Transition-based parsing: Formulation

Given an input sentence, we want to obtain its dependency graph.

We assume we have labeled data (dependency graph for several sentences already known).

Deterministic Parsing

Basic idea

Derive a single syntactic representation (dependency graph) through a deterministic sequence of elementary parsing actions

Deterministic Parsing

Basic idea

Derive a single syntactic representation (dependency graph) through a deterministic sequence of elementary parsing actions

Configurations

A parser configuration is a triple $c = (S, B, A)$, where

- S : a stack $[\ldots, w_i]_S$ of partially processed words,
- B : a buffer $[w_j, \ldots]_B$ of remaining input words,
- A : a set of labeled arcs (w_i, d, w_j).

Sentence (partly processed): He sent her a letter.

Stack	**Buffer**	**Arcs**
[sent, her, a]$_S$	[letter, .]$_B$	He $\xleftarrow{\text{SBJ}}$ sent

Transition System

A transition system for dependency parsing is a quadruple $S = (C, T, c_s, C_t)$, where

- C is a set of configurations,
- T is a set of transitions, such that $t : C \rightarrow C$,
- c_s is an initialization function
- $C_t \subseteq C$ is a set of terminal configurations.

Initialization function takes the given sentence as input, and gives the initial configuration (element of set C).

Each transition (element of set T) takes one configuration to another configuration.

Transition System

A transition system for dependency parsing is a quadruple $S = (C, T, c_s, C_t)$, where

- C is a set of configurations,
- T is a set of transitions, such that $t : C \rightarrow C$,
- c_s is an initialization function
- $C_t \subseteq C$ is a set of terminal configurations.

A transition sequence for a sentence x is a set of configurations
$C_{0,m} = (c_o, c_1, \ldots, c_m)$ such that
$c_o = c_s(x)$, $c_m \in C_t$, $c_i = t(c_{i-1})$ for some $t \in T$

Transition System

A transition system for dependency parsing is a quadruple $S = (C, T, c_s, C_t)$, where

- C is a set of configurations,
- T is a set of transitions, such that $t : C \to C$,
- c_s is an initialization function
- $C_t \subseteq C$ is a set of terminal configurations.

A transition sequence for a sentence x is a set of configurations
$C_{0,m} = (c_o, c_1, \ldots, c_m)$ such that
$c_o = c_s(x)$, $c_m \in C_t$, $c_i = t(c_{i-1})$ for some $t \in T$

Initialization: $([]_S, [w_1, \ldots, w_n]_B, \{\})$
Termination: $(S, []_B, A)$

Initial configuration: Empty stack, all words yet to be processed, no arcs yet.
Final configuration: Buffer is empty.

Transitions for Arc-Eager Parsing

Name of transition | **Input configuration / Resulting configuration** | **Necessary condition for applying this transition**

$$\text{Left-Arc}(d)\ \frac{([\ldots, w_i]_S,\ [w_j, \ldots]_B,\ A)}{([\ldots]_S,\ [w_j, \ldots]_B,\ A \cup \{(w_j, d, w_i)\})}\ \neg\text{HEAD}(w_i)$$

$$\text{Right-Arc}(d)\ \frac{([\ldots, w_i]_S,\ [w_j, \ldots]_B,\ A)}{([\ldots, w_i, w_j]_S,\ [\ldots]_B,\ A \cup \{(w_i, d, w_j)\})}$$

$$\text{Reduce}\ \frac{([\ldots, w_i]_S,\ B,\ A)}{([\ldots]_S,\ B,\ A)}\ \text{HEAD}(w_i)$$

$$\text{Shift}\ \frac{([\ldots]_S,\ [w_i, \ldots]_B,\ A)}{([\ldots, w_i]_S,\ [\ldots]_B,\ A)}$$

Transitions for Arc-Eager Parsing

Name of transition

Input configuration / **Resulting configuration**

Necessary condition for applying this transition

$$\text{Left-Arc}(d)\ \frac{([\ldots, w_i]_S,\ [w_j, \ldots]_B,\ A)}{([\ldots]_S,\ [w_j, \ldots]_B,\ A \cup \{(w_j, d, w_i)\})}\ \neg\text{HEAD}(w_i)$$

Wi is the top of the stack

Wj is the first (yet to be processed) word in the buffer

Remove Wi from stack and add a left-arc Wi <— Wj only if Wi does not have a Head already. Here Wj will be the Head and Wi will be the dependent.

(Single Head condition: every word can have at most one syntactic Head)

Transitions for Arc-Eager Parsing

Name of transition — **Input configuration / Resulting configuration** — **Necessary condition for applying this transition**

$$\text{Left-Arc}(d)\ \frac{([\ldots, w_i]_S,\ [w_j, \ldots]_B,\ A)}{([\ldots]_S,\ [w_j, \ldots]_B,\ A \cup \{(w_j, d, w_i)\})}\ \neg\text{HEAD}(w_i)$$

Any relation that Wi has with a word appearing earlier (than Wj), has already been captured in A.

Now we are removing Wi from stack. So we will not be able to capture any relation between Wi and a word Wk that comes after Wj (in the buffer).

Can a word Wk coming after Wj have any relation with Wi?
- can Wk be the Head of Wi?
- can Wi be the Head of Wk?

Transitions for Arc-Eager Parsing

Name of transition | $\dfrac{\textbf{Input configuration}}{\textbf{Resulting configuration}}$ | **Necessary condition for applying this transition**

$$\text{Left-Arc}(d)\ \frac{([\ldots, w_i]_S,\ [w_j, \ldots]_B,\ A)}{([\ldots]_S,\ [w_j, \ldots]_B,\ A \cup \{(w_j, d, w_i)\})}\ \neg\text{HEAD}(w_i)$$

$$\text{Right-Arc}(d)\ \frac{([\ldots, w_i]_S,\ [w_j, \ldots]_B,\ A)}{([\ldots, w_i, w_j]_S,\ [\ldots]_B,\ A \cup \{(w_i, d, w_j)\})}$$

Wj pushed onto stack. Wi —> Wj added.

$$\text{Reduce}\ \frac{([\ldots, w_i]_S,\ B,\ A)}{([\ldots]_S,\ B,\ A)}\ \text{HEAD}(w_i)$$

If Wi (top of the stack) already has a Head, remove it from stack.

$$\text{Shift}\ \frac{([\ldots]_S,\ [w_i, \ldots]_B,\ A)}{([\ldots, w_i]_S,\ [\ldots]_B,\ A)}$$

The word at the beginning of the buffer is pushed onto the stack.

Now we will take an example sentence and its Dependency Tree, and derive the transition sequence and configurations.

Basically, we want to derive the transitions such that we can get all the arcs in the given Dependency Tree.

DT: given Dependency Tree

In a certain configuration (Stack S, Buffer B, Arcs A):
top(S) - the word at the top of the Stack
first(B) - the first word in the buffer

Rules for adding transitions to A:

Left-Arc (LA) if DT contains top(S) <— first(B). [to be used if top(S) does not have a head]

Right-Arc (RA) if DT contains top(S) —> first(B)

Reduce (RE) if there exists w < top(S) such that DT contains either w —> first(B) or w <— first(B) [to be used if top(S) already has a head]

Shift (SH) otherwise

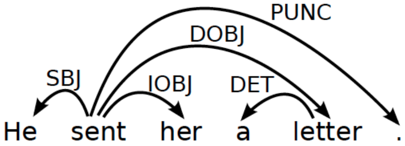
Parse Example

Sentence: He sent her a letter

Initial configuration:
Empty Stack and Arcs
All words in the buffer

Transitions:

Stack	**Buffer**	**Arcs**
$[\ ]_S$	$[\text{He, sent, her, a, letter, .}]_B$	

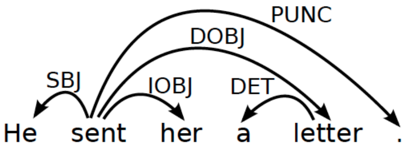


Parse Example

What should be the first transition? At this stage, only Shift can be applied.

Transitions:

Stack	Buffer	Arcs
$[\ ]_S$	$[\text{He, sent, her, a, letter, .}]_B$	

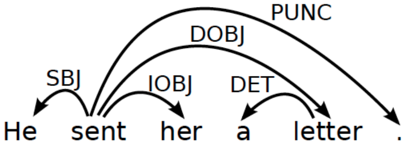


Parse Example

See the dependency tree. There is a Left-arc from “sent” to “he”. So apply Left-arc transition.

Transitions: SH

Stack	**Buffer**	**Arcs**
$[\text{He}]_S$	$[\text{sent, her, a, letter, .}]_B$	



Example

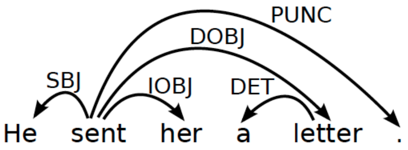
What will be the next transition?

If the stack is empty, the only possible transition is Shift.

Parse Example

Transitions: SH-LA

Stack	**Buffer**	**Arcs**
$[\ ]_S$	$[\text{sent, her, a, letter, .}]_B$	He $\xleftarrow{\text{SBJ}}$ sent

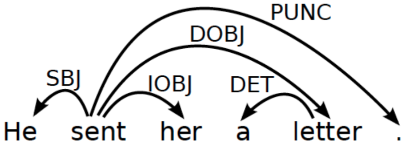


Parse Example

**From the given parse tree, we see a Right-Arc from “sent” to “her”.
So next transition will be a Right-Arc.**

Transitions: SH-LA-SH

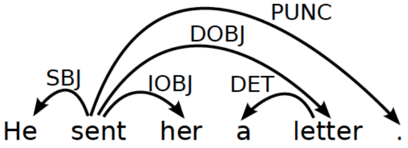
Stack	**Buffer**	**Arcs**
$[\text{sent}]_S$	$[\text{her, a, letter, .}]_B$	He $\xleftarrow{\text{SBJ}}$ sent



Parse Example

Transitions: SH-LA-SH-RA

Stack	Buffer	Arcs
$[\text{sent, her}]_S$	$[\text{a, letter, .}]_B$	He $\xleftarrow{\text{SBJ}}$ sent sent $\xrightarrow{\text{IOBJ}}$ her



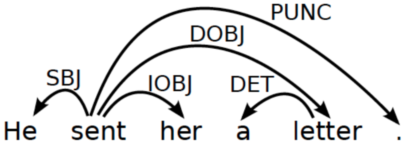
What should be the next transition?

There is no relation between "her" and "a". So next transition cannot be Left-arc or Right-arc. How to choose between "Reduce" and "Shift"?

Parse Example

Transitions: SH-LA-SH-RA

Stack	Buffer	Arcs
$[\text{sent, her}]_S$	$[\text{a, letter, .}]_B$	He $\xleftarrow{\text{SBJ}}$ sent sent $\xrightarrow{\text{IOBJ}}$ her



Example

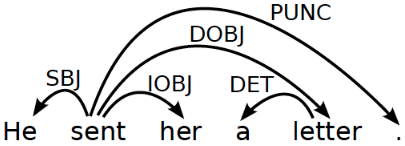
Parse Example

How to choose between "Reduce" and "Shift"?

If there exists w < top(S) that has an arc with first(B), then use "Reduce" (provided Head of top(S) has been found); otherwise use "Shift".

Transitions: SH-LA-SH-RA

Stack	**Buffer**	**Arcs**
[sent, her]$_S$	[a, letter, .]$_B$	He $\xleftarrow{\text{SBJ}}$ sent sent $\xrightarrow{\text{IOBJ}}$ her



Here we have to use "Shift".

What will be the next transition?

There is a Left-Arc connecting "a" and "letter".

Parse Example

Transitions: SH-LA-SH-RA-SH

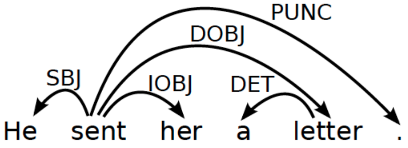
Stack [sent, her, a]$_S$

Buffer [letter, .]$_B$

Arcs

He $\xleftarrow{\text{SBJ}}$ sent

sent $\xrightarrow{\text{IOBJ}}$ her



Example

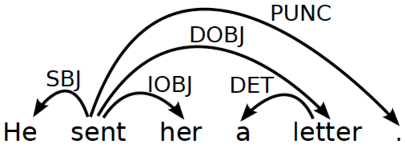
What will be the next transition?

Parse Example

There is no relation between "her" and "letter". So we have to choose either Shift or Reduce.

Transitions: SH-LA-SH-RA-SH-LA

Stack	**Buffer**	**Arcs**
$[\text{sent, her}]_S$	$[\text{letter, .}]_B$	He $\xleftarrow{\text{SBJ}}$ sent sent $\xrightarrow{\text{IOBJ}}$ her a $\xleftarrow{\text{DET}}$ letter

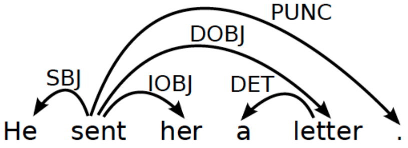


This time, we use Reduce

Parse Example

Transitions: SH-LA-SH-RA-SH-LA-RE

Stack	**Buffer**	**Arcs**
$[\text{sent}]_S$	$[\text{letter}, .]_B$	He $\xleftarrow{\text{SBJ}}$ sent sent $\xrightarrow{\text{IOBJ}}$ her a $\xleftarrow{\text{DET}}$ letter



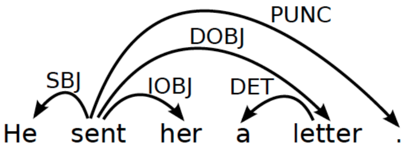
Example

Parse Example

Again, no relation between "letter" and "." This time we will use Reduce.

Transitions: SH-LA-SH-RA-SH-LA-RE-RA

Stack	**Buffer**
$[\text{sent, letter}]_S$	$[.]_B$



Arcs

$\text{He} \xleftarrow{\text{SBJ}} \text{sent}$

$\text{sent} \xrightarrow{\text{IOBJ}} \text{her}$

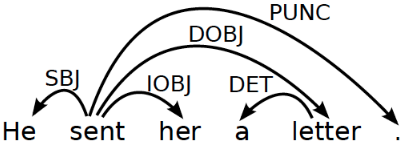
$\text{a} \xleftarrow{\text{DET}} \text{letter}$

$\text{sent} \xrightarrow{\text{DOBJ}} \text{letter}$

Parse Example

Transitions: SH-LA-SH-RA-SH-LA-RE-RA-RE

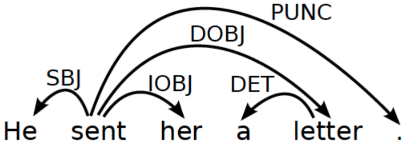
Stack	**Buffer**	**Arcs**
$[\text{sent}]_S$	$[.]_B$	He $\xleftarrow{\text{SBJ}}$ sent
		sent $\xrightarrow{\text{IOBJ}}$ her
		a $\xleftarrow{\text{DET}}$ letter
		sent $\xrightarrow{\text{DOBJ}}$ letter



Parse Example

Transitions: SH-LA-SH-RA-SH-LA-RE-RA-RE-RA

Stack	**Buffer**	**Arcs**
$[\text{sent}, .]_S$	$[\ ]_B$	He $\xleftarrow{\text{SBJ}}$ sent
		sent $\xrightarrow{\text{IOBJ}}$ her
		a $\xleftarrow{\text{DET}}$ letter
		sent $\xrightarrow{\text{DOBJ}}$ letter
		sent $\xrightarrow{\text{PUNC}}$.



Parse Example

Now buffer is empty. So we have reached a terminal configuration.

Transitions: SH-LA-SH-RA-SH-LA-RE-RA-RE-RA

Stack

$[\text{sent}, .]_S$

Buffer

$[\ ]_B$

Arcs

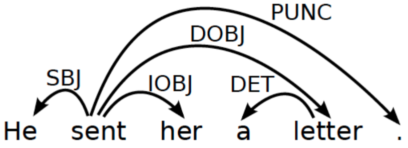
He $\xleftarrow{\text{SBJ}}$ sent

sent $\xrightarrow{\text{IOBJ}}$ her

a $\xleftarrow{\text{DET}}$ letter

sent $\xrightarrow{\text{DOBJ}}$ letter

sent $\xrightarrow{\text{PUNC}}$.



Till now we discussed: Given a sentence and its dependency graph, we can find the sequence of transitions.

But how can we get the dependency graph / transitions for a new sentence? … next lecture

Briefly, the technique we discussed can help build training data for LEARNING how to predict transitions for a new sentence