np : 1er jet = 15'15
⚠ utiliser moins de place au début

CYK [Carton p 199 pour l'algo]

L'algo de Cocke Younger Kasami prend en entrée

un mot $w = w_1 \dots w_n$ et une grammaire G sous forme normale de Chomsky (d'axiome S)

(car ses productions sont de la forme $A \rightarrow a$ ou $A \rightarrow B_1 B_2$; terminaux en minuscules et non terminaux en majuscules)

En sortie, il donne oui si $w \in \mathcal{L}(G)$ et non sinon.
= w est engendré par G

$\forall\ 1 \leq i \leq j \leq n$, on note $E_{ij} = \{$non terminaux qui engendrent $w_i \dots w_j\}$

On rq que $w \in \mathcal{L}(G) \iff S \in E_{1n}$: on veut donc K E_{1n}

Pour ce faire, on calcule les E_{ij} en utilisant la relation suivante :

$$\forall\ 1 \leq i < j \leq n, \quad E_{ij} = \bigcup_{\substack{A \rightarrow B_1 B_2 \\ B_1 \in E_{ik} \\ B_2 \in E_{k+1\,j} \\ i \leq k < j}} \{A\}$$

En effet, si ($\exists B_1, B_2, k$) $B_1 \xrightarrow{*} w_i \dots w_k$, $B_2 \xrightarrow{*} w_{k+1} \dots w_j$ et $A \rightarrow B_1 B_2$ alors

$A \rightarrow w_i \dots w_j$ car $A \in E_{ij}$; et réciproquement, si $A \rightarrow w_i \dots w_j$ (mot d'au-moins 2 lettres car $i \neq j$) alors $\exists B_1, B_2, k$ tq $B_1 \in E_{ik}$, $B_2 \in E_{k+1\,j}$ et $i \leq k < j$

car G est sous forme de Chomsky

Pour calculer E_{ij} $(i<j)$ on a donc besoin des E_{ik} et des $E_{k+1\,j}$ avec $i \leq k < j$ (→ donne l'ordre de K des E_{ij})

On représente cette dépendance ds le tableau suivant :

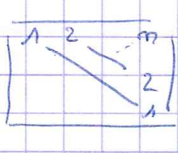
E_{11} ...	E_{1i} ...	E_{1j} ...	E_{1n}
$E_{i1} = \emptyset$ ...	[E_{ii} ...	(E_{ij})] ...	E_{in}
$E_{j1} = \emptyset$ ...	$E_{ji} = \emptyset$ ...	[E_{jj}] ...	E_{jn}
$E_{n,1} = \emptyset$ ...	$E_{ni} = \emptyset$ ...	$E_{nj} = \emptyset$ ...	E_{nn}

en [ ] = ce qu'il faut pour K E_{ij}

Les E_{kj} sous la diagonale sont $\emptyset$ car pour eux, l'indice de début de mot est > à celui de fin de mot

On calcule donc les E_{ij} dans l'ordre suivant : la diagonale (= cas de base et on peut car $\forall i$, $E_{ii} = \{A \mid A \rightarrow w_i \in$ production de $G\}$) puis la surdiagonale ... jusque E_{1n}

Ordre des calculs :



D'où l'algorithme :

Initialiser tous les E_{ij} à $\emptyset$

Pour $i = 1$ à n } calcule la diagonale
  Pour $A \to a \in G$
    Si $a = w_i$ alors $E_{ii} = E_{ii} \cup \{A\}$

Pour $d = 2$ à n } pour les diagonales suivantes
  Pour E_{ij} sur la diagonale d
    Pour $k = i$ à $j-1$
      pour $A \to B_1 B_2 \in G$
        si $B_1 \in E_{ik}$ et $B_2 \in E_{k+1,j}$ alors $E_{ij} = E_{ij} \cup \{B\}$

Renvoyer $S \in E_{1n}$

On obtient une complexité en $n|G| + n^3|G| = O(n^3|G|)$ où $|G| = nb$ de règles de G
et $O(n^3|G|)$ où $|G| = nb$ de symboles des règles

(rq) Mise sous forme de Chomsky d'1 grammaire propre (= Pas de $A \to \varepsilon$ ou de $A \to A'$)

On commence par obtenir des productions de la forme $A \to a$ et $A \to B_1 \dots B_m$
en introduisant pour tout terminal b_i 1 non terminal T_i et en remplaçant
les t_i par T_i dans les productions de G → ajout $T_i \to t_i$
(les anciennes droites $\neq t_i$ tout seul)

Puis, pour chaque règle de la forme $A \to B_1 \dots B_m$ on introduit des non terminaux
$B'_2, \dots B'_{m-1}$ et on remplace cette règle par $A \to B_1 B'_2$, $B'_2 \to B_2 B'_3$ $B'_i \to B_i B'_{i+1} \dots$
$B'_{m-1} \to B_{m-1} B_m$.

ex : $S \to a \mid bSb \mid Sb$ devient $S \to a \mid ASA \mid SB$ puis $S \to a \mid AA' \mid SB$
$A \to a$ $A \to a$
$B \to b$ $B \to b$
$A' \to SB$
(terminaux et non terminaux)

Complexité de la mise sous forme de Chomsky ($|G|$ = nombre de symboles utilisés pr écrire les règles de G)

1ère étape : En ajoutant les $T_i \to t_i$, on ajoute au plus $2 \times nb$ de terminaux $\leq 2|G|$ symboles
(ceux apparaissant dans les règles)
Modifier les $A \to Bt_1t_2C$ en $A \to BT_1T_2C$ n'augmente pas le nb de symboles

2e étape : Passer de $A \to B_1 \dots B_m$ à $A \to B_1 B'_2 \dots B'_{m-1} \to B_{m-1} B_m$ remplace $m+1$ symboles par $\leq 3m$

Donc complexité en $O(|G|)$