

---

## Plus long facteur commun et plus long sous-mot commun

---

### Notions abordées

- facteur, sous-mot, préfixe, suffixe
- programmation dynamique

## Exercice 1 : Plus long sous-mot/facteur commun

**Q. 1** Rappeler les définitions de **facteur** d'un mot et **sous-mot** d'un mot. Donner un exemple illustrant la différence entre les deux.

### Solution

On dit que  $y$  est un **sous-mot** de  $x$  ssi il existe  $\varphi$  une fonction strictement croissante de  $\llbracket 1, |y| \rrbracket$  dans  $\llbracket 1, |x| \rrbracket$  telle que  $y = x_{\varphi(1)}x_{\varphi(2)} \dots x_{\varphi(m)}$ .

On dit que  $y$  est un **facteur** de  $x$  ssi il existe deux mots  $u$  et  $v$  sur  $\Sigma$  tels que  $x = u \cdot y \cdot v$ .

$jrd$  est un sous-mot de  $jardin$  mais pas un facteur.

**Q. 2** Définir le problème du plus long sous-mot commun.

### Solution

$$\text{PLSMC} \begin{cases} \text{Entrée} : x \in \Sigma^*, y \in \Sigma^* \\ \text{Sortie} : \max \{ |u| \mid u \in \Sigma^*, u \text{ est sous-mot de } x \text{ et de } y \} \end{cases}$$

**Q. 3** Définir le problème du plus long facteur commun.

### Solution

$$\text{PLFC} \begin{cases} \text{Entrée} : x \in \Sigma^*, y \in \Sigma^* \\ \text{Sortie} : \max \{ |u| \mid u \in \Sigma^*, u \text{ est facteur de } x \text{ et de } y \} \end{cases}$$

**Q. 4** Proposer un algorithme de programmation dynamique résolvant le problème du plus long sous-mot commun.

- la définition d'une quantité bien choisie, qui se calcule récursivement et permet de résoudre le problème ;
- la preuve que la relation de récurrence annoncée est correcte ;
- le pseudo-code d'un algorithme qui calcule la valeur optimale, muni d'un invariant ;
- le pseudo-code d'un algorithme qui calcule une solution optimale, muni d'un invariant.

## Solution

Soit  $(x, y) \in \Sigma^* \times \Sigma^*$ . On note  $n = |x|$  et  $m = |y|$ .

On pose, pour tout  $(i, j) \in \llbracket 0, n \rrbracket \times \llbracket 0, m \rrbracket$  :

$$\begin{aligned} \mathcal{E}_{i,j} &\stackrel{\text{d\u00e9f}}{=} \{ u \mid u \in \Sigma^*, u \text{ est sous-mot de } x_1 \dots x_i \text{ et de } y_1 \dots y_j \} \\ B_{i,j} &\stackrel{\text{d\u00e9f}}{=} \max\{|u| \mid u \in \mathcal{E}_{i,j}\} \end{aligned}$$

La valeur optimale est alors donn\u00e9e par  $B_{n,m}$ .

Le calcul des valeurs  $B_{i,j}$  peut se faire gr\u00e2ce aux relations suivantes.

$$\begin{aligned} \forall i \in \llbracket 0, n \rrbracket, B_{i,0} &= 0 && \text{car } y_1 \dots y_0 = \varepsilon \text{ n'a que } \varepsilon \text{ comme sous-mot, et } |\varepsilon| = 0 \\ \forall j \in \llbracket 0, m \rrbracket, B_{0,j} &= 0 && \text{idem} \\ \forall (i, j) \in \llbracket 1, n \rrbracket \times \llbracket 1, m \rrbracket, B_{i,j} &= \begin{cases} 1 + B_{i-1,j-1} & \text{si } x_i = y_j \\ \max(B_{i-1,j}, B_{i,j-1}) & \text{sinon} \end{cases} \end{aligned} \quad (\star)$$

Justifions la relation de r\u00e9currence  $(\star)$  par disjonction de cas.

- Cas  $x_i = y_j$ .
  - Si  $v \in \mathcal{E}_{i-1,j-1}$ , alors  $v \cdot x_i \in \mathcal{E}_{i,j}$ , donc  $B_{i,j} \geq |v \cdot x_i| = 1 + |v|$ . Cette relation pour  $v^* \in \mathcal{E}_{i-1,j-1}$  optimal, donne en particulier  $B_{i,j} \geq 1 + B_{i-1,j-1}$ .
  - Si  $u^* \in \mathcal{E}_{i,j}$  est optimal, n\u00e9cessairement  $|u^*| \geq 1$  car  $x_i \in \mathcal{E}_{i,j}$ . On peut donc parler de la derni\u00e8re lettre de  $u^*$  et d\u00e9composer  $u^* = v \cdot z$  avec  $v \in \Sigma^*$  et  $z \in \Sigma$ . Si  $z \neq x_i$ , alors  $u^*$  serait sous-mot de  $x_1 \dots x_{i-1}$  et de  $y_1 \dots y_{j-1}$ , et on aurait alors  $|u^*| \leq B_{i-1,j-1} < B_{i,j}$ . **ABSURDE.** Ainsi  $u^* = v \cdot x_i$  et  $v \in \mathcal{E}_{i-1,j-1}$ , donc  $B_{i,j} = |u^*| = |v| + 1 \leq B_{i-1,j-1} + 1$ .

Par double in\u00e9galit\u00e9 on a bien  $B_{i,j} = 1 + B_{i-1,j-1}$  dans ce cas.

- Cas  $x_i \neq y_j$ .
  - Puisqu'un sous-mot d'un sous-mot est un sous-mot,  $\mathcal{E}_{i,j-1} \subseteq \mathcal{E}_{i,j}$  donc  $B_{i,j} \geq B_{i,j-1}$ . De m\u00eame  $B_{i,j} \geq B_{i-1,j}$ , d'o\u00f9  $B_{i,j} \geq \max(B_{i-1,j}, B_{i,j-1})$ .
  - Soit  $u^* \in \mathcal{E}_{i,j}$  optimal. Si  $u^* = \varepsilon$ ,  $u^* \in \mathcal{E}_{i,j-1}$ , donc  $B_{i,j} = |u^*| \leq B_{i,j-1} \leq \max(B_{i-1,j}, B_{i,j-1})$ . Sinon,  $u^* = v \cdot z$  avec  $v \in \Sigma^*$  et  $z \in \Sigma$ . Puisque  $x_i \neq y_j$ , on a  $z \neq x_i$  ou  $z \neq y_j$ . Si  $z \neq x_i$ , alors  $u^* \in \mathcal{E}_{i-1,j}$ , et donc  $B_{i,j} = |u^*| \leq B_{i-1,j} \leq \max(B_{i-1,j}, B_{i,j-1})$ . De m\u00eame, si  $z \neq y_j$  on a  $B_{i,j} = |u^*| \leq B_{i,j-1} \leq \max(B_{i-1,j}, B_{i,j-1})$ . Ainsi on a bien  $B_{i,j} \leq \max(B_{i-1,j}, B_{i,j-1})$  d\u00e8s lors que  $x_i \neq y_j$ .

Par double in\u00e9galit\u00e9 on a bien  $B_{i,j} = \max(B_{i-1,j}, B_{i,j-1})$  dans ce cas.

---

**Algorithme 1** : Algorithme pour PLSMC (valeur/solution optimale)

---

**Entrée** : Deux mots de longueurs respectives  $n$  et  $m$  :  $x = x_1 \dots x_n$  et  $y = y_1 \dots y_m$

**Sortie** : La longueur d'un plus long sous-mot commun à  $x$  et  $y$  ou un tel sous-mot

```
1 B ← tableau d'entiers indexé par  $\llbracket 0, n \rrbracket \times \llbracket 0, m \rrbracket$  ;
2 M ← tableau de couples d'entiers indexé par  $\llbracket 0, n \rrbracket \times \llbracket 0, m \rrbracket$  ;
3 pour  $i = 0$  à  $n$  faire
4   B[i][0] ← 0 ;
5   M[i][0] ← (i - 1, 0) ;
6 pour  $j = 0$  à  $m$  faire
7   B[0][j] ← 0 ;
8   M[0][j] ← (0, j - 1) ;
9 pour  $i = 1$  à  $n$  faire
10  pour  $j = 1$  à  $m$  faire
11    si  $x_i = y_j$  alors
12      B[i][j] ← 1 + B[i - 1][j - 1] ;
13      M[i][j] ← (i - 1, j - 1) ;
14    sinon
15      si B[i - 1][j] > B[i][j - 1] alors
16        B[i][j] ← B[i - 1][j] ;
17        M[i][j] ← (i - 1, j) ;
18      sinon
19        B[i][j] ← B[i][j - 1] ;
20        M[i][j] ← (i, j - 1) ;
21 retourner B[n][m] ;
22 (i, j) ← (n, m) ;
23 u ← ε ;
24 tant que i > 0 et j > 0 faire
25   si M[i][j] = (i - 1, j - 1) alors
26     u ← xi · u
27   (i, j) ← M[i][j] ;
28 retourner u ;
```

---

**Justifions la correction de l'algorithme 1** Les trois premières boucles remplissent le tableau B avec les valeurs de A grâce aux relations précédemment démontrées, autrement dit à la ligne 28 on a  $\forall (i, j) \in \llbracket 0, n \rrbracket \times \llbracket 0, m \rrbracket, B[i][j] = A_{i,j}$ . Ainsi la version de l'algorithme qui calcule la valeur est correct.

Pour montrer que l'algorithme qui calcule une solution est correct, on s'appuie sur l'invariant suivant pour la boucle ligne 24.

$$u \text{ est sous-mot de } x_{i+1} \dots x_n \text{ et de } y_{j+1} \dots y_m \text{ et } |u| = B[n][m] - B[i][j]$$

- Initialement  $(i, j) = (n, m)$  donc  $x_{i+1} \dots x_n = \varepsilon$  et de  $y_{j+1} \dots y_m = \varepsilon$ , ainsi  $u = \varepsilon$  est bien un sous-mot de ces deux mots. De plus, on a alors  $B[n][m] - B[i][j] = B[n][m] - B[n][m]$  et  $|u| = |\varepsilon| = 0$ . L'invariant proposé est donc vrai avant la boucle ligne 24.

- Supposons l'invariant vérifié à l'entrée d'un tour de boucle, montrons qu'il l'est toujours en sortie de ce tour. On note  $i^a$  (resp.  $j^a$  et  $u^a$ ) la valeur de la variable  $i$  (resp.  $j$  et  $u$ ) à l'entrée du tour de boucle. On note  $i^b$  (resp.  $j^b$  et  $u^b$ ) les valeurs en sortie.
  - Si  $M[i^a][j^a] = (i^a - 1, j^a - 1)$ , alors on a  $i^b = i^a - 1$  et  $j^b = j^a - 1$  d'après la l.27, et  $u^b = x_{i^a} \cdot u^a$  d'après la l.???. De plus, vu comment est construit M entre les lignes 28 et 28, on a nécessairement  $x_{i^a} = y_{j^a}$  et  $B[i^a][j^a] = 1 + B[i^a - 1][j^a - 1]$ .  
On a alors

$$\begin{aligned}
 |u^b| &= 1 + |u^a| \\
 &= 1 + (B[n][m] - B[i^a][j^a]) && \text{par hypothèse} \\
 &= \cancel{1} + B[n][m] - (\cancel{1} + B[i^a - 1][j^a - 1]) \\
 &= B[n][m] - \underbrace{B[i^a - 1]}_{i^b} \underbrace{[j^a - 1]}_{j^b}
 \end{aligned}$$

et comme  $u^a$  est sous-mot de  $x_{i^a+1} \dots x_n$  par hypothèse,  $u^b = x_{i^a} \cdot u^a$  est sous-mot de  $x_{i^a} \dots x_n$  soit de  $x_{i^b+1} \dots x_n$ . De même  $u^b = y_{j^a} \cdot u^a$  est sous-mot de  $y_{j^a} \dots y_m$  soit de  $y_{j^b+1} \dots y_m$ . Ainsi les valeurs  $i^b, j^b$  et  $u^b$  vérifient bien l'invariant dans ce cas.

- Sinon, on a  $u^b = u^a$ , or par hypothèse  $u^a$  est un sous-mot de  $x_{i^a+1} \dots x_n$  et  $y_{j^a+1} \dots y_m$ , donc  $u^b$  aussi, et a fortiori  $u^b$  est sous-mot de  $x_{i^a} \dots x_n$  et  $y_{j^a} \dots y_m$ .  
Vu comment est construit M entre les lignes et on a deux sous-cas possibles.

- Si  $M[i^a][j^a] = (i^a - 1, j^a)$ , on a aussi  $B[i^a][j^a] = B[i^a - 1][j^a]$  par construction, et d'après la ligne 27,  $i^b = i^a - 1, j^b = j^a$ .
- Si  $M[i^a][j^a] = (i^a, j^a - 1)$ , on a aussi  $B[i^a][j^a] = B[i^a][j^a - 1]$  par construction, et d'après la ligne 27,  $i^b = i^a, j^b = j^a - 1$ .

Dans les deux sous-cas  $B[n][m] - B[i^b][j^b] = B[n][m] - B[i^a][j^a]$ , or par hypothèse  $B[n][m] - B[i^a][j^a] = |u^a|$ , et comme  $u^b = u^a$  on a bien  $B[n][m] - B[i^b][j^b] = |u^b|$ .

Ainsi l'invariant est vérifié aussi dans ce cas.

- Q. 5** Proposer un algorithme de programmation dynamique résolvant le problème du plus long facteur commun. On attend :
- la définition d'une quantité bien choisie, qui se calcule récursivement et permet de résoudre le problème ;
  - la preuve que la relation de récurrence annoncée est correcte ;
  - le pseudo-code d'un algorithme qui calcule la valeur optimale, muni d'un invariant ;
  - le pseudo-code d'un algorithme qui calcule une solution optimale, muni d'un invariant.

### Solution

Soit  $(x, y) \in \Sigma^* \times \Sigma^*$ . On note  $n = |x|$  et  $m = |y|$ .

**On pose,** pour tout  $(i, j) \in \llbracket 0, n \rrbracket \times \llbracket 0, m \rrbracket$  :

$$\begin{aligned}
 \mathcal{S}_{i,j} &\stackrel{\text{déf}}{=} \{ u \mid u \in \Sigma^*, u \text{ est suffixe de } x_1 \dots x_i \text{ et de } y_1 \dots y_j \} \\
 A_{i,j} &\stackrel{\text{déf}}{=} \max\{|u| \mid u \in \mathcal{S}_{i,j}\}
 \end{aligned}$$

**La valeur optimale** est alors donnée par  $\max\{A_{i,j} \mid (i,j) \in \llbracket 0, n \rrbracket \times \llbracket 0, m \rrbracket\}$ .

En effet, un facteur est toujours un suffixe d'un préfixe, ainsi un facteur de  $x$  est un suffixe de  $x_1 \dots x_i$  pour un certain  $i \in \llbracket 0, n \rrbracket$ , et de même un facteur de  $y$  est un suffixe de  $y_1 \dots y_j$  pour un certain  $j \in \llbracket 0, m \rrbracket$ , donc on a

$$\begin{aligned} \text{PLFC}(x, y) &= \max \left\{ |u| \mid u \in \Sigma^*, u \text{ est facteur de } x \text{ et de } y \right\} \\ &= \max \left\{ |u| \mid u \in \Sigma^*, \exists (i, j) \in \llbracket 0, n \rrbracket \times \llbracket 0, m \rrbracket, u \text{ est suffixe de } x_1 \dots x_i \text{ et de } y_1 \dots y_j \right\} \\ &= \max \left\{ |u| \mid u \in \bigcup_{(i,j) \in \llbracket 0, n \rrbracket \times \llbracket 0, m \rrbracket} \mathcal{S}_{i,j} \right\} \\ &= \max_{(i,j) \in \llbracket 0, n \rrbracket \times \llbracket 0, m \rrbracket} \max \left\{ |u| \mid u \in \mathcal{S}_{i,j} \right\} \\ &= \max_{(i,j) \in \llbracket 0, n \rrbracket \times \llbracket 0, m \rrbracket} A_{i,j} \end{aligned}$$

**Le calcul des valeurs  $A_{i,j}$**  peut se faire grâce aux relations suivantes.

$$\forall i \in \llbracket 0, n \rrbracket, A_{i,0} = 0 \quad \text{car } y_1 \dots y_0 = \varepsilon \text{ n'a que } \varepsilon \text{ comme suffixe, et } |\varepsilon| = 0$$

$$\forall j \in \llbracket 0, m \rrbracket, A_{0,j} = 0 \quad \text{idem}$$

$$\forall (i, j) \in \llbracket 1, n \rrbracket \times \llbracket 1, m \rrbracket, A_{i,j} = \begin{cases} 1 + A_{i-1,j-1} & \text{si } x_i = y_j \\ 0 & \text{sinon} \end{cases} \quad (\star)$$

**Justifions la relation de récurrence  $(\star)$**  par disjonction de cas.

Si  $x_i \neq y_j$  le seul suffixe commun à  $x_1 \dots x_i$  et  $y_1 \dots y_j$  est  $\varepsilon$ , car un suffixe ayant au moins une lettre aurait pour dernière lettre  $x_i$  en tant que suffixe de  $x_1 \dots x_i$  et pour dernière lettre  $y_j$  en tant que suffixe de  $y_1 \dots y_j$ , on aurait alors  $x_i = y_j$ . **ABSURDE.**

Si  $x_i = y_j$  on a

$$\begin{aligned} \mathcal{S}_{i,j} &= \{\varepsilon\} \cup \left\{ v \cdot x_i \mid v \in \Sigma^*, \begin{array}{l} v \cdot x_i \text{ est suffixe de } x_1 \dots x_{i-1} x_i \\ v \cdot y_j \text{ est suffixe de } y_1 \dots y_{j-1} y_j \end{array} \right\} \\ &= \{\varepsilon\} \cup \left\{ v \cdot x_i \mid v \in \Sigma^*, \begin{array}{l} v \text{ est suffixe de } x_1 \dots x_{i-1} \\ v \text{ est suffixe de } y_1 \dots y_{j-1} \end{array} \right\} \\ &= \{\varepsilon\} \cup \mathcal{S}_{i-1,j-1} \cdot \{x_i\} \end{aligned}$$

$$\begin{aligned} \text{donc } A_{i,j} &= \max \left( |\varepsilon|, \max_{|v|+1} \left\{ |v \cdot x_i| \mid v \in \mathcal{S}_{i-1,j-1} \right\} \right) \\ &= \max \left( 0, 1 + \max \{ |v| \mid v \in \mathcal{S}_{i-1,j-1} \} \right) \\ &= \max \left( 0, \underbrace{1 + A_{i-1,j-1}}_{\geq 1} \right) \\ &= 1 + A_{i-1,j-1} \end{aligned}$$

---

**Algorithme 2** : Algorithme pour PLFC (valeur/solution optimale)

---

**Entrée** : Deux mots de longueurs respectives  $n$  et  $m$  :  $x = x_1 \dots x_n$  et  $y = y_1 \dots y_m$

**Sortie** : La longueur d'un plus long facteur commun à  $x$  et  $y$  ou un tel facteur

1  $A \leftarrow$  tableau d'entiers indexé par  $\llbracket 0, n \rrbracket \times \llbracket 0, m \rrbracket$ ;

2 **pour**  $i = 0$  à  $n$  **faire**

3    $A[i][0] \leftarrow 0$ ;

4 **pour**  $j = 0$  à  $m$  **faire**

5    $A[0][j] \leftarrow 0$ ;

6 **pour**  $i = 1$  à  $n$  **faire**

7   **pour**  $j = 1$  à  $m$  **faire**

8     **si**  $x_i = y_j$  **alors**

9        $A[i][j] \leftarrow 1 + A[i-1][j-1]$ ;

10    **sinon**

11      $A[i][j] \leftarrow 0$ ;

12  $\max \leftarrow 0$ ;

13  $i_{\max} \leftarrow 0$ ;

14 **pour**  $i = 1$  à  $n$  **faire**

15   **pour**  $j = 1$  à  $m$  **faire**

16     **si**  $A[i][j] > \max$  **alors**

17        $\max \leftarrow A[i][j]$ ;

18        $i_{\max} \leftarrow i$ ;

19 **retourner**  $\max$ ;

20  $f \leftarrow i_{\max}$ ;  $d \leftarrow (f - \max) + 1$ ;

21 **retourner**  $x_d \dots x_f$ ;

---