Inférence de Types

Soit $\mathcal{A} = \{A, B, C, \dots\}$ un ensemble infini de *variables de types*.
On considère $\mathcal{T}$ l'ensemble des *types* définis inductivement par :

- $\mathbb{N} \in \mathcal{T}$. ($\mathbb{N}$ est un type)
- $\mathcal{A} \subseteq \mathcal{T}$. (une variable de types est un type)
- si $T_1 \in \mathcal{T}$ et $T_2 \in \mathcal{T}$ alors $T_1 \to T_2 \in \mathcal{T}$. (la "flèche" de deux types est un type)

Une *substitution de types* $\sigma : \mathcal{A} \longrightarrow \mathcal{T}$ est une fonction qui vaut l'identité presque partout, c'est-à-dire telle que $\{A \in \mathcal{A} \mid \sigma(A) \neq A\}$ est fini.
Si σ est une substitution de types et $T \in \mathcal{T}$ un type, on note $\sigma_\uparrow(T)$ (par abus de notation, on s'autorisera à écrire $\sigma(T)$) le type obtenu inductivement par :

- $\sigma_\uparrow(\mathbb{N}) = \mathbb{N}$.
- pour $\mathcal{A}$, $\sigma_\uparrow(A) = \sigma(A)$.
- pour tout $T_1, T_2 \in \mathcal{T}^2$, $\sigma_\uparrow(T_1 \to T_2) = \sigma_\uparrow(T_1) \to \sigma_\uparrow(T_2)$

Un *système d'équations de types* (ou juste *système*) est un ensemble fini $\{(T_k, T'_k)\}_{1 \leq k \leq n}$ de couples de types.
Un *unificateur* d'un système $\{(T_k, T'_k)\}_{1 \leq k \leq n}$ est une substitution de types σ telle que $\forall k, \sigma_\uparrow(T_k) = \sigma_\uparrow(T'_k)$.

Par exemple, on pose $S_0 = \{(B \to A, C),\ (\mathbb{N}, B)\}$

et σ_0 définie par $\begin{cases} \sigma_0(B) & = & \mathbb{N} \\ \sigma_0(C) & = & \mathbb{N} \to A \\ \sigma_0(X) & = & X \text{ pour tout} X \notin \{B, C\} \end{cases}$

Alors σ_0 est un unificateur de S_0 car :

1. $\sigma_0(B \to A) = \mathbb{N} \to A$ et $\sigma_0(C) = \mathbb{N} \to A$
2. $\sigma_0(\mathbb{N}) = \mathbb{N}$ et $\sigma_0(B) = \mathbb{N}$

Question 1. Trouver si possible un unificateur des systèmes suivants :

1. $\{(\mathbb{N} \to A, B \to \mathbb{N})\}$
2. $\{(\mathbb{N} \to A, B \to (C \to \mathbb{N})),\ (\mathbb{N} \to \mathbb{N}, C)\}$
3. $\{(A \to B), (B \to A)\}$
4. $\{(\mathbb{N} \to (A \to \mathbb{N}), \mathbb{N} \to B),\ (B, A)\}$

Question 2. Un système S admet-il toujours un unificateur ?

Quand un unificateur pour S est-il unique ?

Question 3. En considérant les différentes possibilités pour les formes des types T_1 et T'_1, exprimer l'existence d'un unificateur pour $\{(T_k, T'_k)\}_{1 \leq k \leq n}$ en fonction de l'existence d'un unificateur pour un autre système, bien choisi.

Question 4. Rédiger l'algorithme induit par la question précédente, étudier sa terminaison.

Soit $\mathcal{X} = \{x, y, z, \dots\}$ un ensemble de *variables de termes*.
On considère un langage d'expressions fonctionnelles $\mathcal{E}$ défini inductivement par :

- $\mathbb{N} \subseteq \mathcal{E}$, (les entiers sont des expressions)
- $\mathcal{X} \subseteq \mathcal{E}$ (les variables de termes sont des expressions)
- Si $(E_1, E_2) \in \mathcal{E}^2$, $E_1 + E_2 \in \mathcal{E}$ (la somme de deux expressions est une expression)
- Si $(E_1, E_2) \in \mathcal{E}^2$, $E_1(E_2) \in \mathcal{E}$ (l'application d'une expression à une expression est une expression)
- Si $x \in \mathcal{X}$ et $E \in \mathcal{E}$ alors $(x \mapsto E) \in \mathcal{E}$ (si E est une expression et x une variable, l'expression fonctionnelle $x \mapsto E$ est une expression)

On supposera que les variables x apparaissant dans des sous-expressions $x \mapsto E'$ d'une expression E sont toutes distinctes et distinctes deux à deux des variables de E qui n'apparaissent jamais directement à gauche d'un $\mapsto$. On note $FV(E)$ ce dernier ensemble de variables.

Par exemple, $f_0 = (x \mapsto (y \mapsto x(y) + 1))$ est un élément de $\mathcal{E}$.

Les contextes de types sont des fonctions de $D \subseteq \mathcal{X}$ dans T.
On note $\emptyset$ le contexte de domaine vide et $(\Gamma, x : T)$, le contexte Γ' défini par $\Gamma'(x) = T$ et pour tout y dans le domaine de Γ, $y \neq x \Rightarrow \Gamma'(y) = \Gamma(y)$.

Dans la suite on s'intéresse au problème de donner un type de $\mathcal{T}$ à une expression de $E \in \mathcal{E}$ avec un contexte Γ qui sert d'oracle donnant le type des variables de $FV(E)$, quand c'est possible. On dit qu'on *infère un type de E dans le contexte Γ*.

Question 5. Donner deux types différents qu'il semble légitime de donner à f_0 dans le contexte $\emptyset$.

Question 6. Donner des règles qui formalisent quand il est légitime de donner un type T à E dans le contexte Γ.

Question 7. En déduire un algorithme qui infère un type d'une expression E.

Question 8. Utiliser cet algorithme pour trouver un type de $\mathbf{S} = x \mapsto (\ y \mapsto (\ z \mapsto x(z)(y(z))\ )\ )$ dans le contexte $\emptyset$.

Zones

Soit $\mathcal{V} = \{V_1, \ldots, V_n\}$ un ensemble de variables, et $\mathbb{I} \in \{\mathbb{Z}, \mathbb{Q}, \mathbb{R}\}$.
Une contrainte de potentiel est une inégalité de la forme $V_i - V_j \leq c$, avec $c \in \mathbb{I}$.
Une conjonction de contraintes de potentiel sur $\mathcal{V}$ est représentée par une matrice M de taille $n \times n$ telle que :

- $M_{ij} = c$ si la contrainte $V_j - V_i \leq c$ est présente dans la conjonction
- $M_{ij} = +\infty$ sinon

Dans la suite, on appelle ces matrices des matrices de potentiel.
Les solutions des contraintes de potentiel définies par une matrice M sont :

$$\textsc{sols}(M) = \{(v_1, \ldots, v_n) \in \mathbb{I}^n | v_j - v_i \leq M_{i,j}\}$$

Question 1. Soit M une matrice de potentiel de taille n, et $(v_1, \ldots, v_n) \in \textsc{sols}(M)$ une solution à une conjonction de contraintes de potentiel. Montrer que $\forall c \in \mathbb{I}$, $(v_1 + c, \ldots, v_n + c) \in \textsc{sols}(M)$.

Solution : Les $+c$ vont s'annuler dans toutes les inégalités.

Question 2. Expliquer comment encoder des contraintes de la forme $V_i \leq c$ et $V_i \geq c$ en étendant $\mathcal{V}$.

Solution : Ajouter une variable $V_0 \in \mathcal{V}$. On a ensuite $V_i \leq c \rightsquigarrow V_i - V_0 \leq c$, et $V_i \geq c \Leftrightarrow -V_i \leq -c \rightsquigarrow V_0 - V_i \leq -c$.
Il faut ensuite définir une variante des solutions sur ces contraintes étendues.

Question 3. On suppose que $M_{i_1,i_2} = c_1, \ldots, M_{i_{n-1},i_n} = c_n$. Peut-on en déduire une nouvelle contrainte ?

Solution : On a donc $V_{i_2} - V_{i_1} \leq c_1, \ldots, V_{i_n} - V_{i_{n-1}} \leq c_n$ On obtient $V_{i_n} - V_{i_1} \leq c_1 + \ldots + c_n$.

Question 4. On considère les contraintes suivantes :

$$1 \leq V_1; V_1 \leq 5; 1 \leq V_2; V_2 \leq 3; V_1 - V_2 \leq 1$$

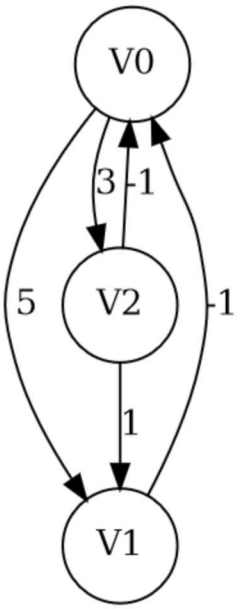
1. Donner la représentation matricielle de cet ensemble de contraintes.
2. Montrer qu'il est possible de déduire une contrainte sur V_1 qui est plus précise que les contraintes initiales.

Solution : On encode en :

$$\begin{aligned} V_0 - V_1 &\leq -1 \\ V_1 - V_0 &\leq 5 \\ V_0 - V_2 &\leq -1 \\ V_2 - V_0 &\leq 3 \\ V_1 - V_2 &\leq 1 \end{aligned}$$

La matrice est donc : $\begin{pmatrix} +\infty & 5 & 3 \\ -1 & +\infty & +\infty \\ -1 & 1 & +\infty \end{pmatrix}$

Pour la question suivante, il est important de noter que la matrice représente un graphe orienté pondéré



Oui, on peut sommer $V_2 - V_0 \leq 3$ et $V_1 - V_2 \leq 1$ pour obtenir $V_1 - V_0 \leq 4$. Graphiquement, cela correspond à trouver le chemin de plus faible poids entre V_0 et V_1

> Une matrice de potentiel est dite close s'il n'est pas possible de déduire de contraintes plus précises. La clôture d'une matrice M, est la matrice M^* telle que $\text{SOLS}(M) = \text{SOLS}(M^*)$ et M^* est close. On admet que lorsque le système de contraintes est satisfiable, M^* existe et est unique.

Question 5. On suppose que le système de contraintes est satisfiable. Donner un algorithme calculant M^*. Quelle est sa complexité ?

Solution : Il faut donc utiliser l'algorithme de Floyd Warshall (au programme de MPI), dans une version en place ici. Il n'y a pas de problèmes de cycles de poids négatif grâce à l'hypothèse de cette question.

```
1: fonction FW(M  matrice de taille n × n)
2:     pour k = 1 à n faire
3:         pour i = 1 à n faire
4:             pour j = 1 à n faire
5:                 M_{i,j} = min(M_{i,j}, M_{i,k} + M_{k,j})
6:             fin pour
7:         fin pour
8:     fin pour
9:     renvoyer M
10: fin fonction
```

On peut parler de complexité temporelle (cubique) et spatiale (ici, on a calculé en place la transformation).

Question 6. Montrer que le système de contraintes est insatisfiable si et seulement si $\exists i, M^*_{i,i} < 0$.

Solution : On montre que le système de contraintes est insatisfiable ssi le graphe possède un cycle de poids strictement négatif, par double implication. Étant donné une matrice de potentiel M, on note

$$\textsc{sols}(M) = \{(v_0, \ldots, v_n) \in \mathcal{I}^{n+1} | v_j - v_i \leq M_{i,j}\}$$

- $\Leftarrow$ Supposons que G possède un cycle de poids strictement négatif. On note $v_{i_1}, \ldots, v_{i_l}$ les arêtes de ce cycle. Les poids correspondent à $M_{i_2,i_1}, ..., M_{i_l,i_1}$, et leur somme, notée c, est strictement négative. Par l'absurde, supposons qu'il existe $(v_0, \ldots, v_n) \in \textsc{sols}(M)$. Par définition, $v_{i_2} - v_{i_1} \leq M_{i_2,i_1}$, ..., $v_{i_1} - v_{i_l} \leq M_{i_l,i_1}$. En sommant, on obtient $0 \leq c$, mais $c < 0$, contradiction.
- $\Rightarrow$ Par contrapposée, supposons qu'il n'existe pas de cycle de poids strictement négatif. On note $G = (V, E)$ le graphe initial, $G' = (V', E')$, avec $V' = V \sqcup \{x\}, E' = E \cup \{(x, v, 0) | v \in E\}$. On note s_i le plus court chemin de x à v_i, qui est bien défini car G n'a pas de cycles de poids négatif. Par définition, $s_j \leq s_i + M_{j,i}$. Donc $s_j - s_i \leq M_{j,i}$. Ainsi le point $(s_0, \ldots, s_n)$ est une solution des contraintes de potentiel.

Question 7. Comment adapter l'algorithme de la question 5 à une implémentation en langage C sur des entiers ?

Solution : Il faut réfléchir à deux choses :

- La représentation de $+\infty$ comme poids des arêtes non connectées (enum ou `MAX_INT`).
- Les cycles de poids négatif peuvent créer des entiers négatifs qui grandissent très vite. Il vaut mieux changer l'algorithme pour que si $i = j$ et $M_{i,i} < 0$ alors l'algorithme s'arrête, vu qu'il n'y a de toute façon pas unicité de la clôture dans ce cas.

  Il est possible d'exhiber des matrices d'adjacence dont les coefficients de la matrice d'adjacence lors des différentes itérations de l'algorithme croissent très rapidement. On peut par exemple prendre la matrice $M_{i,j} = -1$.

  On considère l'algorithme de Floyd Warshall qui n'est pas en place, définit sous la forme de récurrence suivante :

$$M^{(0)} = M$$
$$M^{(n+1)}_{i,j} = \min(M^{(n)}_{i,j}, M^{(n)}_{i,n} + M^{(n)}_{n,j})$$

  On montre par récurrence que $M^{(k)}_{i,j} = -2^k$.

Question 8. Comment transformer les contraintes de la forme $\varepsilon_i V_i + \varepsilon_j V_j \leq c$, avec $\varepsilon_i, \varepsilon_j \in \{-1, 1\}$ en contraintes de potentiel ?

Solution : Cette fois-ci, on duplique les variables, (V_i^+ représente V_i et V_i^- représente $-V_i$)

- $V_i - V_j \leq c \rightsquigarrow V_i^+ - V_j^+ \leq c \wedge V_j^- - V_i^- \leq c$
- $V_i + V_j \leq c \rightsquigarrow V_i^+ - V_j^- \leq c \wedge V_j^+ - V_i^- \leq c$
- $-V_i - V_j \leq c \rightsquigarrow V_i^- - V_j^+ \leq c \wedge V_j^- - V_i^+ \leq c$
- $V_i \leq c \rightsquigarrow V_i^+ - V_i^- \leq 2c$
- $V_i \geq c \rightsquigarrow V_i^- - V_i^+ \leq -2c$

$\textsc{SolsOct}(M) = \{(v_1, \ldots, v_n) \in \mathbb{I}^n | (v_1, -v_1, \ldots, v_n, -v_n) \in \textsc{Sols}(M)\}$

Elimination des coupures dans MLL et MALL

La logique linéaire additive-multiplicative $MALL$ a pour syntaxe :
$A, B ::= p \mid p^\perp \mid A \otimes B \mid A \parr B \mid A \oplus B \mid A \& B$
avec p appartenant à un ensemble infini de *formules atomiques*.
L'opération de *dualité* $\cdot^\perp$ est définie inductivement sur les formules :
$(A \otimes B)^\perp = A^\perp \parr B^\perp \qquad (A \oplus B)^\perp = A^\perp \& B^\perp$
$(A \parr B)^\perp = A^\perp \otimes B^\perp \qquad (A \& B)^\perp = A^\perp \oplus B^\perp \qquad (p^\perp)^\perp = p$
On note Γ, Δ des multi-ensembles $A_1, \ldots, A_n$ de formules (c'est-à-dire que les A_i ne sont pas ordonnées, mais on tient compte de leur multiplicité). Un *séquent linéaire* (ou seulement *séquent*, dans ce sujet) $\vdash \Gamma$ est prouvable quand on peut le déduire des règles suivantes (c'est-à-dire construire avec ces règles une *preuve* dont $\vdash \Gamma$ est la conclusion) :

$$\frac{}{\vdash A, A^\perp}(\texttt{Ax}) \qquad \frac{\vdash \Gamma, A^\perp \qquad \vdash A, \Delta}{\vdash \Gamma, \Delta}(\texttt{Cut}) \qquad \frac{\vdash \Gamma, A \qquad \vdash B, \Delta}{\vdash \Gamma, A \otimes B, \Delta}(\otimes) \qquad \frac{\vdash \Gamma, A, B}{\vdash \Gamma, A \parr B}(\parr)$$

$$\frac{\vdash \Gamma, A \qquad \vdash \Gamma, B}{\vdash \Gamma, A \& B}(\&) \qquad \frac{\vdash \Gamma, A}{\vdash \Gamma, A \oplus B}(\oplus_1) \qquad \frac{\vdash \Gamma, B}{\vdash \Gamma, A \oplus B}(\oplus_2)$$

Une occurence de (`Cut`) dans une preuve qui ajoute les formules A et $A^\perp$ dans ses séquents prémisses est appelée une *coupure sur* A.

Usuellement on appelle "tenseur" le connecteur $\otimes$, "par" le $\parr$, "plus" le $\oplus$ et "avec" le $\&$.
On définit la formule $A \multimap B$ comme $A^\perp \parr B$.

Question 1. Montrer que $\vdash A \otimes (B \oplus C) \multimap (A \otimes B) \oplus (A \otimes C)$ est prouvable

Question 2. Montrer qu'il n'existe pas de preuve de $\vdash A \multimap (A \otimes A)$ sans coupure.

Question 3. Informellement, on interprète la formule A comme "un exemplaire de la ressource A".

Donner une interprétation informelle des symboles $\cdot^\perp$, $\otimes$, $\oplus$, $\&$, $\multimap$.

On dit qu'une règle *élimine* la formule A, si A est une formule qui apparaît dans le séquent conclusion de la règle mais dans aucun des séquents prémisses. Par exemple, la règle $(\otimes)$ élimine la formule $A \otimes B$.
Dans une preuve, on dit qu'une coupure sur la formule C est *principale* si $C^\perp$ et C sont toutes les deux éliminées par (respectivement) la première règle appliqué à la prémisse gauche et la première règle appliquée à la prémisse droite de la coupure.

Question 4. Soit Π une preuve de $\vdash \Gamma_0$ contenant une coupure principale sur la formule C.

Montrer qu'on peut supprimer la coupure sur C de Π, c'est-à-dire obtenir une preuve de $\vdash \Gamma_0$ sans coupure sur C.

Question 5. Soit Π une preuve de $\vdash \Gamma_0$ contenant une coupure non-principale sur la formule C. Montrer qu'il existe une preuve Π' de $\vdash \Gamma_0$ sans coupure sur C.

Le théorème d'*élimination des coupures* dans un système de preuve (un ensemble de règles) $\mathcal{L}$ contenant (`Cut`) est donné par :
Tout séquent $\vdash \Gamma$ *prouvable dans* $\mathcal{L}$ *est prouvable dans* $\mathcal{L}$ *privé de la règle* (`Cut`)

Question 6. Les transformations induites par les questions 4. et 5. forment-elle un bon algorithme pour prouver l'élimination des coupures dans MALL ?

> MLL est la logique obtenue à partir de MALL en n'utilisant que les formules atomiques, la dualité $\cdot^{\perp}$ et les connecteurs $\otimes$ et $\parr$, et seulement les règles (`Ax`), (`Cut`), ($\otimes$) et ($\parr$).
> Un *réseau de MLL* (ou simplment *réseau*) est un graphe orienté avec arêtes pendantes (des arêtes sans destination dont la source est un sommet du graphe), dont les sommets (appelés *noeuds*) sont étiquetés par (`Ax`), (`Cut`), ($\parr$), ($\otimes$). Chaque noeud étiqueté par ($\otimes$) ou ($\parr$) a autant d'arêtes entrantes que le nombre de prémisses de la règle associée à son étiquette. Les noeuds étiquetés par (`Ax`) n'ont pas d'arêtes entrante et deux arêtes sortantes. Les noeuds étiquetés par (`Cut`) n'ont pas d'arête sortante et deux arêtes entrantes.

Question 7. Donner un algorithme qui prend une preuve Π de MLL et la transforme en un réseau correspondant.

> Un *réseau de preuve* est un réseau dans l'image de l'algorithme précédent (obtenu à partir d'une preuve d'un séquent de MLL).
> Un *interrupteur* d'un réseau $\mathcal{R}$ est un graphe non-orienté obtenu en supprimant les arêtes pendantes de $\mathcal{R}$, en supprimant, pour chaque noeud étiqueté par $\parr$ dans $\mathcal{R}$, une des deux arêtes entrantes dans ce noeud, et en désorientant les arêtes restantes.
> On admet le *théorème de Danos-Régnier* :
> *Un réseau $\mathcal{R}$ est un réseau de preuve si et seulement si tout interrupteur de $\mathcal{R}$ est connexe et acyclique.*

Question 8.

1. Donner deux preuves différentes d'un même séquent linéaire envoyées par l'algorithme de la question 7. sur le même réseau.
2. Donner un réseau qui n'est pas un réseau de preuve.
3. Donner un algorithme prend un réseau de preuve obtenu à partir d'une preuve prouvant le séquent $\vdash \Gamma$ et renvoie une preuve de $\vdash \Gamma$.

Question 9. Montrer le théorème d'élimination des coupures dans MLL.

Question 10. Peut-on appliquer cette technique à MALL ?

Structures pliables et traversables

On se place dans un langage récursif, fonctionnel, avec des définitions de types inductifs et un système de types polymorphes similaire à OCaml. On pourra utiliser indifféremment du pseudocode fonctionnel ou la syntaxe OCaml.
On suppose l'existence :

- d'une fonction identité `id` de type $A \to A$;
- d'un opérateur $(\circ) : (A \to B) \to (B \to C) \to (A \to C)$ de composition de fonctions, noté $\circ$ dans sa version infixe ; et
- de types de bases usuels, comme `int` le type des entiers et `unit` le type de l'unique élément ().

On donne la définition d'un type polymorphe `liste` A (où A est une variable de type) et, à titre d'exemple, de la fonction longueur de type `liste` $A \to$ `int` en pseudocode (par exemple) :

`liste` $A =$ `Nil` | `Cons` A (`liste` A)
`longueur` : `liste` $A \to$ `int`
  `longueur` `Nil` $= 0$
  `longueur` (`Cons` _ qu) $= 1 +$ (`longueur` qu)

---

et en OCaml :

```
type 'a liste = Nil | Cons of ('a * 'a liste)
let rec longueur : 'a liste -> int = function
        Nil -> 0
      | Cons (_,qu) -> 1 + (longueur qu)
```

Un type T est un *monoïde*, s'il existe (c'est-à-dire, si on peut définir) ε de type T et $(\diamond)$ de type $T \to T \to T$, noté $\diamond$ dans sa version infixe, tels que pour tous a, b, c de type T, $(a \diamond b) \diamond c = a \diamond (b \diamond c)$ et $\varepsilon \diamond a = a \diamond \varepsilon = a$.

Question 1. Montrer que `liste` A est un monoïde pour tout A.

Un *constructeur de type* est une fonction mathématique C qui associe à un type paramètre T un type résultat $C\ T$. Par exemple `liste` est un constructeur de type.
On dit qu'un constructeur de type C est *pliable* quand il existe un `fold` de type

$$(A \to B \to B) \to B \to C\ A \to B$$

pour tous types A et B, tel que `fold` $f\ e\ t$ applique de manière répétée la fonction f à chacun des éléments de type A de la structure t (elle-même de type CA) pour produire une valeur de type B, en partant initialement de l'élément e de type B.

Question 2. Montrer que `liste` est pliable.

Un `fold` pour `liste` est-il unique ?

Question 3.

1. Donner la définition d'un constructeur de type `barbre` tel que les éléments de `barbre` A sont des arbres binaires dont les nœuds sont des éléments de A.
2. Montrer que `barbre` est pliable.
3. Expliquer comment différentes définitions de `fold` permettent de générer les parcours infixe et préfixe d'un arbre.

Question 4. Utiliser `fold` pour écrire `arbreTaille` : `int` $\to$ `liste` (`barbre` `unit`), la fonction générant une liste des arbres binaires de taille n (c'est-à-dire, à n nœuds) avec les nœuds prenant leur valeur dans `unit`.

Une définition alternative d'un constructeur de type pliable est l'existence d'un `foldmap` de type

$$(A \to M) \to C\ A \to M$$

pour tout type A et pour tout monoïde M, tel que `foldmap` f t parcourt la structure t en accumulant une valeur de type M.

Question 5. Montrer que `liste` et `barbre` sont pliables pour cette définition.

Donner deux définitions de `foldmap` pour les `barbre` permettant de générer les parcours infixes et préfixes.

Question 6. Montrer que les deux définitions de pliabilité sont équivalentes.

Un *foncteur applicatif* est un constructeur de type F pour lequel il existe pour tous types A, B :

1. `fmap` de type $(A \to B) \to F\ A \to F\ B$
2. `pure` de type $A \to F\ A$
3. $(\odot)$ de type $F\ (A \to B) \to F\ A \to F\ B$, noté $\odot$ dans sa version infixe.

qui vérifie (entre autres) les lois suivantes :
`fmap id` $=$ `id` `fmap` $(f \circ g) =$ (`fmap` f) $\circ$ (`fmap` g)
(`pure id`) $\odot$ $v = v$

Question 7. Sans vérifier formellement les lois, donner deux manières différentes de montrer que `liste` est un foncteur applicatif.

Un constructeur de types C est *traversable* s'il existe `traverse` de type

$$(A \to F\ B) \to C\ A \to F\ (C\ B)$$

pour tout foncteur applicatif F et types A et B.

Question 8. Montrer que `barbre` est traversable.

Question 9. On dispose d'un arbre t de type `barbre` A et une fonction `futurs` qui à un élément A associe un une liste de type `liste` A qui correspond aux « futurs possibles » (dans un sens non-déterministe) d'un élément.

A quoi correspond `traverse futurs` t ?

Graphes parfaits

Pour S un ensemble, on note $\mathcal{P}_2(S) = \{\{x, y\} : x, y \in S, x \neq y\}$ les sous-ensembles de S de cardinalité 2. Dans tout le sujet, on considère des graphes non-orientés $G = (V, E)$, avec un ensemble fini de sommets V, et un ensemble d'arêtes $E \subseteq \mathcal{P}_2(V)$.
Coloriage. Un coloriage d'un graphe $G = (V, E)$ est une application $V \to \mathbb{N}$. Dans ce contexte, on appelle les entiers des *couleurs*. Un coloriage est *valide* si toute paire de sommets reliés par une même arête a des couleurs différentes. Le *nombre chromatique* de G, noté $\chi(G)$, est le nombre minimal de couleurs nécessaires pour créer un coloriage valide de G.
Sous-graphe induit. Étant donné un graphe $G = (V, E)$ et un sous-ensemble de sommets $W \subseteq V$, le *sous-graphe induit* par W est le graphe $G[W] = (W, E \cap \mathcal{P}_2(W))$. On dit que c'est un sous-graphe induit *propre* si W est inclus strictement dans V.
Cliques. Une *clique* d'un graphe $G = (V, E)$ est un sous-ensemble de sommets $W \subseteq V$ tel que le sous-graphe $G[W]$ induit par W est un graphe complet, c'est-à-dire : $G[W] = (W, \mathcal{P}_2(W))$. On note $\omega(G)$ la cardinalité de la plus grande clique de G.
Anticliques. Une *anticlique* d'un graphe $G = (V, E)$ est un sous-ensemble de sommets $W \subseteq V$ tel que le sous-graphe $G[W]$ induit par W ne contient pas d'arête, c'est-à-dire : $G[W] = (W, \varnothing)$. On note $\alpha(G)$ la cardinalité de la plus grande anticlique de G.

Question 1. Soit G un graphe quelconque. Montrer $\chi(G) \geq \omega(G)$.

Question 2. Soit $G = (V, E)$ un graphe quelconque.

a. Montrer qu'un coloriage valide de G avec c couleurs existe si et seulement si il existe une partition $\{A_1, \ldots, A_c\}$ de V en c anticliques.

b. Montrer $\chi(G)\alpha(G) \geq |V|$.

Graphe parfait. Un graphe G est dit *parfait* si tous ses sous-graphes induits $G[W]$ satisfont : $\chi(G[W]) = \omega(G[W])$.
Graphe imparfait minimal. Un graphe G est dit *imparfait minimal* s'il n'est pas parfait, et que tous ses sous-graphes induits propres sont parfaits.

Question 3. Donner un exemple de graphe parfait, et de graphe imparfait minimal.

Pour simplifier les notations, dans les questions 4 à 8, on fixe G un graphe imparfait minimal quelconque. Sans perte de généralité, on pose $V = \{1, \ldots, n\}$. On note $\alpha = \alpha(G)$, $\omega = \omega(G)$, $\chi = \chi(G)$.

Question 4. Soit A une anticlique de G. Montrer $\omega(G[V \setminus A]) = \omega$.

Question 5. Soit A_0 une anticlique de G de cardinalité α. Montrer qu'il existe $\alpha\omega$ anticliques $A_1, \ldots, A_{\alpha\omega}$, telles que pour chaque sommet $v \in V$, v fait partie d'exactement α anticliques parmi $A_0, \ldots, A_{\alpha\omega}$. (Formellement : $\forall v \in V, |\{i \in \{0, \ldots, \alpha\omega\} : v \in A_i\}| = \alpha$.)

Question 6. On considère une suite d'anticliques $A_0, \ldots, A_{\alpha\omega}$ définie comme dans la question précédente. Montrer que pour tout i dans $\{0, \ldots, \alpha\omega\}$, il existe une clique C_i telle que $C_i \cap A_i = \varnothing$, et $\forall j \neq i, |C_i \cap A_j| = 1$.
Indication : utiliser la question 4

Matrice d'incidence. Étant donné une suite $V_0, \ldots, V_{\alpha\omega}$ de sous-ensembles de $V = \{1, \ldots, n\}$, on définit la *matrice d'incidence* de la suite (V_i) comme la matrice $M = (M_{i,j})_{1 \leq i \leq n, 0 \leq j \leq \alpha\omega} \in \{0,1\}^{n \times (\alpha\omega+1)}$ définie par $M_{i,j} = 1$ si $i \in V_j$, 0 sinon.

Question 7. Soit M_A la matrice d'incidence de la suite $A_0, \ldots, A_{\alpha\omega}$ de la question 5, et M_C la matrice d'incidence de la suite $C_0, \ldots, C_{\alpha\omega}$ de la question 6. On note $M_A^\top$ la transposée de M_A.

Montrer que $M_A^\top M_C$ est de rang $\alpha\omega + 1$, où les matrices sont vues comme à coefficient dans $\mathbb{Q}$.

Question 8. Montrer $n \geq \alpha\omega + 1$.

Dans les questions suivantes, $G = (V, E)$ est un graphe quelconque.

Question 9. Montrer que G est parfait si et seulement si $\omega(G[W])\alpha(G[W]) \geq |W|$ pour tout sous-graphe induit $G[W]$ de G.

Question 10. Soit $\overline{G} = (V, \mathcal{P}_2(V) \setminus E)$ le graphe *complémentaire* de G. Montrer que G est parfait si et seulement si $\overline{G}$ est parfait.

Théorème des amis

L'objectif du sujet est de montrer que dans tout groupe de personnes, si chaque paire de personnes a exactement un ami en commun, alors il existe quelqu'un (le diplomate) qui est ami de tout le monde. Le problème est modélisé par des graphes.

Graphe. Pour S un ensemble, on note $\mathcal{P}_2(S) = \{\{x, y\} \subseteq S : x \neq y\}$ l'ensemble des paires d'éléments distincts de S. Dans tout le sujet, on considère des graphes non-orientés $G = (V, E)$, composés d'un ensemble fini de sommets V, et d'un ensemble d'arêtes $E \subseteq \mathcal{P}_2(S)$.

Voisins. Étant donné un graphe $G = (V, E)$ et un sommet $x \in V$, l'ensemble des *voisins* de x est $N(x) = \{y \in V : \{x, y\} \in E\}$. Le *degré* de x est le nombre de voisins de x.

Graphe d'amis. Un *graphe d'amis* est un graphe $G = (V, E)$ tel que $|V| \geq 2$ et pour tout $x, y \in V$ avec $x \neq y$, il existe un unique sommet, noté $x \star y$, tel que : $\{x, x \star y\} \in E$ et $\{y, x \star y\} \in E$.

Diplomate. Étant donné un graphe $G = (V, E)$, un *diplomate* est un sommet de degré $|V| - 1$.

Exemple. Le graphe complet à trois sommets $G = (\{1, 2, 3\}, \{\{1, 2\}, \{2, 3\}, \{1, 3\}\})$ est un graphe d'amis contenant 3 diplomates.

Question 1.

a. Montrer qu'un graphe d'amis $G = (V, E)$ ne contient pas de 4-cycle, c'est-à-dire :

$$\neg\exists\{a, b, c, d\} \subseteq V, \{\{a, b\}, \{b, c\}, \{c, d\}, \{d, a\}\} \in E.$$

b. Donner un exemple de graphe d'amis à 5 sommets.

Question 2. Dans cette question, on suppose que G est un graphe d'amis contenant un sommet x de degré 2. On note y, z les deux voisins de x.

a. Montrer $\{y, z\} \in E$.

b. Montrer $V = N(y) \cup N(z)$.

c. Montrer que y ou z est un diplomate.

Dans les questions 3 à 5, on fixe $G = (V, E)$ un graphe d'amis ne contenant pas de sommet de degré 2. (On suppose donc pour l'instant qu'un tel graphe existe ; on verra plus tard si cela amène une contradiction.) Soit $n = |V|$. Sans perte de généralité, on pose $V = \{1, \ldots, n\}$.

Question 3. Soit $\{x, y\}$ une arête de G et $z = x \star y$. Soit $X = N(x) \setminus \{y, z\}$, $Y = N(y) \setminus \{x, z\}$, $Z = N(z) \setminus \{x, y\}$, et $W = V \setminus (X \cup Y \cup Z \cup \{x, y, z\})$.

a. Montrer que l'application suivante est bien définie et bijective :

$$\begin{aligned} f_\star : X \times Y &\to W \\ (a, b) &\mapsto a \star b. \end{aligned}$$

b. Montrer que tous les sommets de G on le même degré (on dit que G est *régulier*).

c. On note δ le degré d'un sommet de G. Montrer $n = \delta^2 - \delta + 1$.

Matrice d'adjacence. La *matrice d'adjacence* de G est la matrice $M_{i,j\in\{1,\ldots,n\}} \in \mathbb{R}^{n\times n}$ définie par $M_{i,j} = 1$ si $\{i, j\} \in E$, 0 sinon.

Question 4. Soit M la matrice d'adjacence du graphe G. On a vu dans la question 3 que G est régulier ; et on note δ le degré d'un sommet de G. Montrer $M^2 = \mathbb{1}_n + (\delta - 1)\mathsf{Id}_n$, où $\mathbb{1}_n \in \mathbb{R}^{n\times n}$ est la matrice ne contenant que des 1, et $\mathsf{Id}_n \in \mathbb{R}^{n\times n}$ est la matrice identité.

On admet (ou rappelle) les faits suivants.

— Toute matrice symétrique réelle est diagonalisable.

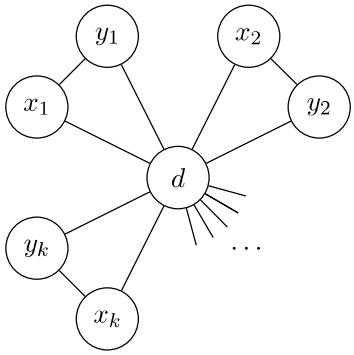
— La trace d'une matrice est invariante par changement de base.

— Si $k \in \mathbb{N}$ et $\sqrt{k} \in \mathbb{Q}$, alors $\sqrt{k} \in \mathbb{N}$ (lemme de Dirichlet).

— Les valeurs propres de $M^2 = \mathbb{1}_n + (\delta - 1)\mathsf{Id}_n$ sont δ^2 (multiplicité 1) et $\delta - 1$ (multiplicité $n - 1$).

Question 5. Montrer que la trace de M doit être nulle, et aboutir à une contradiction.

Question 6. Montrer que pour tout graphe d'amis $G = (V, E)$, il existe un diplomate $d \in V$, et une partition de $V \setminus \{d\}$ en paires $\{\{x_1, y_1\}, \ldots, \{x_k, y_k\}\}$, telle que $E = \bigcup_{i=1}^{k} \{\{x_i, y_i\}, \{d, x_i\}, \{d, y_i\}\}$.



Graphe moulin

Raisonnements ensemblistes

L'ensemble des entiers naturels est noté $\mathbb{N}$, et l'ensemble des parties de $\mathbb{N}$ est noté $\mathcal{P}(\mathbb{N})$.
Soit $\mathcal{T}$ un ensemble fini dont les éléments sont appelés des *variables ensemblistes*. Une *inclusion* est une formule s'écrivant : $(X \subseteq Y)$ où X (resp. Y) est une variable ensembliste, ou $\emptyset$, ou $\mathbb{N}$. Par abus de notation, $\emptyset$ et $\mathbb{N}$ représentent ici respectivement les ensembles vide et plein.
Attention : dans la suite, par convention, les lettres $A, B, C, D \ldots$ désigneront des éléments de $\mathcal{T}$, tandis que les lettres X, Y, Z désigneront des éléments de $\mathcal{T} \cup \{\emptyset, \mathbb{N}\}$.
Une conjonction d'inclusions sur les variables de $\mathcal{T}$ est dite *satisfiable* s'il existe une *valuation* $f : \mathcal{T} \to \mathcal{P}(\mathbb{N})$ des variables qui vérifie toutes les inclusions.
Pour deux formules Φ et Ψ, $\Phi \vDash \Psi$ signifie que toute valuation satisfaisant Φ satisfait également Ψ.

Question 1.
1. Montrer que la conjonction d'inclusions : $(\mathbb{N} \subseteq A_1) \wedge (A_2 \subseteq A_3) \wedge (A_3 \subseteq \emptyset)$ est satisfiable.
2. Montrer qu'en ajoutant l'inclusion : $\mathbb{N} \subseteq A_2$, ce n'est plus satisfiable.

Solution : Prendre $A_1 = \mathbb{N}$, $A_2 = A_3 = \emptyset$. Dans le deuxième cas on a une contradiction entre $A_3 = \mathbb{N}$ et $A_3 = \emptyset$.

Question 2. Soit une conjonction d'inclusions ne contenant aucune inclusion de la forme $\mathbb{N} \subseteq X$. Montrer que cette conjonction est satisfiable.

Solution : Il suffit d'évaluer toutes les variables à $\emptyset$.

Soit une conjonction d'inclusions $\Phi = (X_1 \subseteq Y_1) \wedge \ldots \wedge (X_k \subseteq Y_k)$. On définit un graphe orienté G_Φ tel que :

— Les sommets de G_Φ sont les éléments de $\mathcal{T} \cup \{\emptyset, \mathbb{N}\}$

— Pour toute inclusion $X_i \subseteq Y_i$ de Φ, on crée une arête $X_i \to Y_i$ dans G_Φ

— Pour tout sommet $A \neq \mathbb{N}$, on crée une arête $A \to \mathbb{N}$, et pour tout sommet $A \neq \emptyset$, une arête $\emptyset \to A$

Un *chemin* dans G_Φ est une séquence de sommets $Z_1, \ldots, Z_\ell$ reliés par des arêtes $Z_1 \to Z_2, Z_2 \to Z_3, \ldots, Z_{\ell-1} \to Z_\ell$. Par convention, on autorise des chemins ne comportant qu'un seul sommet (et aucune arête).

Question 3. Montrer l'équivalence entre les propriétés suivantes :
1. Φ est satisfiable
2. Il n'existe pas de chemin dans G_Φ menant de $\mathbb{N}$ à $\emptyset$
3. Φ est satisfiable à valeurs dans $\{\emptyset, \mathbb{N}\}$

Solution : Pour $3 \implies 1$: si Φ est satisfiable à valeurs dans $\{\emptyset, \mathbb{N}\}$, en particulier elle est satisfiable.

Pour $1 \implies 2$: tout chemin $Z_1 \to Z_2 \to \ldots \to Z_k$ dans G se traduit en une chaîne d'inclusions $Z_1 \subseteq \ldots \subseteq Z_k$. Par conséquent, s'il existe un chemin de $\mathbb{N}$ à $\emptyset$, par transitivité de l'inclusion (et récurrence immédiate sur la longueur du chemin) Φ implique $\mathbb{N} \subseteq \emptyset$ ce qui est une contradiction, et elle n'est pas satisfiable.

Pour $2 \implies 3$: Pour un sommet X de G_Φ, soit $s(X)$ l'ensemble des sommets Y tels que $X \to^* Y$ (successeurs, par clôture transitive), et $p(X)$ l'ensemble des sommets Y tels que $X \to^* Y$ par des chemins dans le graphe (prédécesseurs).

Supposons qu'il n'existe pas de chemin de $\mathbb{N}$ à $\emptyset$. On montre qu'il existe une valuation qui fonctionne, qui sera à valeurs dans $\{\emptyset, \mathbb{N}\}$. On propose d'évaluer à $\mathbb{N}$ pour $V_N = s(\mathbb{N})$ et $\emptyset$ pour $V_0 = p(\emptyset)$. Par hypothèse V_0 et V_N sont disjoints. Quant aux sommets restants, on les évalue tous à $\emptyset$. Pour vérifier que cette valuation fonctionne, il faut vérifier qu'elle satisfait toutes les inclusions de Φ.

Soit $X \subseteq Y$ une inclusion dans Φ. On étudie les cas possibles : si $X \in V_N$, alors $Y \in V_N$ et l'inclusion est satisfaite. Si $Y \in V_0$, alors $X \in V_0$ et de même. Si $X \in V_0$, alors $X \subseteq D$ sera toujours satisfait. Si $Y \in V_N$, de même. Reste le cas où : $X \notin (V_0 \cup V_N)$ et $Y \notin (V_0 \cup V_N)$. Comme on a évalué ces deux variables à $\emptyset$ cette inclusion est satisfaite.

Question 4. Montrer que si Φ est satisfiable et X et Y sont des sommets de G_Φ, alors $\Phi \vDash (X \subseteq Y)$ si et seulement s'il existe un chemin dans G_Φ menant de X à Y.

Solution : Une des deux implications est facile : s'il existe un chemin de X à Y, on a $X \subseteq Y$.

Supposons qu'il n'existe pas de chemin de X à Y (en particulier $X \neq Y$, car $X \to^* X$ est aussi un chemin). Par hypothèse Φ est satisfiable, donc il n'y a pas de chemin de $\mathbb{N}$ à $\emptyset$. Cette fois, on va trouver une valuation de Φ qui ne satisfait pas $X \subseteq Y$.

L'idée est d'assigner $\emptyset$ à Y et $\mathbb{N}$ à X, et de voir si on peut encore satisfaire toutes les inclusions de Φ. (Notons qu'on ne peut pas avoir $Y = \mathbb{N}$ ni $X = \emptyset$ ici puisque cela contredirait l'hypothèse selon laquelle il n'y a aucun chemin). On propose la valuation suivante : $\mathbb{N}$ pour $V_N = s(X)$ (qui contient aussi les successeurs de $\mathbb{N}$ puisque $\mathbb{N}$ est successeur de X) ; $\emptyset$ pour $V_0 = p(Y)$ (qui contient aussi les prédécesseurs de $\emptyset$) ; $\emptyset$ pour les sommets restants. Par hypothèse V_0 et V_N sont disjoints, sinon on aurait un chemin de X à Y. On réutilise le raisonnement ci-dessus pour montrer que toutes les inclusions de Φ sont satisfaites.

On peut remarquer que par exemple une formule Φ force $\emptyset \subseteq A$ pour toute variable A, même si A n'apparaît pas dans Φ.

Question 5. En déduire un algorithme Résolution qui prend en entrée une conjonction d'inclusions Φ et une inclusion $(X \subseteq Y)$, et détermine si $\Phi \vDash (X \subseteq Y)$. Montrer qu'avec la bonne structure de données, cet algorithme est de complexité linéaire en le nombre d'inclusions de Φ et de variables de $\mathcal{T}$.

Solution : Il faut d'abord déterminer si Φ est satisfiable (dans ce cas on renvoie Vrai tout le temps).

Ensuite, il faut déterminer s'il existe un chemin entre X et Y dans le graphe G_Φ. On fait simplement un parcours en profondeur à partir de X (il n'est pas demandé de le détailler, puisqu'ils sont censés le connaître). Quand le graphe est représenté comme une liste d'adjacence, la complexité est linéaire en le nombre d'arêtes. Ce nombre d'arêtes dépend du nombre d'inclusions dans Φ et de variables dans $\mathcal{T}$.

On considère maintenant des formules logiques dont les littéraux sont des inclusions, par exemple :

$$(X_1 \subseteq X_2) \vee \neg(X_2 \subseteq X_3) \vee \neg(X_4 \subseteq X_3)$$

Ces formules sont écrites en forme normale conjonctive. On suppose que toutes les clauses sont *Horn*, c'est-à-dire qu'elles contiennent au plus un seul littéral positif (sans négation). On les interprète alors comme des implications. La formule ci-dessus devient ainsi :

$$\Big((X_2 \subseteq X_3) \wedge (X_4 \subseteq X_3)\Big) \implies (X_1 \subseteq X_2)$$

On appelle une telle formule *inclusion-Horn* et on lui étend la notion de satisfiabilité ci-dessus.

Question 6. Montrer que la formule inclusion-Horn avec les clauses suivantes :

$$\begin{aligned}
&\neg(A_3 \subseteq \emptyset)\\
&\neg(A_1 \subseteq A_3)\\
&\neg(A_2 \subseteq A_3) \vee \neg(A_1 \subseteq A_2)\\
&(\mathbb{N} \subseteq A_1)\\
&(A_2 \subseteq \emptyset)
\end{aligned}$$

est satisfiable, mais qu'aucune de ses valuations satisfaisantes n'est à valeurs dans $\{\emptyset, \mathbb{N}\}$.

Solution : On est obligé de prendre $A_1 = \mathbb{N}$, $A_2 = \emptyset$. Ensuite, comme $A_1 \subseteq A_2$ est faux, il ne reste plus que les contraintes : $\neg(A_1 \subseteq A_3) \implies A_3 \neq \mathbb{N}$ et $\neg(A_3 \subseteq \emptyset) \implies A_3 \neq \emptyset$. Pour A_3 on peut prendre tout ensemble non vide qui n'est ni $\emptyset$ ni $\mathbb{N}$.

On admet la propriété (P) :

> Soit Ψ une conjonction d'inclusions. Si Ψ est satisfiable, il existe une valuation f telle que pour tous sommets X, Y dans le graphe G_Ψ, s'il n'existe pas de chemin entre X et Y, alors $f(X)$ et $f(Y)$ sont incomparables (i.e., $f(X) \not\subseteq f(Y)$ et $f(Y) \not\subseteq f(X)$).

Question 7. Soit Φ une formule inclusion-Horn, et soit Ψ la conjonction de toutes les clauses de Φ ne contenant aucun littéral négatif. C'est donc une conjonction d'inclusions de la forme : $\Psi = (X_1 \subseteq Y_1) \wedge \ldots \wedge (X_k \subseteq Y_k)$.

Montrer que si Ψ est satisfiable, et si pour tout littéral négatif $\neg(X \subseteq Y)$ dans les clauses de Φ, $\Psi \not\models (X \subseteq Y)$, alors Φ est satisfiable.

Solution : Noter que, puisque le graphe G_Ψ contient toutes les variables de $\mathcal{T}$ (y compris les variables qui ne sont pas dans Ψ), la valuation donne bien des valeurs à toutes ces variables.

Supposons que Ψ est satisfiable. Utilisons la valuation ci-dessus. Pour toutes variables X, Y, si $\Psi \not\models (X \subseteq Y)$ alors il n'existe pas de chemin entre X et Y par la question 4. Donc $f(X)$ et $f(Y)$ sont incomparables, donc le terme $\neg(X \subseteq Y)$ est vrai.

Par conséquent cette valuation rend tous les termes négatifs dans les clauses de Φ vrais. Toutes les clauses sans termes négatifs étant vraies (puisque Ψ l'est), toutes les clauses de Φ sont vraies. Elle est donc satisfiable.

Question 8. En déduire un algorithme déterminant, en temps polynomial, si une formule inclusion-Horn est satisfiable.

Solution : On définit un algorithme récursif.

Soit Φ une formule inclusion-Horn et soit Ψ l'ensemble de ses clauses positives. On effectue une disjonction de cas. On voit avec la question 7 qu'on va avoir par exemple le cas où Ψ est satisfiable et n'implique aucun littéral négatif, dans ce cas on peut s'arrêter et la formule est satisfiable. Sinon il reste d'autres cas :

1. Si Φ contient une clause vide, on renvoie « Non satisfiable »
2. Si Ψ n'est pas satisfiable, on renvoie « Non satisfiable »
3. Sinon, il existe un littéral négatif $\neg(X \subseteq Y)$ dans une des clauses de Φ tel que $\Psi \vDash X \subseteq Y$. Il faut observer que la formule obtenue en enlevant $\neg(X \subseteq Y)$ de la clause correspondante est équivalente à Φ.

Cela nous donne un algorithme récursif qui transforme Φ, Ψ en Φ', Ψ' où Φ' est plus petite que Φ (quand on ne s'arrête pas) d'un littéral négatif. Chaque appel récursif réduit la taille de la formule, donc le temps est polynomial. La correction de l'algorithme découle des questions précédentes. On peut raffiner le temps d'exécution de quadratique à linéaire en utilisant des structures de données adaptées.

Question 9. Prouver la propriété (P).

Solution : La valuation se construit en résolvant d'abord les cycles de G_Ψ, de façon à enlever toutes les inclusions triviales. On enlève aussi les successeurs de $\mathbb{N}$ et les prédécesseurs de $\emptyset$ (dont les valuations sont évidentes).

On fait ensuite un tri topologique sur les sommets restants, de façon à les parcourir dans l'ordre suivant : on doit s'assurer que pour chaque sommet, on a d'abord étudié tous ses prédécesseurs. On démarre de $\emptyset$. Soit $A_0, \ldots, A_k, \ldots$ la séquence des sommets rencontrés, alors l'ensemble $S(A_k)$ correspondant à A_k est calculé de la manière suivante :

$$S(A_k) = \{k\} \cup \bigcup_{X, X \to A_k} S(X)$$

Cette valuation satisfait bien toutes les inclusions, et de plus, elle a la propriété demandée par construction.

Permutations triables par pile

On s'intéresse aux permutations $\mathfrak{P}(n)$ de $\{1,\dots,n\}$. Une permutation $\pi \in \mathfrak{P}(n)$ est assimilée à la suite $(\pi(1),\pi(2),\dots,\pi(n))$.

Machine à pile. Une *machine à pile* maintient en interne une structure de pile, initialement vide. Elle prend en entrée une permutation $(\pi(1),\dots,\pi(n))$, et effectue une suite fixée d'opérations empile et dépile.

- empile : lit le premier élément non encore lu de la permutation, et l'ajoute à la pile. La i-ième opération empile ajoute donc $\pi(i)$ à la pile.
- dépile : enlève le dernier élément ajouté à la pile, et le renvoie en sortie.

La sortie de la machine est une suite d'éléments de $\{1,\dots,n\}$ renvoyés par les opérations dépile, pris dans l'ordre où ils sont renvoyés. *Exemple :* la machine EEDEDD qui effectue (empile, empile, dépile, empile, dépile, dépile) avec en entrée la permutation $(3,1,2)$ renvoie $(1,2,3)$:

$$\mathsf{EEDEDD}(3,1,2) = (1,2,3).$$

On remarque que le comportement d'une machine à pile est entièrement défini par la suite (fixe) d'opérations empile et dépile qu'elle effectue.

Suite valide. Une telle suite d'opérations est dite *valide* si elle contient exactement n opérations empile et n opérations dépile, et si à tout instant, le nombre d'opérations dépile effectuées est inférieur ou égal au nombre d'opérations empile effectuées. Dans tout le sujet, on ne considère que des machines effectuant des suites valides d'opération.

Permutation triable par pile. Une permutation $\pi \in \mathfrak{P}(n)$ est dite *triable par pile* s'il existe une machine à pile qui prend en entrée $(\pi(1),\dots,\pi(n))$, et qui renvoie $(1,\dots,n)$.

Exemple : l'égalité $\mathsf{EEDEDD}(3,1,2) = (1,2,3)$ plus haut montre que $(3,1,2)$ est triable par pile.

Question 1. Montrer que les permutations $(1,2,3,4)$, $(4,3,2,1)$, $(1,3,2,4)$ sont triables par pile.

Motif. On dit que $\pi \in \mathfrak{P}(n)$ *contient le motif* $\rho \in \mathfrak{P}(k)$ pour $k \leq n$ s'il existe $x_1 < \cdots < x_k$ dans $\{1,\dots,n\}$ tel que $(\pi(x_1),\dots,\pi(x_k)) = (x_{\rho(1)},\dots,x_{\rho(k)})$ (ou plus informellement : tel que $(\pi(x_1),\dots,\pi(x_k))$ et $(\rho(1),\dots,\rho(k))$ sont « dans le même ordre »).

Question 2. Montrer que si π est triable par pile, alors elle ne contient pas le motif $(2,3,1)$.

Question 3. Montrer que si π ne contient pas le motif $(2,3,1)$, alors elle est triable par pile.

Question 4. L'ensemble des permutations triables par pile est-il clos par composition ? Par inverse ?

Question 5. Proposer un algorithme qui détermine en temps linéaire en n si une permutation $\pi \in \mathfrak{P}(n)$ est triable par pile.

Indication : on peut utiliser les questions 2 et 3, mais ce n'est pas obligatoire.

Graphe associé à une permutation. À une permutation $\pi \in \mathfrak{P}(n)$, on associe le graphe non-orienté $G(\pi) = (V, E)$ de sommets $V = \{1, \ldots, n\}$, et d'arêtes $E = \{\{a, b\} \in V : a < b \text{ et } \pi(a) > \pi(b)\}$.
Graphe triable par pile. On dit qu'un graphe $G = (V, E)$ avec $V = \{1, \ldots, n\}$ est *triable par pile* s'il existe $\pi \in \mathfrak{P}(n)$ tel que $G = G(\pi)$ et π est triable par pile.
Graphe réalisable. On dit qu'un graphe $G = (V, E)$ est réalisable s'il est possible de renommer ses sommets V avec les entiers $\{1, \ldots, n\}$, de telle sorte que le graphe obtenu est triable par pile.

Question 6. Montrer que $\pi \mapsto G(\pi)$ est une injection de $\mathfrak{P}(n)$ vers l'ensemble des graphes de sommets $\{1, \ldots, n\}$. Est-ce une bijection ?

Question 7. Donner un exemple de graphe qui n'est pas réalisable.

Question 8. On considère l'ensemble des graphes formés en partant du graphe vide $(\varnothing, \varnothing)$, et en utilisant uniquement les deux opérations suivantes : ajout d'un sommet universel à un graphe de l'ensemble (un sommet est dit *universel* s'il est relié à tous les autres sommets), union disjointe de deux graphes de l'ensemble. Montrer que l'ensemble obtenu est exactement l'ensemble des graphes réalisables.

Question 9. En s'inspirant de la question précédente, proposer un algorithme qui détermine si un graphe $G = (V, E)$ est réalisable, en temps $O(|V|^3)$.