Corrigé DM 3, sujet Mines-Ponts

3.
Le langage reconnu par cet automate est celui reconnu par l'automate ayant une transition de A vers B étiquetée par $(a|b)$ et une transition de B vers B étiquetée par $(a|b)^2$, lui-même équivalent au langage à une seule transition de A vers B étiquetée par $(a|b).((a|b)^2)^*$.
4.
Le langage reconnu par cet automate est celui reconnu par l'automate ayant une transition de C vers D avec étiquettes de C vers C étiquetée par a, de C vers D étiquetée par b, de D vers D étiquetée par $a|(ba^*b)$. Ce langage est aussi celui reconnu par l'automate dont la seule transition de C vers D est $a^*b\big(a|(ba^*b)\big)^*$.
5.

```
let aut=(2,[|(0,1);(1,0)|],[|false;true|]);;
```

6.

```
let numero n l = let t=Array.make n (-1) in
    let rec aux l i = match l with
    | [] -> ()
    | x::q -> t.(x) <- i; aux q (i+1) in aux l 0;;
```

7.

```
let etats_accessibles aut = let (n,delta,f)=aut in let etats=ref [] and parcours=Array.make n false in
    let rec aux l = match l with
        | [] -> ()
        | x::q when not(parcours.(x)) -> parcours.(x)<- true; etats:= x:: !etats; let qa,qb=delta.(x) in aux (qa::qb::q)
        | _::q -> aux q in aux [0]; !etats;;
```

Cette fonction a une complexité proportionnelle au nombre d'états ajoutés à la liste `l` des états mis dans la liste de parcours. Il y a autant de tels états que de transitions. Il y a au plus 2 transitions par états (une étiquetée par a et une par b). Il y a au pire $2n$ éléments ajoutés à la liste de parcours. Au total, la complexité est donc en $O(n)$.
8.

```
let partie_accessible aut = let l=etats_accessibles aut in
    let (autn,autd,autf)=aut and n=List.length l in
    let delta=Array.make n (0,0) and tab=numero autn l and f=Array.make n false in
    for i=0 to autn-1 do
        if tab.(i)>-1 then
        begin
            let (qa,qb)=autd.(i) in delta.(autn-1-tab.(i)) <- (autn-1-tab.(qa),autn-1-tab.(qb));
            if autf.(i) then f.(autn-1-tab.(i)) <- true
        end
    done;
    (n,delta,f);;
```

9.
$\varphi(E) = C$, $\varphi(F) = C$ et $\varphi(G) = D$.
10.
$\varphi(H) = C$, $\varphi(I) = C$, $\varphi(J) = D$ et $\varphi(K) = D$.
11.
Si un tel morphisme existe φ, d'après la propriété (1), $\varphi(A) = C$ et, d'après la propriété (4), $\varphi(B) = D$. En vertu de la propriété (3), on devrait avoir $\varphi(\delta_{\mathcal{A}_1}(A, a)) = \delta_{\mathcal{A}_2}(\varphi(A), a)$ donc $\varphi(B) = C$, ce qui contredit que $\varphi(B) = D$. Un tel morphisme ne peut donc exister.
12.
Supposons qu'un tel morphisme φ existe. Alors $\varphi(L) = C$ d'après (1) et $\varphi(M) = D$, $\varphi(N) = C$ d'après (4). En outre, d'après (3), $\varphi(\delta_{\mathcal{A}_5}(M, a)) = \delta_{\mathcal{A}_2}(\varphi(M), a)$ donc $\varphi(N) = D$. Ceci est impossible donc il ne peut exister de morphisme de $\mathcal{A}_5$ vers $\mathcal{A}_2$.
13.
Supposons qu'il existe un morphisme $\varphi : \mathcal{A} \to \mathcal{B}$. Pour tout $q \in Q_{\mathcal{A}}$, $\delta^*_{\mathcal{A}}(q, \epsilon) = q$ et $\delta^*_{\mathcal{B}}(\varphi(q), \epsilon) = \varphi(q)$ donc $\varphi(\delta^*_{\mathcal{A}}(q, \epsilon)) = \delta^*_{\mathcal{B}}(\varphi(q), \epsilon)$. Soit $w \in \Sigma^*$ tel que, pour tout $q \in Q_{\mathcal{A}}$, $\varphi(\delta^*_{\mathcal{A}}(q, w)) = \delta^*_{\mathcal{B}}(\varphi(q), w)$. Soit $\sigma \in \Sigma$. Alors, pour tout $q \in Q_{\mathcal{A}}$, $\delta^*_{\mathcal{A}}(q, \sigma w) = \delta^*_{\mathcal{A}}(\delta_{\mathcal{A}}(q, \sigma), w)$ donc, d'après l'hypothèse de récurrence faite sur w, $\varphi(\delta^*_{\mathcal{A}}(q, \sigma w)) = \delta^*_{\mathcal{B}}\big(\varphi(\delta_{\mathcal{A}}(q, \sigma)), w\big)$ donc, d'après la propriété (3) d'un morphisme, $\varphi(\delta^*_{\mathcal{A}}(q, \sigma w)) = \delta^*_{\mathcal{B}}(\delta_{\mathcal{B}}(\varphi(q), \sigma), w) = \delta^*_{\mathcal{B}}(\varphi(q), \sigma w)$.
Par récurrence, pour tout $q \in Q_{\mathcal{A}}$, pour tout $w \in \Sigma^*$, $\varphi(\delta^*_{\mathcal{A}}(q, w)) = \delta^*_{\mathcal{B}}(\varphi(q), w)$.
Par conséquent, $w \in L_{\mathcal{A}}$ si et seulement si $\delta^*_{\mathcal{A}}(i_{\mathcal{A}}, w) \in F_{\mathcal{A}}$. D'après la propriété (4) d'un morphisme, $\delta^*_{\mathcal{A}}(i_{\mathcal{A}}, w) \in F_{\mathcal{A}}$ si et seulement si $\varphi(\delta^*_{\mathcal{A}}(i_{\mathcal{A}}, w)) \in F_{\mathcal{B}}$; par conséquent, $w \in L_{\mathcal{A}}$ si et seulement si $\delta^*_{\mathcal{B}}(\varphi(i_{\mathcal{A}}), w) \in F_{\mathcal{B}}$ donc, d'après (1), $w \in L_{\mathcal{A}}$ si et seulement si $\delta^*_{\mathcal{B}}(i_{\mathcal{B}}, w) \in F_{\mathcal{B}}$ si et seulement si $w \in L_{\mathcal{B}}$. Ainsi, $L_{\mathcal{A}} = L_{\mathcal{B}}$.
14.
Supposons que $\varphi : \mathcal{A} \to \mathcal{B}$ est un morphisme d'automates ayant même nombre d'états. Comme $\varphi : Q_{\mathcal{A}} \to Q_{\mathcal{B}}$ est une application surjective entre ensembles finis de même cardinal, $\varphi : Q_{\mathcal{A}} \to Q_{\mathcal{B}}$ est bijective. On dispose donc de $\varphi^{-1} : Q_{\mathcal{B}} \to Q_{\mathcal{A}}$. Comme φ^{-1} est bijective, φ^{-1} est surjective (1). En outre, $\varphi(i_{\mathcal{A}}) = i_{\mathcal{B}}$ donc $\varphi^{-1}(i_{\mathcal{B}}) = i_{\mathcal{A}}$ (2). Soit $q \in Q_{\mathcal{B}}$. Alors $q = \varphi(\varphi^{-1}(q))$ donc $q \in F_{\mathcal{B}}$ si et seulement si $\varphi(\varphi^{-1}(q)) \in F_{\mathcal{B}}$ si et seulement si $\varphi^{-1}(q) \in F_{\mathcal{A}}$ (4). Enfin, pour tout $q \in Q_{\mathcal{B}}$, pour tout $\sigma \in \Sigma$, $\varphi(\delta_{\mathcal{A}}(\varphi^{-1}(q), \sigma)) = \delta_{\mathcal{B}}(\varphi(\varphi^{-1}(q)), \sigma)$ donc $\varphi(\delta_{\mathcal{A}}(\varphi^{-1}(q), \sigma)) = \delta_{\mathcal{B}}(q, \sigma)$ donc $\delta_{\mathcal{A}}(\varphi^{-1}(q), \sigma) = \varphi^{-1}(\delta_{\mathcal{B}}(q, \sigma))$ (3). Par conséquent, φ^{-1} est bien un morphisme d'automates.
15.
Soient $f : \mathcal{A} \to \mathcal{B}$ et $g : \mathcal{B} \to \mathcal{C}$ deux morphismes d'automates. Comme $f : Q_{\mathcal{A}} \to Q_{\mathcal{B}}$ et $g : Q_{\mathcal{B}} \to Q_{\mathcal{C}}$ sont surjectives, que la composition de deux surjections est une surjection, $g \circ f : Q_{\mathcal{A}} \to Q_{\mathcal{B}}$ est une surjection (1). $f(i_{\mathcal{A}}) = i_{\mathcal{B}}$ et $g(i_{\mathcal{B}}) = i_{\mathcal{C}}$ donc $(g \circ f)(i_{\mathcal{A}}) = i_{\mathcal{C}}$ (2).

Pour tout $q' \in Q_{\mathcal{B}}$, pour tout $\sigma \in \Sigma$, $g(\delta_{\mathcal{B}}(q',\sigma)) = \delta_{\mathcal{C}}(g(q'),\sigma)$. Pour tout $q \in Q_{\mathcal{A}}$, pour tout $\sigma \in \Sigma$, $f(\delta_{\mathcal{A}}(q,\sigma)) = \delta_{\mathcal{B}}(f(q),\sigma)$ donc $(g\circ f)(\delta_{\mathcal{A}}(q,\sigma)) = g\big(\delta_{\mathcal{B}}(f(q),\sigma)\big) = \delta_{\mathcal{C}}(g(f(q)),\sigma) = \delta_{\mathcal{C}}((g\circ f)(q),\sigma)$ (3). Enfin, pour tout $q' \in Q_{\mathcal{B}}$, $q' \in F_{\mathcal{B}}$ si et seulement si $g(q') \in F_{\mathcal{C}}$ donc pour tout $q \in Q_{\mathcal{A}}$, $q \in F_{\mathcal{A}}$ si et seulement si $f(q) \in F_{\mathcal{B}}$ si et seulement si $g(f(q)) \in F_{\mathcal{C}}$ si et seulement si $(g\circ f)(q) \in F_{\mathcal{C}}$ (4). En conclusion, $g \circ f$ est bien un morphisme d'automates.

16.

Soit $\varphi : \mathcal{A} \to \mathcal{B}$ une application vérifiant les points (2), (3) et (4) et supposons que $\mathcal{A}$ et $\mathcal{B}$ sont accessibles.
Soit $q' \in Q_{\mathcal{B}}$. Comme $\mathcal{B}$ est accessible, il existe $w \in \Sigma^*$ tel que $q' = \delta^*_{\mathcal{B}}(i_{\mathcal{B}},w)$; alors $q' = \delta^*_{\mathcal{B}}(\varphi(i_{\mathcal{A}}),w)$. On a démontré dans la question 13 que $\delta^*_{\mathcal{B}}(\varphi(i_{\mathcal{A}}),w) = \varphi(\delta^*_{\mathcal{A}}(i_{\mathcal{A}},w))$. Ceci prouve que, pour tout $q' \in Q_{\mathcal{B}}$, il existe $q \in Q_{\mathcal{A}}$ tel que $q' = \varphi(q)$ donc que φ est surjective.

17.

```
let existe_morphisme aut1 aut2 = let (n1,delta1,f1)=aut1 and (n2,delta2,f2)=aut2 in
    let phi=Array.make n1 (-1) and boo=ref true in phi.(0) <- 0;
    let rec aux l = match l with
        | [] -> ()
        | x::q -> let (qa,qb)=delta1.(x) and (qpa,qpb)=delta2.(phi(x)) in
            if (phi.(qa)>-1 && phi.(qa)<>qpa)||(phi.(qb)>-1 && phi.(qb)<>qpb) then boo:=false
            else begin phi.(qa)<-qpa; phi.(qb)=qpb; aux qa::qb::q end in aux [0];
    for k=0 to n1-1 do if f1.(k)<>f2.(phi.(k)) then boo:=false done;
    (!boo,phi);;
```

18.

Les états accessibles de $\mathcal{A}_3 \times \mathcal{A}_4$ sont : (E,H), (F,I), (G,J), (F,H), (G,K) ; ses états finaux sont (G,J), (G,K) ; l'état initial est (E,H). Il y a une transition de (E,H) vers (F,I) étiquetée par a, une transition de (E,H) vers (G,J) étiquetée par b, une transition de (F,I) vers (F,H) étiquetée par a, une transition de (F,I) vers (G,K) étiquetée par b, une transition de (G,J) vers (G,K) étiquetée par a, une transition de (G,J) vers (F,H) étiquetée par b, une transition de (F,H) vers (F,I) étiquetée par a, une transition de (F,H) vers (G,J) étiquetée par b, une transition de (G,K) vers (G,J) étiquetée par a et une transition de (G,K) vers (F,I) étiquetée par b.

19.

```
let produit aut1 aut2 = let (n1,delta1,f1)=aut1 and (n2,delta2,f2)=aut2 in
    let n=n1*n2 in let delta=Array.male n (0,0) and f=Array.make n false in
    for k=0 to n1-1 do
        let (qa1,qb1)=delta1.(k) in
        for j=0 to n2-1 do
            let (qa2,qb2)=delta2.(j) in delta.(k*n_2+j)<-(qa1*n2+qa2,qb1*n2+j);
            if f1.(k)&&f2.(j) then f.(k*n2+j)<-true
        done
    done;
    (n,delta,f);;
```

20.

Notons $\mathcal{A}$ et $\mathcal{B}$ les deux automates en jeu. Pour tout $(q,q') \in Q_{\mathcal{A}} \times Q_{\mathcal{B}}$, $\delta^*_{\mathcal{A}\times\mathcal{B}}((q,q'),\epsilon) = (q,q') = \big(\delta^*_{\mathcal{A}}(q,\epsilon), \delta^*_{\mathcal{B}}(q',\epsilon)\big)$. Soit $w \in \Sigma^*$ tel que, pour tout $(q,q') \in Q_{\mathcal{A}} \times Q_{\mathcal{B}}$, $\delta^*_{\mathcal{A}\times\mathcal{B}}((q,q'),w) = \big(\delta^*_{\mathcal{A}}(q,w), \delta^*_{\mathcal{B}}(q',w)\big)$. Soit $\sigma \in \Sigma$. Alors, pour tout $(q,q') \in Q_{\mathcal{A}} \times Q_{\mathcal{B}}$, $\delta^*_{\mathcal{A}\times\mathcal{B}}((q,q'),\sigma w) = \delta^*_{\mathcal{A}\times\mathcal{B}}\big(\delta_{\mathcal{A}\times\mathcal{B}}((q,q'),\sigma),w\big) = \delta^*_{\mathcal{A}\times\mathcal{B}}\big((\delta_{\mathcal{A}}(q,\sigma),\delta_{\mathcal{B}}(q',\sigma)),w\big)$ donc, d'après l'hypothèse de récurrence, $\delta^*_{\mathcal{A}\times\mathcal{B}}((q,q'),\sigma w) = \delta^*_{\mathcal{A}\times\mathcal{B}}\big((\delta_{\mathcal{A}}(q,\sigma),\delta_{\mathcal{B}}(q',\sigma)),w\big) = \big(\delta^*_{\mathcal{A}}(\delta_{\mathcal{A}}(q,\sigma),w), \delta^*_{\mathcal{B}}(\delta_{\mathcal{B}}(q',\sigma),w)\big) = \big(\delta^*_{\mathcal{A}}(q,\sigma w), \delta^*_{\mathcal{B}}(q',\sigma w)\big)$.
Ceci prouve, par récurrence, que, pour tout $w \in \Sigma^*$, $\delta^*_{\mathcal{A}\times\mathcal{B}}((q,q'),w) = \big(\delta^*_{\mathcal{A}}(q,w), \delta^*_{\mathcal{B}}(q',w)\big)$.
(q,q') étant accessible, il existe $w \in \Sigma^*$ tel que $(q,q') = \delta^*_{\mathcal{A}\times\mathcal{B}}\big((i_{\mathcal{A}},i_{\mathcal{B}}),w\big)$; ainsi, $(q,q') = (\delta^*_{\mathcal{A}}(i_{\mathcal{A}},w), \delta^*_{\mathcal{B}}(i_{\mathcal{B}},w))$ donc $q = \delta^*_{\mathcal{A}}(q,w)$ et $q' = \delta^*_{\mathcal{B}}(i_{\mathcal{B}},w)$. Comme $L_{\mathcal{A}} = L_{\mathcal{B}}$, $q \in F_{\mathcal{A}}$ si et seulement si $w \in L_{\mathcal{A}}$ si et seulement si $w \in L_{\mathcal{B}}$ si et seulement si $q' \in F_{\mathcal{B}}$.

21.

Notons $\mathcal{C}$ la partie accessible de $\mathcal{A} \times \mathcal{B}$. Soit $f : Q_{\mathcal{C}} \to Q_{\mathcal{A}}$, $(a,b) \mapsto a$. $f(i_{\mathcal{C}}) = f(i_{\mathcal{A}}, i_{\mathcal{B}}) = i_{\mathcal{A}}$ (2). Soit $(q,q') \in Q_{\mathcal{C}}$. Soit $\sigma \in \Sigma$. Alors $\delta_{\mathcal{C}}((q,q'),\sigma) = \big(\delta_{\mathcal{A}}(q,\sigma),\delta_{\mathcal{B}}(q',\sigma)\big)$ donc $f(\delta_{\mathcal{C}}\big((q,q'),\sigma)\big) = \delta_{\mathcal{A}}(q,\sigma) = \delta_{\mathcal{A}}(f(q,q'),\sigma)$ (3). Enfin, d'après la question précédente, pour tout $(q,q') \in Q_{\mathcal{C}}$, $q \in F_{\mathcal{A}}$ si et seulement si $q' \in F_{\mathcal{B}}$ donc $q \in F_{\mathcal{A}}$ si et seulement si $q \in Q_{\mathcal{A}}$ et $q' \in Q_{\mathcal{B}}$ si et seulement si $(q,q') \in F_{\mathcal{C}}$ donc $f(q,q') \in F_{\mathcal{A}}$ si et seulement si $(q,q') \in F_{\mathcal{C}}$ (4). En outre, les automates $\mathcal{C}$ et $\mathcal{A}$ étant accessibles, la propriété (1) est automatiquement vérifiée. f est donc bien un morphisme d'automates. De même, on montre que $g : Q_{\mathcal{C}} \to Q_{\mathcal{B}}$, $(a,b) \mapsto b$ est un morphisme d'automates.

22.

Soit $p \in Q_{\mathcal{B}}$. Alors $p = q_0, q_0 = p$ est une suite de longueur $0+1$ d'états ; comme $[\![0, 0-1]\!] = \emptyset$, la deuxième condition est vide. Ainsi, $p \equiv p$.
Soit $(p,q) \in Q^2_{\mathcal{B}}$. Supposons que $p \equiv q$. Alors il existe une suite finie de longueur $k+1$: $p = q_0, \cdots, q_k = q$ telle que, $\forall 0 \le j < k$, $\varphi(q_j) = \varphi(q_{j+1})$ ou $\psi(q_j) = \psi(q_{j+1})$. Soit $q'_i = q_{k-i}$. $q'_0 = q, \cdots, q'_k = p$ est une suite finie de longueur $k+1$ et, $\forall 0 \le j < k$, $0 \le k-j-1 < k$ donc $\varphi(q_{k-j-1}) = \varphi(q_{k-j})$ ou $\psi(q_{k-j-1}) = \psi(q_{k-j})$ donc $\varphi(q'_{j+1}) = \varphi(q'_j)$ ou $\psi(q'_{j+1}) = \psi(q'_j)$. Ainsi, $q \equiv p$.
Soit $(p,q,r) \in Q^3_{\mathcal{B}}$ tel que $p \equiv q$ et $q \equiv r$. Il existe deux suites de longueur $k+1$ et $l+1$ notées $p = q_0, \cdots, q_k = q$ et $q = q'_0, \cdots, q'_l = r$ telles que, $\forall 0 \le j < k$, $\varphi(q_j) = \varphi(q_{j+1})$ ou $\psi(q_j) = \psi(q_{j+1})$ et, $\forall 0 \le i < l$, $\varphi(q'_i) = \varphi(q'_{i+1})$ ou $\psi(q'_i) = \psi(q'_{i+1})$. Soit $r_i = q_i$ pour tout $0 \le i \le k$ et $r_i = q'_{i-k}$ pour tout $k+1 \le i \le k+l$; on remarque que l'on a aussi $r_k = q'_{k-k}$. Cette suite a longueur $l+k+1$. Soit $0 \le j < k+l$. Si $j < k$, $\varphi(q_j) = \varphi(q_{j+1})$ ou $\psi(q_j) = \psi(q_{j+1})$ donc $\varphi(r_j) = \varphi(r_{j+1})$ ou $\psi(r_j) = \psi(r_{j+1})$; si $k \le j$, alors $0 \le j-k < l$ donc $\varphi(q'_{j-k}) = \varphi(q'_{j-k+1})$ ou $\psi(q'_{j-k}) = \psi(q'_{j-k+1})$ donc $\varphi(r_j) = \varphi(r_{j+1})$ ou $\psi(r_j) = \psi(r_{j+1})$. Ainsi, pour tout $0 \le j < k+l$, $\varphi(r_j) = \varphi(r_{j+1})$ ou $\psi(r_j) = \psi(r_{j+1})$. Par conséquent, $p \equiv r$.
La relation $\equiv$ est donc une relation d'équivalence sur $Q_{\mathcal{B}}$.

23.

Soit $(p,q) \in Q^2_{\mathcal{B}}$ tel que $p \equiv q$ et soit $\sigma \in \Sigma = \{a,b\}$. Il existe une suite $p = q_0, \cdots, q_k = q$ une suite finie de longueur $k+1$ telle que, $\forall 0 \le j < k$, $\varphi(q_j) = \varphi(q_{j+1})$ ou $\psi(q_j) = \psi(q_{j+1})$. Soit, pour tout $0 \le j \le k$, $q'_j = \delta_{\mathcal{B}}(q_j,\sigma)$. Alors $q'_0 = \delta_{\mathcal{B}}(p,\sigma), \cdots, q'_k = \delta_{\mathcal{B}}(q,\sigma)$ est une suite de longueur finie $k+1$. En outre, $\forall 0 \le j < k$, $\varphi(q_j) = \varphi(q_{j+1})$ ou $\psi(q_j) = \psi(q_{j+1})$ donc $\varphi(q'_j) = \varphi(\delta_{\mathcal{B}}(q_j,\sigma)) = \delta_{\mathcal{A}}(\varphi(q_j),\sigma) = \delta_{\mathcal{A}}(\varphi(q_{j+1}),\sigma) = \varphi(\delta_{\mathcal{B}}(q_{j+1},\sigma)) = \varphi(q'_{j+1})$ ou $\psi(q'_j) = \psi(\delta_{\mathcal{B}}(q_j,\sigma)) = \delta_{\mathcal{A}}(\psi(q_j),\sigma) = \delta_{\mathcal{A}}(\psi(q_{j+1}),\sigma) = \psi(\delta_{\mathcal{B}}(q_{j+1},\sigma)) = \psi(q'_{j+1})$.

Ainsi, $\delta_{\mathcal{B}}(p,\sigma) \equiv \delta_{\mathcal{B}}(q,\sigma)$.
24.
Supposons que $p \equiv q$. Soit $p = q_0, \cdots, q_k = q$ une suite de longueur finie $k+1$ telle que, $\forall 0 \le j < k$, $\varphi(q_j) = \varphi(q_{j+1})$ ou $\psi(q_j) = \psi(q_{j+1})$. Supposons que $p \in F_{\mathcal{B}}$. Alors $q_0 \in F_{\mathcal{B}}$. Soit $0 \le j < k$ tel que $q_j \in F_{\mathcal{B}}$. Alors $\varphi(q_j) \in F_{\mathcal{A}}$ et $\psi(q_j) \in F_{\mathcal{A}'}$; en outre, $\varphi(q_j) = \varphi(q_{j+1})$ ou $\psi(q_j) = \psi(q_{j+1})$ donc $\varphi(q_{j+1}) \in F_{\mathcal{A}}$ ou $\psi(q_{j+1}) \in F_{\mathcal{A}'}$ donc $q_{j+1} \in F_{\mathcal{B}}$ ou $q_{j+1} \in F_{\mathcal{B}}$ donc $q_{j+1} \in F_{\mathcal{B}}$. Par récurrence, pour tout $0 \le j \le k$, $q_j \in F_{\mathcal{B}}$. Par conséquent, $q_k = q \in F_{\mathcal{B}}$. Ainsi, $p \equiv q \Rightarrow (p \in F_{\mathcal{B}} \Rightarrow q \in F_{\mathcal{B}})$ donc $p \equiv q \Rightarrow q \equiv p \Rightarrow (q \in F_{\mathcal{B}} \Rightarrow p \in F_{\mathcal{B}})$; finalement, $p \equiv q \Rightarrow (p \in F_{\mathcal{B}} \Leftrightarrow q \in F_{\mathcal{B}})$.
25.
Soit $Q_{\mathcal{C}}$ l'ensemble des classes d'équivalence de $Q_{\mathcal{B}}$. Soit $i_{\mathcal{C}}$ la classe d'équivalence de $i_{\mathcal{B}}$. Soit $S_i \in Q_{\mathcal{C}}$; soit $(p,q) \in Q_{\mathcal{B}}^2$ tel que $S_i = [p] = [q]$; alors $p \equiv q$; d'après la question 23, pour tout $\sigma \in \Sigma$, $\delta_{\mathcal{B}}(p,\sigma) \equiv \delta_{\mathcal{B}}(q,\sigma)$ donc $[\delta_{\mathcal{B}}(p,\sigma)] = [\delta_{\mathcal{B}}(q,\sigma)]$; ainsi, $[\delta_{\mathcal{B}}(p,\sigma)]$ ne dépend pas du choix de p tel que $S_i = [p]$. On peut donc poser $\delta_{\mathcal{C}}(S_i,\sigma) = [\delta_{\mathcal{B}}(p,\sigma)]$; ceci définit une application $\delta_{\mathcal{Q}} : Q_{\mathcal{C}} \times \Sigma \to Q_{\mathcal{C}}$ tel que, pour tout $q \in Q_{\mathcal{B}}$, pour tout $\sigma \in \Sigma$, $\delta_{\mathcal{C}}([p],\sigma) = [\delta_{\mathcal{B}}(q,\sigma)]$. Enfin, soit $S_i \in Q_{\mathcal{C}}$. Soit $(p,q) \in Q_{\mathcal{B}}^2$ tel que $S_i = [p] = [q]$; alors $p \equiv q$. D'après la question 24, $p \in F_{\mathcal{B}}$ si et seulement si $q \in F_{\mathcal{B}}$. Par conséquent, la propriété "$p \in Q_{\mathcal{B}}$" ne dépend pas du choix de p tel que $S_i = [p]$. On peut donc définir $F_{\mathcal{C}}$ par la propriété : $S_i \in F_{\mathcal{C}}$ si et seulement si $p \in F_{\mathcal{B}}$ lorsque $S_i = [p]$.
De plus, pour tout $q \in Q_{\mathcal{B}}$, $\delta_{\mathcal{C}}^*([q],\epsilon) = [q] = [\delta_{\mathcal{B}}^*(q,\epsilon)]$. Soit $w \in \Sigma^*$ tel que, pour tout $q \in Q_{\mathcal{B}}$, $\delta_{\mathcal{C}}^*([q],w) = [\delta_{\mathcal{B}}^*(q,w)]$. Alors, pour tout $q \in Q_{\mathcal{B}}$, pour tout $\sigma \in \Sigma$, $\delta_{\mathcal{C}}^*([q],\sigma w) = \delta_{\mathcal{C}}^*(\delta_{\mathcal{C}}([q],\sigma),w) = \delta_{\mathcal{C}}^*([\delta_{\mathcal{B}}(q,\sigma)],w)$ donc, d'après l'hypothèse de récurrence, $\delta_{\mathcal{C}}^*([q],\sigma w) = [\delta_{\mathcal{B}}^*(\delta_{\mathcal{B}}(q,\sigma),w)] = [\delta_{\mathcal{B}}^*(q,\sigma w)]$.
Ceci prouve, par récurrence, que, pour $q \in Q_{\mathcal{B}}$, pour tout $w \in \Sigma^*$, $\delta_{\mathcal{C}}^*([q],w) = [\delta_{\mathcal{B}}^*(q,w)]$.
Soit $S_i \in Q_{\mathcal{C}}$. Soit $p \in Q_{\mathcal{B}}$ tel que $S_i = [p]$. $\mathcal{B}$ étant accessible, il existe $w \in \Sigma^*$ tel que $\delta_{\mathcal{B}}^*(i_{\mathcal{B}},w) = p$; par conséquent, $S_i = [p] = [\delta_{\mathcal{B}}^*(i_{\mathcal{B}},w)] = \delta_{\mathcal{C}}^*([i_{\mathcal{B}}],w) = \delta_{\mathcal{C}}^*(i_{\mathcal{C}},w)$ donc S_i est accessible donc $\mathcal{C}$ est accessible.
Enfin, soit $\eta : Q_{\mathcal{B}} \to Q_{\mathcal{C}}$, $p \mapsto [p]$. Alors, par définition de $i_{\mathcal{C}}$, $i_{\mathcal{C}} = [i_{\mathcal{B}}] = \eta(i_{\mathcal{B}})$ (2) ; pour tout $p \in Q_{\mathcal{B}}$, pour tout $\sigma \in \Sigma$, $\delta_{\mathcal{C}}([p],\sigma) = [\delta_{\mathcal{B}}(p,\sigma)]$ donc $\delta_{\mathcal{C}}(\eta(p),\sigma) = \eta(\delta_{\mathcal{C}}(p,\sigma))$ (3) ; enfin, pour tout $p \in Q_{\mathcal{B}}$, par définition de $F_{\mathcal{C}}$, $[p] \in F_{\mathcal{C}}$ si et seulement si $p \in F_{\mathcal{B}}$ donc $\eta(p) \in F_{\mathcal{C}}$ si et seulement si $p \in F_{\mathcal{B}}$. Ainsi, η vérifie (2), (3) et (4). Comme $\mathcal{B}$ et $\mathcal{C}$ sont accessibles, η est un morphisme d'automates.
26.
Soit $p \in \mathcal{A}$. φ est un morphisme d'automates donc est surjective. Il existe donc $q \in Q_{\mathcal{B}}$ tel que $\varphi(q) = p$. Supposons que $q' \in Q_{\mathcal{B}}$ tel que $\varphi(q') = p$. Alors $\varphi(q) = \varphi(q')$ donc $q = q_0$, $q_1 = q'$ est une suite de longueur $1+1$ qui prouve que $q \equiv q'$. Par conséquent, $[q]$ ne dépend pas du choix de q tel que $\varphi(q) = p$. On peut donc poser $\varphi'(p) = [q]$. En particulier, pour tout $p \in Q_{\mathcal{B}}$, $\varphi'(\varphi(p)) = [p] = \eta(p)$ donc $\varphi' \circ \varphi = \eta$.
φ est un morphisme d'automates donc $\varphi(i_{\mathcal{B}}) = i_{\mathcal{A}}$. Ainsi, $\varphi'(i_{\mathcal{A}}) = [i_{\mathcal{B}}] = i_{\mathcal{C}}$ (2). Pour tout $q \in Q_{\mathcal{A}}$, il existe $p \in Q_{\mathcal{B}}$ tel que $q = \varphi(p)$. Alors, pour tout $\sigma \in \Sigma$, $\delta_{\mathcal{A}}(q,\sigma) = \delta_{\mathcal{A}}(\varphi(p),\sigma) = \varphi(\delta_{\mathcal{B}}(p,\sigma))$ donc $\varphi'(\delta_{\mathcal{A}}(q,\sigma)) = \varphi'(\varphi(\delta_{\mathcal{B}}(p,\sigma))) = [\delta_{\mathcal{B}}(p,\sigma)] = \delta_{\mathcal{C}}([p],\sigma) = \delta_{\mathcal{C}}(\varphi'(q),\sigma)$ (3). Enfin, pour tout $q \in Q_{\mathcal{A}}$, il existe $p \in Q_{\mathcal{B}}$ tel que $\varphi(p) = q$; alors $q \in F_{\mathcal{A}}$ si et seulement si $p \in F_{\mathcal{B}}$ si et seulement si $[p] \in F_{\mathcal{C}}$ si et seulement si $\varphi'(q) \in F_{\mathcal{C}}$ (4). Comme $\mathcal{A}$ et $\mathcal{C}$ sont accessibles, φ' est un morphisme accessible. En outre, par définition, $\varphi' \circ \varphi = \eta$
Par symétrie des hypothèses faites sur $\mathcal{A}$ et $\mathcal{A}'$, on définit de même un morphisme d'automate $\psi' : \mathcal{A}' \to \mathcal{C}$ tel que $\psi' \circ \psi = \eta$.
27.

```
let renomme t =
    let tp=Array.make (Array.length t) (-1) and i=ref 0 in
    for k=0 to Array.length t -1 do
        if tp.(k)=-1 then
        begin
        tp.(k)<-!i;
        for j=k+1 to Array.length t -1 do
            if t.(j)=t.(k) then tp.(j)<-!i done;
        i:= !i +1 end done;
    tp;;
```

Cette fonction contient deux boucles imbriquées donc est en $O(n^2)$ où n est la taille du tableau.
28.

```
let relation phi psi = let n=Array.length phi in let eta=Array.make n (-1) in
    for k=0 to n-1 do
        if eta.(k)=-1 then eta.(k)<-phi.(k);
        for l=k+1 to n-1 do
            if phi.(l)=phi.(k) || psi.(l)=psi.(k) then eta.(l)<-eta.(k)
        done
    done;
    renomme eta;;
```

29.
Supposons que $\mathcal{A}$ et $\mathcal{A}'$ sont accessibles et acceptent le même langage. Soit $\mathcal{B}$ la partie accessible de $\mathcal{A} \times \mathcal{A}'$. D'après la partie 4.1., il existe des morphismes d'automates $\varphi : \mathcal{B} \to \mathcal{A}$ et $\psi : \mathcal{B} \to \mathcal{A}'$. D'après la partie 4.2., il existe un automate $\mathcal{C}$ et deux morphismes $\varphi' : \mathcal{A} \to \mathcal{C}$, $\psi' : \mathcal{A}' \to \mathcal{C}$.
30.
D'après la question 18, les états de $\mathcal{B}$, la partie accessible de $\mathcal{A}_3 \times \mathcal{A}_4$ sont : (E,H), (F,I), (G,J), (F,H), (G,K). En outre, $(E,H) \equiv (F,H) \equiv (F,I)$ et $(G,J) \equiv (G,K)$. Les états sont donc 0 qui correspond à la classe de (E,H) et 1 qui correspond à la classe de (G,J). L'état initial est 0. Comme G et K sont finaux, 1 est final. Enfin, il y a une transition de 0 vers 0 étiquetée par a, une transition de 0 vers 1 étiquetée par b, une transition de 1 vers 1 étiquetée par a et une transition de 1 vers 0 étiquetée par b. Quitte à renommer 0 en C et 1 en D, il s'agit de l'automate $\mathcal{A}_2$. Les morphismes φ' et ψ' sont les morphismes des questions 9 et 10.
31.
Soient $\mathcal{A}$ et $\mathcal{A}'$ deux tels automates. D'après la question 29, on peut construire un automate $\mathcal{C}$ et des morphismes $\varphi : \mathcal{A} \to \mathcal{C}$, $\psi' : \mathcal{A}' \to \mathcal{C}$; $\mathcal{C}$ reconnaît donc le langage L. En outre, par minimalité de m_L, le nombre d'états $n_{\mathcal{C}}$ de $\mathcal{C}$ vérifie $n_{\mathcal{C}} \ge m_L$. Comme $\varphi' : Q_{\mathcal{A}} \to Q_{\mathcal{C}}$ et $\psi' : Q_{\mathcal{A}'} \to Q_{\mathcal{C}}$ sont surjectives, $n_{\mathcal{C}} \le \mathrm{card}(Q_{\mathcal{A}}) = m_L$ donc $n_{\mathcal{C}} = m_L$. D'après la partie 3.2., φ' et ψ' sont donc des isomorphismes

d'automates. Par conséquent, $(\psi')^{-1} \circ \varphi'$ est, par composition (cf. 3.2.), un morphisme d'automates de $\mathcal{A}$ vers $\mathcal{A}'$ qui est, comme $\mathcal{A}$ et $\mathcal{A}'$ ont le même nombre d'états, bijectif ; $\mathcal{A}$ et $\mathcal{A}$' sont donc isomorphes.

32.

Soit $\mathcal{A}$ un automate dans $\mathfrak{K}_L$. Soit $\mathcal{M}_L$ un automate à m_L états. Alors il existe un automate $\mathcal{C}$ tel que $\varphi' : \mathcal{A} \to \mathcal{C}$ et $\psi' : \mathcal{M}_L \to \mathcal{C}$ sont des morphismes d'automates. Comme $\mathcal{C}$ reconnaît L, il a, par minimalité, au plus m_L états ; comme $\psi' : \mathcal{M}_L \to \mathcal{C}$ est surjective pour les ensembles d'états, $\mathcal{C}$ a également au plus m_L états ; $\mathcal{C}$ a donc m_L états donc, d'après la partie 3.2., ψ' est un isomorphisme d'automates donc $(\psi')^{-1} \circ \varphi'$ est un morphisme d'automates de $\mathcal{A}$ vers $\mathcal{M}_L$.

33.

On considère l'automate d'états O', Q', R', S', avec état final O', état initial O' transitions $(O', a) \mapsto O'$, $(O', b) \mapsto Q'$, $(Q', a) \mapsto R'$, $(Q', b) \mapsto O'$, $(R', a) \mapsto Q'$, $(R', b) \mapsto S'$, $(S', a) \mapsto Q'$, $(S', b) \mapsto S'$.
L'application $\varphi : O \mapsto O'$, $P \mapsto O'$, $Q \mapsto Q'$, $R \mapsto R'$, $S \mapsto S'$, $T \mapsto Q'$ est alors un morphisme d'automates.

34.

S'il existait un tel automate $\mathcal{A}_6^{Q,R} = \mathcal{C}$ avec un morphisme $\mathcal{A}_6 \to \mathcal{A}_6^{Q,R}$, alors $\delta_{\mathcal{C}}(i_{\mathcal{C}}, b) = \varphi(\delta_{\mathcal{A}}(P, b)) = \varphi(Q) = \varphi(R)$ donc $\delta^*_{\mathcal{C}}(i_{\mathcal{C}}, bb) = \delta_{\mathcal{C}}(\delta_{\mathcal{C}}(i_{\mathcal{C}}, b), b) = \delta_{\mathcal{C}}(\varphi(R), b) = \varphi(\delta_{\mathcal{A}}(R, b)) = \varphi(S) \notin F_{\mathcal{C}}$ mais $\delta^*_{\mathcal{C}}(i_{\mathcal{C}}, bb) = \delta_{\mathcal{C}}(\delta_{\mathcal{C}}(i_{\mathcal{C}}, b), b) = \delta_{\mathcal{C}}(\varphi(Q), b) = \varphi(\delta_{\mathcal{A}}(Q, b)) = \varphi(P) \in F_{\mathcal{C}}$ ce qui est absurde puisqu'un état de $\mathcal{C}$ ne peut être à la fois final et non final.

35.

On peut fusionner S' et R'. Ceci donne un automate à trois états O'', Q'' et R'' avec O'' état final et initial, transitions $(O'', a) \mapsto O''$, $(O'', b) \mapsto Q''$, $(Q'', a) \mapsto R''$, $(Q'', b) \mapsto O''$, $(R'', a) \mapsto Q''$ et $(R'', b) \mapsto R''$. On note $\mathcal{M}_{L_6}$ cet automate.
On a de nouveau un morphisme $\mathcal{A}_6^{O,P} \to \mathcal{M}_{L_6}$ donc, par composition (question 15), un morphisme $\mathcal{A}_6 \to \mathcal{M}_{L_6}$. D'après la question 13, les automates $\mathcal{A}_6$ et $\mathcal{M}_{L_6}$ acceptent le même langage.

36.

```
let table_de_predecesseurs aut = let (n,delta,f)=aut in let table=Array.make_matrix n n false in
    let rec aux l =
        | [] -> ()
        | (p,q)::suite when table.(p).(q) -> aux suite
        | (p,q)::suite -> table.(p).(q)<-true;
    let (pa,pb)=delta.(p) and (qa,qb)=delta.(q) in aux (pa,qa)::(pb,qb)::suite in
    for p=0 to n-1 do for q=0 to n-1 do
        if (f.(p) && not(f.(q)))||(f.(q) && not(f.(p))) then aux [(p,q)] done done; table;;
```

37.

On considère la relation sur $Q_{\mathcal{A}}$ définie par $p \equiv q$ si et seulement si il n'existe pas de chemin de (p, q) vers (p_0, q_0) où $(p_0, q_0) \in (F_{\mathcal{A}} \times Q_{\mathcal{A}} \backslash F_{\mathcal{A}}) \cup (Q \backslash F_{\mathcal{A}} \times F_{\mathcal{A}})$. On montre que c'est une relation d'équivalence.
On considère $Q_{\mathcal{B}}$ l'ensemble des classes d'équivalences de cette relation. On définit $F_{\mathcal{B}}$ par $[p] \in F_{\mathcal{B}}$ si et seulement si $p \in F_{\mathcal{A}}$ (on montre que cette propriété ne dépend pas du représentant p). On pose, pour tout $[p] \in Q_{\mathcal{B}}$, $\sigma \in \Sigma$, $\delta_{\mathcal{B}}([p], \sigma) = [\delta_{\mathcal{B}}(p, \sigma)]$ (on montre que cette définition ne dépend pas du choix du représentant p). Enfin, on pose $i_{\mathcal{B}} = [i_{\mathcal{A}}]$.
On montre également que l'application $[.] : \mathcal{A} \to \mathcal{B}$ est un morphisme d'automate.
Si $\mathcal{B}$ n'est pas minimal, il existe un morphisme d'automate $f : \mathcal{B} \to \mathcal{M}_L$ et $(p, q) \in Q^2_{\mathcal{A}}$ tel que $[p] \neq [q]$ et $f([p]) = f([q])$. Par composition, $f \circ [.]$ est un morphisme d'automates. Ainsi, pour tout $w \in \Sigma^*$, $\delta^*_{\mathcal{M}_L}(f([p]), w) = \delta^*_{\mathcal{M}_L}(f([q]), w)$ donc $f([p]) \in F_{\mathcal{M}_L}$ si et seulement si $f([q]) \in F_{\mathcal{M}_L}$ donc $p \in F_{\mathcal{A}}$ si et seulement si $q \in F_{\mathcal{A}}$ donc $[p] = [q]$, ce qui est absurde.
Passons à la mise en oeuvre en Caml. Commençons par écrire une fonction analogue à celle de la question précédente, mais inversant les chemins cherchés.

```
let liste_de_successeurs aut = let (n,delta,f)=aut in let table=Array.make_matrix n n false in
    let antecedent_a=Array.make n [] and antecedent_b=Array.make n [] in
    for k=0 to n-1 do
        let (qa,qb)=delta.(k) in
        antecedent_a.(qa)<-k::antecedent_a.(qa); antecendent_b.(qb)<-k::antecedent_b.(qb) done;
    let rec produit_liste l1 l2 = match l1 with
        | [] -> []
        | x::q -> (aux1 x l2)@(produit q l2)
    and aux1 x l = match l with
        | [] -> []
        | y::q -> (x,y)::(aux1 x q) in
    let rec aux l =
        | [] -> ()
        | (p,q)::suite when table.(p).(q) -> aux suite
        | (p,q)::suite when table.(p).(q) -> table.(p).(q)<-true;
let la=produit_liste antecedent_a.(p) antecedent_a.(q) and lb=produit_liste antecedent_b.(p) antecedent_b.(q) in
aux (la@lb@suite) in for p=0 to n-1 do for q=0 to n-1 do
            if (f.(p) && not(f.(q)))||(f.(q) && not(f.(p))) then aux [(p,q)] done done;
    table;;
```

On peut désormais passer à la fonction principale.

```
let reduit aut = let (n,delta,f)=aut in let phi=Array.make n (-1) and phi_inv=Array.make n (-1) in
    let t=liste_de_successeurs aut and n_red=ref 0 in
    for k=0 to n-1 do
        if phi.(k)=-1 then begin
            phi_inv.(!n_red)<-k;
            n_red:=!n_red+1;
            phi.(k)<-k;
```

```
        for j=k+1 to n-1 do
            if not(t.(k).(j)) then phi.(j)<-k
        done end done;
let delta_red=Array.make (!n_red) (-1,-1) and f_red=Array.make (!n_red) false in
for k=0 to !n_red-1 do
    let (qa,qb)=delta.(phi_inv.(k)) in delta_red.(k)<-(phi.(qa),phi.(qb));
    if f.(phi_inv.(k)) then f_red.(k)<-true done;
(!n_red,delta_red,f_red);;
```