Documents permis - Automates **Dr. Jihad Saab**

1. A_d =AFD=$(\Sigma, E, q_0, F, \delta)$: $\Sigma = \{\text{alphabet}\}, E = \{q_0, q_1, \cdots, q_n\}$ ensemble fini d'états, q_0 : état initial, $F \subset E$ ensemble d'états finaux ou acceptants, δ est une transition où chaque symbole de Σ agit d'une façon unique sur chaque état de E càd, $\forall a_i \in \Sigma, \forall j$,il existe un et un seul état q_k tel que $\delta(a_i, q_j) = q_k$

2. On se donne un $A_d = AFD$ on cherche un langage $L = L(A_d)$. Soit on construit le système de départ, soit on construit le système d'arrivée:

   (a) Système de départ:

   1-A chaque état q_i on associe un langage de départ (inconnu) D_i

   2-Si les symboles $\alpha_j,\ j = 1 \cdots k$, font passer q_i vers les états q_j,alors on établit l'équation

$$D_i = \begin{cases} \sum_{j=1}^{k} \alpha_j D_j & \text{si} \quad q_i \text{ n'est pas acceptant} \\ \sum_{j=1}^{k} \alpha_j D_j + \varepsilon & \text{si} \quad q_i \text{ est acceptant} \end{cases}$$

   3-$L(A_d) = D_0$ le langage associé à l'état initial.

   4-La résolution du système de départ se base sur la formule:

$$X = uX + v \text{ alors } X = \begin{cases} u^*v & \text{si} \quad \varepsilon \notin u \\ u^*(v+w) & \text{si} \quad \varepsilon \in u \end{cases} \text{ où } w \text{ est quelconque}$$

   (b) Système d'arrivée:

   1-A chaque état q_i on associe un langage d'arrivée (inconnu) A_i

   2-Si l'état q_i reçoit de flèches de la part des états q_i ,$j = 1 \cdots k$, via les symboles α_j alors on lui associe l'équation

$$A_i = \begin{cases} \sum_{j=1}^{k} A_j \alpha_j & \text{si} \quad A_i \text{ n'est pas initial} \\ \sum_{j=1}^{k} A_j \alpha_j + \varepsilon & \text{si} \quad A_i \text{ est initial} \end{cases}$$

   3-$L(A_d) = \sum A_k$ tel que q_k est un état acceptant

   4- La résolution du système de d'arrivée se base sur la formule:

$$X = Xu + v \text{ alors } X = \begin{cases} vu^* & \text{si} \quad \varepsilon \notin u \\ (v+w)u^* & \text{si} \quad \varepsilon \in u \end{cases} \text{ où } w \text{ est quelconque}$$

3. Algorithme de Gluskov pour la construction d'un AFD, A_d qui reconnait une expression régulière, sachant qu'une expression régulière est une chaîne finie de symboles de Σ dont certains ayant l'exposant $*$ ou $+$. (Par exemple aab^*ab^+)

   1-On linéarise l'expression régulière aab^*ab^+ sera $x_1x_2x_3^*x_4x_5^+$

   2-On déterminer les objets suivants:

   2-1- Premier: Ensemble des x_i pouvant commencer un mot (Premier = $\{x_1\}$)

   2-2- Dernier: Ensemble des x_i pouvant finir un mot (Dernier = $\{x_5\}$)

   2-3-Pour tout x_i on détermine Suivant (x_i) : Ensemble des x_j pouvant suivre x_i comme ds le tableau relatif à l'expression traitée

	Suivant
x_1	x_2
x_2	x_3, x_4
x_3	x_3, x_4
x_4	x_5
x_5	x_5

3-On a la règle suivante: $\omega = \omega_1 \cdots \omega_n \in L(A)$ ssi $\omega_1 \in$ Premier, $\omega_n \in$ Dernier, $\omega_{i+1} \in$ Suivant (ω_i) $\forall i = 1...n-1$.

4-$A = (\Sigma, Q \cup \{i\}, \{i\}, F, \delta)$: On a ajouté un état initial $\{i\}$, $Q = \{x_i\}$, $F =$ Dernier et on ajoute à F l'etat $\{i\}$ lorsque $\varepsilon \in L$ et δ est donnée par

$$\begin{cases} \delta(i, x_k) & = & x_k & \text{si } x_k \in \text{Premier} \\ \delta(x_i, x_j) & = & x_j & \text{si } x_j \in \text{ Suivant}(x_i) \end{cases}$$

5- On remplace les etats x_i par des états q_i et les symboles de transitions x_i par les symboles qui leur sont équivalent dans l'expression régulière.

$$\begin{array}{ccccccc} \to (i) & \xrightarrow{a} & (x_1) & \xrightarrow{a} & (x_2) & \xrightarrow{b} & \overset{\frown b}{(x_3)} \\ & & & & & a \searrow & \downarrow a \\ & & & & \underset{\circlearrowleft b}{((x_5))} & \xleftarrow{b} & \longleftarrow \quad (x_4) \end{array}$$

4. Algorithme de construction d'un $A_d = A^{'}$ qui reconnait le langage L^* à partir d'un $A_d = A = (\Sigma, E, q_0, F, \delta)$ qui reconnait L :

   $1 - A^{'} = (\Sigma, E, q_0, F \cup \{q_0\}, \delta^{'})$

   $2- \delta^{'}$ contient toutes les transitions de δ auxquelles on ajoute d'autres transitions de la façon suivante:

   $$\text{si } q_0 a_j q_j \in \delta \text{ alors } q_i a_j q_j \in \delta^{'} \ \forall q_i \in F$$

5. Un AFN, $A_n = (\Sigma, E, I, F, \delta)$ où $I \subset E$ ensemble des états initiaux, $F \subset E$ ensemble des états finaux, δ est une transition qui permet à un symbole de Σ d'agir de plusieurs façons sur un état de E

   $$\delta(a_i, q_j) = q_k \text{ et } \delta(a_i, q_j) = q_l \text{ avec } q_k \neq q_l$$

6. Noter que le système de départ et le système d'arrivée pour un A_n se font de la même manière que dans le cas d'un A_d

7. Algorithme de déterminisation d'un AFN : A_n est donné, on cherche A_d tel que $L(A_d) = L(A_n)$:

   1-On pose $q_0 = I$

   2-construire le tableau de multiplication: Si $\Sigma = \{a_1, \cdots, a_n\}$, alors le tableau est

δ	q_0	q_1	$\cdots$
a_1	q_1		
$\vdots$	$\vdots$		
a_n			

   où q_1 est obtenu de la façon suivante: soit $q_0 = I = \{p_i, \cdots, p_{i+k}\}$ états initiaux de A_n alors $q_1 = \{\delta(a_1, p_i), \delta(a_1, p_{i+1}), \cdots, \delta(a_1, p_{i+k})\}$ et ainsi de suite .

   3-Les états acceptants de A_d sont les q_l qui contiennent au moins un état acceptant de A_n.

8. Un automate spontanné $A_s = AFN - \varepsilon = (\Sigma \cup \{\varepsilon\}, E, I, F, \delta)$ c'est un AFN où en plus les symboles de Σ, ε agit sur les états de A_s.

   $$\delta(\varepsilon, q_i) = q_j$$

9. Algorithme de déterminisation d'un A_s : On cherche A_d tel que $L(A_d) = L(A_s)$:

   1- $q_0 = I \cup \{$ les états de A_s,recevant une $\varepsilon-$ transition depuis les éléments de $I\}$

   2-Construire le tableau de multiplication mais attention un symbole $a \in \Sigma$ peut prendre la forme $a = \varepsilon^n a \varepsilon^m$, $n, m \in \mathbb{N}$.

   3-Un état acceptant de A_d est l'état qui contient au moins un état acceptant de A_s.

10. Algorithme de transformation d'un A_s en A_n : On cherche A_n tel que $L(A_n) = L(A_s)$:

    1-On regarde l'état k de plus grand nombre recevant au moins une ε-transition. Si de k ne sort aucune flèche, on enlève l'état k et recommence la procedure, sinon, on passe à l'etape 2

    2-Si $l \neq k$ et $(l, \varepsilon, k) \in \delta$ et si $l^{'} \in E$ tel que $(k, x, l^{'}) \in \delta$, pour $x \in \Sigma \cup \{\varepsilon\}$

    $$l \xrightarrow{\varepsilon} k \xrightarrow{x} l^{'}$$

    on crée la transition de A_n : $(l, x, l^{'}) \in \delta(A_n)$, ensemble des transitions de A_n.

    3-On enlève toutes les transitions $l \xrightarrow{\varepsilon} k$

    4-Si k est acceptant, on ajoute l aux états acceptants de A_n.

    Algorithmes de Kleen:

11. Algo1: $A = A_d$ ou A_n ou A_s qui reconnait L, on cherche un A_s qui recnnait L^* :

    1-Reconstruire la même automate A

    2-Ajouter un état accepant S

    3-Envoyer vers S une ε-transition depuis tout état initial de A

    4-Envoyer une ε-transition depuis tout état acceptant vers tout état initial

12. Algo2: Constuction de A tel que $L(A) = L_1 + L_2$

    1-Construire A_i tel que $L(A_i) = L_i,\ i = 1, 2$

    2-Changer la numerotation de A_2 de façon a completer la numerotation de A_1

    3-L'état initial de A est l'ensembles des états initiaux de A_1 et A_2 auxquels on ajoute les états de A_1 et A_2 ayant reçu une ε-transition depuis les états initiaux.

    4-Constrire la table de multiplication

    5-Les états acceptants de A sont ceux qui contiennent au moins un état acceptant de A_1 ou A_2.

13. Algo3: Constuction de A tel que $L(A) = L_1.L_2$

    1-Construire A_i tel que $L(A_i) = L_i,\ i = 1, 2$

    2-Changer la numerotation de A_2 de façon a completer la numerotation de A_1

    3-Ajouter un état non acceptant r

    4-Introduire des ε-transitions depuis les états acceptants de A_1 vers r et de r vers les états initiaux de A_2

    $$F_1 \xrightarrow{\varepsilon} r \xrightarrow{\varepsilon} I_2$$

    5-Rendre non acceptants dans A, les états acceptants de A_1.

14. Algo4: Construction d'un automate $\overline{A}$ reconnaissant $\overline{L} = \Sigma^* - L$ le complémentaire de $L = L(A)$

    Il suffit de mettre les états acceptants dans A comme non acceptants dans $\overline{A}$ et les états non acceptants dans A comme des états acceptants dans $\overline{A}$

15. Algo5: Construction de A qui reconnait $L_1 \cap L_2$

    1-On construit les AFD sous forme complète tel que $L(A_1) = L_1$ et $L(A_2) = L_2$ et on numérote les états de A_2 de façon à completer la numérotation de A_1

    2-L'état initial de A est le couple (i_1, i_2) avec i_k est l'état initial de A_k

    3-La transition δ sur A est donnée par

    $$((q_1, q_2), x, (r_1, r_2)) \in \delta \text{ ssi } (q_1, x, r_1) \in \delta_1 \text{ et } (q_2, x, r_2) \in \delta_2$$

    pour tous q_1, r_1 états de A_1 et q_2, r_2 états de A_2

    4-L'ensemble d'états acceptants F de A est l'ensemble d'états $(p_i, p_j) \in F_1 \times F_2$.

16. Minimisation ou réduction d'un automate A:

    1-On élimine les états qui ne reçoivent aucune transition

    2-On écrit E l'ensemble des états, comme l'union de deux sous ensembles, l'un contient les états acceptants et l'autre contient les non acceptants $E = E_1 \cup E_2$

    3-On cherche les états obtenus par action des symboles de Σ sur les états de E_1 puis de E_2 et on regroupe les états équivalents ensemble, sachant que q_1 est équivalent à q_2 si et seulement si $\forall \alpha_i \in \Sigma,\ \alpha_i(q_1)$ appartient au même sous ensemble E_1 ou E_2 que $\alpha_i(q_2)$.

    4-On reécrit comme union de sous ensembles E_1 et E_2 selon que les états de E_i sont ou ne sont pas stables par action des symboles de Σ

    $$E_1 = E_{11} \cup E_{12} \cup \cdots ; \quad E_2 = E_{21} \cup E_{22} \cup \cdots$$

    où les éléments de chaque sous ensemble sont équivalents

    5-On repete cette itération jusqu'à ne plus pouvoir ajouter de sous ensembles. Finalement $E = G_1 \cup G_2 \cup \cdots \cup G_k$

    6-L'automate minimal, ayant le même langage que A, a k états, $G_1, \cdots, G_k$, l'état initial étant celui qui contient l'état initial de A et tout état contenant un état acceptant de A est un état acceptant pour l'autonate minimal.