Algorithmique 2 — TD n° 5.

Q1. (a) on fait un DFS ⤳ $G(n)$

(b) on calcule les degrés et on donne les sommets de degré 1:
↳ $G(n)$ ↳ $G(2n)$

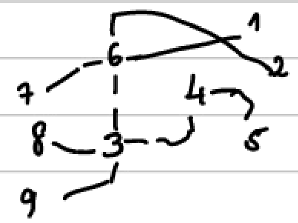
Q2. (a) On fait de la programmation dynamique : (on peut aussi procéder à un tri à l'aide d'un bucket sort)

$$\text{mwvc}[x] := \min \left\{ \text{poids}(x) + \sum_{x \to z \to y} \text{mwvc}[y] \; , \; \sum_{x \to y} \text{mwvc}[y] \right\}.$$

(b) On fait un 1er DFS pour trouver le sommet x le plus loin du sommet choisi arbitrairement. Ensuite, avec un 2nd DFS, on part de x et on regarde le sommet y le plus loin de x.
diam(T) = profondeur de y dans le 2nd parcours

Q3. 6643633

Q4.

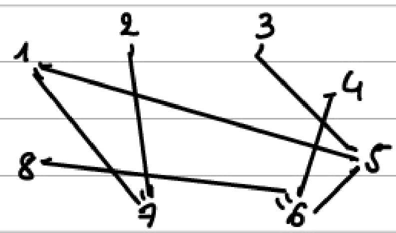


On crée une file de priorité sur $[\![1, |w|+2]\!]$ où les priorités sont les nb d'occ dans w, plus un.
$i \leftarrow 0$
Tant que la file de prio n'est pas vide faire
- Extraire le min x.
- Relier x à w_i
- $i \leftarrow i+1$.
- Retirer 1 à la prio de w_i et x.
- Si prio = 0 alors on le retire.

<u>Solution</u> : On barre des sommets. On lit le mot, à une lettre on la relie au plus petit qui n'est pas barré et qui n'est pas dans la séquence restante ; puis on barre la lettre.

Q5. $|T_n| = n^{n-2}$

7 5 6 1 5 6



TD n° 6.

Q 1. (a) Soient T_1 et T_2 deux arbres couvrant de poids minimum.
Supposons $T_1 \neq T_2$ d'où $E(T_1) \Delta E(T_2) \neq \emptyset$.
Soit $e \in E(T_1) \Delta E(T_2)$ de poids min.

Sans perdre en généralité, supposons $e \in E(T_1)$.
Le graphe $T_2 + e$ a un cycle C.

Soit $e' \in (C \setminus \{e\}) \cap (E(T_2) \setminus E(T_1))$.

Alors, $T_2 + e' - e$ est un arbre couvrant de poids < poids de T_2.
Absurde.

On conclut $T_1 = T_2$.

(b) Soit $E = \{e_1, \dots, e_n\}$ tels que $w(e_1) \leq \dots \leq w(e_n)$.
On pose $w'(e_i) := i$.

Comme les poids $(w'(e))_{e \in E}$ sont tous différents, alors on peut appliquer l'algorithme et avoir T.

Et, comme l'ordre défini par w' est un raffinement de l'ordre défini par w, on a que T est un ACPM pour w.

Q 2. On considère $G = (V, \mathcal{P}_2(V), w)$ où $w(v, v') := d(v, v')$.
On fait $n - k$ étapes de Kruskal.

Complexité en $O((n-k)\,\alpha(n))$.

Soit C le résultat d'espacement ε.
Soit C' un autre k-clustering.
Il existe u, v dans 2 composantes différentes de C' et dans la même composante de C.
Montrons que $d(u,v) \leq \varepsilon$ (ce qui implique espacement $(C') \leq \varepsilon$).
Soit st tel que $d(s, t) = \varepsilon$.

Si $d(s,t) = \varepsilon < d(u,v)$ et on sait que st ne crée pas de cycle alors absurde car Kruskal aurait choisit st.

Q3. Arbres non enracinés

- Réflexivité : $\phi = id$
- Symétrie : $\phi' = \phi^{-1}$
- Transitivité : $\phi'' = \phi' \circ \phi$

$$\begin{tikzcd} V_1 \arrow[r, "\phi"] \arrow[rr, bend right, "\phi'' = \phi' \circ \phi"'] & V_2 \arrow[r, "\phi'"] & V_3 \end{tikzcd}$$

Arbre enraciné :

- Réflexivité : $\phi = id$
- Symétrie : $\phi' = \phi^{-1}$
- Transitivité : $\phi'' = \phi' \circ \phi$

$$\begin{tikzcd} r_1 \arrow[r, mapsto] \arrow[rr, bend right] & r_2 \arrow[r, mapsto] & r_3 \end{tikzcd}$$

Q4.

$A \sim D$ avec l'isomorphisme

a	w	f	u	k	r
b	x	g	s		
c	y	h	t		
d	z	i	q		
e	v	j	p		

$C \nsim A, D$ car il existe un sommet de degré 4 relié à trois feuilles dans C mais pas dans A.

$B \nsim A, C, D$ car $\deg_B(2) = 2$ et $\deg_A(\cdot) \neq 2$ $\deg_C(\cdot) \neq 2$.

Q5. $\forall x \in V(T-F)$, $R_{T-F}(x) = R_T(x) - 1$

$$\begin{aligned} C(T-F) &= \{x \in V(T-F) \mid R_{T-F}(x) = R(T-F)\} \\ &= \{x \in V(T-F) \mid R_T(x) = R(T)\} \\ &= C(T) \end{aligned}$$

Q6. Par récurrence forte sur $\#T$. Complexité en $O(n)$, c.f. TD 5.

Q7 $T \sim T' \iff \exists r \in C(T), \exists r' \in C(T'), (T,r) \sim (T',r')$

"$\Leftarrow$" oui

"$\Rightarrow$" $R(\phi(x)) = R(x)$

d'où $C(T') = C(\phi(T)) = \phi(C(T))$.

Q8. On calcule $C(T)$ et $C(T')$ en $O(n)$.

Soit $x \in C(T)$.

Pour tout $x' \in C(T')$, tester $(T,x) \sim (T',x')$.

Complexité en $O(n + 2f(n)) = O(f(n) + n)$.