Les correcteurs attendent des réponses précises et concises aux questions posées. On demande à plusieurs reprises de proposer des algorithmes. On exprimera ces algorithmes avec un point de vue de haut niveau, sans décrire leur implantation effective ni les structures de données utilisées (cf l'algorithme proposé dans l'énoncé de la question 1.5). La complexité d'un algorithme doit toujours être comprise comme le nombre d'opérations élémentaires (calculs, comparaisons, ...) nécessaires à son exécution.

1 Algorithmique des graphes

Un *multi-graphe orienté fini* (*graphe* dans la suite) est un couple (N, E) où N est un ensemble fini dont les éléments sont appelés les *nœuds* et où E est un ensemble fini dont les éléments sont appelés les *arcs*. À chaque arc $e \in E$ est associé un *nœud initial* $i(e) \in N$ et un *nœud final* $f(e) \in N$. Soit K un ensemble, un *graphe valué dans* K est un graphe (N, E) muni d'une application $\varphi : E \to K$. Les éléments de K sont appelés les *étiquettes*.

Graphiquement, un nœud est représenté par un point, un arc e par une flèche orientée du point $i(e)$ vers le point $f(e)$, éventuellement surmontée de l'étiquette $\varphi(e)$. Il peut y avoir plusieurs flèches entre deux mêmes nœuds, ainsi que des flèches bouclant sur un nœud.

Un *chemin (fini)* p du graphe est une suite finie d'arcs dont les extrémités sont consécutives : $p = (e_1, e_2, \ldots, e_k)$, $e_i \in E$, et $f(e_i) = i(e_{i+1})$. Le chemin p *commence* en $i(e_1)$ et *se termine* en $f(e_k)$. On dit également que le chemin va de $i(e_1)$ à $f(e_k)$. On utilise la notation condensée $x \to y$ pour indiquer qu'il existe un chemin allant du nœud x au nœud y. La *longueur* du chemin, notée $|p|$, est égale au nombre d'arcs. Par convention, on considère qu'il existe un chemin de longueur nulle allant de u à u, pour tout nœud u. Un *circuit* est un chemin de longueur non nulle qui commence et se termine en le même nœud.

Soit un graphe (N, E) et un couple de nœuds distingués (i, j). Le graphe *émondé* correspondant est le graphe (N', E') défini comme suit :

$$N' = \{u \in N \ : \ i \to u, u \to j\} \text{ et } E' = \{e \in E \ : \ i(e) \in N', f(e) \in N'\}.$$

Question 1.1.

Proposer et justifier un algorithme prenant en entrée un graphe et un couple de nœuds distingués et retournant en sortie le graphe émondé. On réduira autant que possible la complexité, et on évaluera son ordre de grandeur.

Soit un graphe $\mathcal{G} = (N, E)$ valué dans les réels par $\varphi : E \to \mathbb{R}$. On suppose que l'on a $E \subset N \times N$; pour $e = (u, v) \in E$, on suppose naturellement que $i(e) = u$ et $f(e) = v$. Il y a donc au plus un arc pour chaque couple de nœuds. On pose $n = \text{Card}(N)$ et $m = \text{Card}(E)$ (où $\text{Card}(S)$ est le cardinal de l'ensemble S). À un chemin $p = (e_1, \ldots, e_k)$, on associe son *poids* $w(p) = \varphi(e_1) + \cdots + \varphi(e_k)$. Par convention, un chemin de longueur nulle a pour poids 0.

On définit $W = (W_{ij})_{i,j \in N}$, matrice à valeurs dans $\mathbb{R} \cup \{-\infty\}$, de la façon suivante (avec $\max(a, -\infty) = \max(-\infty, a) = a$, pour tout $a \in \mathbb{R} \cup \{-\infty\}$) :

$$W_{ij}^{(0)} = \begin{cases} 0 & \text{si } i = j \\ -\infty & \text{sinon} \end{cases}, \quad W_{ij}^{(1)} = \begin{cases} \varphi((i,j)) & \text{si } (i,j) \in E \\ -\infty & \text{sinon} \end{cases}, \quad W_{ij} = \max\left(W_{ij}^{(0)}, W_{ij}^{(1)}\right).$$

Le coefficient W_{ij} est donc égal au poids maximal d'un chemin de longueur 0 ou 1 de i à j ($-\infty$ s'il n'existe pas de tel chemin).

Question 1.2.

a) Dans le graphe $\mathcal{G}$, montrer qu'il existe un circuit de poids strictement positif si et seulement s'il existe un circuit de poids strictement positif et de longueur au plus n.

b) On définit pour $i, j \in N, k \geq 2$ (avec $a + (-\infty) = (-\infty) + a = -\infty$, pour tout $a \in \mathbb{R} \cup \{-\infty\}$) :

$$
\begin{aligned}
U_{ij}^{(1)} &= W_{ij} \\
U_{ij}^{(k)} &= \max \left(U_{ij}^{(k-1)}, \max_{l \in N} \left(U_{il}^{(k-1)} + W_{lj}\right) \right) \\
&= \max_{l \in N} \left(U_{il}^{(k-1)} + W_{lj}\right), \qquad (1)
\end{aligned}
$$

où la dernière égalité provient de ce que $\forall j,\ W_{jj} \geq 0$.

Montrer qu'il existe un circuit c tel que $w(c) > 0$ si et seulement s'il existe $i \in N$ et $k \in \{1, \ldots, n\}$ tels que $U_{ii}^{(k)} > 0$.

c) En déduire qu'il existe un algorithme de complexité $O(n^4)$ prenant en entrée le graphe $\mathcal{G}$, et répondant 'oui' s'il existe un circuit de poids strictement positif, 'non' dans le cas contraire.

Question 1.3.

Soit P et Q deux matrices carrées de même dimension et à coefficients dans $\mathbb{R} \cup \{-\infty\}$. On note $P \otimes Q$ la matrice définie par

$$(P \otimes Q)_{ij} = \max_{l} (P_{il} + Q_{lj}) .$$

On remarque que l'équation (1) peut aussi s'écrire sous la forme matricielle : $U^{(k)} = U^{(k-1)} \otimes W$.

Vérifier que l'opération $\otimes$ est associative. Montrer que, grâce à cette propriété, on peut modifier l'algorithme de la question précédente de façon à obtenir un algorithme de complexité $O(n^3 \log n)$.

Question 1.4.

On ordonne les nœuds de N de façon à identifier N et $\{1, \ldots, n\}$. On définit pour $i, j, k \in N$,

$$
\begin{aligned}
V_{ij}^{(0)} &= W_{ij} \\
V_{ij}^{(k)} &= \max \left(V_{ij}^{(k-1)}, V_{ik}^{(k-1)} + V_{kj}^{(k-1)}\right) .
\end{aligned}
$$

Montrer qu'il existe un circuit c tel que $w(c) > 0$ si et seulement s'il existe $i \in N, k \in N \cup \{0\}$ tels que $V_{ii}^{(k)} > 0$.

En déduire qu'il existe un algorithme de complexité $O(n^3)$ prenant en entrée le graphe $\mathcal{G}$, et répondant 'oui' s'il existe un circuit de poids strictement positif, 'non' dans le cas contraire.

Question 1.5.

On définit l'algorithme suivant, prenant en entrée le graphe $\mathcal{G}$, et retournant 'oui' s'il existe un circuit de poids strictement positif, 'non' dans le cas contraire.

```
DÉTECTION-CIRCUIT-STRICTEMENT-POSITIF (N, E, φ)
    pour tout u ∈ N faire
        valeur[u] ← 0
    pour i ← 1 à Card(N) faire
        pour tout (u, v) ∈ E faire
            ACTUALISATION(u, v, φ)
    si ∃(u, v) ∈ E, valeur[v] < valeur[u] + φ(u, v)
        alors retourner 'oui'
        sinon retourner 'non'

ACTUALISATION (u, v, φ)
    si valeur[v] < valeur[u] + φ(u, v)
        alors valeur[v] ← valeur[u] + φ(u, v)
```

Justifier la validité de cet algorithme et montrer que sa complexité est $O(nm)$.

2 Transducteurs

Étant donnés deux ensembles E et F et une fonction $f : E \to F$, on note $\mathrm{dom}(f)$ le *domaine* de f, c'est-à-dire le sous-ensemble de E sur lequel f est définie. Un alphabet est toujours supposé fini et non vide. Étant donné un alphabet A, l'ensemble des mots finis sur A muni de l'opération de concaténation est noté A^*; l'élément neutre de cette opération, le *mot vide*, est noté $\mathbf{1}_{A^*}$. La longueur d'un mot w est notée $|w|$.

Un *transducteur (fini)* $\mathcal{T}$ est un sextuplet $\mathcal{T} = (Q, A, B, T, I, F)$, où A et B sont deux alphabets, Q est un ensemble fini, et où $I : Q \to B^*$, $F : Q \to B^*$ et $T : (Q \times A \times Q) \to B^*$ sont des fonctions. Les éléments de Q sont appelés les *états*. Un état q est dit *initial* si $q \in \mathrm{dom}(I)$ et *final* si $q \in \mathrm{dom}(F)$. On appelle ensemble des *transitions*, l'ensemble défini par $E_T = \{(p, a, u, q) \in (Q \times A \times B^* \times Q) \ : \ (p, a, q) \in \mathrm{dom}(T),\ T(p, a, q) = u\}$.

On associe au transducteur $\mathcal{T}$ le graphe (Q, E_T) valué par $\varphi : E_T \to A \times B^*$. Si $e = (p, u, v, q) \in E_T$, on a $i(e) = p$, $f(e) = q$ et $\varphi(e) = (u, v)$. Graphiquement, on représente un transducteur à l'aide de son graphe associé auquel on ajoute, pour chaque nœud initial q, une flèche entrante étiquetée $I(q)$, et, pour chaque nœud final q, une flèche sortante étiquetée $F(q)$.

Le transducteur hérite de la terminologie et des notations du graphe associé. On définit ainsi la notion de chemin dans le transducteur. On représente le chemin $p = (q_0, u_1, v_1, q_1), (q_1, u_2, v_2, q_2), \ldots, (q_{n-1}, u_n, v_n, q_n)$ sous la forme

$$p = q_0 \xrightarrow{u_1|v_1} q_1 \xrightarrow{u_2|v_2} q_2 \longrightarrow \cdots \longrightarrow q_{n-1} \xrightarrow{u_n|v_n} q_n \,.$$

On écrira également p sous la forme condensée $q_0 \xrightarrow{u|v} q_n$, avec $u = u_1u_2\cdots u_n \in A^*$ et $v = v_1v_2\cdots v_n \in B^*$. On dira que le mot u est l'*étiquette (mot) d'entrée* du chemin et que le mot v est l'*étiquette (mot) de sortie* du chemin. Par convention, pour un chemin de longueur nulle, l'étiquette d'entrée est $\mathbf{1}_{A^*}$ et celle de sortie est $\mathbf{1}_{B^*}$. Un chemin est dit *réussi* s'il mène d'un état initial à un état final.

Étant donnés deux alphabets A et B, une *transduction* est une application de A^* dans $\mathcal{P}(B^*)$, l'ensemble des parties de B^*. À un transducteur $\mathcal{T} = (Q, A, B, T, I, F)$ est associée une transduction $f_{\mathcal{T}} : A^* \to \mathcal{P}(B^*)$ qui est dite *réalisée* par $\mathcal{T}$ et qui est définie comme suit :

$f_{\mathcal{T}}(u)$ est l'ensemble des $I(p)vF(q)$ tels qu'il existe un chemin réussi $p \xrightarrow{u|v} q$.

En particulier, s'il n'existe aucun chemin réussi de mot d'entrée u, on a $f_{\mathcal{T}}(u) = \emptyset$.

En résumé, un transducteur peut être vu comme une machine prenant en entrée un mot de A^* et retournant en sortie un langage de B^*.

Exemple. On considère le transducteur $\mathcal{T} = (\{1,2\}, \{a,b\}, \{a,b,c\}, T, I, F)$ avec $\mathrm{dom}(I) = \{1\}$, $\mathrm{dom}(F) = \{1\}$, $I(1) = \mathbf{1}_{B^*}$, $F(1) = c$, et avec $T(1,a,1) = a$, $T(1,a,2) = c^3$, $T(1,b,1) = b$, $T(2,b,1) = \mathbf{1}_{B^*}$, et T non défini pour les autres triplets. Ce transducteur est représenté sur la figure 1 (lorsqu'il y a plusieurs transitions pour un même couple d'états, on représente une seule flèche avec plusieurs étiquettes).

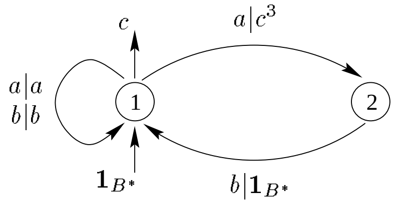


FIG. 1: Le transducteur $\mathcal{T}$.

À un mot u est associé l'ensemble $f_{\mathcal{T}}(u)$ des mots obtenus en remplaçant certaines des occurrences de ab par c^3 et en ajoutant c à la fin des mots. Par exemple, on a $f_{\mathcal{T}}(baba^3b^2) = \{baba^3b^2c, baba^2c^3bc, bc^3a^3b^2c, bc^3a^2c^3bc\}$.

Question 2.1.

Construire un transducteur sur les alphabets $A = \{a\}$ et $B = \{b,c\}$ réalisant la transduction suivante (ici, comme dans la suite, on identifie le mot w et le singleton $\{w\}$) :

$$f(\mathbf{1}_{A^*}) = \mathbf{1}_{B^*} \quad et \quad f(a^n) = \begin{cases} b^n & si\ n\ pair \\ c^n & si\ n\ impair\,. \end{cases}$$

Question 2.2.

On considère la transduction $f : A^* \to \mathcal{P}(A^*)$ qui à un mot associe l'ensemble de ses facteurs (soit un mot $w = w_1w_2\cdots w_n$, $w_i \in A$, un *facteur* de w est soit le mot vide, soit un mot de la forme $w_iw_{i+1}\cdots w_j$ avec $1 \le i \le j \le n$). *Construire un transducteur réalisant f.*

Question 2.3.

On associe à un entier non nul n sa *représentation binaire* $\langle n \rangle \in \{0,1\}^*$ définie de façon unique comme suit :

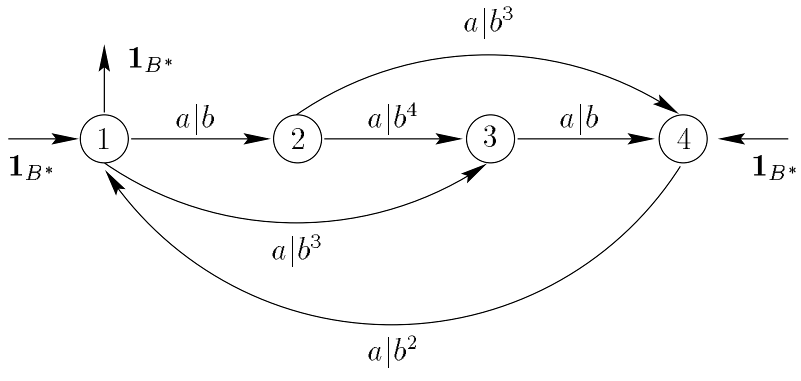
$$\textit{Si} \;\; n = \sum_{i=0}^{k} w_i 2^i, \;\; w_i \in \{0,1\}, w_k = 1, \;\; \textit{alors} \;\; \langle n \rangle = w_k w_{k-1} \cdots w_0 \,.$$

L'objectif est d'obtenir un transducteur calculant la représentation binaire de la somme de deux entiers. Soit n et m deux entiers non nuls avec $\langle n \rangle = u_k \cdots u_0$ et $\langle m \rangle = v_l \cdots v_0$. Si $\langle n \rangle$ et $\langle m \rangle$ sont de longueurs différentes, supposons par exemple que $l < k$, alors on pose $v_{l+1} = \cdots = v_k = 0$. On définit le mot w sur l'alphabet $\{0,1,2\}$ comme la somme chiffre à chiffre de u et de v, c'est-à-dire $w = w_k \cdots w_0$ avec $w_i = u_i + v_i$.
Proposer un transducteur à deux états sur les alphabets $\{0,1,2\}$ *et* $\{0,1\}$, *prenant en entrée le mot miroir de* w *et retournant en sortie le mot miroir de* $\langle n+m \rangle$ *(on appelle* mot miroir *du mot* $x_1 \cdots x_k$, *le mot* $x_k \cdots x_1$*). En déduire un transducteur prenant en entrée* w *et retournant en sortie* $\langle n{+}m \rangle$.

Une transduction $f : A^* \to \mathcal{P}(B^*)$, vérifiant $\mathrm{Card}(f(u)) \leq 1$ pour tout u dans A^*, peut être vue comme une fonction $f : A^* \to B^*$. Par exemple, la transduction de la question 2.1 est une fonction, mais celle de la question 2.2 n'en est pas une. Un transducteur $\mathcal{T}$ est *fonctionnel* si $f_{\mathcal{T}}$ est une fonction.

Question 2.4.

On a représenté ci-dessous un transducteur sur les alphabets $A = \{a\}$ et $B = \{b\}$. *Établir, en justifiant la réponse, si ce transducteur est fonctionnel ou non. Déterminer la transduction associée.*



Question 2.5.

Construire un transducteur $\mathcal{T}$ *tel que* $\mathit{Card}(f_{\mathcal{T}}(u)) = |u|^2$ *pour tout mot d'entrée* u. *Établir s'il est encore possible de construire un tel transducteur si l'on impose que l'alphabet des mots de sortie ne contienne qu'une lettre.*

3 Transducteurs séquentiels

Dans un transducteur, un état q est dit *accessible* s'il existe un chemin menant d'un état initial à q ; il est dit *co-accessible* s'il existe un chemin menant de q à un état final. Un transducteur dont tous les états sont à la fois accessibles et co-accessibles est dit *émondé*. Partant d'un transducteur, on peut, à l'aide d'un algorithme semblable à celui de la question 1.1, obtenir un transducteur émondé réalisant la même transduction. Dans toute la suite, on ne considère que des transducteurs émondés.

On peut aisément transformer un transducteur en un autre réalisant la même transduction et tel que, pour tout état initial q, l'étiquette $I(q)$ soit le mot vide. Dans toute la suite, on suppose que les transducteurs vérifient cette dernière propriété.

Un transducteur $\mathcal{T} = (Q, A, B, T, I, F)$ est dit *séquentiel* si :

1. $\mathrm{dom}(I)$ est un singleton ;
2. $\forall p \in Q, \forall a \in A,\ \ \mathrm{Card}\{q \in Q\ :\ (p, a, q) \in \mathrm{dom}(T)\} \leq 1$.

La définition d'un transducteur séquentiel est à rapprocher de celle d'un automate déterministe. Il est immédiat qu'un transducteur séquentiel est fonctionnel. Une fonction $f : A^* \to B^*$ est dite *séquentielle* si elle est réalisable par un transducteur séquentiel. Étant donné un transducteur fonctionnel mais non séquentiel, un problème important est de déterminer s'il existe un autre transducteur, séquentiel celui-ci, réalisant la même fonction.

Note culturelle : les transducteurs séquentiels peuvent être implantés de façon effective ce qui les rend très importants dans la pratique. Ils interviennent notamment en compilation, en arithmétique des ordinateurs ou encore dans l'étude du génome.

Pour $u, v \in A^*$, on note $u \wedge v$ le plus long préfixe (facteur gauche) commun de u et de v. On définit, $\forall u, v \in A^*$,

$$d(u, v) = |u| + |v| - 2|u \wedge v|\,.$$

La fonction $f : A^* \to B^*$ est dite à *variation bornée* si (on utilise la même notation $d(.,.)$ sur A^* et B^*)

$$\forall k \geq 0, \exists K \geq 0, \forall u, v \in \mathrm{dom}(f), \quad d(u, v) \leq k \implies d(f(u), f(v)) \leq K\,.$$

Question 3.1.

> *Montrer que $d(.,.)$ définit une distance sur A^*. Montrer que la fonction de la question 2.1 n'est pas à variation bornée.*

Soit $\mathcal{T} = (Q, A, B, T, I, F)$ un transducteur fini. On dit que deux états p et q sont *jumelés* si pour toute paire de chemins

$$i \xrightarrow{u_1|v_1} p \xrightarrow{u_2|v_2} p \quad \text{et} \quad j \xrightarrow{u_1|w_1} q \xrightarrow{u_2|w_2} q\,,$$

où i et j sont des états initiaux, les étiquettes de sortie vérifient la propriété suivante : soit $v_2 = w_2 = \mathbf{1}_{B^*}$, soit il existe $x \in B^*$ tel que $(w_1 = v_1 x,\ x w_2 = v_2 x)$ ou $(v_1 = w_1 x,\ x v_2 = w_2 x)$. Cette propriété peut s'exprimer de la façon équivalente suivante :

(i) $|v_2| = |w_2|$;

(ii) si $v_2 \neq \mathbf{1}_{B^*}, w_2 \neq \mathbf{1}_{B^*}$, alors les mots infinis $(v_1 v_2 v_2 v_2 \cdots)$ et $(w_1 w_2 w_2 w_2 \cdots)$ sont égaux.

Un transducteur $\mathcal{T}$ vérifie la *propriété de jumelage* si tous ses couples d'états sont jumelés.

Question 3.2.

Soit $\mathcal{T}$ un transducteur sur A et B, où B est un alphabet à une seule lettre. On note $n = \mathrm{Card}(Q)$, où Q est l'ensemble des états du transducteur. *Proposer et justifier un algorithme de complexité $O(n^6)$ prenant en entrée $\mathcal{T}$, et retournant 'oui' si $\mathcal{T}$ vérifie la propriété de jumelage, 'non' dans le cas contraire.* **Indication** *: on pourra construire un graphe valué d'ensemble de nœuds $Q \times Q$ et utiliser les algorithmes de la partie 1.*

On se replace dans le cadre d'alphabets de cardinalité finie quelconque. On va démontrer, en plusieurs étapes, le théorème qui suit.

Théorème de séquentialité (Choffrut, 1978). *Soit f une fonction de A^* dans B^* réalisée par un transducteur $\mathcal{T}$. Les propriétés suivantes sont équivalentes :*

1. *la fonction f est séquentielle*;
2. *la fonction f est à variation bornée*;
3. *le transducteur $\mathcal{T}$ vérifie la propriété de jumelage.*

Question 3.3.

Soit f une fonction séquentielle de A^* dans B^*. Montrer que f est à variation bornée.

On en déduit que la fonction de la question 2.1 n'est pas séquentielle.

Question 3.4.

Les transducteurs obtenus en réponse à la question 2.3 sont notés respectivement $\mathcal{T}_1$ pour celui opérant sur les mots miroirs et $\mathcal{T}_2$ pour l'autre. *Déterminer, en justifiant la réponse, si les fonctions $f_{\mathcal{T}_1}$ et $f_{\mathcal{T}_2}$ sont séquentielles ou non.*

Question 3.5.

On considère quatre mots v_1, v_2, w_1 et w_2 dans A^*. On suppose qu'il existe $x \in A^*$ tel que $(w_1 = v_1 x,\ x w_2 = v_2 x)$ ou $(v_1 = w_1 x,\ x v_2 = w_2 x)$. *Montrer que l'on a, $\forall v_3, w_3 \in A^*$, $d(v_1 v_2 v_3, w_1 w_2 w_3) = d(v_1 v_3, w_1 w_3)$.*

Question 3.6.

Soit $\mathcal{T} = (Q, A, B, T, I, F)$ un transducteur vérifiant la propriété de jumelage. On note M la longueur maximale de l'étiquette de sortie d'un arc, c'est-à-dire $M = \max\{\ |T(p,u,q)|\ :\ (p,u,q) \in \mathrm{dom}(T)\}$. On pose $n = \mathrm{Card}(Q)$. *Pour tout couple de chemins*

$$i \xrightarrow{u|v} p \quad et \quad j \xrightarrow{u|w} q\,,$$

où i et j sont des états initiaux, montrer que $d(v,w) \leq 2n^2 M$.

Question 3.7.

Montrer l'équivalence entre les propriétés 2 et 3 du théorème de séquentialité.

Soit u, v et w des mots tels que $u = vw$, alors par définition $v^{-1}u = w$. Soit U une partie non vide de B^*, on définit $\mathrm{pr}(U) = \{v \in B^* \ : \ \forall u \in U, \ v \text{ préfixe de } u\}$ et on définit $\bigwedge U$ comme l'ensemble des mots de longueur maximale dans $\mathrm{pr}(U)$. On montre aisément que $\bigwedge U$ est réduit à un seul élément et que pour tout $u \in U$, il existe $v \in U$ tel que $u \wedge v = \bigwedge U$.

Soit $\mathcal{T} = (Q, A, B, T, I, F)$ un transducteur. On va définir ci-dessous à l'aide d'un procédé itératif éventuellement infini le quintuplet $\mathcal{T}_\mathrm{s} = (Q_\mathrm{s}, A, B, T_\mathrm{s}, I_\mathrm{s})$. Cette construction se comprend mieux à l'aide de la figure 2, ou encore en l'appliquant directement à la résolution de la question 3.8. L'ensemble Q_s est inclus dans l'ensemble des parties finies non vides de $Q \times B^*$, et $T_\mathrm{s} : (Q_\mathrm{s} \times A \times Q_\mathrm{s}) \to B^*$ et $I_\mathrm{s} : Q_\mathrm{s} \to B^*$ sont des fonctions.

1. On commence par définir $\mathbf{q_0} = \{(i, \mathbf{1}_{B^*}) \ : \ i \in \mathrm{dom}(I)\}$. On pose $\mathbf{q_0} \in Q_\mathrm{s}$, $\mathrm{dom}(I_\mathrm{s}) = \{\mathbf{q_0}\}$ et $I_\mathrm{s}(\mathbf{q_0}) = \mathbf{1}_{B^*}$.
2. Soit $\mathbf{q} = \{(q_1, w_1), \ldots, (q_k, w_k)\}$ un élément déjà construit de Q_s et soit $a \in A$. S'il existe $i \in \{1, \ldots, k\}$ et $r \in Q$, tels que $(q_i, a, r) \in \mathrm{dom}(T)$, alors on définit $\mathbf{q} \cdot a$ et $\mathbf{q} \star a$ comme suit. On pose pour $i \in \{1, ..., k\}$,

$$\begin{array}{rcl} V(q_i, w_i, a) & = & \{(r, v) \in Q \times B^* \ : \ T(q_i, a, r) = v\} \\ W(q_i, w_i, a) & = & \{w_i v \in B^* \ : \ \exists r \in Q, T(q_i, a, r) = v\}\,. \end{array}$$

   Soit $W = \bigcup_i W(q_i, w_i, a)$ et $w = \bigwedge W$. On définit

$$\mathbf{q} \cdot a = \bigcup_i \{(r, w^{-1} w_i v) \ : \ (r, v) \in V(q_i, w_i, a)\}, \quad \text{et} \quad \mathbf{q} \star a = w\,.$$

   On a alors $\mathbf{q} \cdot a \in Q_\mathrm{s}$, $(\mathbf{q}, a, \mathbf{q} \cdot a) \in \mathrm{dom}(T_\mathrm{s})$ et $T_\mathrm{s}(\mathbf{q}, a, \mathbf{q} \cdot a) = \mathbf{q} \star a$.

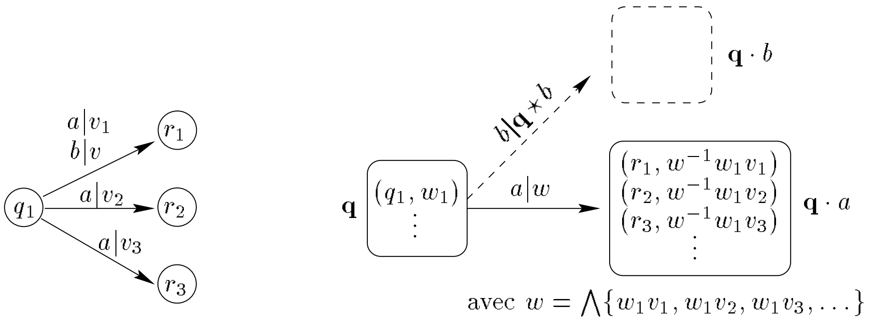


FIG. 2: Une partie d'un transducteur $\mathcal{T}$ et du quintuplet $\mathcal{T}_\mathrm{s}$ associé.

Question 3.8.

En utilisant la construction décrite ci-dessus, obtenir un transducteur séquentiel réalisant la même fonction que le transducteur de la question 2.4.

Question 3.9.

Soit $\mathcal{T}$ un transducteur et soit $\mathcal{T}_{\mathrm{s}}$ le quintuplet associé. Soit $\mathbf{q} \in Q_{\mathrm{s}}$. Montrer que pour tout $(p, v) \in \mathbf{q}$, il existe $(q, w) \in \mathbf{q}$ tel que $v \wedge w = \mathbf{1}_{B^*}$.
On suppose de plus que $\mathcal{T}$ vérifie la propriété de jumelage. Montrer que pour tout $\mathbf{q} \in Q_{\mathrm{s}}$ et tout $(p, v) \in \mathbf{q}$, on a $|v| \leq 2n^2 M$ (où n et M sont définis comme à la question 3.6). En déduire que l'ensemble Q_{s} est fini.

Question 3.10.

Soit $\mathcal{T}$ un transducteur fonctionnel vérifiant la propriété de jumelage. Construire un transducteur séquentiel réalisant la fonction $f_{\mathcal{T}}$.
Proposer et justifier un algorithme prenant en entrée un transducteur $\mathcal{T}$, retournant 'non' si $f_{\mathcal{T}}$ n'est pas une fonction séquentielle et retournant dans le cas contraire un transducteur séquentiel $\mathcal{T}'$ réalisant $f_{\mathcal{T}}$.

On a ainsi démontré, à l'aide des questions 3.3 à 3.10, le théorème de séquentialité. Ce faisant on a obtenu un algorithme permettant de 'séquentialiser' un transducteur (lorsque cela est possible). La complexité de cet algorithme est exponentielle. En effet, lorsque tous les mots de sortie sont égaux au mot vide, cet algorithme se réduit à l'algorithme classique de déterminisation d'un automate dont la complexité est exponentielle.

Si on ne cherche pas à séquentialiser effectivement le transducteur $\mathcal{T}$ mais seulement à décider de la séquentialité de $f_{\mathcal{T}}$ alors ceci peut être réalisé à l'aide d'un algorithme de complexité polynômiale. Dans le cas où l'alphabet des mots de sortie ne contient qu'une lettre, l'algorithme de la question 3.2 permet de décider de la séquentialité lorsqu'il est appliqué à un transducteur fonctionnel. D'autre part, il existe un algorithme basé sur la même construction et de même complexité permettant de tester la fonctionnalité. Lorsque les alphabets sont de cardinalité quelconque, il existe encore des algorithmes polynômiaux permettant de décider de la fonctionnalité et de la séquentialité.