Etudes des analyses et algorithmes :

Résumé des formules et interprétations à savoir de janvier – février :

2 exemples explications pour comprendre concrètement le cours seront fournis

1. Complexité temporelle :

- $W(i;I) = W(i) + W(I)$
- $W(x = e) = W(e)$
- $W(if\ (C)\ S1;\ S2) = W(C) + \max\ (W(S1), W(S2))$
- $W\big(while(C)(S)\big) = \sum_{i \in \mathbb{N}}\ (W(C) + W(S))$
- $W(for(S1;S2;S3)S4) = W(S1) + \sum_{i \in \mathbb{N}}\ (W(S2) + W(S3) + W(S4))$
- $W(recursion) =$ cf **4.**

2. Complexité spatiale :

- $S(A) = \sum_{i \in \mathbb{N}} S(x_i)$
- $S(X) = \max\ (size(v)/\ v\ in\ any\ value\ stored\ in\ x)$

On decide $n = \max\ (\log x, \log y)$
Et en comparant avec la complexité temporelle trouvé, on retrouve la complexité spatiale avec n le plus grand des nombres de bits

3. Complexité des opérations arithmétiques :

- $W(x + y) = \mathcal{O}(\log x + \log y) = \mathcal{O}(n)$
- $W(x - y) = \mathcal{O}(\log x + \log y) = \mathcal{O}(n)$
- $W(x \geq y) = \mathcal{O}(\log x + \log y) = \mathcal{O}(n)$
- $W(x * y) = \mathcal{O}(\log x * \log y) = \mathcal{O}(n^2)$
- $W(x * y) = \mathcal{O}(\log x * \log y) = \mathcal{O}(n^2)$
- $S(x + y) = \mathcal{O}(n)$
- $S(x - y) = \mathcal{O}(n)$
- $S(x \geq y) = \mathcal{O}(1)$
- $S(x * y) = \mathcal{O}(n)$
- $S(x/y) = \mathcal{O}(n)$

Exemple concret de complexité sans récurrence :

1. $r \leftarrow 1$
2. $while(y\ != 0)$
3. $r \leftarrow r * x$
4. $y \leftarrow y - 1$
5. $return\ x$

On a donc : $W(Pr) = W(1) + W(2) + W(5)$

$W(1) = \mathcal{O}(1)$
$W(2) = \sum_{i=1}^{y}[W(C) + W(3) + W(4)]$
$W(C) = \mathcal{O}(1)$
$W(3) = \mathcal{O}(\log r * \log x) = \mathcal{O}(\log x^i * \log x) = \mathcal{O}(i * \log^2 x)$
$W(4) = \mathcal{O}(\log y)$
$W(2) = \sum_{i=1}^{y}[\mathcal{O}(1) + \mathcal{O}(i * \log^2 x) + \mathcal{O}(\log y)] = \sum_{i=1}^{y}\mathcal{O}(i \log^2 x) + \mathcal{O}(\log y)$
$= \mathcal{O}(\sum_{i=1}^{y}(i * \log^2 x)) + \mathcal{O}(y \log y) = \mathcal{O}(y^2 * \log^2 x) + \mathcal{O}(y \log y)$
↪ mais comme $y^2 > y \log y$ on le supprime
$= \mathcal{O}(y^2 * \log^2 x)$

On décide $n = \max(\log x, \log y)$
$\log x = n \implies \log^2 x = n^2 \quad | \quad \log y = n \implies y = 2^n \implies y^2 = 2^{2n}$
Donc $\mathcal{O}(y^2 * \log^2 x) = \mathcal{O}(2^n * n) \leftarrow \mathcal{O}(2^{2n}) = \mathcal{O}(2^n)$

$W(5) = \mathcal{O}(y \log x)$ car la taille des bits étant $\log x^y$ = $\mathcal{O}(2^n * n)$

4. Calcul récursif

A. Construire un programme récursif
   1. Nombre d'appel récursif → m
   2. Valeur de l'appel → a_i
   3. Ordre sans récursivité → $f(n)$

B. Résoudre le programme récursif
   1. Et 2. On cherche k et l tel que $\mathcal{O}(f(n)) = \mathcal{O}(n^k * (\log n)^l)$
   3. $\gamma = \sum_{i=1}^{m} a_i^k$
      Si $\gamma < 1 \rightarrow \mathcal{O}(n^k * (\log n)^l)$
      Si $\gamma = 1 \rightarrow \mathcal{O}(n^k * (\log n)^{l+1})$
      Si $\gamma > 1 \rightarrow \mathcal{O}(n^c)$ avec c tel que $\sum_{i=1}^{m} a_i^c = 1$

T : $\mathbb{N} \rightarrow \mathbb{N}$ satisfait $T(n) = \sum_{i=1}^{m} T(a_i n + \mathcal{O}(1)) + f(n)$

Exemple concret de complexité avec récurrence :

$Mult(x, y)\{$

1. $if\ y = 0$
2. $\quad return\ 0$
3. $z = Mult\left(x, \left\lfloor \frac{y}{2} \right\rfloor\right)$
4. $if\ pair(y)$
5. $\quad return\ 2z$
6. $else\ return\ 2z + x$

$\}$

On suppose $n = y > x$

A. Construire un programme récursif
   1. $m = 1$
   2. $a_1 = \frac{1}{2}$
   3. Ordre $O(f(n))$
      $\mathcal{O}(\log 2z + \log x) = \mathcal{O}(\log 1 + \log z + \log x) = \mathcal{O}(\log n)$

      $T(n) = T\left(\frac{n}{2}\right) + \mathcal{O}(\log n)$

B. Résoudre le programme récursif
   $\mathcal{O}(\log n) = \mathcal{O}(n^k * (\log n)^l)$
   1. k = 0
   2. l = 1
   3. $\gamma = \sum_{i=1}^{m} a_i^k = \left(\frac{1}{2}\right)^0 = 1$
      Donc $\mathcal{O}\left(n^k * (\log n)^l\right) = \mathcal{O}(\log^2 n)$