Composition d'informatique n°6

Corrigé

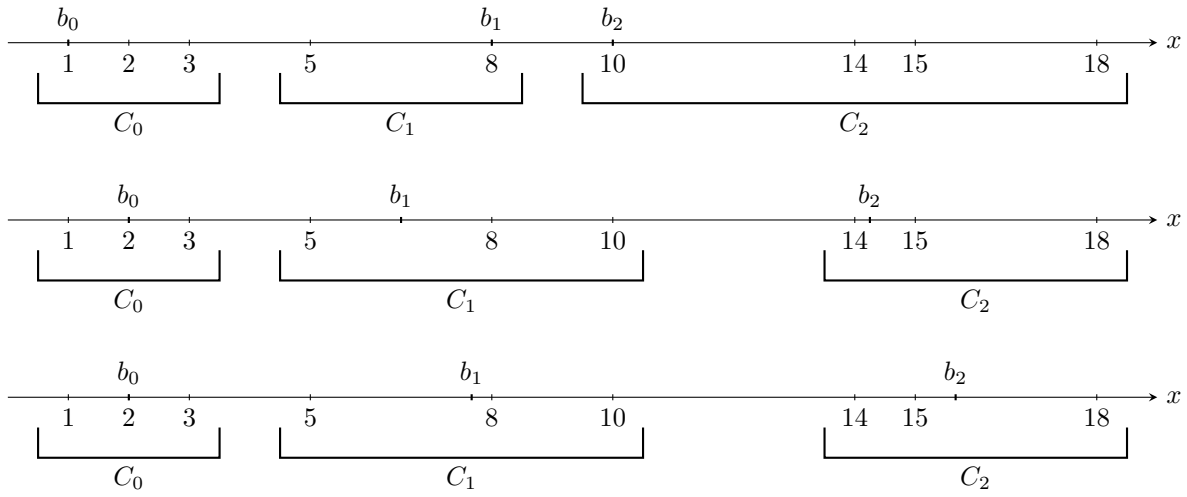
⋆ ⋆ ⋆

1 Clustering en dimension 1

1.1 Préliminaires

Question 1 Si $K = N$, chaque classe d'équivalence est de cardinal 1 (sinon l'une est vide) et le score est nul, ce qui est bien minimal. Si $K = N - 1$, toutes les classes sont de cardinal 1, sauf une qui est de cardinal 2 (par le principe des tiroirs). Pour minimiser le score, il faut mettre dans la même classe les deux éléments les plus proches de E.

Question 2 On a l'exécution suivante :



Question 3 Les erreurs sont :

- ligne 5 : il faut vérifier que ni `i1`, ni `i2` n'a atteint la dernière valeur. On doit remplacer la condition booléenne par `if (i2 == n2 || (i1 < n1 && tab1[i1] <= tab2[i2])){`
- entre les lignes 9 et 10 : il faut penser à incrémenter `i2` en rajoutant `i2++;`
- ligne 15 : le cas d'arrêt doit aussi prendre en compte le cas du tableau à un seul élément, sinon l'algorithme ne termine pas. Il faut remplacer cette ligne par `if (n <= 1) return;`
- entre les lignes 26 et 27 : il faut penser à libérer la mémoire. Il faut rajouter `free(tab1); free(tab2);`.

Question 4 Il suffit juste de faire attention aux indices.

```
double moyenne(double* E, int i, int j){
    double mu = 0;
    for (int k=i; k<j; k++){
        mu += E[k];
    }
    return mu / (j - i);
}
```

Question 5 Le code ressemble au précédent en faisant d'abord le calcul de la moyenne.

```
double somme_emc(double* E, int i, int j){
    double mu = moyenne(E, i, j);
    double S = 0;
    for (int k=i; k<j; k++){
        S += (E[k] - mu) * (E[k] - mu);
    }
    return S;
}
```

Question 6 Le tableau associé à la partition $\mathcal{P} = \{\{0,1,2\},\{3,4\},\{5,6,7\},\{8\}\}$ est `{0, 3, 5, 8}`. La partition associée au tableau `{0, 1, 4, 5}` est $\mathcal{P} = \{\{0\},\{1,2,3\},\{4\},\{5,6,7,8\}\}$.

Question 7 Il faut ici faire attention à la manière dont sont délimitées les classes d'équivalence. On traite la dernière classe à part, qui contient les valeurs jusqu'à x_{N-1}.

```
double score(double* E, int N, int* P, int K){
    double SP = somme_emc(E, P[K - 1], N);
    for (int i=0; i<K-1; i++){
        SP += somme_emc(E, P[i], P[i + 1]);
    }
    return SP;
}
```

Question 8 Il suffit d'une double boucle pour déterminer les indices des plus petits éléments de C_1 et C_2. On calcule le score à chaque étape et on garde la partition de score minimal. On fait attention à différencier l'usage d'un tableau statique et d'un tableau dynamique (pour pouvoir renvoyer le tableau, il faut l'allouer sur le tas).

```
int* clustering3(double* E, int N){
    int Pmin = malloc(3 * sizeof(int));
    Pmin[0] = 0; Pmin[1] = 1; Pmin[2] = 2;
    for (int i=1; i<N-1; i++){
        for (int j=i+1; j<N; j++){
            int P[3] = {0, i, j};
            if (score(E, N, P, 3) < score(E, N, Pmin, 3)){
                Pmin[1] = i; Pmin[2] = j;
            }
        }
    }
    return Pmin;
}
```

Question 9 On donne les complexités des fonctions précédentes :

- `moyenne(E, i, j)` est en $\mathcal{O}(j-i)$, au même titre que `somme_emc(E, i, j)` ;
- `score(E, N, P, K)` est en $\mathcal{O}(N)$, car on calcule `somme_emc` pour chaque classe, dont la somme des cardinaux est N (attention à ne pas faire une analyse trop grossière ici) ;
- on en déduit que `clustering3(E)` est en $\mathcal{O}(N^3)$, car on fait le calcul d'un score de l'ordre de $\mathcal{O}(N^2)$ fois. La création et copie des tableaux n'est pas à prendre en compte (car ce sont des tableaux de taille 3).

1.2 Clustering hiérarchique ascendant

Question 10 On garde en mémoire l'indice i_{opt} et l'écart de moyenne entre $C_{i_{\text{opt}}}$ et $C_{i_{\text{opt}}+1}$. Ensuite, on parcourt tous les indices jusqu'à $K-2$ et on vérifie si l'écart est strictement inférieur pour éventuellement mettre à jour. On fait toujours attention à traiter la dernière classe à part. C'est pour cette raison qu'il y a un opérateur ternaire. On rappelle que `b?x:y` renvoie `x` si `b` vaut `true` et `y` sinon.

```
int classes_plus_proches(double* E, int N, int* P, int K){
    int iopt = 0;
    double moy1 = moyenne(E, P[1], (K==2)?N:P[2]);
    int ecartopt = moy1 - moyenne(E, P[0], P[1]);
    for (int i=1; i<K-1; i++){
        double moy2 = moyenne(E, P[i + 1], (K==i+2)?N:P[i + 2]);
        double ecart = moy2 - moy1;
        if (ecart < ecartopt){
            iopt = i;
            ecartopt = ecart;
        }
        moy1 = moy2;
    }
    return iopt;
}
```

À noter, on aurait pu partir de la fin du tableau pour éviter les tests supplémentaires.

Question 11 On crée un nouveau tableau, qu'on remplit avec les éléments d'indice différent de `iopt + 1`.

```
int* fusion_classes(int* P, int K, int iopt){
    int* nouvP = malloc((K - 1) * sizeof(int));
    for (int i=0; i<K; i++){
        if (i <= iopt){
            nouvP[i] = P[i];
        } else if (i > iopt + 1){
            nouvP[i - 1] = P[i];
        }
    }
    return nouvP;
}
```

Question 12 On se contente d'appliquer l'algorithme décrit précédemment avec les fonctions déjà écrites. On pense à libérer la mémoire du tableau `P` lorsqu'on fusionne des classes.

```
int* CHA(double* E, int N, int K){
    int* P = malloc(N * sizeof(int));
    for (int i=0; i<N; i++) P[i] = i;
    int taille = N;
    while (taille > K){
        int iopt = classes_plus_proches(E, N, P, taille);
        int* nouvP = fusion_classes(P, taille, iopt);
        free(P);
        P = nouvP;
        taille--;
    }
    return P;
}
```

Question 13

- La fonction `classe_plus_proches` calcule de l'ordre de K moyennes, pour une somme des tranches d'indices égale à N. La complexité est donc en $\mathcal{O}(N)$.
- La fonction `fusion_classes` se contente de créer et remplir un tableau de taille $K-1$, donc en $O(K)$ (avec $K \leqslant N$).
- Finalement, la fonction `CHA` fait appel aux deux fonctions précédentes, pour $k \in [\![K, N]\!]$, soit une complexité en $O((N-K) \times N)$.

Question 14 On considère $N = 4$ et $K = 2$. On pose $E = \{0, 2, 4, 7\}$. L'algorithme de CHA donnera les classes $\mathcal{P}_1 = \{\{0, 2, 4\}, \{7\}\}$. Avec une telle partition, on aurait un score $S(\mathcal{P}_1) = (4+4) + 0 = 8$. Cependant, avec la partition $\mathcal{P}_2 = \{\{0, 2\}, \{4, 7\}\}$, on obtient un score de $S(\mathcal{P}_2) = (1+1) + (2,25 + 2,25) = 6,5$.

1.3 Solution optimale en programmation dynamique

Question 15 Lorsque $k = 1$, $D(n, k) = S_{\text{emc}}(0, n)$ (il n'y a qu'une seule classe). Ce score peut être calculé en $\mathcal{O}(n)$.

Question 16 Une partition des n éléments de taille k consiste en une partition de $i < n$ éléments en $k-1$ classes, à laquelle on rajoute une classe formée des $n-i$ derniers éléments. Comme aucune classe ne doit être vide, on considère $i \geqslant k-1$. La partition optimale atteint le minimum parmi toutes les partitions possibles de cette forme, ce qui donne bien la formule voulue.

Question 17 On écrit une fonction auxiliaire `cluster_rec` qui prend en argument l'ensemble E, des entiers n et k et un dictionnaire (ici codé par une matrice) mémoïsant les résultats, et renvoie $D(n, k)$. L'initialisation et l'hérédité se font selon les deux questions précédentes.

```
double cluster_rec(double* E, int n, int k, double** dic){
    if (dic[n][k] < 0){
        if (k == 1){
            dic[n][k] = somme_emc(E, 0, n);
        } else {
            dic[n][k] = cluster_rec(E, k - 1, k - 1, dic) + somme_emc(E, k - 1, n);;
            for (int i=k-1; i<n; i++){
                double d = cluster_rec(E, i, k - 1, dic) + somme_emc(E, i, n);
                if (d < dic[n][k]){
                    dic[n][k] = d;
                }
            }
        }
    }
    return dic[n][k];
}
```

Une fois cette fonction écrite, il suffit de lancer un appel avec $n = N$ et $k = K$, en utilisant un dictionnaire vide. On pense à libérer la mémoire avant de renvoyer la valeur.

```
double clustering_dynamique(double* E, int N, int K){
    double** dic = malloc((N + 1) * sizeof(double*));
    for (int n=0; n<=N; n++){
        dic[n] = malloc((K + 1) * sizeof(double));
        for (int k=0; k<=K; k++){
            dic[n][k] = -1;
        }
    }
    double d = cluster_rec(E, N, K, dic);
    for (int n=0; n<=N; n++){
        free(dic[n]);
    }
    free(dic);
    return d;
}
```

Question 18 On remarque qu'il y a de l'ordre de $N \times K$ valeurs qui sont calculées dans le dictionnaire. De plus, chaque valeur nécessaire de calculer le minimum par la boucle `for`. Cette boucle, de taille $n - k$, fait un appel à `somme_emc`, de complexité $\mathcal{O}(n - i)$. En combinant tout ça, on obtient une complexité totale en $\mathcal{O}(K \times N^3)$.

Question 19 Dans le calcul du minimum, on peut garder en mémoire l'indice `i` qui permet d'atteindre ce minimum, ce qui correspond au plus petit élément de la classe C_{k-1} dans une partition de taille k. On peut alors reconstruire une solution complète en utilisant les valeurs présentes dans le dictionnaire.

Question 20 On remarque les formules suivantes :

- $\mu(i, i+1) = x_i$;
- $\mu(i, n+1) = \dfrac{(n-i)\mu(i,n) + x_n}{n+1-i}$.

Ces formules peuvent être utilisées pour calculer tous les $\mu(i, n)$, $i < n$, en temps $\mathcal{O}(N^2)$ au total. Dès lors, on remarque, en adaptant la formule admise que :

- $S_{\text{emc}}(i, i+1) = 0$;
- $S_{\text{emc}}(i, n+1) = S_{\text{emc}}(i, n) + \frac{n-i}{n+1-i}(x_n - \mu(i, n))^2$.

À nouveau, on peut utiliser ces formules pour calculer les $S_{\text{emc}}(i, n)$ en temps $\mathcal{O}(N^2)$ au total.

2 Logique linéaire

2.1 Présentation de la logique linéaire multiplicative

2.1.1 Ensemble des formules

2.1.2 Règles de déduction de la logique linéaire

2.1.3 Premiers arbres de preuve

Question 21 On propose :

$$\dfrac{\dfrac{\dfrac{}{\vdash x, x^\perp}\,\text{ax} \quad \dfrac{}{\vdash y, y^\perp}\,\text{ax}}{\vdash x \otimes y^\perp, x^\perp, y}\,\otimes}{\vdash (x \otimes y^\perp) \parr x^\perp, y}\,\parr}{\vdash ((x \otimes y^\perp) \parr x^\perp) \parr y}\,\parr$$

Question 22 On a les deux preuves suivantes. On utilise une notation similaire à une règle d'inférence, mais sans nom, lorsqu'on se contente de réécrire la formule par une formule qui lui est égale syntaxiquement.

– $\vdash ((x \parr y) \parr z) \multimap (x \parr (y \parr z))$ prouvable par :

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\vdash x, x^\perp}\,\text{ax} \quad \dfrac{}{\vdash y, y^\perp}\,\text{ax}}{\vdash x^\perp \otimes y^\perp, x, y}\,\otimes \quad \dfrac{}{\vdash z, z^\perp}\,\text{ax}}{\vdash (x^\perp \otimes y^\perp) \otimes z^\perp, x, y, z}\,\otimes}{\vdash ((x^\perp \otimes y^\perp) \otimes z^\perp) \parr (x \parr (y \parr z))}\,\parr \times 3}{\vdash ((x \parr y) \parr z)^\perp \parr (x \parr (y \parr z))}}{\vdash ((x \parr y) \parr z) \multimap (x \parr (y \parr z))}$$

– $\vdash (x \parr (y \parr z)) \multimap ((x \parr y) \parr z)$ prouvable par :

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\vdash y, y^\perp}\,\text{ax} \quad \dfrac{}{\vdash z, z^\perp}\,\text{ax}}{\vdash y^\perp \otimes z^\perp, y, z}\,\otimes \quad \dfrac{}{\vdash x, x^\perp}\,\text{ax}}{\vdash x^\perp \otimes (y^\perp \otimes z^\perp), x, y, z}\,\otimes}{\vdash (x^\perp \otimes (y^\perp \otimes z^\perp)) \parr ((x \parr y) \parr z)}\,\parr \times 3}{\vdash (x \parr (y \parr z))^\perp \parr ((x \parr y) \parr z)}}{\vdash (x \parr (y \parr z)) \multimap ((x \parr y) \parr z)}$$

Question 23 On raisonne par induction sur A :

– si $A = x \in \mathcal{V}$, alors $\dfrac{}{\vdash x, x^\perp}\,\text{ax}$ est une preuve de $\vdash A, A^\perp$.

– supposons que $\vdash B, B^\perp$ et $\vdash C, C^\perp$ soient prouvables. Distinguons selon la formule A :

  * si $A = B^\perp$, alors $\dfrac{\dfrac{\vdash B^\perp, B}{\vdash B^\perp, (B^\perp)^\perp}}{\vdash A, A^\perp}$ est une preuve de $\vdash A, A^\perp$;

$*$ si $A = B \parr C$, alors $\dfrac{\dfrac{\vdash B, B^\perp \quad \vdash C, C^\perp}{\vdash B \parr C, B^\perp \otimes C^\perp}\otimes}{\dfrac{\vdash B \parr C, (B \parr C)^\perp}{\vdash A, A^\perp}}$ est une preuve de $\vdash A, A^\perp$;

$*$ si $A = B \otimes C$, alors $\dfrac{\dfrac{\vdash B, B^\perp \quad \vdash C, C^\perp}{\vdash B \otimes C, B^\perp \parr C^\perp}\otimes}{\dfrac{\vdash B \otimes C, (B \otimes C)^\perp}{\vdash A, A^\perp}}$ est une preuve de $\vdash A, A^\perp$.

On conclut par induction.

Question 24 On a l'arbre de preuve suivant :

$$\dfrac{\vdash \Gamma, A \multimap B \qquad \dfrac{\vdash \Delta, A \qquad \dfrac{}{\vdash B, B^\perp}\,\text{ax}}{\dfrac{\vdash \Delta, B, A \otimes B^\perp}{\vdash \Delta, B, (A \multimap B)^\perp}}\otimes}{\vdash \Gamma, \Delta, B}\,\text{cut}$$

Question 25 On a l'égalité $|A^\perp|_x = |A|_{x^\perp}$, qui se montre par induction sur A. C'est vrai pour les variables, la négation linéaire transforme les x en $x^\perp$ et réciproquement, et les constructeurs $\parr$ et $\otimes$ se contentent de faire une somme.

Montrons un résultat un peu plus fort que celui demandé par l'énoncé, à savoir que si $\vdash \Gamma$ est prouvable, alors $|\Gamma|_x = |\Gamma|_{x^\perp}$, où $|\Gamma|_x = \sum_{A \in \Gamma} |A|_x$ (de même pour $|\Gamma|_{x^\perp}$). Montrons ce résultat par induction sur l'arbre de preuve de $\vdash \Gamma$. Pour ce faire, montrons que pour toute règle, si les prémisses vérifient l'égalité, alors la conclusion aussi.

- La règle (ax) vérifie l'égalité trivialement.
- Supposons que $|\Gamma, A|_x = |\Gamma, A|_{x^\perp}$ et $|\Delta, A^\perp|_x = |\Delta, A^\perp|_{x^\perp}$. Alors :

  $$|\Gamma, \Delta|_x = |\Gamma, A|_x - |A|_a + |\Delta, A^\perp|_x - |A^\perp|_x = |\Gamma, A|_{x^\perp} - |A^\perp|_{x^\perp} + |\Delta, A^\perp|_{x^\perp} - |A|_{x^\perp} = |\Gamma, \Delta|_{x^\perp}$$

  On a ici utilisé le résultat prouvé précédemment. La règle (cut) vérifie donc la propriété voulue.
- Supposons que $|\Gamma, A|_x = |\Gamma, A|_{x^\perp}$ et $|\Delta, B|_x = |\Delta, B|_{x^\perp}$. Alors :

  $$|\Gamma, A \otimes B, \Delta|_x = |\Gamma, A|_x + |\Delta, B|_x = |\Gamma, A|_{x^\perp} + |\Delta, B|_{x^\perp} = |\Gamma, A \otimes B, \Delta|_{x^\perp}$$

  La règle ($\otimes$) vérifie donc la propriété.
- Supposons que $|\Gamma, A, B|_x = |\Gamma, A, B|_{x^\perp}$. Alors :

  $$|\Gamma, A \parr B|_x = |\Gamma, A, B|_x = |\Gamma, A, B|_{x^\perp} = |\Gamma, A \parr B|_{x^\perp}$$

  La règle ($\parr$) vérifie donc la propriété.

On conclut par induction et en posant $\Gamma = \{A\}$.

Question 26 Si on pose $A = (x \otimes x) \multimap x$, alors $A = (x \otimes x)^\perp \parr x = x^\perp \parr x^\perp \parr x$. On a donc $|A|_x = 1 \neq 2 = |A|_{x^\perp}$. Par contraposée de la question précédente, on en déduit que A n'est pas prouvable.

2.2 Réseaux de preuve

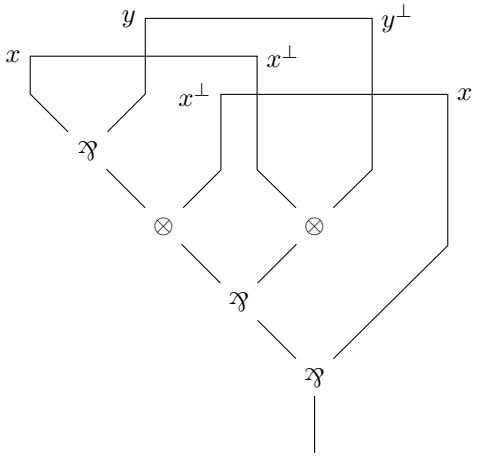
2.2.1 Structures de preuve

Question 27 On remarque que les seuls sommets qui ont à la fois des arêtes entrantes et sortantes sont les sommets par et tens. Pour chacun de ces sommets, on remarque que si A et B sont les étiquettes des arêtes entrantes et C l'étiquette de l'arête sortante, alors $|C| > |A|$ et $|C| > |B|$ (où $|A|$ est la taille de la formule, c'est-à-dire le nombre de symboles).

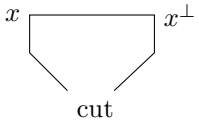
S'il existe un cycle orienté, ce cycle est de la forme $s_0 \xrightarrow{A_1} s_1 \xrightarrow{A_2} \dots \xrightarrow{A_n} s_n$, avec $s_n = s_0$. Par ce qui a été dit plus haut, on a, pour $i \in [\![1, n-1]\!]$, $|A_i| < |A_{i+1}|$. De même, $|A_n| < |A_1|$. Mais alors, par transitivité, $|A_1| < |A_1|$, ce qui est absurde.

2.2.2 Réseaux de preuve

Question 28 On obtient le réseau de preuve suivant. Attention à ne pas inverser les deux arêtes sortantes a (si on les inverse, l'arbre de preuve associé est incorrect).



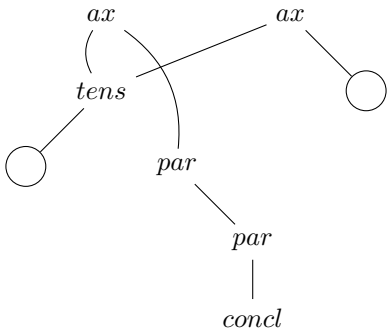
Question 29 La structure de preuve suivante n'est pas un réseau de preuve :



En effet, en accord avec la construction des réseaux de preuve, cela correspondrait à un arbre de preuve de la forme $\dfrac{\vdash x \quad \vdash x^\perp}{\vdash}$ cut qui est évidemment incorrect ($\vdash x$ et $\vdash x^\perp$ ne sont pas prouvables d'après la question 25).

2.2.3 Critère de Danos et Régnier

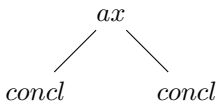
Question 30 Le graphe suivant est un interrupteur :



Question 31 Un nœud *par* ayant deux arêtes entrantes, on en déduit directement que le nombre d'interrupteurs est 2^p.

Question 32 Par induction sur les arbres de preuve :

- pour un axiome $\dfrac{}{\vdash x, x^\perp}\ \text{ax}$, tous les interrupteurs sont :



  C'est effectivement un graphe connexe sans cycle.
- pour une règle tens $\dfrac{\vdash \Gamma, A \qquad \vdash \Delta, B}{\vdash \Gamma, A \otimes B, \Delta}\ \otimes$, supposons que les réseaux $\mathcal{S}$ et $\mathcal{S}'$ de conclusions Γ, A et Δ, B ont des interrupteurs connexes sans cycle. Alors un interrupteur associé au réseau de conclusions $\Gamma, A \otimes B, \Delta$ s'obtient en supprimant les nœuds *concl* de A et B et en rajoutant un nœud *concl* adjacent à un nœud *tens* adjacent à un nœud d'un interrupteur de $\mathcal{S}$ et à un nœud d'un interrupteur de $\mathcal{S}'$. Cela forme bien un graphe connexe sans cycle.
- pour une règle par $\dfrac{\vdash \Gamma, A, B}{\vdash \Gamma, A \parr B}\ \parr$, supposons que le réseau $\mathcal{S}$ de conclusions Γ, A, B ait des interrupteurs connexes sans cycle. Alors un interrupteur associé au réseau de conclusions $\Gamma, A \parr B$ s'obtient en remplaçant un nœud *concl* de A ou B par un nœud factice et en remplaçant l'autre par un nœud *par* adjacent à un nouveau nœud *concl*. Cela donne bien un graphe connexe sans cycle.

On conclut par induction.

2.2.3.1 Constitution d'un empire

Question 33 On remarque que les arêtes conclusions d'un nœud axiome ne sont jamais déconnectées de ce nœud axiome. Elles sont donc soit toutes les deux dans un empire, soit aucune des deux, ce qui donne l'équivalence.

Question 34

- Supposons $b \in \text{emp}(a)$ et $c \in \text{emp}(a)$. Comme dans tout a-interrupteur, d est reliée soit à b, soit à c (car a n'est pas une prémisse du nœud), on en déduit que $d \in \text{emp}(a)$.
- supposons $d \in \text{emp}(a)$. Soit I un interrupteur de $\mathcal{S}$. Par connexité, il existe un chemin contenant b et c. Par acyclicité, ce chemin ne contient pas l'arête d. Par ailleurs, ce chemin ne contient pas l'arête a. En effet, si c'était le cas, il existerait dans cet interrupteur une suite d'arêtes incidentes de la forme (sans perte de généralité) $(d, b, \ldots, a, \ldots, c)$. Quitte à considérer l'interrupteur où on inverse les rôles de b et c, supposons que dans l'a-interrupteur associé, la déconnexion de a crée les deux chemins $(d, b, \ldots)$ et $(a, \ldots, c)$. Alors $d \notin \text{emp}(a)$ (sinon il existerait un cycle dans I, car il existerait deux chemins distincts de d à a). On conclut par l'absurde.

Question 35 Il existe un interrupteur I de $\mathcal{S}$ tel que a et b sont incidentes. Si $b \in \text{emp}(a)$, alors dans l'a-interrupteur associé à I, a et b ne sont pas incidentes. Si $b \in \text{emp}(a)$, alors il existe un autre chemin reliant b à a, donc il existe un cycle dans I. Par l'absurde, $b \notin \text{emp}(a)$.

Question 36 Si $c \in \text{emp}(a)$, alors par la question précédente, a n'est pas prémisse de ce nœud. On distingue :

- pour un nœud *par*, par la question 34, $b \in \text{emp}(a)$;
- pour un nœud *tens*, comme a n'est pas prémisse du nœud, b n'est pas déconnectée du nœud dans un a-interrupteur, donc b et c sont toujours incidentes, donc $b \in \text{emp}(a)$.

Question 37 Supposons que a n'est pas prémisse de s. Alors $a \neq b$. Si s est un nœud *tens*, alors dans tout a-interrupteur, b et c sont incidentes, donc sont soit toutes les deux dans l'empire, soit aucune des deux, ce qui est absurde. On en déduit que s est un nœud *par*. Dès lors, si l'autre arête est dans $\text{emp}(a)$, par la question 34, $c \in \text{emp}(a)$, ce qui est absurde. On en déduit le résultat voulu.

Question 38

- si $a \notin \text{emp}(b)$, alors soit $c \in \text{emp}(a) \cap \text{emp}(b)$. Dans tous les a-interrupteurs de $\mathcal{S}$, il existe un chemin de a à c. Si un tel chemin passe par b, alors $b \in \text{emp}(a)$. De même, dans tous les b-interrupteurs, il existe un chemin de b à c qui ne passe pas par a. Pourtant, cela implique que dans tous les a-interrupteurs, il existe un chemin de a à b (en concaténant les chemins). C'est absurde. On en déduit que $\text{emp}(a) \cap \text{emp}(b) = \emptyset$;
- sinon, $a \in \text{emp}(b)$, commençons par montrer qu'il existe un b-interrupteur I_b tel que toutes les arêtes connectées à b sont exactement celles de $\text{emp}(b)$. On le construit de la manière suivante :
  - si b est prémisse d'un nœud *par*, on déconnecte l'autre arête prémisse de ce nœud *par* ;
  - si a est prémisse d'un nœud *par* dont la conclusion est dans $\text{emp}(b)$ (ce qui ne peut pas arriver dans le cas précédent d'après la question 36), on déconnecte l'autre arête prémisse de ce nœud *par* ;
  - pour tout autre nœud *par* dont la conclusion n'est pas dans $\text{emp}(b)$, par la question 34, l'une des deux prémisses n'est pas dans $\text{emp}(b)$. On déconnecte l'autre prémisse du nœud *par* (qui peut être ou non dans $\text{emp}(b)$) ;
  - pour tout autre nœud *par* dont la conclusion est dans $\text{emp}(b)$ mais pas dans $\text{emp}(a)$, l'une des deux prémisses n'est pas dans $\text{emp}(a)$. On déconnecte l'autre prémisse du nœud *par* ;
  - pour tout autre nœud *par* dont la conclusion est dans $\text{emp}(a) \cap \text{emp}[b)$, on choisit arbitrairement.

  Les arêtes de I_b sont exactement celles de $\text{emp}(b)$. En effet, une arête qui relie $\text{emp}(b)$ à son complémentaire dans un b-interrupteur est soit l'arête b elle-même, soit une prémisse d'un nœud *par*. Par construction, ce n'est pas le cas pour I_b.

  Notons I l'interrupteur associé à I_b et I_a le a-interrupteur associé à I. Par construction, les seules arêtes de $\text{emp}(b)$ dans I qui peuvent relier $\text{emp}(a)$ à son complémentaire sont a et b, mais $b \notin \text{emp}(a)$.

  Dès lors, soit $c \in \text{emp}(a)$. Dans I_a, il existe un chemin reliant a à c. Si ce chemin passe par b, alors soit ce chemin relie c à b par le nœud dont b est conclusion, auquel cas $c \in \text{emp}(b)$, soit ce chemin relie c à b par le nœud dont b est prémisse, auquel cas $b \in \text{emp}(a)$, ce qui est absurde par hypothèse. Si le chemin de a à c ne passe pas par b, alors il existe dans I_b, donc $c \in \text{emp}(b)$.

  Dans tous les cas, on a bien $\text{emp}(a) \subseteq \text{emp}(b)$.

Question 39 On vérifie les différents points :

- c'est bien une structure de preuve. En effet, s'il possède une arête conclusion d'un nœud, alors il possède l'autre conclusion (nœud *ax*) ou les deux prémisses (nœuds *tens* et *parr*) ;
- a est une conclusion. En effet, a est prémisse d'un nœud dont la conclusion n'est pas dans l'empire. C'est donc une conclusion dans ce sous-graphe ;
- il vérifie le critère de Danos et Régnier : $\text{emp}(a)$ est un sous-graphe connexe de tout interrupteur de $\mathcal{S}$. En particulier, tout interrupteur de $\text{emp}(a)$ est connexe. De plus, en tant que sous-graphe d'un graphe sans cycle, il est sans cycle ;
- il est maximal pour l'inclusion : si b est une arête dans une structure de preuve vérifiant les conditions, alors dans un interrupteur I, il existe un chemin de b à a. Ce chemin existe aussi dans le a-interrupteur as-

socié, car a est une conclusion (donc ne se déconnecte que de son nœud conclusion dans le a-interrupteur). On en déduit que $b \in \text{emp}(a)$.

Question 40 La question était incomplète pour pouvoir être utilisée correctement à la question suivante. On s'attendait ici à prouver qu'en plus d'être disjoints, les deux empires formaient une partition de la structure $\mathcal{S}$ privée du nœud *tens* et de sa conclusion. On fait la preuve en prenant cela en compte.

On choisit l'un de ces nœuds *tens*, de prémisse a, tel que $\text{emp}(a)$ est maximal pour l'inclusion.

Alors les conclusions de $\text{emp}(a)$ (autres que a) sont des conclusions de $\mathcal{S}$. En effet, s'il existe une arête $b' \in \text{emp}(a) \setminus \{a\}$ qui est conclusion de $\text{emp}(a)$ mais qui n'est pas une conclusion de $\mathcal{S}$, alors cette arête est prémisse d'un nœud s de conclusion c qui n'est pas dans $\text{emp}(a)$. Par le lemme de quasi clôture par le bas, s est un nœud *par*. Par hypothèse, la conclusion c de ce nœud *par* n'est pas une conclusion de $\mathcal{S}$, donc mène à la prémisse c' d'un nœud *tens* terminal. Par le lemme de clôture par le haut, cette arête c' n'est pas dans $\text{emp}(a)$. Mais sachant que $b' \in \text{emp}(a)$ et $b' \in \text{emp}(c')$, et que $c' \notin \text{emp}(a)$, par le lemme d'emboîtement d'empires, $\text{emp}(a) \subsetneq \text{emp}(c')$, ce qui contredit la maximalité.

On en déduit qu'en supprimant ce nœud *tens*, on obtient deux structures de preuves : $\text{emp}(a)$, qui est une structure de preuve vérifiant le critère de Danos et Régnier, par la question précédente, et le reste, qui vérifie les mêmes conditions, car on n'a pas supprimé d'arête prémisse.

Question 41 On montre ce résultat par récurrence sur le nombre de nœuds de la structure de preuve en considérant les cas suivants :

- s'il existe un nœud *par* terminal, on peut supprimer ce nœud et recommencer sur la structure obtenue (qui vérifie bien les conditions car tout interrupteur reste connexe et acyclique) ;
- s'il n'y a pas de nœud *par* terminal mais au moins un nœud *tens* terminal, par le lemme du tenseur scindent, on peut supprimer ce nœud et recommencer sur les deux structures obtenues ;
- sinon, les seuls nœuds terminaux sont des nœuds ax, mais alors la structure n'a qu'un seul nœud a par connexité, donc est bien un réseau de preuve.

2.2.4 Un théorème d'élimination des coupures

Question 42 On reprend la preuve du sens direct du théorème et on traite les nœuds *cut* comme les nœuds *tens*.

Question 43 On sait qu'un nœud *cut*, dans le cas général, a une prémisse A et une prémisse $A^\perp$. Ainsi, si l'un des deux côté est un nœud *tens*, de conclusion $A \otimes B$, alors l'autre côté est nécessairement de conclusion $(A \otimes B)^\perp = A^\perp \parr B^\perp$, qui est bien conclusion d'un nœud *par*.

Question 44 Dans un premier temps, remarquons que l'élimination d'une coupure reliée à un nœud ax ne brise ni la connexité, ni l'acyclicité. On considère une élimination d'une coupure *tens*/*par* dans une structure $\mathcal{S}$ qui vérifie le critère de Danos et Régnier. On suppose que $a(A)$ et $b(B)$ sont les arêtes prémisses du nœud *tens*, et que $c(A^\perp)$ et $d(B^\perp)$ sont les arêtes prémisses du nœud *par*.

Soit I un interrupteur de la structure $\mathcal{S}'$ obtenue après élimination de la coupure.

- I est acyclique : s'il existe un cycle dans I, s'il passe uniquement par la coupure $a - c$ (resp. $b - d$), alors dans l'interrupteur dans $\mathcal{S}$ où on déconnecte d (resp. c) du nœud *par*, ce cycle existe dans $\mathcal{S}$, ce qui est absurde. Sinon, c'est un cycle de toute façon dans tout interrupteur de $\mathcal{S}$;
- I est connexe : soit J l'interrupteur de $\mathcal{S}$ obtenu à partir de I où on déconnecte c du nœud *par*. Par connexité, toute arête est reliée à a, b, c ou d. Dans I, il y a donc au plus deux composantes connexes : celle reliée à $a - c$ et celle à $b - d$. Cependant, s'il existe un autre chemin de c à a ou b que l'arête $a - c$, il existerait un cycle. c est donc reliée à b via l'arête $c - d$, donc le graphe est connexe.

On conclut par récurrence.

Question 45 L'algorithme termine car le nombre de nœuds de la structure est un variant : il diminue strictement (de 1 ou 2) à chaque élimination de coupure. Par construction, une coupure est toujours reliée soit

à un nœud ax, soit à un nœud $tens$ (resp. par) et donc un nœud par (resp. $tens$), soit à un nœud cut. Dans ce dernier cas, on peut toujours appliquer l'une des deux règles au nœud cut qui apparaît en premier dans un ordre topologique de la structure. Lorsque l'algorithme termine, il n'y a donc plus de coupure.

Question 46 On remarque que s'il est appliqué à une structure de preuve, l'algorithme d'élimination des coupures ne change pas les arêtes conclusions de la structure. Ainsi, s'il existe une preuve d'un séquent $\vdash \Gamma$, on considère le réseau de preuve associé. Par la question 42, il vérifie le critère de Danos et Régnier. En appliquant l'algorithem d'élimination des coupures, on obtient une structure de preuve vérifiant le critère de Danos et Régnier, ne contenant pas de nœud cut, et ayant les mêmes conclusions. Par le théorème de Danos et Régnier, c'est un réseau de preuve. La preuve correspondante est bien une preuve du séquent $\vdash \Gamma$ n'utilisant pas la coupure, qui est donc admissible.

$\star\star\star$