TD Théorie n°2

# NP-difficulté par réduction

## Correction

Un exercice par jour, pendant 24 jours, comme un calendrier de l'Avent (mais en février, pas en décembre).

Lorsque des exercices sont similaires, une démonstration complète est faite pour le premier exercice, et les autres sont présentés de manière plus concise.

Dernière mise à jour : 15/02/2024, jour #15

Certains exercices seront corrigés plus tard, Pas dans l'ordre du sujet :

- Q6. AUTO-MOT-NON-ACCEPTÉ :
- Q7. LANG-REG-FINIS-DIFFÉRENTS ;
- Q11. ENSEMBLES-DISJOINTS.

*Hugo SALOU*

## — Question Q1. —

Montrons que PAS-TOUT-ÉGAL-3SAT est **NP**-complet.

(1) Premièrement, ce problème est dans **NP**. Vérifier que $\varphi$ peut être satisfaite par une valuation « pas tout égal » peut se vérifier en temps polynomial. En effet, la vérification de $[\![\varphi]\!]^\rho = \boldsymbol{V}$ se fait en temps polynomial (comme dans la réduction de 3SAT), et vérifier que toutes les clauses contiennent deux littéraux ayant une valuation différente peut se faire en procédant clause par clause. Ce problème est vérifiable en temps polynomial avec un certificat de taille polynomiale (la valuation demandée), il est donc dans **NP**.

(2) Deuxièmement, réduisons 3SAT à PAS-TOUT-ÉGAL-4SAT. Soit $\varphi$ une entrée de 3SAT. Soit, de plus, $y \notin \operatorname{vars} \varphi$. Pour chaque clause $c$ de $\varphi$, on la modifie en $c' = c \vee y$. On construit donc une nouvelle formule $\varphi'$ sous forme 4CNF, et cette construction se fait en temps polynomial. Montrons maintenant l'équivalence $\varphi' \in \text{PAS-TOUT-ÉGAL-4SAT}^+ \iff \varphi \in \text{3SAT}^+$.

« $\Longleftarrow$ ». Supposons $\varphi$ satisfiable. Soit ainsi $\rho \in \mathbb{B}^{\operatorname{vars}\varphi}$ une valuation qui la satisfait, autrement dit, tel que $[\![\varphi]\!]^\rho = \boldsymbol{V}$. Ainsi, en posant $\rho' = \rho \uplus (y \mapsto \boldsymbol{F})$, on obtient une valuation qui satisfait la formule $\varphi'$. Cette valuation est bien « pas tout égal », on en déduit donc que $\varphi' \in \text{PAS-TOUT-ÉGAL-4SAT}^+$.

« $\Longrightarrow$ ». Supposons $\varphi' \in \text{PAS-TOUT-ÉGAL-4SAT}^+$. Soit ainsi $\rho' \in \mathbb{B}^{(\operatorname{vars}\varphi)\cup\{y\}}$ une valuation « pas tout égal » satisfaisant $\varphi'$. Si $y$ est valué à $\boldsymbol{F}$, il suffit de poser $\rho = \rho'|_{\operatorname{vars}\varphi}$ une valuation qui satisfait $\varphi$. Si $y$ est valué à $\boldsymbol{V}$ alors, par définition du problème PAS-TOUT-ÉGAL-4SAT, dans chaque clause $c$, il existe un autre littéral $\ell_c \neq y$ valué à $\boldsymbol{F}$. On peut en déduire une valuation $\mu$ telle que $[\![\ell_c]\!]^\mu = \boldsymbol{V}$, pour toute clause $c$. L'existence de la valuation $\mu$ est assurée par la Remarque A.1 (page 10). On en déduit donc que $\mu$ satisfait $\varphi$.

(3) Troisièmement, réduisons PAS-TOUT-ÉGAL-4SAT à PAS-TOUT-ÉGAL-3SAT. Pour cela, soit $\varphi$ une entrée de PAS-TOUT-ÉGAL-4SAT. Pour chaque clause $c$ de $\varphi$, on considère une variable $a_c \notin \operatorname{vars}\varphi$ (qui sont différentes pour chaque clause), et on transforme la 4-clause $c = \ell_1 \vee \ell_2 \vee \ell_3 \vee \ell_4$ en deux 3-clauses $c'_1 = \ell_1 \vee \ell_2 \vee a_c$ et $c'_2 = \ell_3 \vee \ell_4 \vee \neg a_c$. On définit donc une nouvelle formule $\varphi'$, et cette construction se réalise en temps polynomial : une 4CNF avec $m$ clauses et $n$ variables sera transformée en une 3CNF avec $2m$ clauses et avec $n + m$ variables. Il ne reste qu'à montrer l'équivalence suivante :

$$\varphi \in \text{PAS-TOUT-ÉGAL-4SAT}^+ \iff \varphi' \in \text{PAS-TOUT-ÉGAL-3SAT}^+.$$

« $\Longrightarrow$ ». Soit $\rho \in \mathbb{B}^{\operatorname{vars}\varphi}$ une valuation « pas tout égal » de $\varphi$ qui la satisfait. Nous devons créer une valuation $\rho \in \mathbb{B}^{\operatorname{vars}\varphi'}$ « pas tout égal » de $\varphi'$ qui la satisfait. Pour cela, nous devons déterminer la valuation de $a_c$. On procède comme suit. Pour simplifier les notations, on ne se concentrera que sur une clause $c$, mais le passage à de multiples clauses se fait sans problème. Si $[\![\ell_1]\!]^\rho = [\![\ell_2]\!]^\rho = b$, alors on pose $\rho' = \rho \uplus (a_c \mapsto \overline{b})$. Si $[\![\ell_3]\!]^\rho = [\![\ell_4]\!]^\rho = b$, alors on pose $\rho' = \rho \uplus (a_c \mapsto b)$. Sinon, la valuation de $a_c$ n'a pas d'importance et on peut la choisir arbitrairement, on posera donc $\rho' = \rho \uplus (a_c \mapsto \boldsymbol{V})$. Si $c = \ell_1 \vee \ell_2 \vee \ell_3 \vee \ell_4$ a deux littéraux valués différemment, alors les deux clauses $c'_1$ et $c'_2$ aussi.

« $\Longleftarrow$ ». On procède par contraposée en se concentrant sur la difficulté. Supposons que, pour toute valuation $\rho$, il existe une formule sous 4CNF qui contient une clause dont les littéraux sont toujours valués au même booléen. Alors, il est clair qu'une des deux clauses de la 3CNF aura une tous ses littéraux valués au même booléen, quel que soit le choix de la valuation de $a_c$.

On en conclut, par les réductions suivantes sur la **NP**-complétude de PAS-TOUT-ÉGAL-3SAT, car le problème 3SAT est **NP**-difficile et les réductions donnent :

$$\text{3SAT} \preccurlyeq_{\text{p}} \text{PAS-TOUT-ÉGAL-4SAT} \preccurlyeq_{\text{p}} \text{PAS-TOUT-ÉGAL-3SAT}.$$

*Hugo SALOU*

### — Question Q2. —

Il est clair que ce problème est dans **NP**, avec un argument similaire à celui de Q1.

Procédons à la réduction depuis 3SAT à 1-PARMI-AU-PLUS-7SAT. Soit $\varphi$ une formule sous 3CNF. Pour chaque clause, on définit 8 variables $y_{000}, \dots, y_{111}$. L'idée est qu'une valuation d'une clause $\ell_1 \vee \ell_2 \vee \ell_3$ correspond à une valuation où la variable $y_{ijk}$ est valuée à $\boldsymbol{V}$ si et seulement si $i = f([\![\ell_1]\!]^\rho)$, $j = f([\![\ell_2]\!]^\rho)$ et $k = f([\![\ell_3]\!]^\rho)$, où $f$ transforme un booléen en entier.[1] On notera cette condition sur $i, j, k$ (♪). Une 3-clause $\ell_1 \vee \ell_2 \vee \ell_3$ est donc transformée en 7 clauses :

$$
\begin{aligned}
&(1) : y_{001} \vee y_{010} \vee y_{011} \vee y_{100} \vee y_{101} \vee y_{110} \vee y_{111} \\
&(2) : \ell_1 \vee y_{000} \vee y_{001} \vee y_{010} \vee y_{011} \\
&(3) : \neg\ell_1 \vee y_{100} \vee y_{101} \vee y_{110} \vee y_{111} \\
&(4) : \ell_2 \vee y_{000} \vee y_{001} \vee y_{100} \vee y_{101} \\
&(5) : \neg\ell_2 \vee y_{010} \vee y_{011} \vee y_{110} \vee y_{111} \\
&(6) : \ell_3 \vee y_{000} \vee y_{010} \vee y_{010} \vee y_{110} \\
&(7) : \neg\ell_3 \vee y_{001} \vee y_{011} \vee y_{101} \vee y_{111}.
\end{aligned}
$$

Si $\ell_1 \vee \ell_2 \vee \ell_3$ est valué à $\boldsymbol{V}$, alors on définit $\rho(y_{ijk}) = \boldsymbol{V}$ *si* $i, j, k$ vérifient la condition (♪) et $\rho(y_{ijk}) = \boldsymbol{F}$ sinon. Ainsi, chacune des 7 clauses contient précisément un littéral valué à $\boldsymbol{V}$. Réciproquement, si chaque clause contient exactement un littéral valué à $\boldsymbol{V}$, c'est que précisément l'une des 6 variables de la clause $(1)$ doit être valuée à $\boldsymbol{V}$. Et, par les clauses $(2)$–$(7)$, la variable choisit doit correspondre à une valuation des littéraux $\ell_1$, $\ell_2$ et $\ell_3$. Ceci implique donc que $\ell_1 \vee \ell_2 \vee \ell_3$ est valué à $\boldsymbol{V}$. D'où la réduction.

Pour démontrer la réduction de 1-PARMI-AU-PLUS-7SAT à 1-PARMI-3-SAT, on procède comme dans Q1 en ajoutant des variables pour réduire la taille des clauses.

Ceci implique donc que 1-PARMI-3-SAT est **NP**-difficile, donc **NP**-complet car il est dans **NP**.

### — Question Q3. —

Ce problème est *clairement* vérifiable en temps polynomial. Il suffit d'effectuer le produit $\boldsymbol{Ax}$ et de comparer terme à terme avec $\boldsymbol{b}$. Il est donc dans **NP**.

Pour démontrer la réduction, on transforme une clause $\ell_1 \vee \ell_2 \vee \ell_3$ d'une formule $\varphi$ sous forme 3CNF en une équation linéaire $v_1 + v_2 + v_3 \geq 1$, où $v_i = 1 - x_i$ si $\ell_i = \neg p_i$ et $v_i = x_i$ si $\ell_i = p_i$, où $p_i \in \operatorname{vars} \varphi$. De cette manière, un ensemble de clauses sera transformé en un système d'inéquations linéaires. On peut donc représenter matriciellement ce système. Un vecteur $\boldsymbol{x} \in \mathscr{M}_{1,n}(\{0,1\})$ est solution de $\boldsymbol{Ax} \leq \boldsymbol{b}$ si et seulement si chaque clause contient au moins un littéral valué à $\boldsymbol{V}$. Ceci démontre la réduction.

On en déduit que PROG-ENTIÈRE-0-1 est **NP**-complet.

[1] Explicitement :

$$
\begin{aligned}
f : \mathbb{B} &\longrightarrow \{0,1\} \\
\boldsymbol{V} &\longmapsto 1 \\
\boldsymbol{F} &\longmapsto 0
\end{aligned}
$$

— **Question Q4.** —

Ce problème est *clairement* vérifiable en temps polynomial : il suffit de trois boucles `for`. Ce problème est donc dans **NP**.

Cette réduction n'utilisera pas exactement l'indication donnée. Une réduction est possible en l'utilisant totalement, mais est, je trouve, moins claire. On réduira ce problème à PROG-ENTIÈRE-0-1. On procèdera en deux temps.

Soient $\langle(\boldsymbol{A}, \boldsymbol{b})\rangle$ une entrée de PROG-ENTIÈRE-0-1. Nous allons transformer ces $n$ *inéquations* linéaires en $n$ *équations* linéaires. On note ce problème modifié PROG-ENTIÈRE-0-1-EQ. On supposera que les inéquations sont la forme de celles définies en Q3 : $v_1 + v_2 + v_3 \leq 1$. On transforme chacune des ces inéquations en $n$ équations en introduisant deux variables $y$ et $z$ (différentes pour chaque clause), afin d'obtenir l'équation $v_1 + v_2 + v_3 + y + z = 3$. Les termes $y$ et $z$ sont là pour « compenser » dans le cas où la somme $v_1 + v_2 + v_3 < 3$. Il est clair que $\boldsymbol{x}$ est solution du système d'inéquations *ssi* $\boldsymbol{x}'$ est solution du système d'équations. D'où la réduction.

Soient $\langle(\boldsymbol{A}, \boldsymbol{b})\rangle$ une entrée de PROG-ENTIÈRE-0-1-EQ. Construisons une entrée $\langle(\boldsymbol{A}', \boldsymbol{b}')\rangle$ du problème. Aux équations linéaires définies par le système matriciel $(\boldsymbol{A}, \boldsymbol{b})$, on ajoute les équations quadratiques $v_i^2 - v_i = 0$. Cette transformation est calculable en temps polynomial. Il est clair que $\boldsymbol{x}$ est solution de $(\boldsymbol{A}, \boldsymbol{b})$ *ssi* $\boldsymbol{x}'$ est solution de $(\boldsymbol{A}', \boldsymbol{b}')$. D'où, la réduction.

Par **NP**-difficulté de PROG-ENTIÈRE-0-1, on en déduit que PROG-ENTIÈRE-0-1-EQ est **NP**-complet.

— **Question Q5.** —

Dans cette preuve plus complexe, on ne donnera que les étapes importants de la preuve. Pour plus de détails, on pourra consulter le DS4, Exercice 4 du 7 janvier 2023.

Dans un premier temps, on peut vérifier qu'une solution fonctionne en temps polynomial. En effet, la configuration initial et la configuration finale seront considérées comme certificat. Vérifier que la condition finale découle de la condition initiale est possible en temps polynomial, en $\Theta(m^2)$ car il s'agit du test d'inclusion entre deux sous-ensembles de $[\![1, m]\!]^2$. On en déduit que JEU-SOLVABLE est dans **NP**.

Réduisons ce problème à 3SAT (en réalité, cela fonction pour CNF-SAT, mais on se place dans le cas de 3SAT pour respecter le titre de la section). Soit $\varphi$ une formule avec $m$ variables et $k$ clauses. On notera donc $\operatorname{vars} \varphi = \{x_1, \dots, x_m\}$. On pourra supposer, en pré-traitement, qu'aucune clause de $\varphi$ ne contient $x_i$ et $\neg x_i$, avec $x_i \in \operatorname{vars} \varphi$ ; on peut supprimer $x_i$ ou $\neg x_i$ sans changer la satisfiabilité de la formule $\varphi$. Construisons le tablier de taille $m \times k$ comme suit.

- Si $x_i$ est dans la $j$-ème clause, placer une pierre bleue dans la $i$-ème colonne, à la $j$-ème ligne.
- Si $\neg x_i$ est dans la $j$-ème clause, placer une pierre rouge dans la $i$-ème colonne, à la $j$-ème ligne.

On peut transformer le tablier en un carré en ajoutant des colonnes supplémentaires, ou en recopiant des lignes si nécessaire. Ceci n'affectera pas le caractère gagnant d'un tablier. Montrons que $\varphi$ est satisfiable *ssi* le tablier est solvable.

« $\Longrightarrow$ ». Soit $\rho \in \mathbb{B}^{\operatorname{vars} \varphi}$ une valuation satisfaisant $\varphi$. Si $\rho(x_i)$ est $\boldsymbol{V}$ (resp. $\boldsymbol{F}$), retirer les pierres rouges (resp. bleues) de la $i$-ème colonne. Ainsi, seules les pierres correspondant à des littéraux valués à $\boldsymbol{V}$ restent. Parce que chaque clause a un littéral valué à $\boldsymbol{V}$, chaque ligne a une pierre. De plus, chaque colonne contient une seule couleur, sans quoi, on aurait à la fois $\rho(x_i) = \boldsymbol{V}$ et $\rho(x_j) = \boldsymbol{F}$.

« $\Longleftarrow$ ». Soit une solution du jeu. Si les pierres rouges (resp. bleues) ont été retirées de la colonne $i$, définir $\rho(x_i) = \boldsymbol{V}$ (resp. $\boldsymbol{F}$). Chaque ligne a une pierre restante, donc chaque clause a un littéral valué à $\boldsymbol{V}$. D'où, $[\![\varphi]\!]^\rho = \boldsymbol{V}$.

D'où la réduction. On en déduit donc que JEU-SOLVABLE est **NP**-complet.

— **Question Q6.** —

Cette question est plus compliquée, la correction sera faite plus tard ...

— **Question Q7.** —

Cette question est plus compliquée, la correction sera faite plus tard ...

### — Question Q8. —

Le problème est vérifiable en temps polynomial, les certificats sont les indices des éléments choisis dans la famille qui forme une partition. Si $C = \{X_i \mid i \in I\}$, le certificat est $J$ tel que $\{X_i \mid i \in J\}$ qui forme une partition de $U$. Le problème est donc dans **NP**.

On réduit depuis 1-PARMI-3-SAT. Soit $\varphi$ une formule sous 3CNF avec des variables $x_1, \dots, x_n$ et des clauses $c_1, \dots, c_k$. On choisit $U = \{v_1, \dots, v_n, c_1, \dots, c_\ell\}$. On définit $2n$ sous-ensembles :

$$S_{x_i} = \{v_i\} \cup \{c_j \mid x_i \in c_j\},\text{[2]}$$
$$\text{et } S_{\neg x_i} = \{v_i\} \cup \{c_j \mid \neg x_i \in c_j\}.\text{[2]}$$

Le calcul $\varphi \mapsto \left\langle \left(U, \left\{S_{x_1}, S_{\neg x_1}, \dots, S_{x_n}, S_{\neg x_n}\right\}\right)\right\rangle$ peut être réalisé en temps polynomial.

« $\Longrightarrow$ ». Supposons que $\varphi$ soit satisfiable. Soit $\rho \in \mathbb{B}^{\{v_1, \dots, v_n\}}$ une valuation « 1 parmi 3 »[3] satisfaisant $\varphi$. Alors, la famille

$$\mathscr{P}_\rho = \{S_\ell \mid [\![\ell]\!]^\rho = \boldsymbol{V}\}$$

est une partition de $U$. En effet, pour tout $i \in [\![1, n]\!]$, et $j \in [\![1, k]\!]$,
- chaque élément $v_i$ apparaît seulement dans $S_{x_i}$ et $S_{\neg x_i}$, et on sélectionne précisément *un* des deux ensembles ;
- chaque élément $c_j = \ell_1 \vee \ell_2 \vee \ell_3$ apparaît seulement dans trois ensembles $S_{\ell_1}$, $S_{\ell_2}$ et $S_{\ell_3}$. Comme $\rho$ est une valuation « 1 parmi 3 », précisément un littéral sera valué à $\boldsymbol{V}$ et donc précisément *un* $S_\ell$ est un élément de $\mathscr{P}_\rho$.

« $\Longleftarrow$ ». Supposons avoir une partition $\mathscr{P}$. Alors, pour $i \in [\![1, n]\!]$, les ensembles $S_{x_i}$ et $S_{\neg x_i}$ contiennent l'élément $v_i$ et ainsi, précisément un d'entre eux appartient à $\mathscr{P}$. Posons $\rho(x_i) = \boldsymbol{V}$ si $S_{x_i} \in \mathscr{P}$ et sinon, on pose $\rho(x_i) = \boldsymbol{F}$. Autrement dit, on construit $\rho$ de telle sorte que chaque littéral $\ell$ soit valué à $\boldsymbol{V}$ *ssi* $S_\ell \in \mathscr{P}$. Comme chaque variable $c_j$ apparaît dans précisément un ensemble $S_\ell$, chaque clause contient précisément une variable valuée à $\boldsymbol{V}$. On en conclut que $\rho$ est une valuation « 1 en 3 ».

D'où la réduction. On en déduit que CONTIENT-PARTITION est **NP**-complet.

### — Question Q9. —

Ce problème est clairement dans **NP**. Il suffit de calculer la somme. Le certificat est le sous-ensemble $I$.

Réduisons depuis CONTIENT-PARTITION. Soit $b$ le nombre d'ensembles dans $C$ *plus 1*.[4] Supposons que $U = [\![0, n-1]\!]$.[5] Pour chaque ensemble $S \in C$, on associe l'entier $x_S = \sum_{u \in S} b^u$. Ainsi, la décomposition dans la base $b$ contient seulement des 0 et des 1, et les 1 apparaissent précisément aux positions $u \in S$. On choisit $k = \sum_{i=0}^{n-1} b^i$.

Remarquons que si on ajoute $b-1$ nombres à $x_S$ en utilisant le développement dans la base $b$, la $u$-ième décimale vaut la somme des $u$-ièmes décimales des nombres (on n'a pas besoin de retenues pour la décimale suivante). Ainsi, l'ensemble $\{x_S \mid S \in C\}$ contient un sous-ensemble de somme $k$ *ssi* $C$ contient une partition de $U$. D'où la réduction.

On en déduit que le problème SUBSET-SUM est **NP**-complet.

---

[2] On dit que $\ell \in c_j$ si $\ell$ est *exactement* dans $c_j$, *i.e.* $c_j$ peut s'écrire $c_j = \dots \vee \ell \vee \dots$. Attention, si $\ell = p$, si $c = \neg p \vee q$, on n'a pas $p \in c$, il y a un « $\neg$ ».

[3] Par « 1 parmi 3 », on veut dire que, pour toute 3-clause de $\varphi$, un *seul* littéral sera valué à $\boldsymbol{V}$, pas plus pas moins.

[4] Pas nécessaire, mais il simplifie la réduction.

[5] On pourra éventuellement numéroter les indices des éléments de $U$ par des entiers de cet intervalle.

— **Question Q10.** —

Ce problème est clairement vérifiable en temps polynomial, il suffit de calculer les sommes. Il est donc dans **NP**.

Réduisons depuis SUBSET-SUM. Soit $(X, k)$ une entrée du problème SUBSET-SUM. On pose $S$ la somme de l'ensemble $X$, *i.e.* $S = \sum_{x \in X} x$. On considère $X' = X \cup \{k + S, 2S - k\}$, qui est toujours un sous-ensemble de $\mathbb{Z}$. Cette réduction peut être calculée en temps polynomial.

Le problème SET-PARTITION est défini à l'aide de familles. On préfère considérer un ensemble fini plutôt, afin de simplifier les notations dans la réduction.

« $\Longrightarrow$ ». S'il existe $I \subseteq X$, de somme $k$, alors on partitionne $X'$ en $I \cup \{2S - k\}$ et $(X \setminus I) \cup \{S + k\}$. La somme de la première partie vaut $k + (2S - k) = 2S$ ; la somme de la deuxième partie vaut $(S - k) + (S + k) = 2S$.

« $\Longleftarrow$ ». S'il existe une partie $I \subseteq X$ telle que $\sum_{i \in I} i = \sum_{i \notin I} i = 2S$. Alors, il existe un sous-ensemble de $X$ de somme $k$. En effet, $(S + k) + (2S - k) = 3S$, ces deux éléments sont donc dans deux parties différentes, un dans $I$, l'autre dans $X \setminus I$. Sans perdre en généralité, nous supposerons que $2S - k \in I$. Alors, $I' = I \setminus \{2S - k\} \subseteq X$, et a pour somme $k$.

D'où la réduction. On en déduit que le problème est **NP**-complet.

— **Question Q11.** —

J'ai envie de changer, on passe à des exercices de graphes . . . Ce problème sera corrigé plus tard

— **Question Q12.** —

Ce problème est dans la classe **NP**. En effet, il suffit de prendre la partie $U$ en certificat, de taille $k$. On peut vérifier que les $k$ sommets sont deux-à-deux non connectés et ce, en temps polynomial.

Soit $(G, k)$ une entrée de CLIQUE. On pose $G = (V, E)$. On considère le graphe $G'$ définit par

$$G' = (V, (V \times V) \setminus E).$$

On peut construire un tel graphe en temps polynomial.

Remarquons que l'on a l'équivalence : $G$ contient un ensemble indépendant de taille $k$ *ssi* $G'$ contient une clique de taille $k$. Cette équivalence est assurée par la dualité des définitions entre une clique et un ensemble indépendant (« sommets deux à deux non connectés » devient « sommets deux à deux connectés »).

D'où la réduction. On en déduit que le problème est **NP**-complet.

— **Question Q13.** —

Vérifier qu'un ensemble est une couverture par sommets peut se faire en temps polynomial. Il suffit de vérifier que chaque arête a une extrémité dans $V'$. On peut également vérifier aisément la cardinalité de $V'$. On en déduit que le problème est dans **NP**.

Réduisons depuis ENSEMBLE-INDÉPENDANT. Soit $((V, E), k)$ une entrée du problème ENSEMBLE-INDÉPENDANT. Nous avons l'équivalence suivante : $U \subseteq V$ est un ensemble indépendant *ssi* $V \setminus U$ est une couverture par sommets. Ceci est assuré par définition. L'ensemble indépendant $U$ a un cardinal d'au plus $k$ *ssi* $V \setminus U$, la couverture par sommets, a un cardinal d'au moins $|V| - k$.

D'où la réduction. On en déduit que le problème est **NP**-complet.

— **Question Q14.** —

Vérifier qu'il y a bien $k$ ensembles et que l'union de ces ensembles est bien $E$ est faisable en temps polynomial. On en déduit que le problème est dans **NP**.

On réduit depuis COUVERTURE-SOMMETS. Soit $G = (V, E)$ et $k$ une entrée du problème COUVERTURE-SOMMETS. On numérote les sommets : $V = \{v_1, \dots, v_m\}$. Pour chaque sommet $v_i \in V$, on pose $S_i$ l'ensemble des arêtes connectées à $v_i$ :

$$S_i = \{e \in E \mid v_i \in e\}.$$

Cette construction est faisable en temps polynomial.

On peut couvrir $E$ avec au moins $k$ ensembles *ssi* $G = (V, E)$ a une couverture par sommets de taille au moins $k$. Ceci est assuré par définition des ensembles $S_i$ et d'une couverture par sommets.

D'où la réduction. On en déduit que le problème est **NP**-complet.

— **Question Q15.** —

Vérifier qu'il y a bien $k$ ensembles et que l'union de ces ensembles est bien $E$ est faisable en temps polynomial. On en déduit que le problème est dans **NP**.

On réduit depuis ENSEMBLE-INDÉPENDANT. Soit $G = (V, E)$ et $k$ une entrée du problème ENSEMBLE-INDÉPENDANT. On numérote les sommets : $V = \{v_1, \dots, v_m\}$. Pour chaque sommet $v_i \in V$, on pose $S_i$ l'ensemble des arêtes connectées à $v_i$ :

$$S_i = \{e \in E \mid v_i \in e\}.$$

Cette construction est faisable en temps polynomial.

Un ensemble $U$ est un ensemble indépendant *ssi* les ensembles $S_i$ et $S_j$ sont disjoints pour tout couple de sommets $(v_i, v_j) \in U^2$. Les contraintes sur les cardinalités correspondent également.

D'où la réduction. On en déduit que le problème est **NP**-complet.

— **Question Q16.** —

Vérifier ce problème est possible en temps polynomial. En effet, avec un tel chemin $\gamma$ en certificat, vérifier qu'il va de $s$ à $t$ se fait en $\mathrm{O}(|\gamma|)$. De plus, vérifier qu'il visite au plus un sommet de chaque paire $(u_i, v_i)$ se fait en $\mathrm{O}(|\gamma| \cdot n)$. On en déduit que le problème est dans **NP**.

Réduisons le depuis 3SAT. Soit $\varphi$ une formule sous 3CNF. On note $n$ le nombre de clauses de $\varphi$, et $m$ le nombre de variables. Ainsi, on peut écrire $\varphi = \bigwedge_{i=1}^{n} \phi_i$, où $\phi_i = \ell_{i,1} \vee \ell_{i,2} \vee \ell_{i,3}$. Chaque littéral $\ell_{i,j}$ est, ou bien $x_k$, ou bien $\neg x_k$, pour $x_k \in \mathrm{vars}\ \varphi$. On construit le graphe $G$ comme ci-dessous.

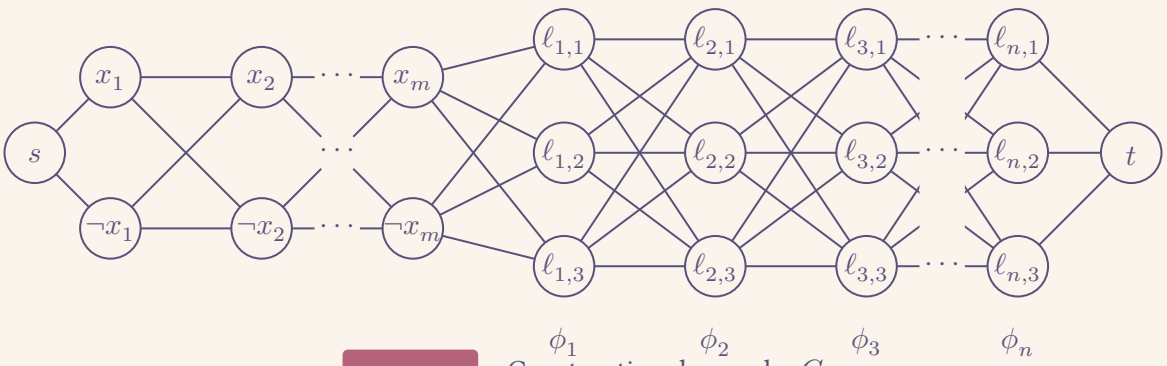


**Figure 1.** Construction du graphe $G$

On considère le $m$-uplet $F$ de couples suivant : $(x_1, \neg x_1), (x_2, \neg x_2), \ldots, (x_m, \neg x_m)$. Le graphe $G$ et le $m$-uplet $F$ sont construits de telle sorte qu'il existe un chemin entre $s$ et $t$ *ssi* $\varphi$ est satisfiable. En effet, un chemin dans ce graphe correspond à une valuation. La contrainte $F$ implique que l'on ne puisse passer que par $x_k$ ou $\neg x_k$, mais pas les deux, on peut donc construire la valuation « sans contradictions ».[6]

D'où la réduction. On en déduit que le problème est **NP**-complet.

— **Question Q17.** —

Cette réduction demande beaucoup de construction, de bonnes idées, et de la justification. Je pourrai essayer de la représenter sur un schéma, mais la réduction serait encore moins claire. C'est pour cela que je recommande cette vidéo Youtube (en Anglais). Elle explique clairement les étapes de construction du graphe, et la démonstration de la réduction. C'est possible de formaliser cette réduction, mais ça ne permettrai pas de faire comprendre clairement la réduction.

— **Question Q18.** —

[6] Ce que j'entends par là, c'est qu'on n'a pas à avoir $\rho(x_k) = \boldsymbol{V}$ d'une part, et $\rho(x_k) = \boldsymbol{F}$ d'autre part.

# A. Remarques complémentaires.

## A.1. Q1. Existence d'une valuation $\mu$.

Rappelons que l'on a une formule $\varphi'$ sous forme CNF, et une valuation $\rho$ « pas tout égal » qui satisfait la formule $\varphi'$. Dans chaque clause $c \in \varphi'$, il existe un littéral $\ell_c \in c$ tel que $[\![\ell_c]\!]^\rho = \boldsymbol{F}$, avec $\ell_c \neq y_c$. Notre but est de construire une valuation $\mu$ qui satisfait la formule $\varphi$. Pour cela, dans chaque clause $c \in \varphi$, on veut valuer un littéral à $\boldsymbol{V}$, procédons par disjonction de cas.

- Si $\ell_c = \neg p$, avec $p \in \operatorname{vars} \varphi$, il suffit de définir $\mu(p) = \boldsymbol{F}$.
- Si $\ell_c = p$, avec $p \in \operatorname{vars} \varphi$, il suffit de définir $\mu(p) = \boldsymbol{V}$.

Il est *crucial* de remarquer que l'on ne peut pas avoir, deux définitions différentes de $\mu(p)$. En effet, par définition de $\ell_c$, on a $[\![\ell_c]\!]^\rho = \boldsymbol{F}$. On ne peut donc pas avoir à la fois $[\![\neg p]\!]^\rho = \boldsymbol{F}$ et $[\![p]\!]^\rho = \boldsymbol{F}$. La définition d'une valuation $\mu$ est donc possible, et elle est calculable en $\mathrm{O}(|\varphi|)$, où $|\varphi|$ est le nombre de clauses. (On voit $\varphi$ comme un ensemble de clauses, et une clause comme un ensemble de littéraux, pour simplifier les notations.)