Algorithmique 2 — TD n° 5.

Q1. (a) on fait un DFS ⤳ $O(n)$

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

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

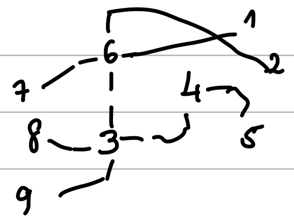
$$mvc[x] := \min \left\{ poids(x) + \sum_{x \to z \to y} mvc[y] \;,\; \sum_{x \to y} mvc[y] \right\}.$$

(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.

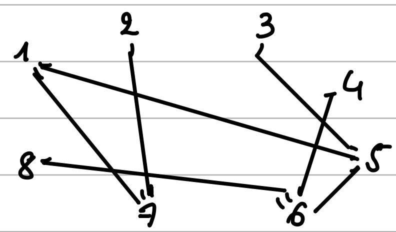
$i \leftarrow 0$

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.

Solution : 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_1.
Absurde.

On conclut $T_1 = T_2$.

(b) Soit $E = \{e_1, \ldots, e_n\}$ tels que $w(e_1) \leq \cdots \leq w(e_n)$.
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 $x \in C(T)$.

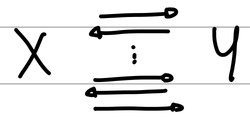
Pour tout $x' \in C(T')$, tester $(T, x) \sim (T', x')$.

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

TD n° 7

I Graphes bipartis

Q1. S'il est biparti et qu'il a un $(2k+1)$-cycle alors



Absurde car $X \neq Y$.

Réciproquement, si G n'est

Q2. DFS en $O(n+m)$ pour avoir un 2-coloriage

II Tri topologique par élagage

Q3. On part que $u \in V$.
Tant que $deg^+(u) > 0$ faire
$\quad u \leftarrow$ un prédécesseur de u

Q4. cycle $\Rightarrow$ $x_1 < \dots < x_n$ dans le tri topo $< x_1$ absurde.
$x_1 \to \dots \to x_n \to x_1$

acyclique $\Rightarrow$ tri topo

Soit v de $deg^+(v) = 0$.

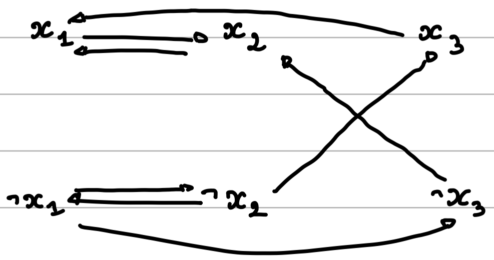
$$v < \underbrace{\boxed{\text{tri topo de } G - v}}_{\tau \text{ acyclique}}$$

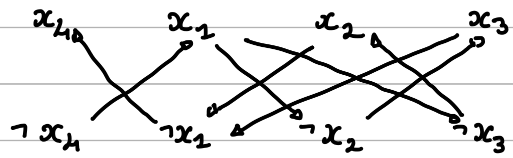
Q5. On calcule tous les deg^+ que l'on maintient.
On extrait tous les sommets de $deg^+ = 0$ (dans une pile)

III Graphe pour 2-SAT

$\neg p \vee q \equiv p \to q$

Q6





Q7. Que le graphe ne contienne pas de cycle avec x_i et $\neg x_i$.

Q8. DFS en O(nombre de clauses)

$\rho : \mathcal{X} \to \mathbb{B}$

$x_i \longmapsto$ VRAI ssi $x_i < \neg x_i$

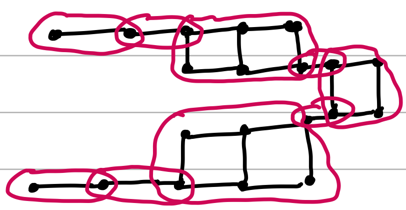
tri topo

IV Points d'articulations, ponts, et composantes 2-connexes.

Q9. Points d'articulation : $\underline{\text{II}}$, $\underline{\text{IX}}$, $\underline{\text{XII}}$, $\underline{\text{XIX}}$

Ponts : $\underline{\text{II}}-\underline{\text{III}}$, $\underline{\text{VIII}}-\underline{\text{IX}}$, $\underline{\text{XIII}}-\underline{\text{XII}}$, $\underline{\text{XIX}}-\underline{\text{XVIII}}$

Comp. bi-connexes :



Q10. Si r est un point d'articulation avec < 2 fils alors retirer r ne déconnecte pas le graphe. Absurde !

Si la racine a ≥ 2 fils alors les sous arbres des fils de r sont des composantes connexes de $G - r$.
D'où point d'articulation.

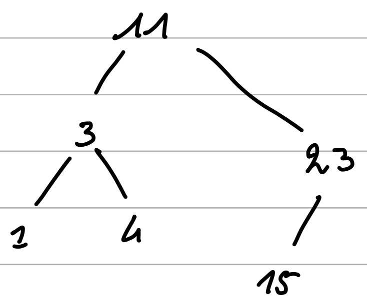
Q11.

1D n° 8

I. Jeu des erreurs

Q1. Vrai

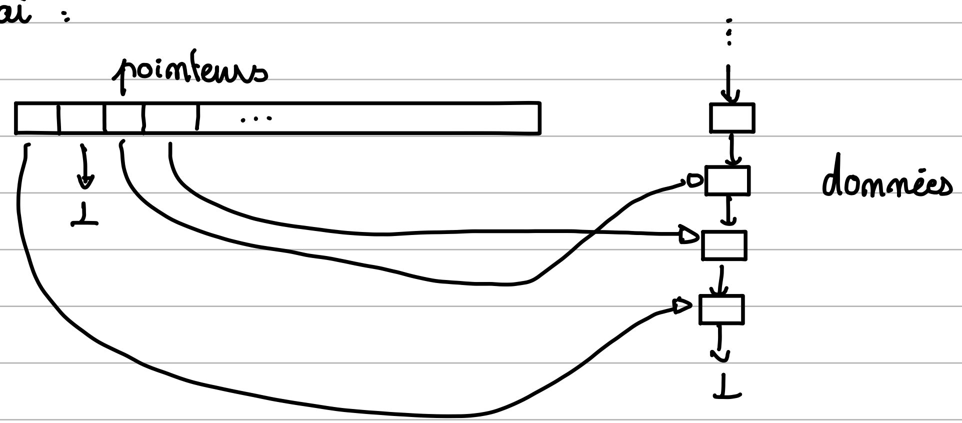
Q2. Faux : 15 < 4



Q3. Faux : 20 18 il faut faire de 7 à 1

Q4. Faux : pas d'hypothèse sur le poids de e

Q5. Vrai :



Q6. On fait l'algorithme de Prim Jarník avec trois buckets

arêtes de poids 1	arêtes de poids 2	arêtes de poids 3	on peut remplir ces arêtes par DFS du graphe

II Coloriage minimal et k-ième minimum

Q7. Par programmation dynamique : $\quad opt : V \times [\![1, \Delta+1]\!] \to \mathbb{N}$

où $\Delta = \max_{v \in V} \deg(v)$

$$opt(u, i) = i + \sum_{u \to v} \min_{1 \leq j \leq \deg v + 1} opt(v, j)$$

Algo en $\underbrace{\sum_{u \in V} \sum_{u \to v}}_{\leq |E|} \underbrace{d(v)}_{\leq \Delta}$

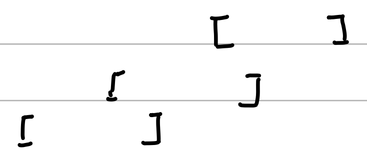
Q8. Tas & k extractions de min

ou... quick select

III Graphes dynamiques

Q9. (a) oui

(b) $\Rightarrow$ oui $\quad \Leftarrow \; c_i = \max(t_i, t_{i+1})$



$\rightsquigarrow$ contre-exemple

(c)

$$TCM := \min_{\substack{\text{chemins} \\ \text{de contagion}}} c_\ell$$

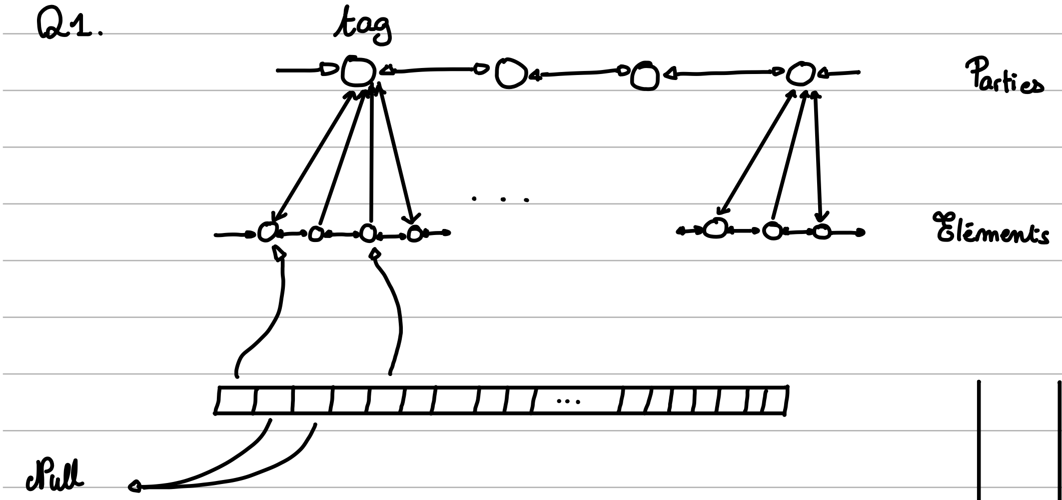
parcours du graphe en $O(|V| + |E|)$

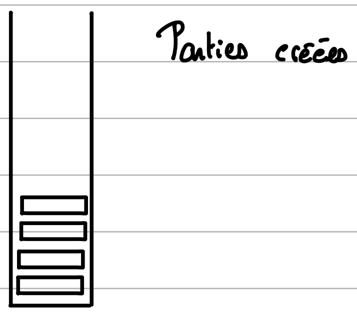
Q 10. On fait un parcours avec une file de priorité.

TD n° 9

I. Raffinement de partition

Q1.





II. Inclusion d'ensembles

Q2. (a) On construit une table de hachage.
On ajoute (i, x) à la table pour tout i et tout $x \in A_i$.
On teste si $(T[i], x)$ est dans la table pour tout i, tout $x \in A_i$.
↳ oui pour tout $i, x \Rightarrow$ oui
↳ non pour un certain $i, x \Rightarrow$ non

D'où une complexité en $O^*(n+m)$

(b) Solution magie magie

$M \leftarrow$ malloc (n^2)
$V \leftarrow$ malloc (n^2) } magie magie c'est un $O(1)$
$cnt \leftarrow 0$

Add$((i, x))$:=
- $M[i \cdot n + x] \leftarrow cnt$
- $V[cnt] \leftarrow i \cdot n + x$
- $cnt \leftarrow cnt + 1$

Membership$((i, x))$:=
- Si $M[i \cdot n + x] \geq cnt$ alors faux
- Sinon si $V[M[i \cdot n + x]] \neq i \cdot n + x$
  - alors faux
  - sinon vrai

D'où une complexité en $O(n+m)$

Autre solution :

Pour tout j, $U[j] \leftarrow \{i \mid T[i]=j\}$

$E \leftarrow$ [faux, ..., faux]

Pour tout j

Pour tout $x \in A_j$, $E[x] \leftarrow$ vrai

Pour tout $i \in U[j]$

Pour tout $y \in A_i$ Tester $E[y]$

Pour tout $x \in A_j$, $E[x] \leftarrow$ faux

III LexBFS

Q3. Si $L[j] \not- L[i] - L[k]$ où $i < j < k$

alors

si, au moment de traiter l'indice i, $L[j]$ et $L[k]$ sont dans la même partie alors on a $k < j$, ce qui n'est pas possible.

D'où $L[j]$ et $L[k]$ sont déjà dans des parties différentes.

Soit t minimal tel que $L[t] - L[j]$ xou $L[t] - L[k]$.

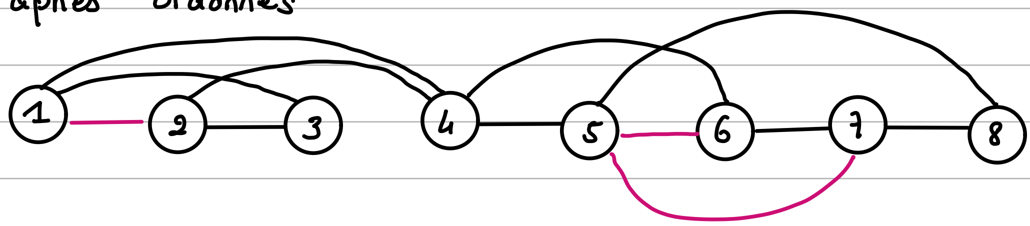
Par $j < k$ alors $L[t] - L[j]$ et $L[t] \not- L[k]$.

Q4. (a) $$\sum_{x \in V} \Big(2 \cdot O(1) + O(\# N(x))\Big) + O(\# V) = O(2n + 2m)$$

(b) On extrait le max avec la structure de Q1.

IV Graphes ordonnés

Q5.



Q6. Si σ' simplicial et $G^*(\sigma') = G$, $\sigma \preceq \sigma'$

alors

$G \subseteq G^*(\sigma) \subseteq G^*(\sigma') \subseteq G$

d'où σ simplicial

Q7.

(a) On vérifie $\underbrace{N^-(i)}_{A_i} \subseteq \underbrace{N^-(j) \cup \{j\}}_{A_{j+n}}$ pour tout i et $j = \min N^-(i)$.

Complexité en $O(n+m)$.

(b) $[\![1, i-1]\!] \setminus N^-(i) \subseteq \left([\![1, j-1]\!] \setminus N^-(j) \right) \cup \{j\}$
ssi $[\![1, i-1]\!] \setminus N^-(i) \subseteq [\![1, i-1]\!] \setminus N^-(j)$
ssi $N^-(i) \supseteq N^-(j)$

On peut tester ça pour tout i, où $j = \max \{ i + j \}$

TD 10

I. Arêtes sur les plus courts chemins

Q1. On fait un Dijkstra et on obtient le poids min p.
On réalise un DFS entre s et t où l'on s'arrête lorsque le poids $> p$.
Sur le retour du DFS, on ajoute les arêtes qui ont abouties (et qui ne sont pas déjà ajoutés).
Co pile Complexité : $O(m + n \log n)$

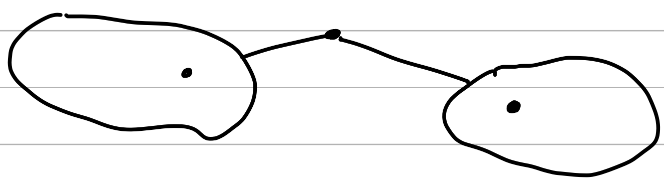
Q2. okay.

Solution 1.
Lance Dijkstra pour calculer $d(s, -)$ dans G
Lance Dijkstra pour calculer $d(-, t)$ dans G^t
On parcourt les arêtes et on ajoute $uv \in E$
si $d(s, u) + d(v, t) + w(uv) = d(s, t)$
$O(3m + 2n \log n)$

Solution 2.
Lance Dijkstra sur G et on retient les sommets du voisinages "duquel on vient".
On obtient un graphe $\tilde{G}$ que l'on transpose.
DFS sur $\tilde{G}^t$.
Complexité en $O(m + n \log n)$

II Sommet le plus lourd

Q3. On parcourt les sommets en pondération croissante.
Lorsqu'on considère un sommet v, on unifie les composantes connexes et on considère les couples dans des comp. connexes distinctes.



$O(n^2)$.

III. Frogger

Q4. Par programmation dynamique, $DP[i,k]$ = énergie minimale qu'il faut pour aller de i à n en au plus k sauts.
Solution: $\min\{k \in \mathbb{N} \mid e \geq E(0,k)\}$.

$$E[i,k] = \min_{j \geq i} \left[c(x_j - x_i)^2 + \max\left(E[j, k-1] - \text{nourriture}(j), 0\right)\right.$$

IV L'argent ne dort jamais

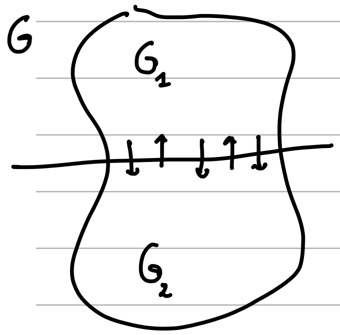
Q5. On passe au $-\log$ et on cherche un cycle de poids négatif.
Ceci peut se faire en $O(n \cdot m)$ avec l'algorithme de Bellman-Ford.

Q6. On part d'une arête et on remonte par les prédécesseurs.

Relâcher (u, v)
  Si $d(u) + w(u,v) < d(v)$
    alors $d(v) \leftarrow d(u) + w(u,v)$ et $pred[v] \leftarrow u$.

V. Chérie, j'ai réduit les enfants.

Matrix multiplication $\multimap$ Transitive closure : $E(n) = O(n^{\omega})$



1) CFC & tri topologique par Kosaraju
2) Coupe G en deux G_1, G_2
3) Appels récursifs

$M(G) = M(G_1) \times M(\text{interface}) \times M(G_2)$

D'où

$$T(n) = 2T(n/2) + 2E(n) + O(n^2)$$
$$= 2T(n/2) + O(E(n))$$
$$= O(E(n))$$