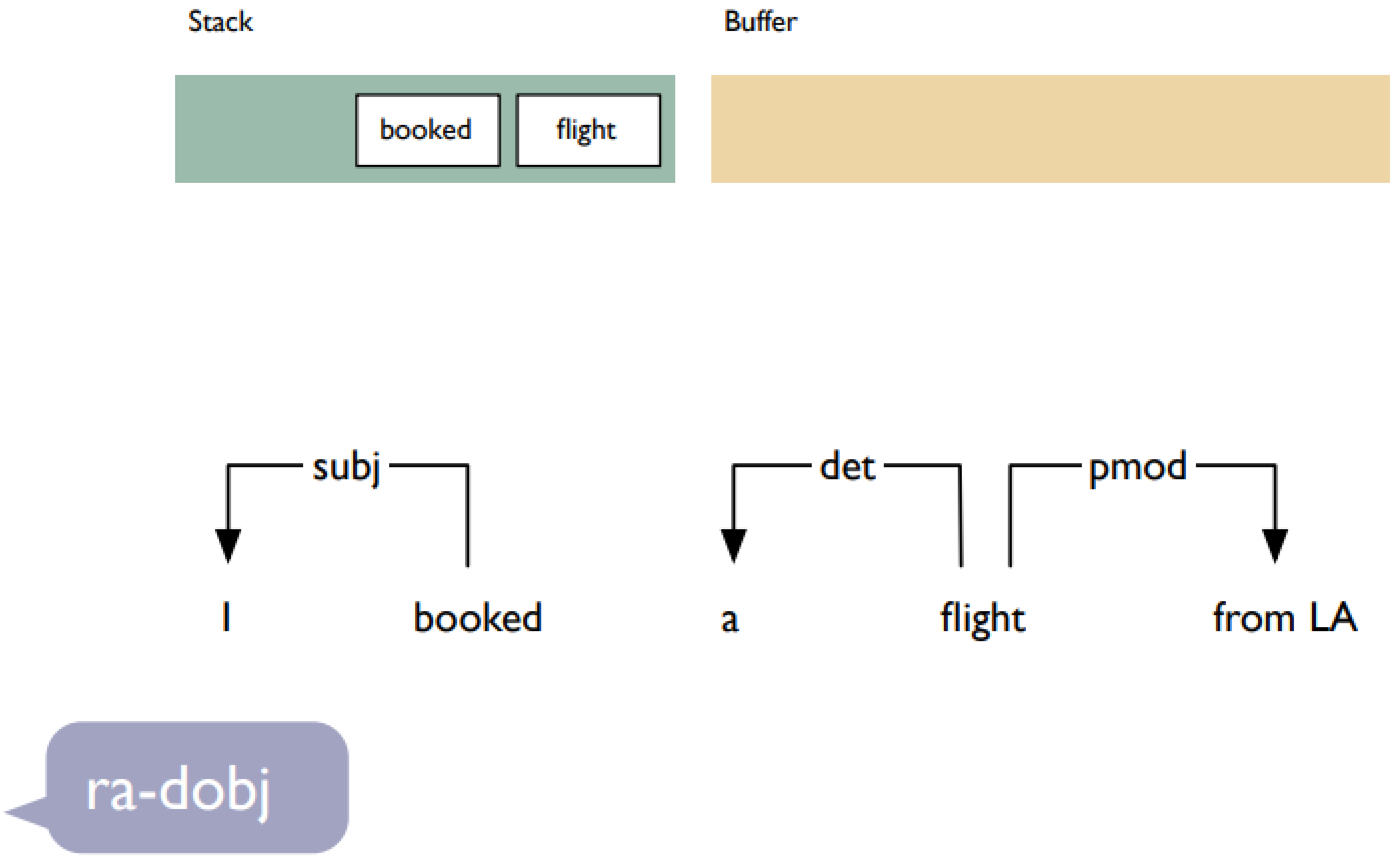


| https://cl.lingfil.uu.se/~sara/kurser/5LN455-2014/lectures/5LN455-F8.pdf. pp.14-33

TRANS-BASED

Ⅱ Dependency Parsing Algorithm

❖ **Arc-standard transition-based parser**

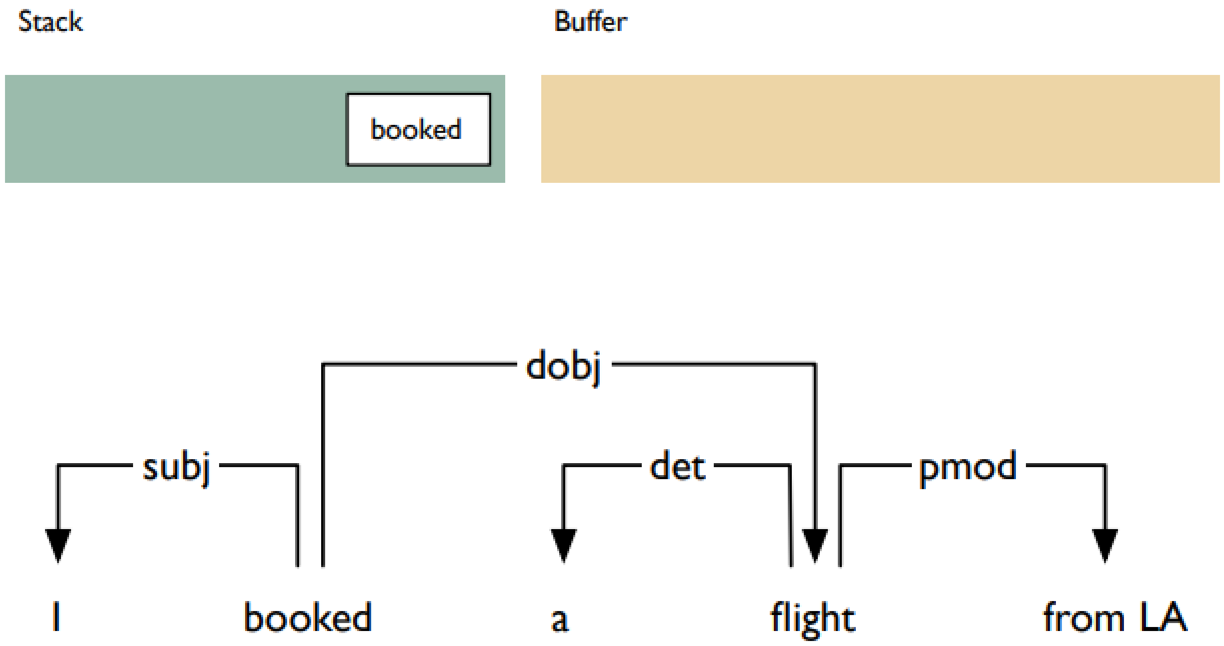


| https://cl.lingfil.uu.se/~sara/kurser/5LN455-2014/lectures/5LN455-F8.pdf. pp.14-33

Ⅱ Dependency Parsing Algorithm

TRANS-BASED

❖ **Arc-standard transition-based parser**

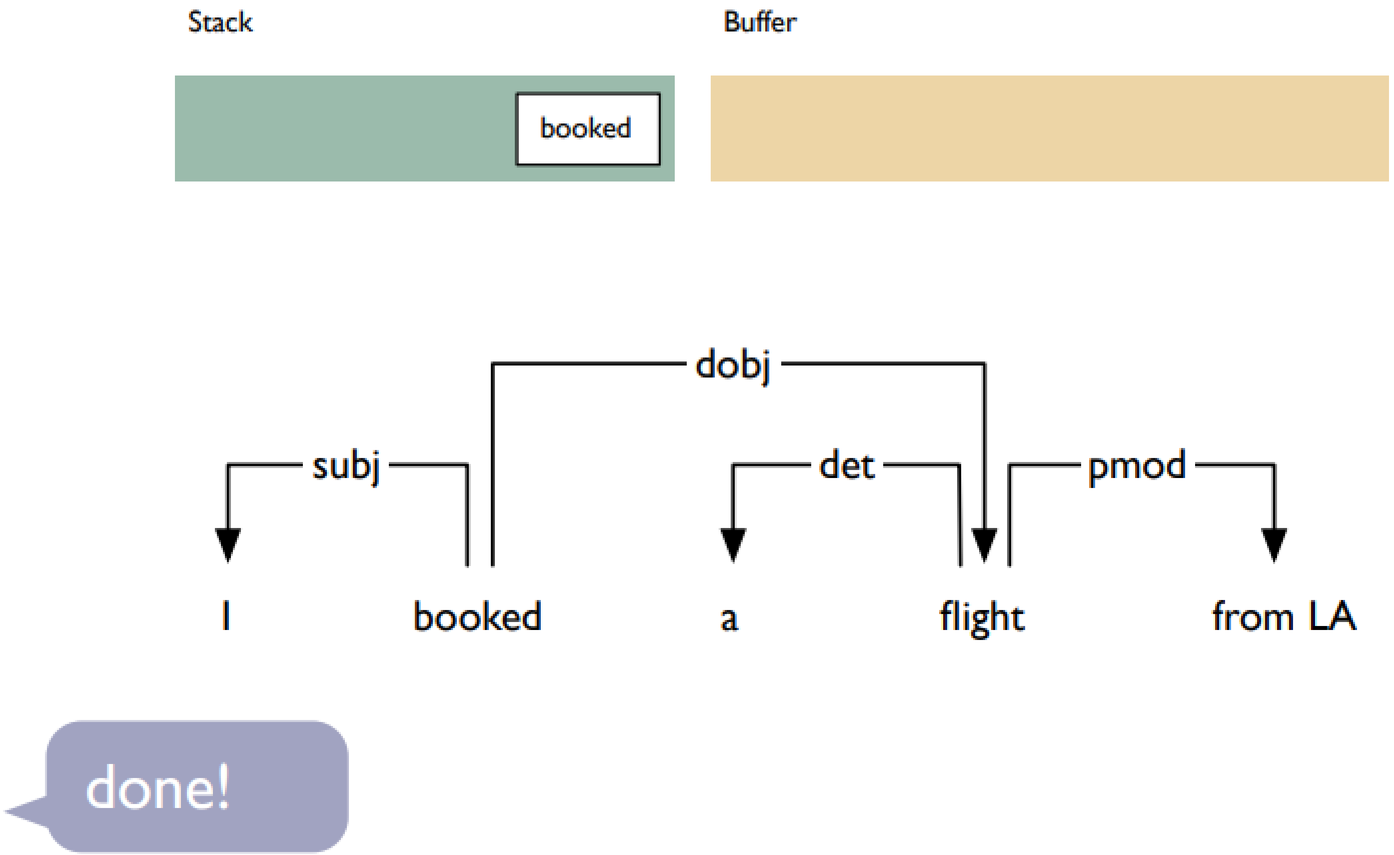


| https://cl.lingfil.uu.se/~sara/kurser/5LN455-2014/lectures/5LN455-F8.pdf. pp.14-33

Ⅱ Dependency Parsing Algorithm

TRANS.-BASED

❖ **Arc-standard transition-based parser**

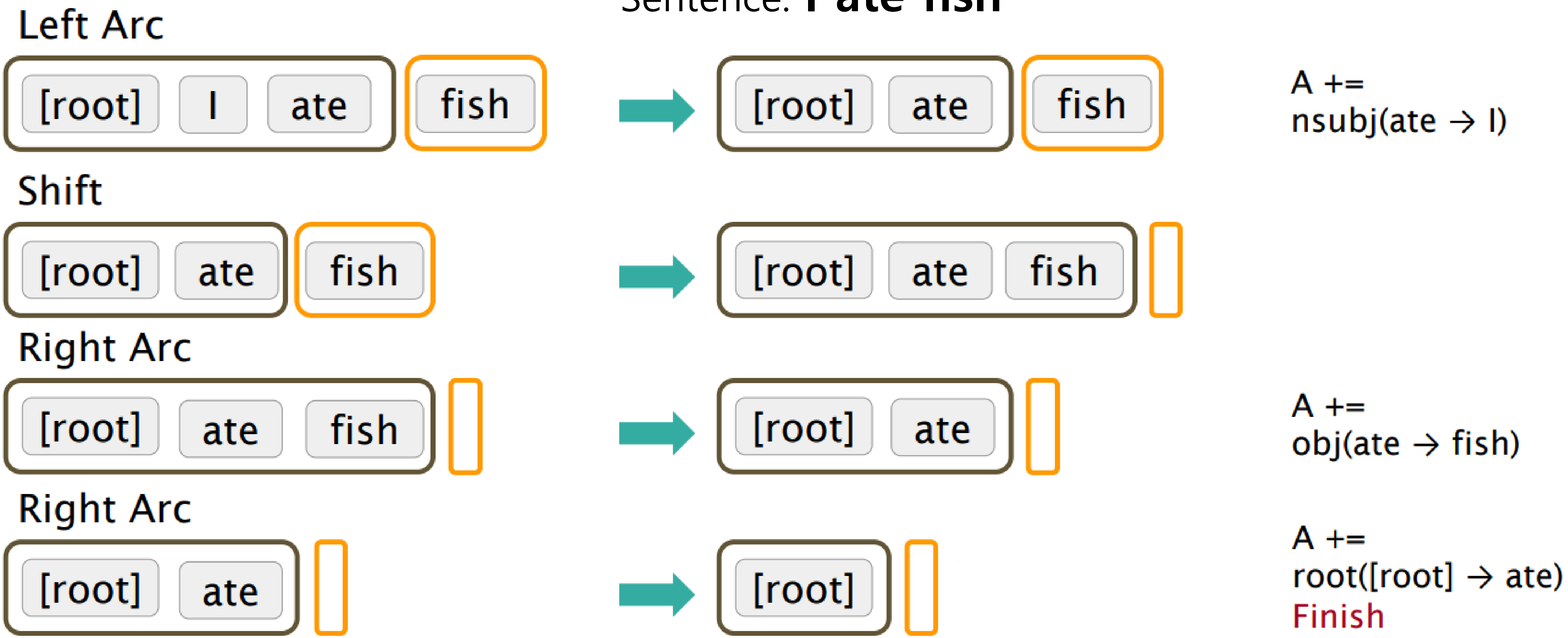


| https://cl.lingfil.uu.se/~sara/kurser/5LN455-2014/lectures/5LN455-F8.pdf. pp.14-33

Ⅱ Dependency Parsing Algorithm

❖ **Arc-standard transition-based parser**

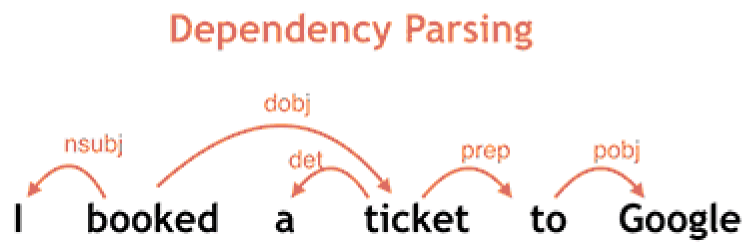
Sentence: **I ate fish**



TRANS.I BASED

II Dependency Parsing Algorithm

❖ **Arc-standard transition-based parser**



- Time complexity is linear, **O(n)**; only treat each word once
- There is **no guarantee** that we will even find the best tree given the model (There is a risk of error propagation)

| https://ai.googleblog.com/2016/05/announcing-syntaxnet-worlds-most.html

TRANS-BASED

Ⅱ Dependency Parsing Algorithm

❖ **Arc-eager transition-based parser**

arc-standard

Shift	$(\sigma, i\vert\beta, A) \Rightarrow (\sigma\vert i, \beta, A)$
LArc	$(\sigma\vert i\vert j, \beta, A) \Rightarrow (\sigma\vert j, \beta, A \cup \{(j \rightarrow i)\})$
RArc	$(\sigma\vert i\vert j, \beta, A) \Rightarrow (\sigma\vert i, \beta, A \cup \{(i \rightarrow j)\})$

arcs are added
between the top two words on the **stack**

arc-eager

Shift	$(\sigma, i\vert\beta, A) \Rightarrow (\sigma\vert i, \beta, A)$
LArc	$(\sigma\vert i, j\vert\beta, A) \Rightarrow (\sigma, j\vert\beta, A \cup \{(j \rightarrow i)\})$
RArc	$(\sigma\vert i, j\vert\beta, A) \Rightarrow (\sigma\vert i\vert j, \beta, A \cup \{(i \rightarrow j)\})$
Reduce	$(\sigma\vert i, \beta, A) \Rightarrow (\sigma, \beta, A)$

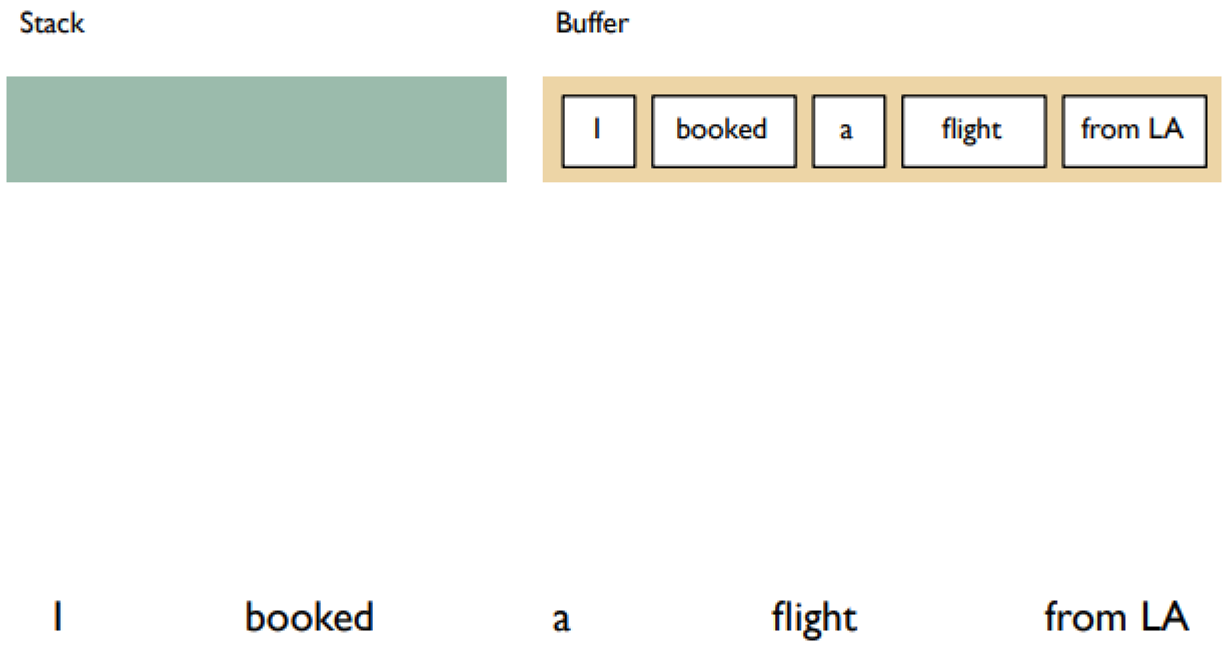
arcs are added
between the topmost word on the **stack**
and the topmost word on the **buffer**

- can **create arcs earlier** than arc-standard model

Ⅱ Dependency Parsing Algorithm

TRANS.-BASED

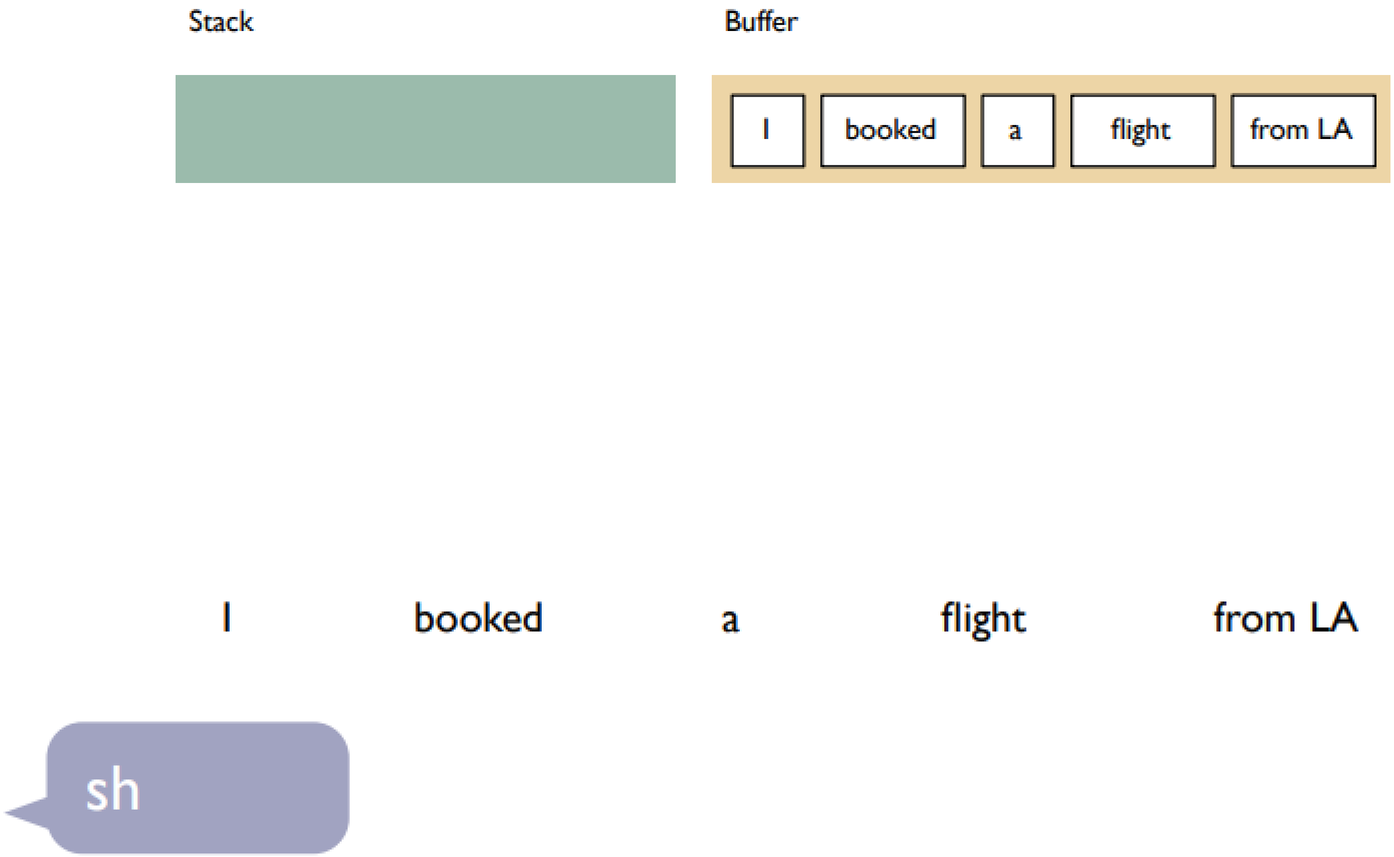
❖ **Arc-eager transition-based parser**



Ⅱ Dependency Parsing Algorithm

TRANS.-BASED

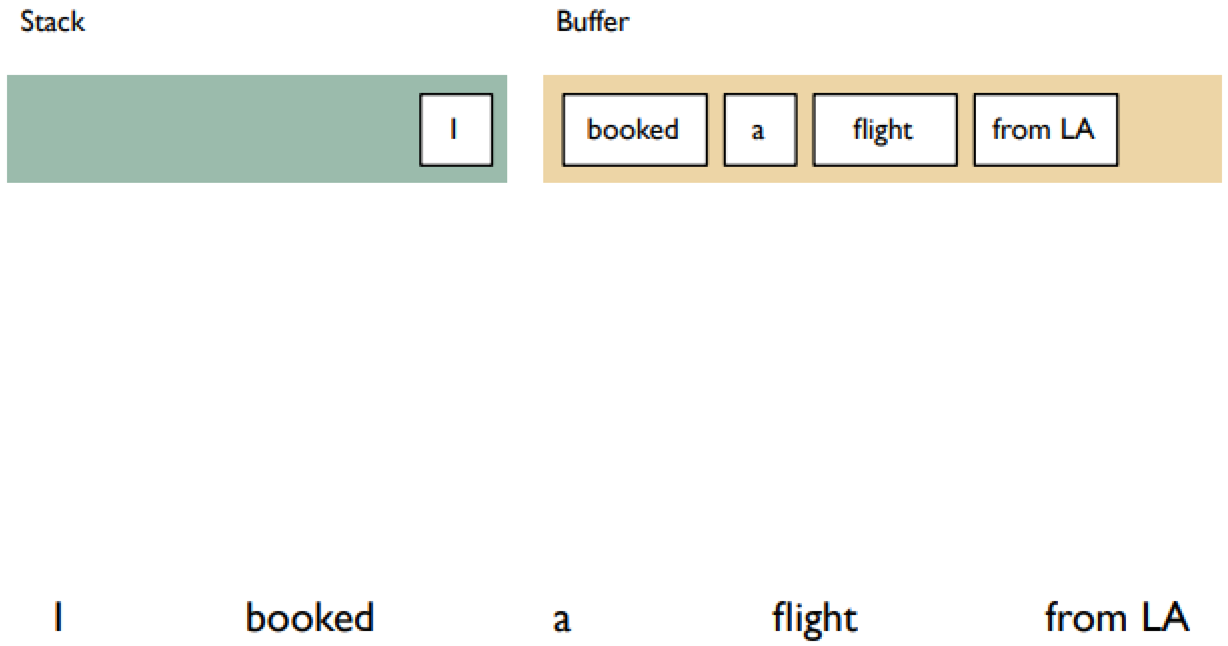
❖ **Arc-eager transition-based parser**



Ⅱ Dependency Parsing Algorithm

TRANS. BASED

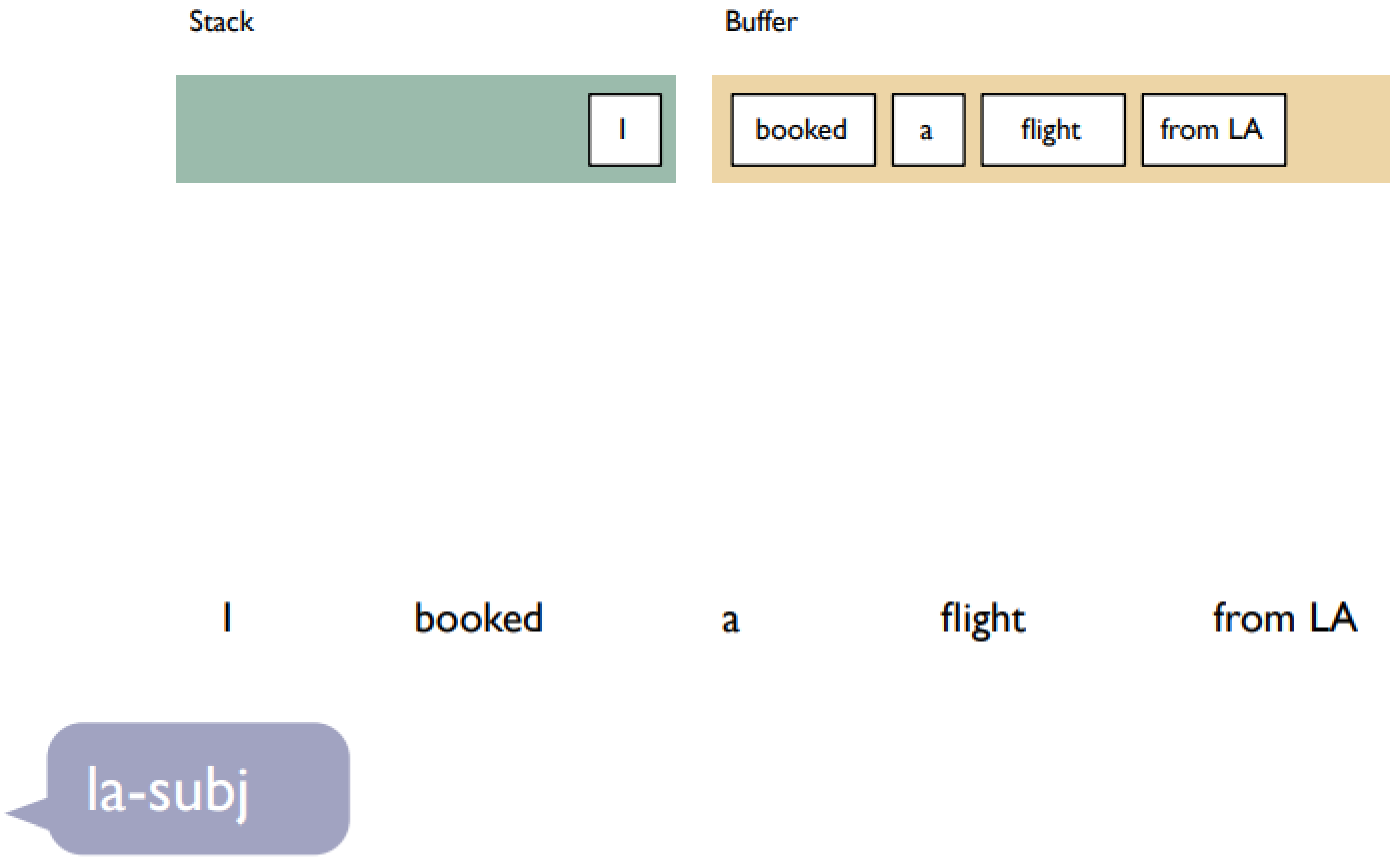
❖ Arc-eager transition-based parser



TRANS-BASED

Ⅱ Dependency Parsing Algorithm

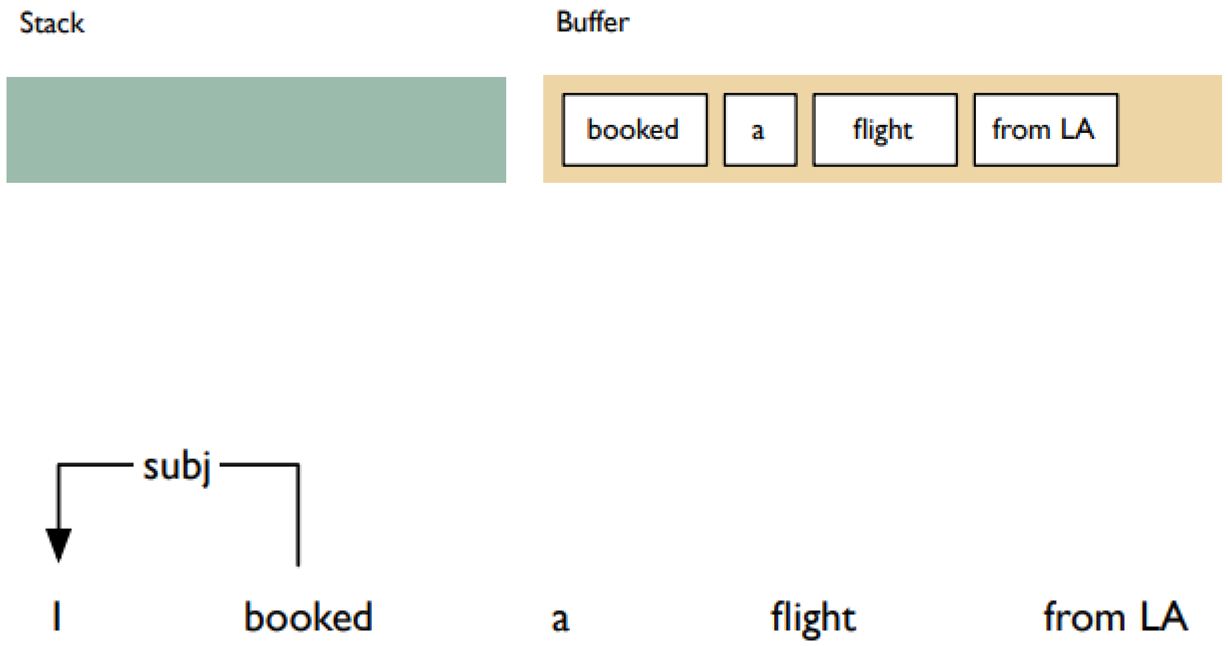
❖ Arc-eager transition-based parser



TRANS.-BASED

Ⅱ Dependency Parsing Algorithm

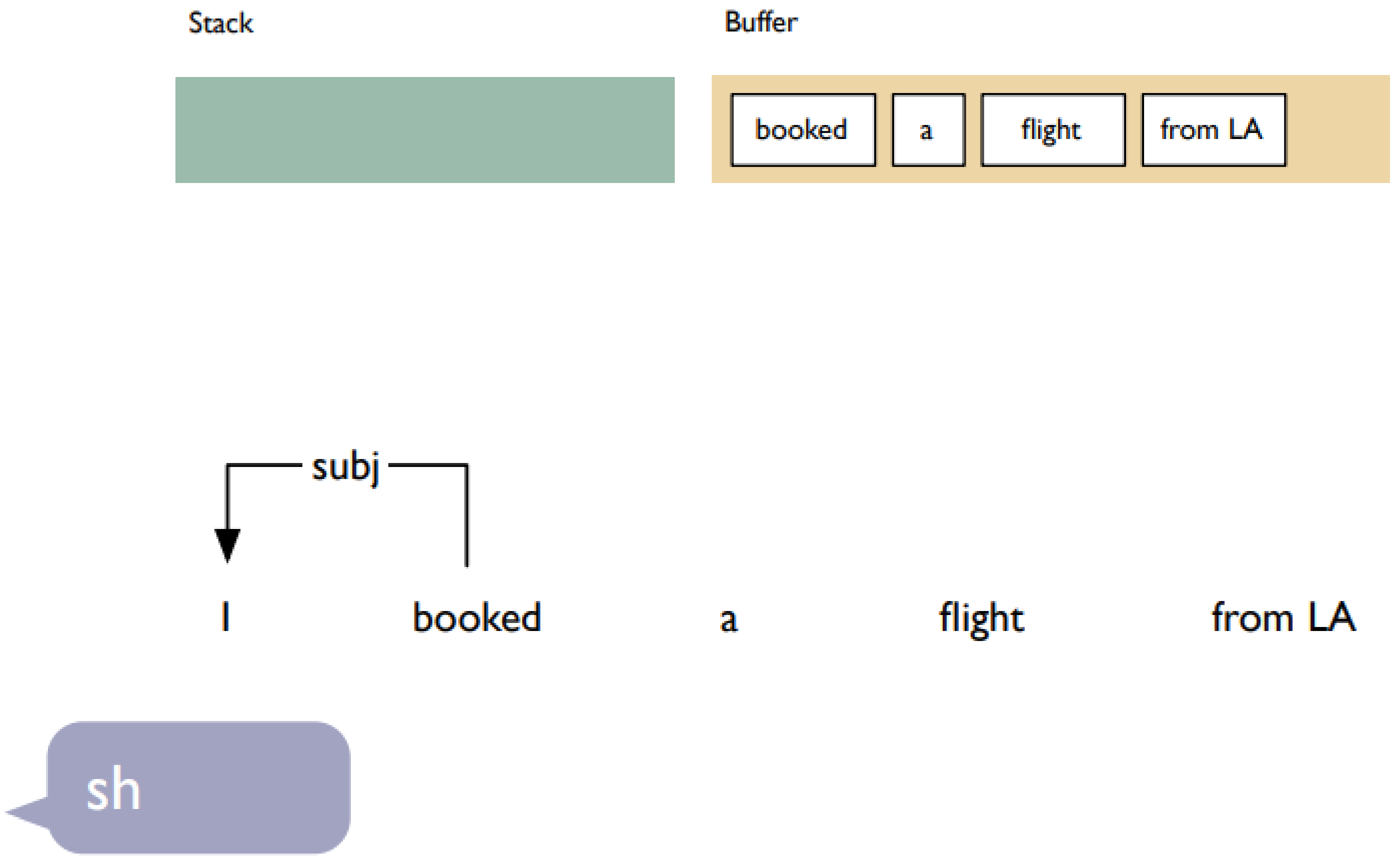
❖ Arc-eager transition-based parser



TRANS-BASED

Ⅱ Dependency Parsing Algorithm

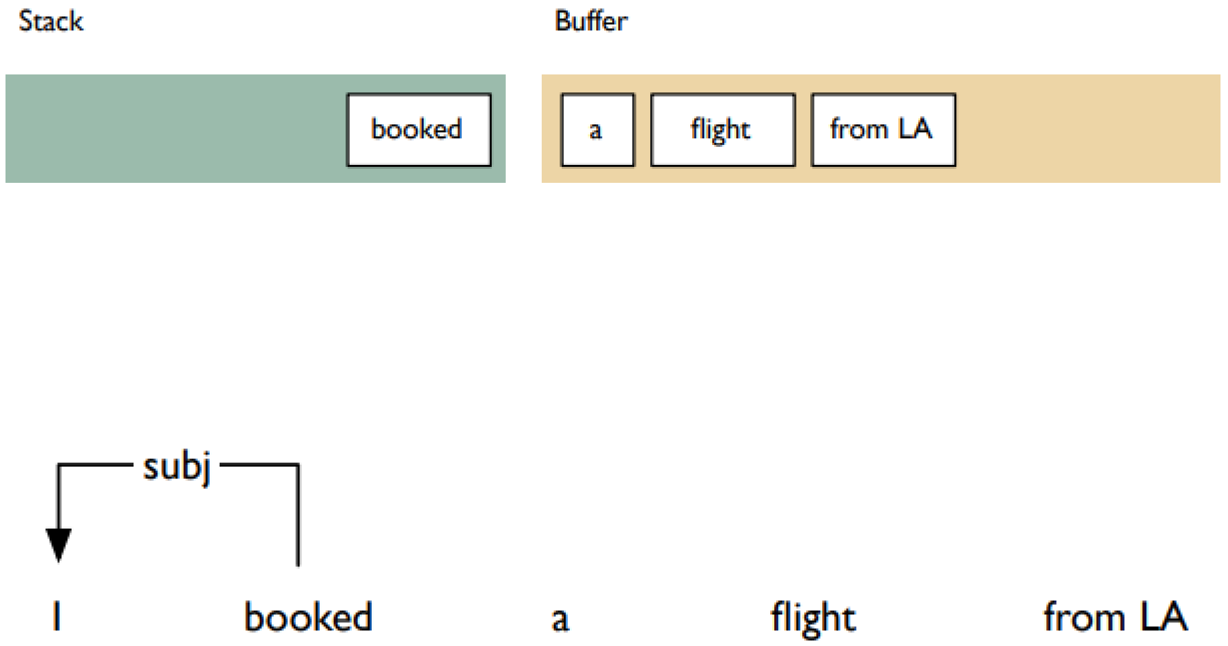
❖ Arc-eager transition-based parser



Ⅱ Dependency Parsing Algorithm

TRANS.-BASED

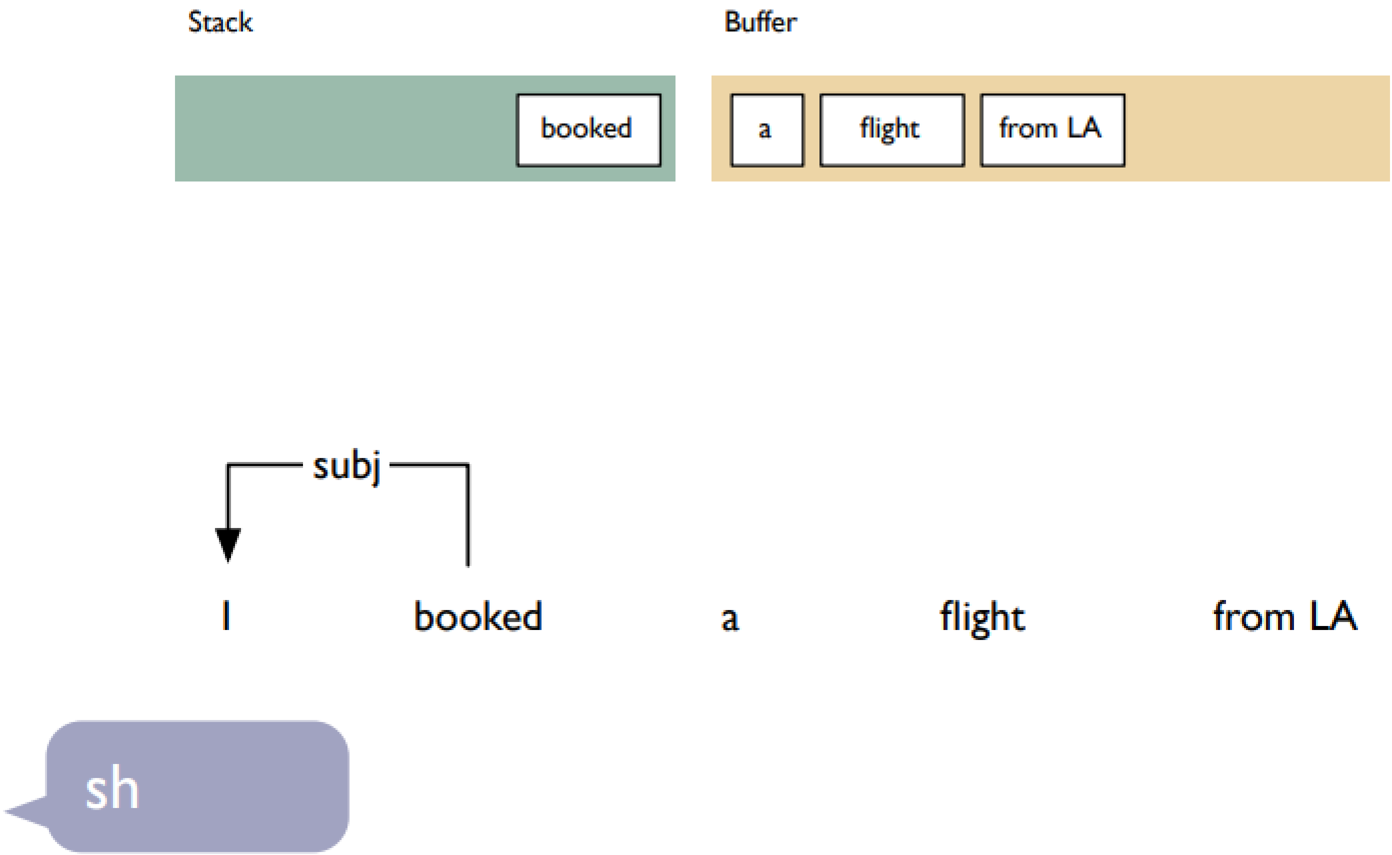
❖ Arc-eager transition-based parser



Ⅱ Dependency Parsing Algorithm

TRANS.-BASED

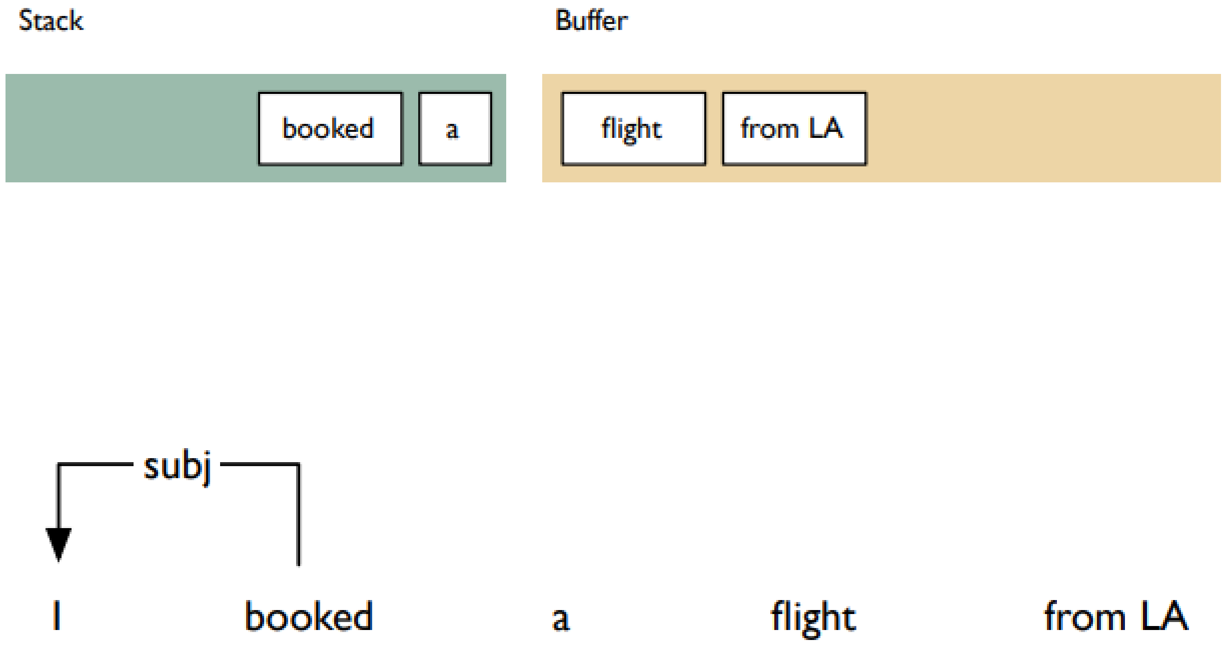
❖ **Arc-eager transition-based parser**



TRANS-BASED

Ⅱ Dependency Parsing Algorithm

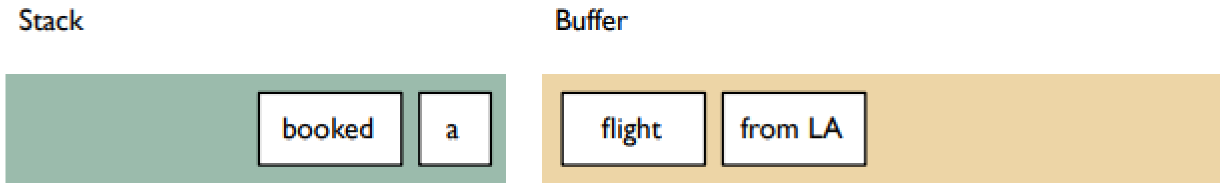
❖ **Arc-eager transition-based parser**

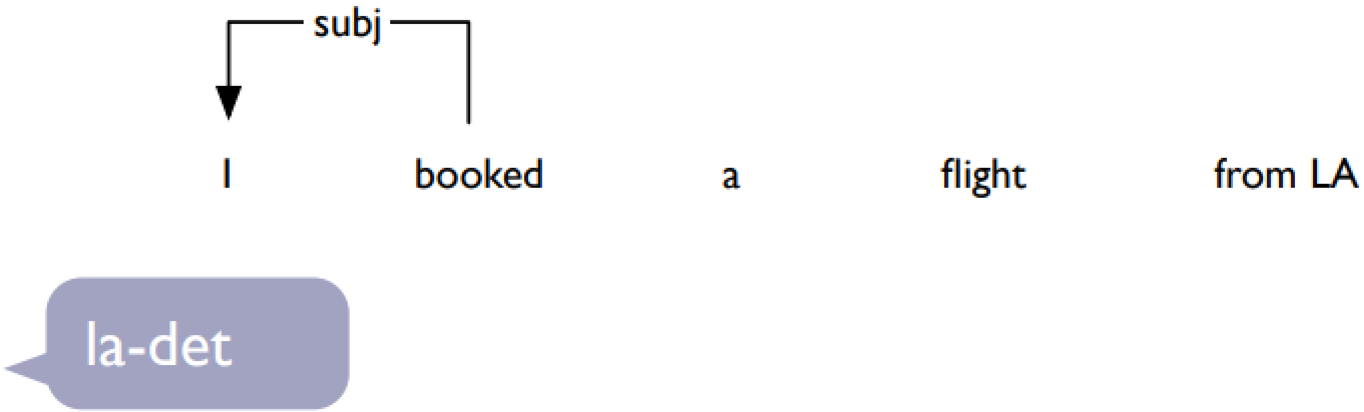


Ⅱ Dependency Parsing Algorithm

TRANS-BASED

❖ Arc-eager transition-based parser

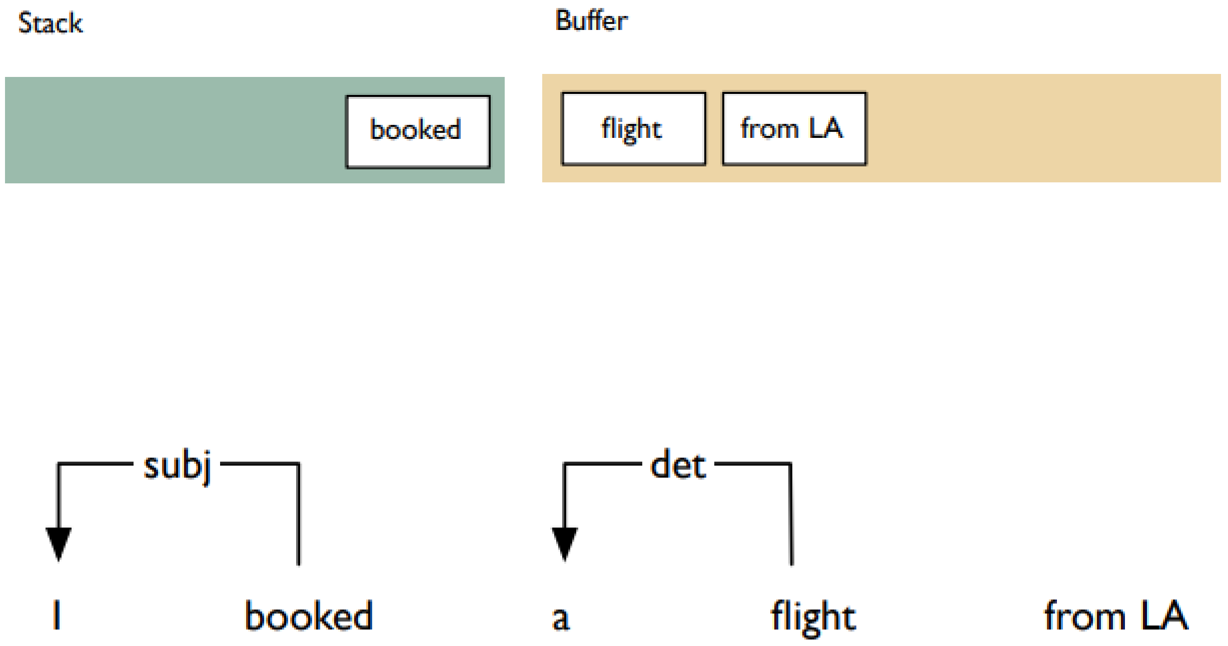




TRANS.-BASED

Ⅱ Dependency Parsing Algorithm

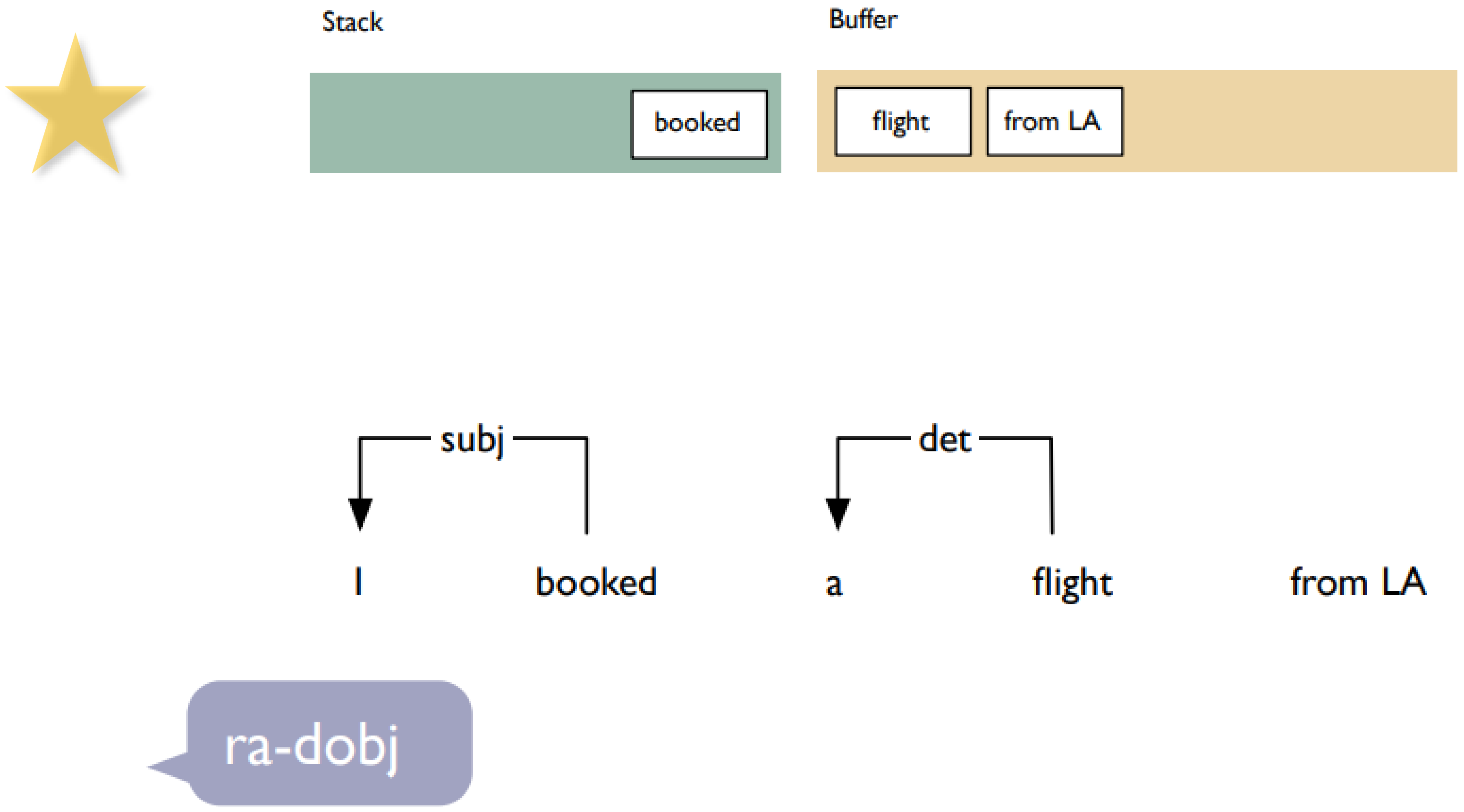
❖ Arc-eager transition-based parser



II Dependency Parsing Algorithm

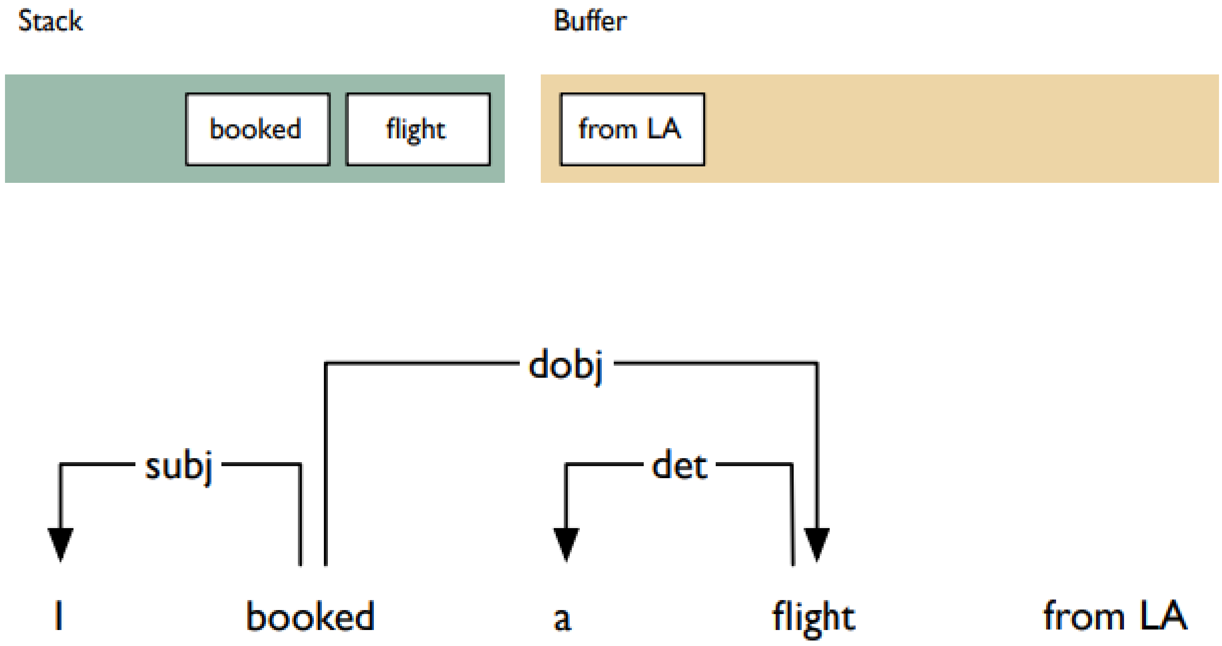
TRANS.-BASED

❖ Arc-eager transition-based parser



Ⅱ Dependency Parsing Algorithm

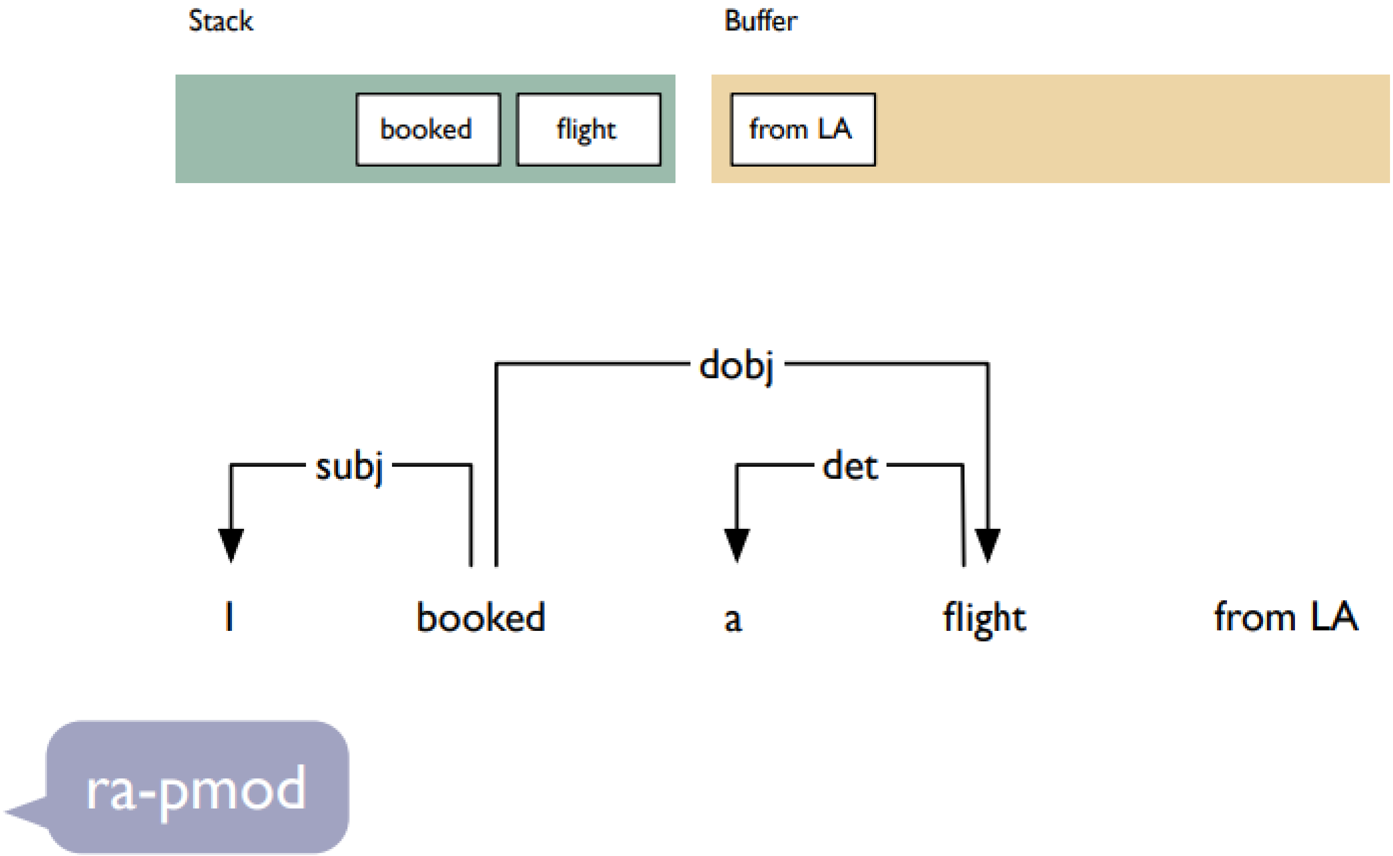
❖ **Arc-eager transition-based parser**



TRANS.-BASED

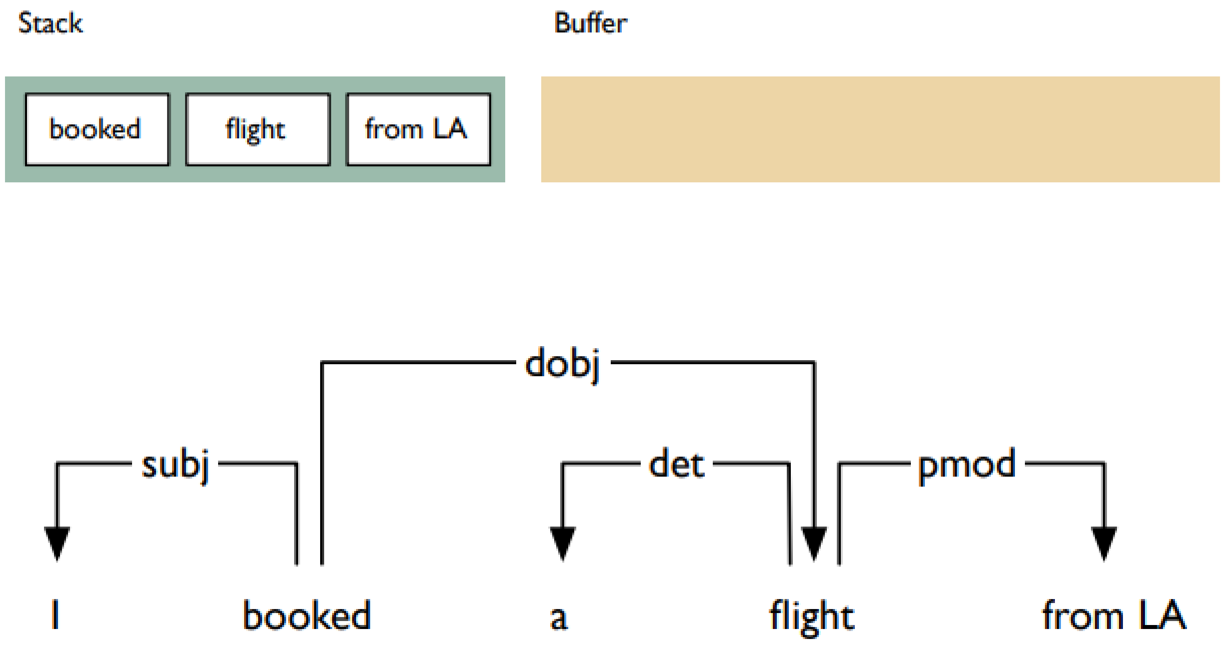
Ⅱ Dependency Parsing Algorithm

❖ Arc-eager transition-based parser



Ⅱ Dependency Parsing Algorithm

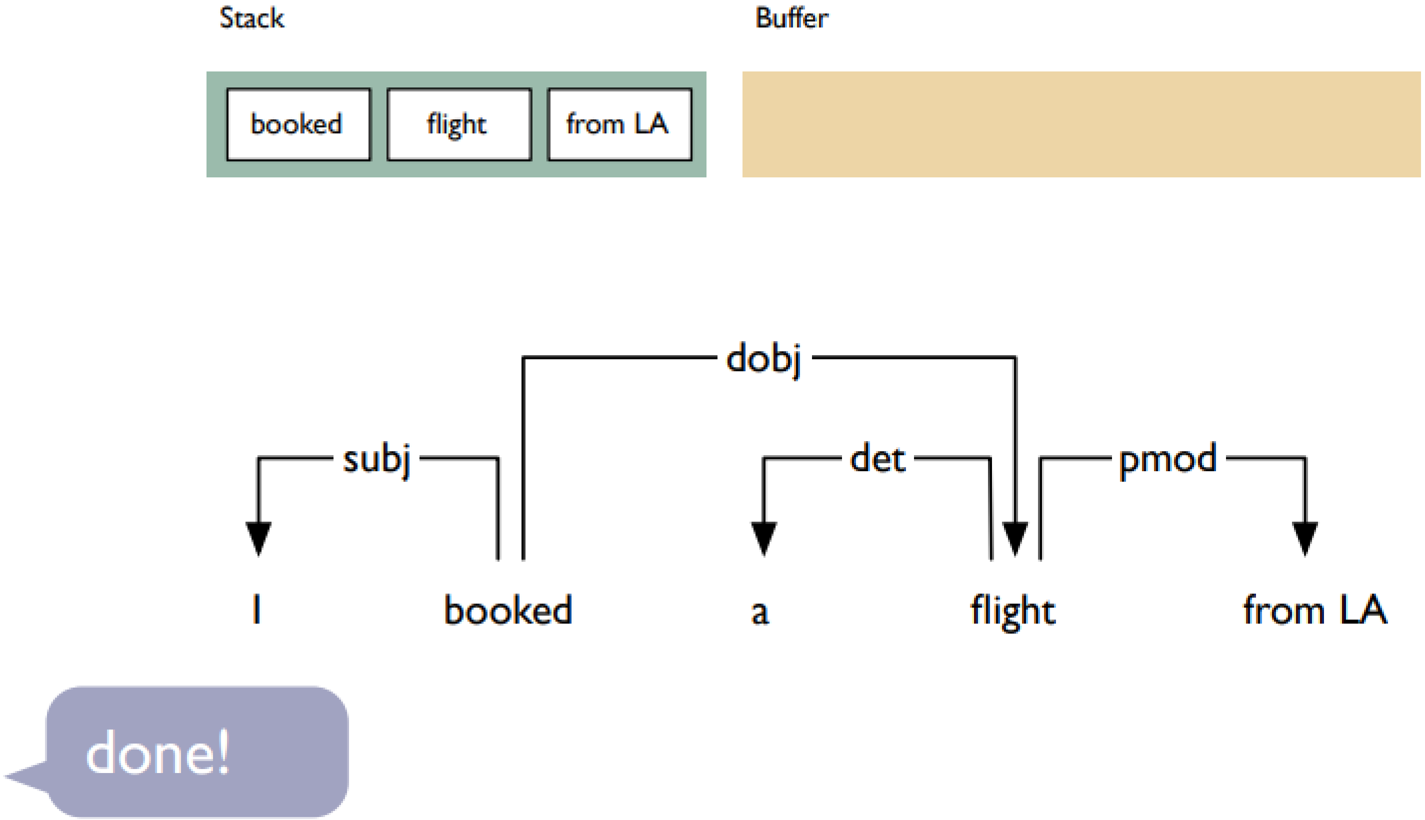
❖ Arc-eager transition-based parser



Ⅱ Dependency Parsing Algorithm

TRANS.-BASED

❖ Arc-eager transition-based parser



- **Top-down style /** can **create arcs earlier** than arc-standard model

Ⅱ Dependency Parsing Resources

❖ Treebank

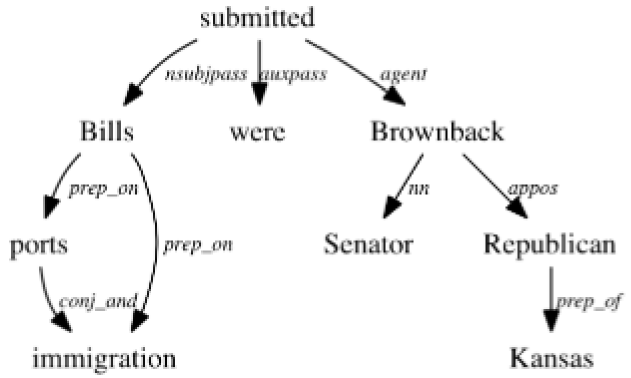
- Treebanks are corpora in which each sentence has been annotated with a syntactic analysis.
- The annotation process requires detailed guidelines and measures for quality control.
- Producing a high-quality treebank is both time-consuming and expensive.

| Sara Stymne. 2014. 'Treebank Grammars and Parser Evaluation', Syntactic analysis. Uppsala univ.
https://cl.lingfil.uu.se/~sara/kurser/5LN455-2014/lectures/5LN455-F4.pdf

Ⅱ Dependency Parsing Resources

❖ **Treebank**

- **Prague Dep. treebank 3.0 (2013)**
  - 1,506,484 words, 87,913 sentences
- **Danish Dep. treebank 1.0 (2004)**
  - 100,200 words, 5,540 sentences
- **SPMRL Shared Task Annotated Dataset (2013-2015)**
  - Penn Treebank format, need license
- **Stanford Dependencies (SD)**
  - https://nlp.stanford.edu/software/dependencies_manual.pdf
  - 50 relations



| https://en.wikipedia.org/wiki/Treebank#syntactic_treebanks
| http://universaldependencies.org/

II Dependency Parsing Resources

❖ Parser

- MaltParser (2007) :: transition-based
  - J. Hall, J. Nilsson and J. Nivre. http://www.maltparser.org/
  - J. Nivre. MaltParser: A language-independent system for data-driven dependency parsing
  - arc-standard/eager, projective/non-projective, …
- MSTParser (2005) :: graph-based
  - J. Baldrige & R. McDonald. https://www.seas.upenn.edu/~strctlrn/MSTParser/MSTParser.html
  - projective / non-projective
- Stanford's Neural Network Dependency Parser
  - https://nlp.stanford.edu/software/nndep.html
  - transition-based(2014) / graph-based(2017)
- SyntaxNet (2016) :: transition-based
  - Google. https://github.com/tensorflow/models/tree/master/research/syntaxnet
  - D, Andor et al., Globally Normalized Transition-Based Neural Networks
  - Parsey McParseface (trained English parser)

Ⅱ Dependency Parsing Resources

❖ Converting Tool

- The LTH Constituent-to-Dependency Conversion Tool for Penn-style Treebanks
  - used to prepare the English dependency treebanks in 2007-2009 CoNLL Sahred Task.
  - CoNLL-X format
  - http://nlp.cs.lth.se/software/treebank_converter/
- Stanford Dependencies
  - offering a converting option in the SD Parser
    - Penn Treebank(constituency) format → UD in CoNLL-U format
  - https://nlp.stanford.edu/software/stanford-dependencies.html

Ⅱ Universal Dependency

❖ What is Universal Dependency

- A **tagset** is a list of part-of-speech tags (POS tags for short), i.e. labels used to indicate the part of speech and sometimes also other grammatical categories (case, tense etc.) of each token in a text corpus.

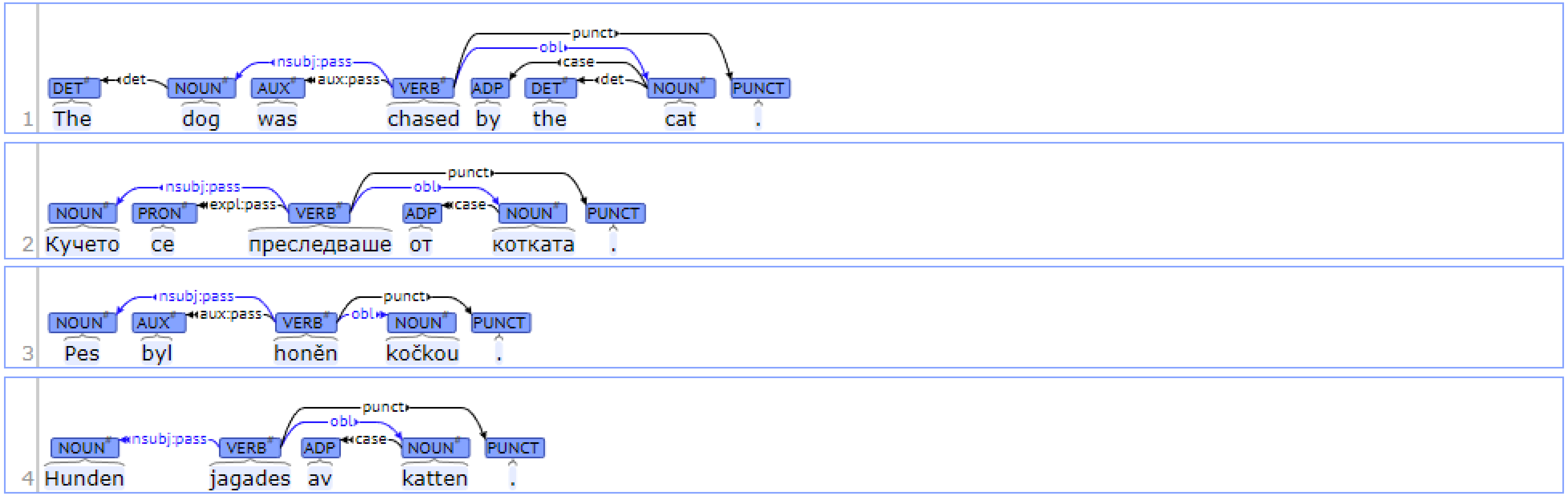
- **Universal POS tags** are part-of-speech marks used in Universal Dependencies (UD) which is a project that is developing cross-linguistically consistent treebank annotation for many languages. The annotation scheme is based on an evolution of (universal) Stanford dependencies (de Marneffe et al., 2006, 2008, 2014), Google universal part-of-speech tags (Petrov et al., 2012), and the Interset interlingua for morphosyntactic tagsets (Zeman, 2008).

| https://universaldependencies.org/u/overview/morphology.html#part-of-speech-tags
| https://www.sketchengine.eu/universal-pos-tags/

Ⅱ Universal Dependency

❖ What is Universal Dependency

- developing cross-linguistically consistent treebank annotation for many languages, with the goal of facilitating multilingual parser development, cross-lingual learning, and parsing research from a language typology perspective.



II Universal Dependency

❖ What is Universal Dependency

- **Universal Dependency Treebank**
  - **(GitHub)** https://github.com/UniversalDependencies
  - **(LINDAT/CLARIN)** https://lindat.mff.cuni.cz/repository/xmlui/handle/11234/1-2895

Ⅱ Universal Dependency

- **(LINDAT/CLARIN >KonText)** https://lindat.mff.cuni.cz/services/kontext/first_form?corpname=ud_23_ko_kaist_a

Hits: **6** | i.p.m.: **17.14** (related to the whole corpus) | ARF: **3.84** | Result is shuffled **1** / **1**

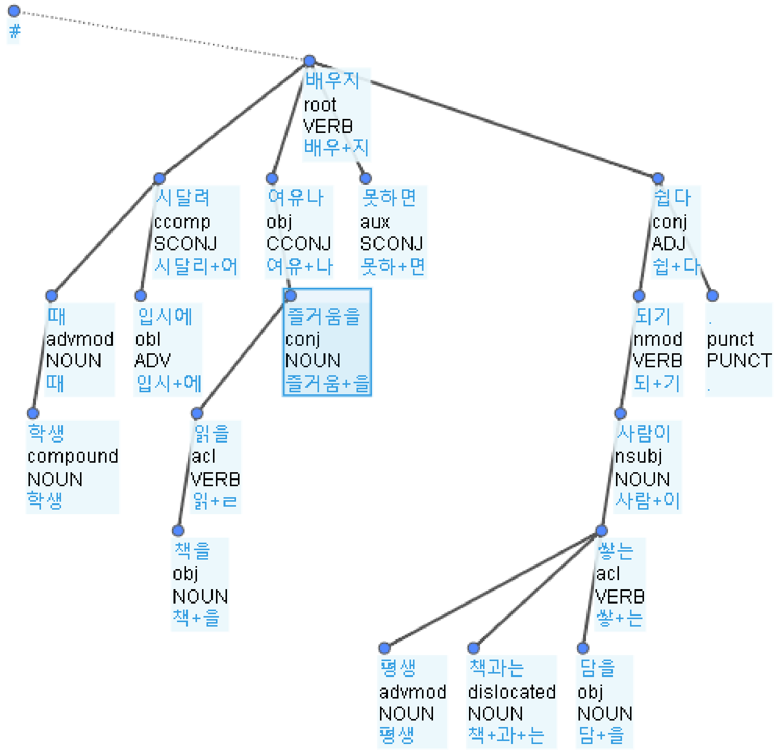
Line selection: simple

train	여러가지가 있지만 독서 교육이 제대로 이루어지지 못한 책임이 크다 .	**학생**	때 입시에 시달려 책을 읽을 여유나 즐거움을 배우지 못하면 평생
train	대중을 주 대상으로 삼고 그들에게 향유되는 노래를 의미한다 . 따라서	**학생**	, 지식인권의 노래는 논의 대상에서 제외될 것이다 . 지금 노동가요의
train	, 맥박과 일체화시킨다 . 그리고 이어 3연과 4연에서 젊은이 또는	**학생**	, 그리고 어린이의 손발과 움직임 , 숨결 속에서 조선의 맥박을

#학생 때 입시에 시달려 책을 읽을 여유나 즐거움을 배우지 못하면 평생 책과는 담을 쌓는 사람이 되기 쉽다 .

Hide empty attributes

deprel	conj
id	8
lc	즐거움을
lemma	즐거움+을
p_deprel	obj
p_id	7
p_lemma	여유+나
p_upos	CCONJ
p_word	여유나
p_xpos	ncn+jcj
parent	7
upos	NOUN
word	즐거움을
xpos	ncn+jco



II Universal Dependency

❖ What is Universal Dependency

▪ UD relations

	Nominals	Clauses	Modifier words	Function Words
Core arguments	nsubj obj iobj	csubj ccomp xcomp		
Non-core dependents	obl vocative expl dislocated	advcl	advmod* discourse	aux cop mark
Nominal dependents	nmod appos nummod	acl	amod	det clf case
Coordination	**MWE**	**Loose**	**Special**	**Other**
conj cc	fixed flat compound	list parataxis	orphan goeswith reparandum	punct root dep

- acl: clausal modifier of noun (adjectival clause)
- advcl: adverbial clause modifier
- advmod: adverbial modifier
- amod: adjectival modifier
- appos: appositional modifier
- aux: auxiliary
- case: case marking
- cc: coordinating conjunction
- ccomp: clausal complement
- clf: classifier
- compound: compound
- conj: conjunct
- cop: copula
- csubj: clausal subject
- dep: unspecified dependency
- det: determiner
- discourse: discourse element
- dislocated: dislocated elements
- expl: expletive
- fixed: fixed multiword expression
- flat: flat multiword expression
- goeswith: goes with
- iobj: indirect object
- list: list
- mark: marker
- nmod: nominal modifier
- nsubj: nominal subject
- nummod: numeric modifier
- obj: object
- obl: oblique nominal
- orphan: orphan
- parataxis: parataxis
- punct: punctuation
- reparandum: overridden disfluency
- root: root
- vocative: vocative
- xcomp: open clausal complement

| de Marneffe et al. 2014. Universal Stanford Dependencies: A cross-linguistic typology
| https://universaldependencies.org/u/dep/

Ⅱ Universal Dependency

❖ Universal POS tags for Universal Dependency

Open class words	Closed class words	Other
ADJ	ADP	PUNCT
ADV	AUX	SYM
INTJ	CCONJ	X
NOUN	DET	
PROPN	NUM	
VERB	PART	
	PRON	
	SCONJ	

- ADJ: adjective
- ADP: adposition
- ADV: adverb
- AUX: auxiliary
- CCONJ: coordinating conjunction
- DET: determiner
- INTJ: interjection
- NOUN: noun
- NUM: numeral
- PART: particle
- PRON: pronoun
- PROPN: proper noun
- PUNCT: punctuation
- SCONJ: subordinating conjunction
- SYM: symbol
- VERB: verb
- X: other

| https://universaldependencies.org/u/pos/index.html#universal-pos-tags
| https://universaldependencies.org/v2/postags.html

II Evaluation Metrics

❖ Methodology for evaluating parsers

- Apply them to a test set taken from a treebank
  and **compare** the output of the parser to the gold standard annotation
  found in the treebank

❖ Metrics

- Attachment score
  - the percentage of words that have the correct head and label
  - **Unlabeled Attachment Score (UAS)**
    - does not consider the semantic relation(e.g. Subj)
  - **Labeled Attachment Score (LAS)**
    - requires a semantic correct label

| S. Kubler, R. McDonald, J. Nivre. 2009. Dependency Parsing. p. 79

II Evaluation Metrics

❖ LAS

- **word-based** : how many words were parsed correctly
- **sentence-based** : calculate for each sentence what the percentage of correct dependencies is and then average over the sentences
- **ex]** a test set containing **2 sentences**
  - the percentage of correct dependencies
    - 1^{st} sentence: 10 中 9 (words)
    - 2^{nd} sentence: 45 中 15 (words)

  - **LAS_w** = (9+15)/(10+45) = 0.436 ← micro-avg LAS

  - **LAS_s** = (9/10 + 15/45) / 2 = 0.617 ← macro-avg LAS

| S. Kubler, R. McDonald, J. Nivre. 2009. Dependency Parsing. p. 80

Deep-Learning Approaches to Dependency Parsing

Neural Basic Theory of Dependency Parsing

Dependency Parsing with Deep-Learning

Ⅲ Neural Basic Theory of Dependency Parsing

Graph-Based Model

Factored **the score** of each head-dependent relation

Transition-Based Model

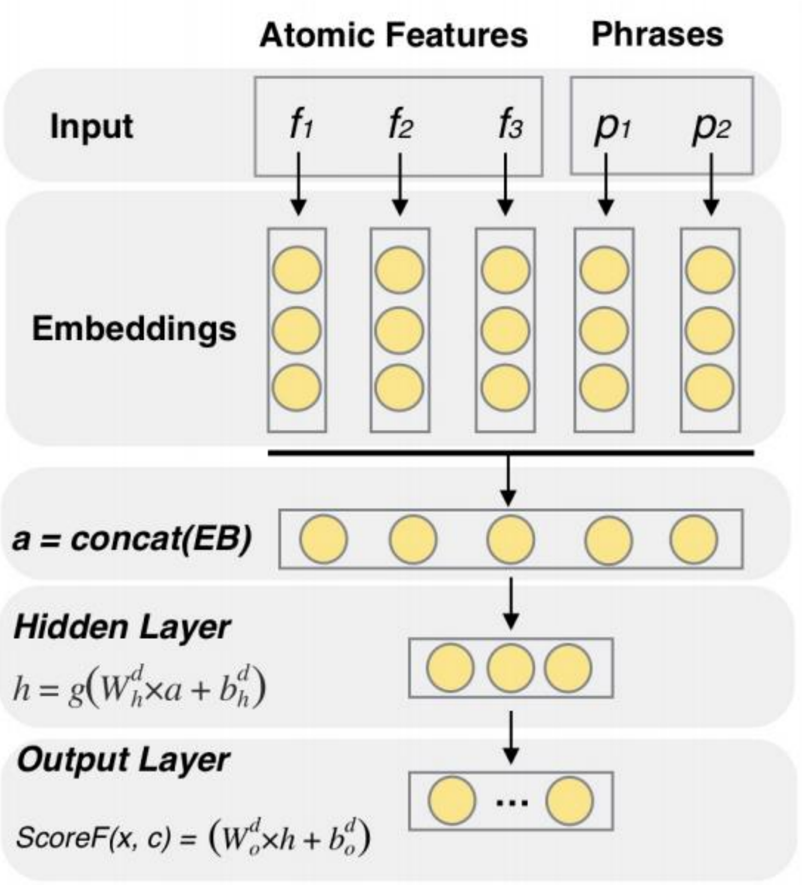
Determine **transition action**

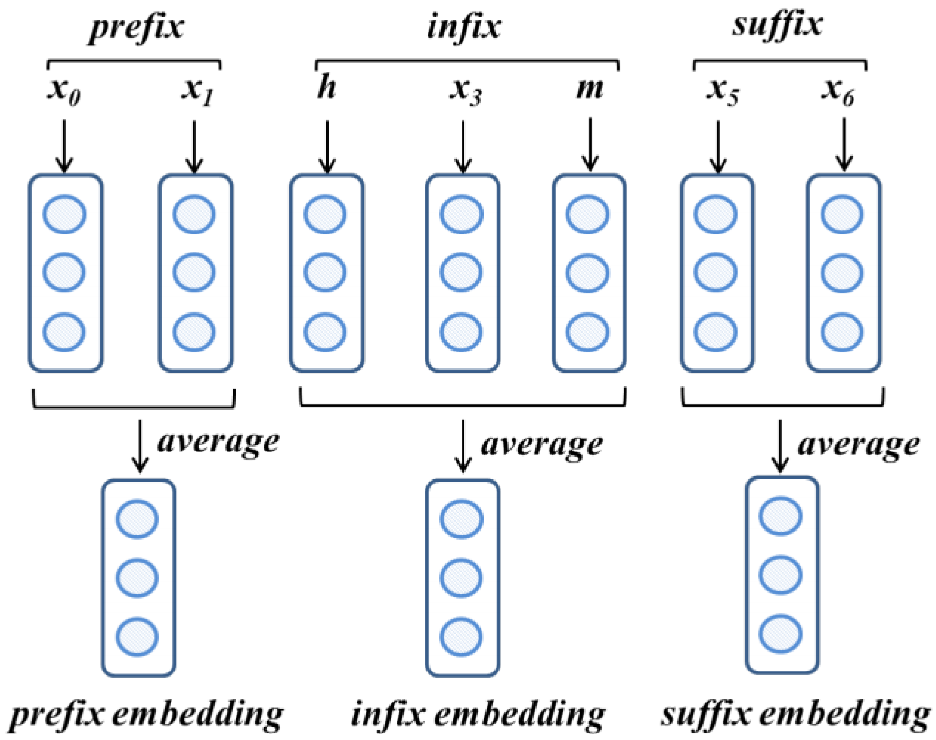
GRAPH-BASED

III Neural Basic Theory of Dependency Parsing

❖ **An Effective Neural Network Model for Graph-based Dependency Parsing**

(Pei et al. 2015. ACL)





III Neural Basic Theory of Dependency Parsing

❖ **A fast and Accurate Dependency Parser using Neural Networks**

(Chen & Manning. 2014. EMNLP)

Softmax layer:

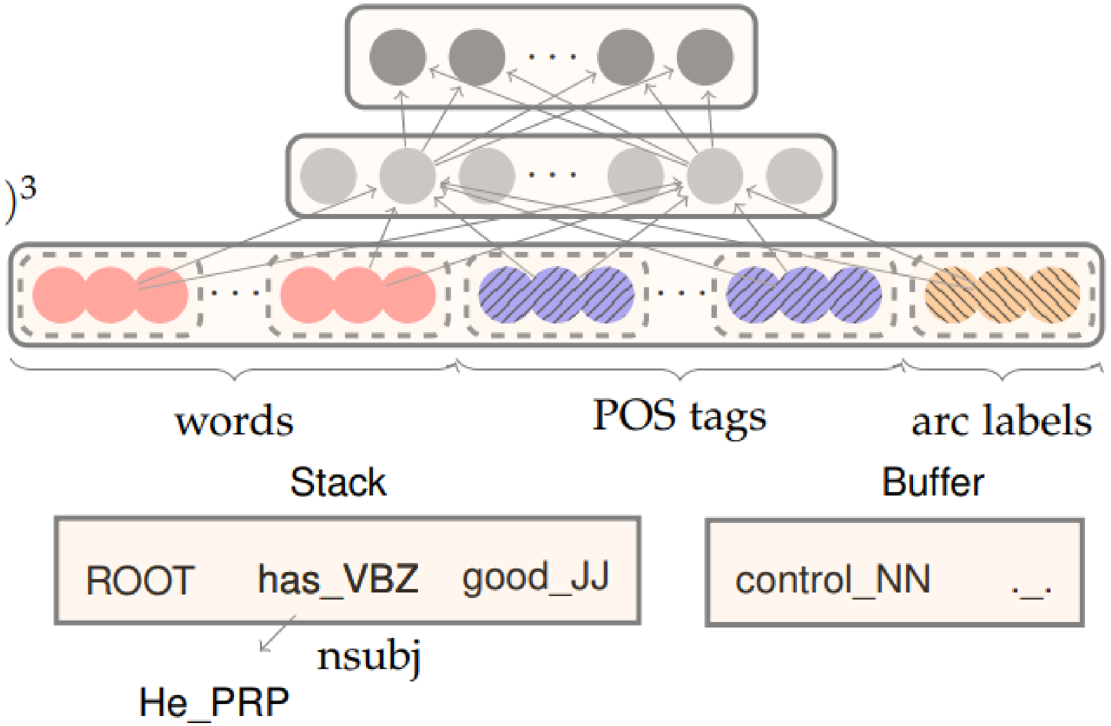
$$p = \texttt{softmax}(W_2 h)$$

Hidden layer:

$$h = (W_1^w x^w + W_1^t x^t + W_1^l x^l + b_1)^3$$

Input layer: $[x^w, x^t, x^l]$

Configuration



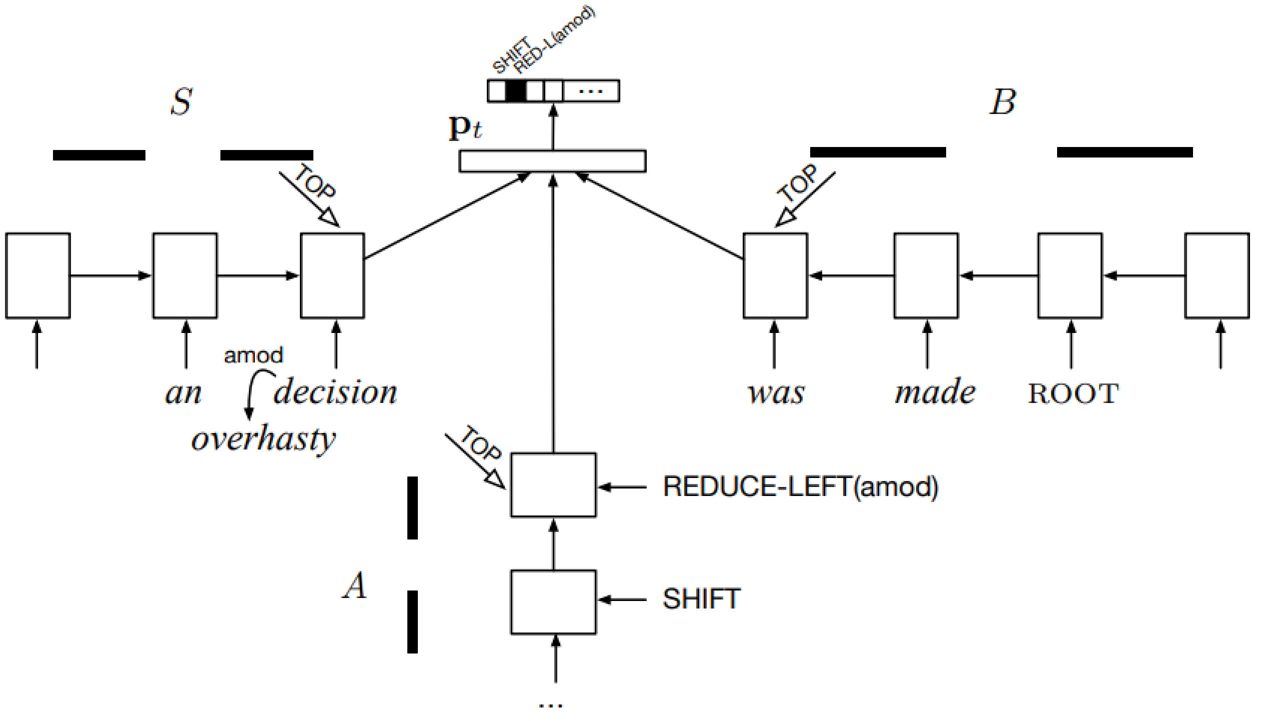
III Dependency Parsing with Deep-Learning

❖ Transition-Based Dependency Parsing with Stack Long Short-Term Memory

(C. Dyer et al. ACL 2015)

	Development		Test	
	UAS	LAS	UAS	LAS
S-LSTM	**93.2**	**90.9**	**93.1**	**90.9**
−POS	93.1	90.4	92.7	90.3
−pretraining	92.7	90.4	92.4	90.0
−composition	92.7	89.9	92.2	89.6
S-RNN	92.8	90.4	92.3	90.1
C&M (2014)	92.2	89.7	91.8	89.6

Table 1: English parsing results (SD)



$$\mathbf{p}_t = \max\left\{\mathbf{0}, \mathbf{W}[\mathbf{s}_t; \mathbf{b}_t; \mathbf{a}_t] + \mathbf{d}\right\}$$

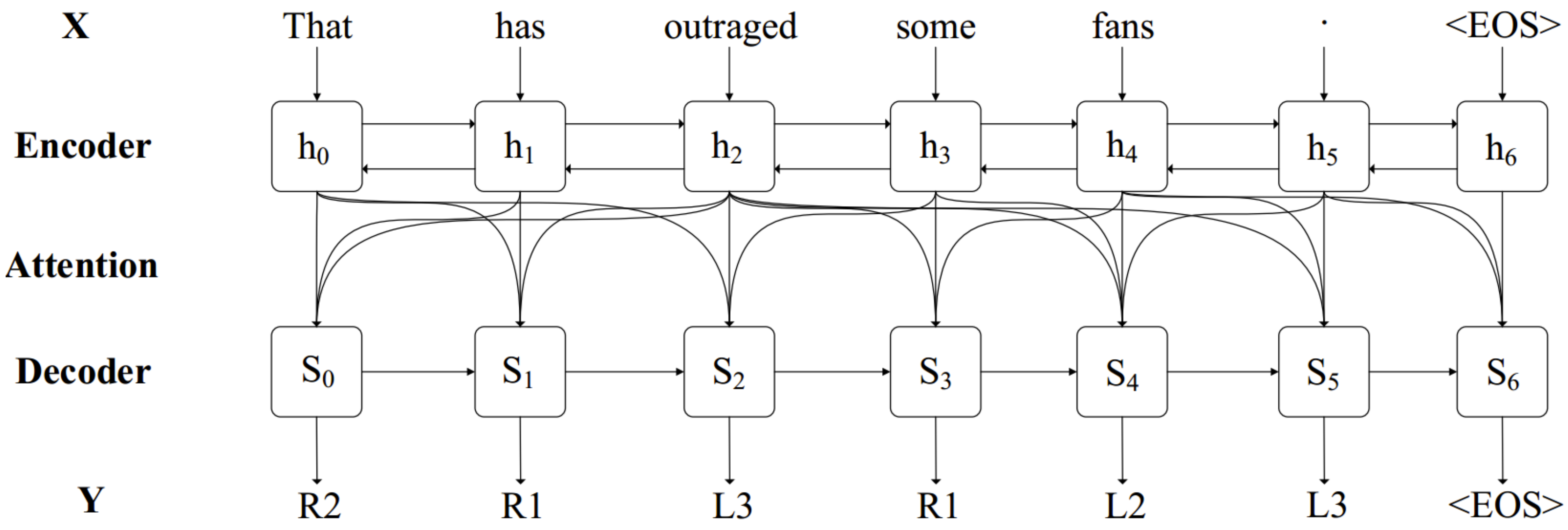
$$\mathbf{c} = \tanh\left(\mathbf{U}[\mathbf{h}; \mathbf{d}; \mathbf{r}] + \mathbf{e}\right)$$

Ⅱ Dependency Parsing with Pointer Network

❖ Seq2seq Dependency Parsing

(Z. LI et al. COLING 2018)

- **Training**
  - convert each **head** into a **position** representation
- **Inferencing**
  - the parser **picks a head** for each word by predicting the **relative position** of the head.



Applied Deep-Learning for Dependency Parsing

Supervised Dependency Parsing

Unsupervised Dependency Parsing

Neural Dependency Parsing for Korean

IV Supervised Dependency Parsing

*** Penn Treebank**

Model		Year	POS	UAS	LAS	Paper
CVT + Multi-Task (Clark et al.)	G	2018	97.74	96.61	95.02	Semi-Supervised Sequence Modeling with Cross-View Training
Deep Biaffine (Dozat and Manning)	G	2017	97.3	95.74	94.08	Deep Biaffine Attention for Neural Dependency Parsing
jPTDP (Nguyen and Verspoor)	G	2018	97.97	94.51	92.87	An improved neural network model for joint POS tagging and dependency parsing
SyntaxNet (Andor et al.)	T	2016	97.44	94.61	92.79	Globally Normalized Transition-Based Neural Networks
Distilled neural FOG (Kuncoro et al.)	G	2016	97.3	94.26	92.06	Distilling an Ensemble of Greedy Dependency Parsers into One MST Parser
Distilled transition-based parser (Liu et al.)	T	2018	97.3	94.05	92.14	Distilling Knowledge for Search-based Structured Prediction
Weiss et al.	T	2015	97.44	93.99	92.05	Structured Training for Neural Network Transition-Based Parsing
BIST transition-based parser (Kiperwasser and Goldberg)	T	2016	97.3	93.9	91.9	Simple and Accurate Dependency Parsing Using Bidirectional LSTM Feature Representations
Arc-hybrid (Ballesteros et al.)	T	2016	97.3	93.56	91.42	Training with Exploration Improves a Greedy Stack-LSTM Parser
BIST graph-based parser (Kiperwasser and Goldberg)	G	2016	97.3	93.1	91.0	Simple and Accurate Dependency Parsing Using Bidirectional LSTM Feature Representations

| https://github.com/sebastianruder/NLP-progress/blob/master/english/dependency_parsing.md

IV Supervised Dependency Parsing

*** CoNLL 2018 Shared Task, UD**

Model	LAS	MLAS	BLEX	Paper
Stanford (Qi et al.)	74.16	62.08	65.28	Universal Dependency Parsing from Scratch
HIT-SCIR (Che et al.)	75.84	59.78	65.33	Towards Better UD Parsing: Deep Contextualized Word Embeddings, Ensemble, and Treebank Concatenation
TurkuNLP (Kanerva et al.)	73.28	60.99	66.09	Turku Neural Parser Pipeline: An End-to-End System for the CoNLL 2018 Shared Task
UDPipe Future (Straka)	73.11	61.25	64.49	UDPipe 2.0 Prototype at CoNLL 2018 UD Shared Task

- **MLAS** (morphology-aware labeled attachment score) : evaluation of POS tags and morphological features
- **BLEX** (bi-lexical dependency score) : combines content-word relations with lemmatization (but not with tags and features)

| https://github.com/sebastianruder/NLP-progress/blob/master/english/dependency_parsing.md

IV Unsupervised Dependency Parsing

* Penn Treebank

Model	Year	UAS	Paper
Iterative reranking (Le & Zuidema)	2015	66.2	Unsupervised Dependency Parsing - Let's Use Supervised Parsers
Combined System (Spitkovsky et al.)	2013	64.4	Breaking Out of Local Optima with Count Transforms and Model Recombination - A Study in Grammar Induction
Tree Substitution Grammar DMV (Blunsom & Cohn)	2010	55.7	Unsupervised Induction of Tree Substitution Grammars for Dependency Parsing
Shared Logistic Normal DMV (Cohen & Smith)	2009	41.4	Shared Logistic Normal Distributions for Soft Parameter Tying in Unsupervised Grammar Induction
DMV (Klein & Manning)	2004	35.9	Corpus-Based Induction of Syntactic Structure - Models of Dependency and Co nstituency

| https://github.com/sebastianruder/NLP-progress/blob/master/english/dependency_parsing.md

IV Neural Dependency Parsing for Korean

* Sejong Treebank

Model	Year	F1	UAS	LAS	Paper
NNLM+ReLU+dropout+MI feat. (이창기, 김준석, 김정희)	2014		90.37	88.17	딥 러닝을 이용한 한국어 의존 구문 분석
LSTM, Transition-based (이건일, 이종혁)	2015		90.33		순환 신경망을 이용한 전이 기반 한국어 의존 구문 분석 (KIBS95, 97)
Stack LSTM, Transition-based (나승훈, 신종훈, 김강일)	2016		90.44	88.17	Stack LSTM 기반 한국어 의존 파싱을 위한 음절과 형태소의 결합 단어 표상 방법
의존 관계명 부착 (안재현, 이호경, 고영중)	2016	96.01			의존 경로와 음절단위 의존 관계명 분포 기반의 Bidirectional LSTM CRFs를 이용한 한국어 의존 관계명 레이블링
Pointer network (박천음, 이창기)	2016		91.65	89.34	멀티 태스크 학습 기반 포인터 네트워크를 이용한 한국어 의존 구문 분석
Graph-based (나승훈, 이건일, 신종훈, 김강일)	2017		91.78	89.76	Deep Biaffine Attention을 이용한 한국어 의존 파싱
Pointer network (박천음, 이창기)	2017		91.79	89.50	포인터 네트워크를 이용한 한국어 의존 구문 분석

| https://github.com/papower1/Awesome-Korean-NLP-Papers#dependency-parsing

Challenges and Future Directions

V Challenges and Future Directions

- Korean UD treebank
  - Dependency treebank
  - Converting tool

- Parser domain adaptation
  - How to capture the domain differences
    and improve the models for the target domain?

- Hybrid way

감 사 합 니 다.

남궁영
한국해양대학교 컴퓨터공학과
자연언어처리실험실
young_ng@kmou.ac.kr

2019. 04. 17.