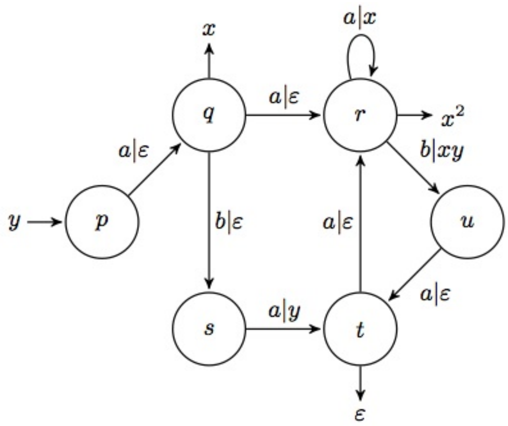
TD 12 : Fonctions séquentielles

Exercice 1 (**À préparer -** Minimisation)**.** Minimiser la fonction séquentielle $\{a,b\}^* \to \{x,y\}^*$ définie par l'automate suivant :



1. Calculer les préfixes minimaux $m_q = \bigwedge [\![A_q]\!](\{a,b\}^*)$ pour chaque état q de l'automate.
2. Donner l'automate normalisé équivalent.
3. Minimiser cet automate par raffinement de partitions.

Exercice 2 (**À préparer -** Résiduels)**.** 1. Montrer que la fonction successeur n'est pas séquentielle sur un codage binaire des entiers commençant par le bit de poids fort.

2. Montrer que la fonction $n \to \lfloor n/3 \rfloor$ n'est pas séquentielle sur un codage binaire des entiers commençant par le bit de poids faible.

Exercice 3 (**À préparer -** Fonction non séquentielle)**.** Soit $B = \{0,1\}$; montrer que la fonction $\cdot^2 : B^* \to B^*$ qui à un entier codé en binaire associe le codage de son carré n'est ni séquentielle ni co-séquentielle (i.e, avec bit de poids faible en premier).

Exercice 4 (Lexical analyzer)**.** Design a lexical analyzer using sequential transducers. It should output each lexeme with its type. You may consider the following lexical grammar :

$$identifier := [a-z]([a-z] \cup [0-9])^*$$
$$keyword := if \mid then \mid else \mid elseif$$
$$number := [1-9]([0-9])^*$$

Other characters, including blank space (␣) may be used as a delimiter.

Exercice 5 (find and replace)**.** How can you extend the sequential transducers to handle 'find and replace' text processing ? For example it should be able to handle
`replace` #1's #2 `with` #2 de #1

Exercice 6 (Calcul de préfixes de langages reconnaissables)**.** Soit $A = \langle Q, \Sigma^*, \delta, I, F\rangle$ un automate fini étendu (ses transitions peuvent être étiquetées par un mot de Σ^*) ; ?on note A_q l'automate ayant q pour unique état initial : $A = \langle Q, \Sigma^*, \delta, \{q\}, F\rangle$. On s'intéresse dans cet exercice au calcul par point fixe de

$$m(q) = \bigwedge L(A_q)$$

pour tous les états q de Q (pour l'ordre préfixe $\leq$ sur Σ^*). On compléte $(\Sigma^*, \geq)$ à l'aide d'un nouvel élément 0, tel que $u \leq 0$ pour tout u de Σ^* ; par convention $|0| = \infty$. Notons $S = \Sigma^* \uplus \{0\}$.

1. Montrer que l'on peut étendre l'opération de concaténation à S en respectant l'ordre préfixe.
2. Montrer que $(S^Q, \geq)$est un ordre partiel complet avec $\bar{0} : q \mapsto 0$ pour plus petit élément.
3. On considère la fonction f de S^Q dans lui-même, définie pour tout m de S^Q et q de Q par
$$(f(m))(q) = (\bigwedge_{(q,u,p)\in\delta} u \cdot m(p)) \wedge \begin{cases} \epsilon & \text{si } q \in F \\ 0 & \text{sinon} \end{cases}$$
Montrer que f est continue sur $(S, \geq)$. Par le théorème du point fixe de Kleene, on en déduit l'existence d'un plus petit point fixe m (pour $\geq$, donc un plus grand point fixe pour $\leq$) tel que $f(m) = m$, obtenu comme la limite de la séquence d'itération de Kleene $(f^i(\bar{0}))_i$.
4. Soit $i \geq 0$; on définit $L^{<i}(A_q)$ comme le sous-ensemble des mots de $L(A_q)$ acceptés par A depuis l'état q en suivant un nombre de transitions strictement inférieur à i. On obtient ainsi $L^{<0}(A_q) = \emptyset$ et $L(A_q) = \cup_{i\geq 0} L^{<i}(A_q)$. Montrer que pour tout $i \geq 0$ et pour tout q de Q,
$$(f^i(\bar{0}))(q) = \bigwedge L^{<i}(A_q)$$
5. Nous avons presque un algorithme : il reste à montrer que $(f^i(\bar{0}))_i$ atteint son point fixe en un nombre fini d'étapes. Nous montrons en fait beaucoup mieux, à savoir qu'il suffit de $2 \cdot |Q|$ itérations.
Supposons A réduit et $n = |Q|$, on note m_q^i pour $(f^i(\bar{0}))(q)$.
   (a) Montrer que $f^n(\bar{0})$ appartient à $(\Sigma^*)^Q$.
   (b) On considère les sous-ensembles P_j de Q définis par récurrence sur $j \geq 1$ par
   $$P_1 = \{p \in Q \mid |m_p^{n+1}| = \min\{|m_q^{n+1}| \mid q \in Q\}\}$$

$$P_{j+1} = \{p \in (Q \setminus \bigcup_{k \leq j} P_k) \mid |m_p^{n+1}| = \min\{|m_q^{n+1}| \mid q \in (Q \setminus \bigcup_{k \leq j} P_k)\}\}$$

Montrer que, pour tout p dans P_j, $m_p^{n+j} = m_p^{n+j+1}$ et conclure que $2n$ est bien une borne supérieure au nombre d'itérations.

6. En considérant l'automate suivant, montrer que la borne de $2 \cdot |Q|$ itérations est atteinte :