

# Extraction des formes dérivées des mots arabes par des automates déterministes

Jamal JAÏT\*, Abderrahim El Qadi\*\*, Driss Aboutajeddine\*

\* GSCM-LRIT, Faculté des Sciences Université Med V, Rabat, Maroc.

Email: jamal\_info@yahoo.fr aboutaj@ieee.org

\*\* GSCM-LRIT, EST Meknès, Maroc. Email: elqadi\_a@yahoo.com

**Résumé**— Etant très complexe et très fluctuée, la langue arabe nécessite un outil de Racinisation robuste, pour une meilleure performance en terme d'indexation et de recherche d'information. La plupart des études d'indexation et de recherche utilisent, pour la reconnaissance des racines des mots, une transcription plus une troncature de plus longs suffixes et préfixes. Ceci, conduit aux pseudo-racines, dans la majorité des cas, ne correspondent pas à leurs mots d'origines et ne permettent pas de les retrouver. L'analyseur morphologique que nous avons élaboré, basé sur des automates déterministes à états finis reconnaissant la langue arabe. Ce système permet d'extraire des formes dérivées des mots arabes, et de reconnaître les mots correctement écrits, ainsi que leurs étiquettes.

**Abstract**— Being very complex and very fluctuate, the Arabic language requires a robust tool stemming, for better performance in terms of indexing and information retrieval. Most studies of indexing and searching using, for recognizing the roots of words, a transcript and add a truncation longer suffixes and prefixes. This leads to pseudo-roots, in most cases do not match their backgrounds and words can not be found. . In this work we present the morphological analyzer that we developed based on deterministic automata finite state recognition of the Arabic language. This system can extract derived forms of Arabic words and to recognize the words correctly written, and their labels.

**Mots clés**—TAL; Automate à états finis ; Racinisation ; Analyseur morphologique ; Automate reconnaissant la langue arabe

**Keywords**—TAL, finite state automaton, Stemming, morphological analyzer, Automate recognizing Arabic.

## I. INTRODUCTION

Le traitement automatique des langues (TAL) et la recherche d'information (RI) sont deux disciplines dont l'interaction s'est renforcée ces dernières années. Le traitement automatique des langues s'intéresse aux traitements automatisés qui mettent en jeu des techniques linguistiques: analyse de texte, traduction automatique, ...etc. L'objectif est la représentation des données textuelles à différents niveaux de compréhension (morphologique, syntaxique,...).

Dans le contexte de l'augmentation importante des documents numériques, notamment dans le cadre du

WEB, le défi réel concerne la recherche et le filtrage des informations réellement pertinentes.

L'indexation à l'aide d'une base de connaissances ou l'indexation à vocabulaire contrôlé avec les termes d'indexation les plus complets et précis, permet d'atteindre une meilleure performance pour un SRI.

Le processus de Racinisation est un processus qui permet de prendre en entrée un mot, pour en tirer la racine d'où il est dérivé. Ceci suppose, bien sûr, que chaque mot ait une racine dans l'ensemble de mots que génère le langage considéré.

Plusieurs travaux ont été réalisés dans cet axe de recherche. Et ce pour différentes langues. Chaque travail adopte une approche selon le contexte où elle s'inscrit.

Les uns adoptent une approche morphologique légère [1] ou lourde, se basant sur les travaux qui ont été effectués selon une approche qui se base sur l'élimination des plus longs préfixes et ou suffixes [2] [3]. Kazem TAGHVA dans [4] a modéré le stemmer de Khoja [2] pour une équivalente performance pour les systèmes de recherche d'information.

D'autres adoptent des méthodes statistiques qui présentent une indépendance vis-à-vis du langage traité [5]. Ces méthodes utilisent les n-grammes pour regrouper en classe les mots ayant la même racine. Ces classes peuvent être raffinées en utilisant des méthodes de classification sur les mots.

Khalid Alsamara et Martha Evens [6] ont élaboré un analyseur morphologique pour reconnaître les noms arabes, en se basant sur les propriétés du genre et du nombre et leurs correspondances en suffixes et préfixes.

Beesley [7] a créé une machine à états finis, à base de lexique existant, qui permet de reconnaître les mots arabes. Ainsi, pour chaque racine, Beesley associe une machine qui permet de reconnaître les dérivées qui lui sont liées.

L'objectif des travaux présentés dans cette contribution est plus précisément l'extraction des formes dérivées des mots arabes, tout en illustrant les règles de productions des automates reconnaissant des différents schèmes des termes arabes.

Dans la suite de notre article nous exposerons la théorie des langages et des automates DETERMINISTES A ETATS FINIS, et en particulier le langage arabe, et nous présenterons ensuite L'ANALYSEUR MORPHOLOGIQUE QUE NOUS AVONS DEVELOPPE, BASE SUR DES REGLES DE PRODUCTION DES AUTOMATES RECONNAISSANT DES FORMES ARABES. Dans la partie IV, nous détaillerons la mise en oeuvre de ce système D'identification et de la reconnaissance des mots arabes sans connaissance préalable de la racine, ainsi que les résultats des expérimentations avant de conclure. Théorie des langages et des automates<sup>1</sup>.

## II. THEORIE DES LANGAGES ET DES AUTOMATES DETERMINISTES A ETATS FINIS

Soit L un langage sur un alphabet de symboles  $\Sigma$ .

Le problème : peut-on décrire une petite machine qui acceptera ou rejettera une chaîne w de  $\Sigma^*$  selon qu'elle appartient, ou non, au langage L ?

Un modèle du genre est un automate distributeur de café, qui fait un test d'appartenance sur la combinaison de pièces de monnaie, introduites par l'utilisateur, dans l'ensemble des combinaisons acceptées. Il répond en distribuant la boisson choisie dans le cas d'acceptation, ou par un message d'erreur dans le cas contraire.

- Un automate en théorie est écrit en précisant :
  - Son alphabet de symboles (les pièces de monnaie),
  - Ses états initiaux,
  - Ses états finaux (café, avec ou sans sucre, chocolat, ...),
  - Les états intermédiaires dans lesquels il peut se trouver pendant l'interaction avec le demandeur.
  - Ainsi que les transitions, c'est-à-dire les règles de passage d'un état à un autre.

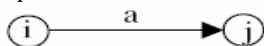
Un automate est défini par un 5-uplet :

$A = (\Sigma, E, E_0, F, \delta)$ , où :

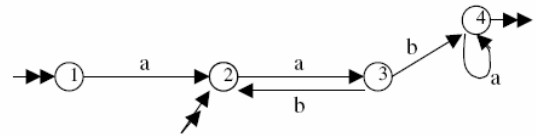
- $\Sigma$  est l'ensemble fini des symboles
- E est un ensemble fini : l'ensemble des états
- $E_0 \subset E$  est le sous-ensemble des états initiaux
- $F \subset E$  : est le sous-ensemble des états finaux
- $\delta$  est un ensemble fini de transitions

Une transition, ou production, est un triplet (i, a, j), où i et j sont des états et a est un symbole.

Une transition (i, a, j) se représente par une flèche de i à j, munie d'une étiquette a.



Exemple : automate reconnaissant une famille des mots binaires.



Pour cet automate, l'alphabet  $\Sigma = \{a=0, b=1\}$ , l'ensemble des états  $E = \{1, 2, 3, 4\}$ , parmi ceux-ci, il y a deux états initiaux et un état final :  $E_0 = \{1, 2\}$  et  $F = \{4\}$ . Les transitions sont décrites dans l'ensemble  $\delta = \{(1,a,2), (2,a,3), (3,b,2), (3,b,4), (4,a,4)\}$ .

Cet automate reconnaît tout mot écrit en binaire ayant u = 0 ou u = 01 comme préfixe suivi d'une suite de « 01 » et v = 10\*2 comme suffixe. Par exemple le mot 0010101000<sub>2</sub>.

Les règles de production de cet automate peuvent être écrites comme suit [8] :

$S \rightarrow aABbC \mid BbC$

$A \rightarrow abB$

$B \rightarrow abB \mid \lambda$

$C \rightarrow 0C \mid \lambda$

$a(0$

$b(1$

Où  $\lambda$  signifie la transition vide.

- Langage reconnu par un automate :

Une chaîne  $w \in \Sigma^*$  est reconnue par un automate A s'il y a un cheminement, c'est-à-dire une suite de transitions, partant d'un état initial, aboutissant à un état final, et dont la suite des étiquettes est w.

Le langage reconnu par un automate A est l'ensemble de toutes les chaînes reconnues. Il est noté L(A).

## III. LANGAGE ARABE

Le langage arabe est l'ensemble des chaînes écrites en alphabet arabe.

**But :** définir un  $\lambda$ -Automate reconnaissant les mots du langage arabe écrit correctement par l'Homme.

### A. Problématique

Pour introduire la problématique dans laquelle s'inscrit ce travail, analysons les mots suivants : لالوان, الاستشعار, بنتاغون.

La plupart des études d'indexation et de recherche utilisent, pour la reconnaissance des racines de ces mots, une transcription plus une troncature de plus longs suffixes et préfixes, ce qui conduit aux pseudo-racines suivants : نتاغ, الو, شعار. On remarque, alors, que les pseudo-racines trouvés, dans la majorité des cas, ne correspondent pas à leurs mots d'origines et ne permettent pas de les retrouver.

Le problème réside essentiellement dans les mots qui ne sont pas d'origine arabe ou qui ne suivent pas un schème (wazn) : mots étrangers ou mots spécifiques. Selon la vision proposée, la seule solution capable de résoudre ce problème est la Racinisation basée sur des outils linguistiques.

<sup>1</sup> Ces notes de mathématiques sont prises de [8]

<sup>2</sup> le terminal 0 se produit plusieurs fois

### B. Etude linguistique

Les mots arabes se décomposent en trois grandes catégories : pronoms (al horouf), les noms (al asma-a) et les verbes (al af3al) [9].

#### 1) Les verbes :

Les verbes se décomposent en deux catégories :

- Non dérivable (jaamid) : نعم, حبذا ليس, ...
- Dérivables (Motassarrif) : علم, جدد, ...

La dérivation des verbes de la 2<sup>ème</sup> catégorie peut donner naissance à d'autres verbes :

- ✓ اكتب -- تكتب -- كاتب ( donner )

Ainsi qu'à des noms :

- ✓ كاتب كاتبة مكاتب ( donner )

#### 2) Les pronoms :

Les pronoms sont non dérivables, limités en un nombre restreint.

#### 3) Les noms

Un nom en arabe est :

- fixé : considéré comme racine

Par exemple : قمر -- شمس (soleil – lune)

- ou obtenu par dérivation d'un verbe. Par exemple : صامد (persistant) tiré de صمد (persister).

### C. Principe de dérivation

En langue arabe, les verbes suivent différents schèmes : 21 schèmes [9]. De chacun d'eux se dérive un ensemble d'autres schèmes correspondants aux : nom du sujet, nom du complément, schème d'exagération, substantif...etc.

Exemple : la racine fa3ala

TABLE 1.

FAMILLE FA3ALA

fa3ala	
1. af3aal	Pluriel de fi3l
2. fa33aal	Forme exagérée
3. fa3oul	Forme exagérée
4. mof3il	Nom sujet
5. fa3aa il	Pluriel fa3il
6. faa3ala	Verbe
7. mofaa3il	Nom sujet fa3ala
8. fi3aal	Substantif
9. Af3al	Nom de préférences
10. fo3laa	Féminin af3al
11. Infi3aal	Substantif
12. faa3il	Non sujet
13. maf3oul	Nom de complément
14. tafaa3ala	Verbe
15. motafaa3il	Non sujet
16. fi3l	Substantif

17. taf3iil	Substantif
18. af3aal	Pluriel fi3l
19. Infa3ala	Verbe
20. Faa3oul	Nom d'outils
21. Fawaa3iil	Pluriel cassé
22. Fa3laa	Forme de préférences.
23. Fawaa 3il	Pluriel faa3ilat
24. Istaf3ala	Verbe
25. Istif3aal	Substantif
26. Mostaf3al	Nom complément
27. Mostaf3il	Nom sujet
28. Ifta3ala	Verbe
29. Ifti3aal	Substantif
30. Mofta3il	Nom sujet
31. Mofta3al	Nom complément

Pour pouvoir lire et comprendre les schèmes du Tableau 1, on adopte la transcription suivante, des lettres arabes :

TABLE 2.  
TRANSCRIPTION DES LETTRES ARABES

Lettre	Code
Alif avec hamza au dessus et fatha	A au debut du mot
Alif avec hamza ou fatha au dessous	I
Fatha	a après une lettre
Kasra	i après une lettre
La lettre ain	3
La lettre taa	t
La lettre faa	f
La lettre Waw après le signe Damma	ou
Kasra suivi de la lettre yaa	li
Alif mamouda	aa
Alif maqsoura	aa à la fin du mot
Tadiif	Deux lettres égales successives (p. ex. : ڤ = 33)
Hamza avec Kasra	_i
Hamza sur la ligne	_a
Tah marabouta	T à la fin du mot
Qaf	Q

Le principe du processus de « Racinisation » abordé, correspond à la représentation de chaque forme par un automate adéquat.

#### IV. AUTOMATE RECONNAISSANT LE LANGAGE ARABE

Soit  $\Sigma$  = l'alphabet arabe  
 $\Sigma_c = \{\text{alef, yaa, waw, alef Maqsoura, tah Marbouta}\} = \{\text{ا ي و ؤ ة}\}$  : l'ensemble des lettres communes aux mots arabes

$\Sigma_m = \{\text{alef avec Hamza, noon, yaa, taa}\} = \{\text{آ ن ي ت}\}$  : lettres introduisant le temps présent.

Soit LA = langage arabe.

Et L0, L1 et L2 les trois sous langages de LA définis comme suit :

- L0 = {m ∈ LA / racine(m) est constitué de lettres li appartenant à  $\Sigma - \Sigma_c$ }
- L2 = {m ∈ LA / m est spécifique}
- L1 = LA - L0 U L2

Il est facile de constater que L0, L1, L2 définissent une partition de LA :

- L0 ∩ L1, L0 ∩ L2 et L1 ∩ L2 sont vide
- LA = L0 U L1 U L2

##### 1) Construction de L0 :

Soit L0 = L3 U L4

Où :

L3 = {m ∈ L0 / |racine(m)|=3}

L4 = {m ∈ L0 / |racine(m)|>3}

Où : |m| = longueur du mot m.

L3 est engendré par les automates correspondants au schème fa3ala et ses dérivés cités dans Tableau 1.

L4 est engendré par les automates correspondants au schème fa3lala et ses dérivés.

##### 2) Construction de L3

Soit fi un élément de F\_Fa3ala, et A (fi) l'automate correspondant.

L3 =  $\bigcup_{fi \in F\_Fa3ala} L(A(fi))$

F\_Fa3ala est la famille des dérivées de la racine fa3ala.

On a : **F\_Fa3ala** = { mof3il, fa3aa il, faa3ala, mofaa3il, fi3aal, Af3al, fo3laa, Infi3aal, faa3il, maf3oul, tafaa3ala, motafaa3il, fi3l, taf3iil, af3aal, Infa3ala, Faa3oul, Fawaa3iil, Fa3laa, Fawaa 3il, Istaf3ala, Istif3aal, Mostaf3al, Mostaf3il, Ifta3ala, Ifi3aal, Mofa3il, Mofa3al }

Les schèmes (awzaane) contenus dans F\_Fa3ala contiennent des verbes comme fa3ala et infa3ala ainsi que des noms comme faa3il et infi3aal.

Pour chaque schème de verbe on associe trois automates A\_fverbe\_present, A\_fverbe\_passé et A\_fverbe\_Amr<sup>3</sup> ; où fverbe prends ses valeurs dans les schèmes de verbes contenus dans la famille F\_Fa3ala.

Pour les noms on associe un seul automate A\_fnom ; où fnom prends ses valeurs dans les schèmes de noms contenus dans la famille F\_Fa3ala.

##### Notation :

Si A est un automate, alors L(A) est le langage engendré par A.

Ainsi :

L3 =

$\{ \bigcup_{fi \in F\_Fa3ala\_Verbe} L(A\_fi\_present) / fi \in F\_Fa3ala\_Verbe \}$

$\cup \{ \bigcup_{fi \in F\_Fa3ala\_Verbe} L(A\_fi\_passé) / fi \in F\_Fa3ala\_Verbe \}$

$\cup \{ \bigcup_{fi \in F\_Fa3ala\_Nom} L(A\_fi) / fi \in F\_Fa3ala\_Nom \}$

Où :

F\_Fa3ala\_Verbe = les verbes de F\_Fa3ala

F\_Fa3ala\_Nom = les noms de F\_Fa3ala\_Nom.

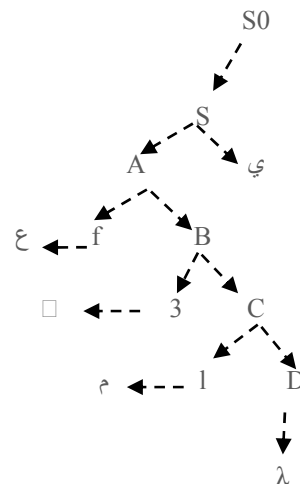
Soit l'automate A\_fa3ala\_present qui permet de reconnaître les mots arabes, suivants le schème fa3ala, conjugués au présent. Les règles de production de cet automate sont définies comme suit :

S0	→	فS1   سS1   س   س   س   س   س
S1	→	سS
S	→	تA   يA   نA   أA
A	→	fB
B	→	3C
C	→	لD
f	→	lettres li appartenant à $\Sigma - \Sigma_c$
3	→	lettres li appartenant à $\Sigma - \Sigma_c$
ل	→	lettres li appartenant à $\Sigma - \Sigma_c$
D	→	لD1   وD1   ؤD2   ةD3   ةD5   λ
D1	→	نD1   ةD5   ةD5   λ
D2	→	لD6   نD4   ةD5   ةD5
D3	→	ةD5   ةD5   λ
D4	→	ةD5   ةD5   λ
D5	→	ةD8   نD7   مD7   λ
D6	→	λ
D7	→	لD9   λ
D8	→	نD11   مD10   λ
D9	→	λ
D10	→	لD9   λ
D11	→	λ

Où λ signifie la transition vide.

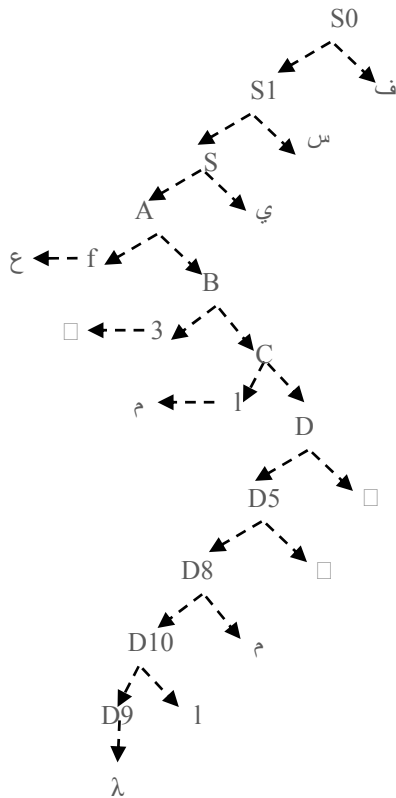
Exemples de dérivation :

- Exemple 1 :



<sup>3</sup> Al-Amr est le temps qui désigne l'impératif

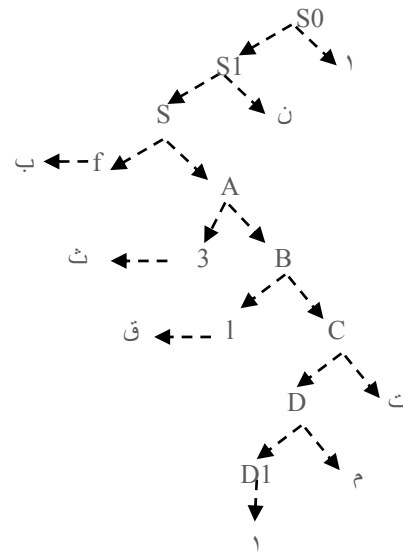
Exemple 2 :



S0	→	S1
S1	→	س
S	→	fA
A	→	3B
B	→	lC
C	→	وD2   D   D3   λ
f	→	lettres li appartenant à $\Sigma - \Sigma_c$
3	→	lettres li appartenant à $\Sigma - \Sigma_c$
l	→	lettres li appartenant à $\Sigma - \Sigma_c$
D	→	مD1   ن
D1	→	وD2   ا
D2	→	ا

Notons que les verbes sont intransitifs suivant ce schème, ce qui implique qu'ils ne prennent pas de suffixes correspondant aux compléments comme les verbes transitifs (ex. نلمسه nous le touchons).

Exemple de dérivation :



Ainsi, on pourra reconnaître toutes les formes dérivées de la racine fa3ala conjuguées au présent, de plus simple dans l'exemple 1 « يعلم » qui veut dire : il sait» au plus complexe dans l'exemple 2 « فسيعلكمهما » ou sous sa forme voyelle « فسيعلكمهما » qui veut dire : il va vous les apprendre»

Comme pour le temps présent, le temps passé (comme pour le temps impératif « Al-Amr ») est représenté par des automates adéquats.

Prenons par exemple le schème du verbe infa3ala (ex : انخرط qui veut dire s'adhérer):

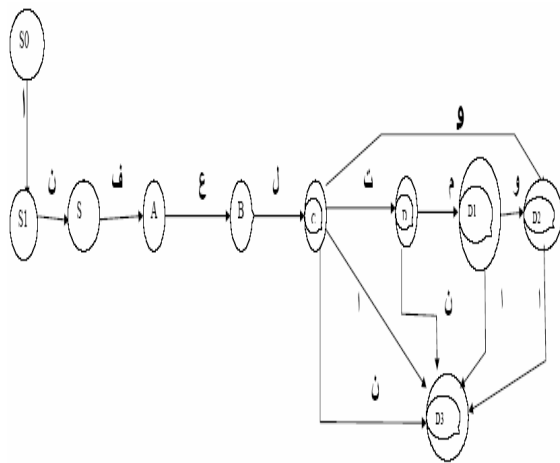


Figure 1. A (infa3ala\_passé)

Cet automate est défini par ses règles de production suivantes :

3) Construction de L4

$$L4 = \bigcup_{f \in F_{Fa3ala}} L(A(f))$$

Où :

$F_{Fa3ala} = \{fa3lala, mofa3lil, mofa3lal, fa3lalat, tafa3lala, motafa3lil, tafa3lol, if3anlala, mof3anlil, if3inlaal, if3alalla, mof3allil, if3illal\}$

4) Construction de L1

L1 est le sous langage de LA constitué de tous les schèmes ayant une racine constituée d'un mélange de consonnes et de lettres communes ( $\Sigma_c$ ).

Par exemples :

❖ وعد, فاق, تلا, وقى, عوى

Ces verbes suivent le schème fa3ala, mais sont irréguliers en dérivation.

Exemples :

TABLE 3.  
VERBE WA3ADA « PROMETTRE »

promettre	وعد		
Promets	أعد	Je	أنا
Promet	يعد	Il	هو
Promet	تعد	Elle	هي
Promettent	يعدان	Ils/elles	هما
Promettent	يعدون	Ils / elles	هم
Promettent	يعدن	elles	هن
Promets	تعد	Tu	أنت
Promets	تعدين	Tu	أنتِ
Promettez	تعدان	Vous	أنتما
Promettez	تعدون	Vous	أنتم
Promettez	تعدن	Vous	أنتن
promettons	نعد	nous	نحن

TABLE 4.  
VERBE WAQAA « PROTÉGER »

Protéger	وقى		
Protège	أقي	Je	أنا
Protège	يقي	Il	هو
Protège	تقي	Elle	هي
Protègent	يقيان	Ils/elles	هما
Protègent	يقون	Ils / elles	هم
Protègent	يقين	elles	هن
Protèges	تقي	Tu	أنت
Protèges	تقين	Tu	أنتِ
Protégez	تقيان	Vous	أنتما
Protégez	تقون	Vous	أنتم
Protégez	تقين	Vous	أنتن
Protégeons	نقي	nous	نحن

On remarque qu'à la dérivation, la lettre faa de la racine de وعد disparaît complètement.

Pour وقى, la lettre faa(و) disparaît, et la lettre laam(ي) disparaît aussi dans ce cas, ou se convertit en yaa (ي) dans d'autres.

Ceci conduit à la conclusion suivante :

Ses verbes ne peuvent pas être vus comme des verbes à racine trilatérale fa3ala. Chose qui renforce notre choix de considérer le sous langage L1 qui est généré par les familles des schèmes du tableau5.

Notons que les verbes comportant une ou deux lettres communes, sont dits des verbes « naqissa », et ils se décomposent en cinq groupes : mitaal مثل – ajwaf أجوف - naaqis ناقص – lafif mafrouq لفيف مفروق – lafif maqroun لفيف مقرون.

Ces verbes doivent être mieux représentés en leurs associant de nouveaux schèmes [9].

TABLE 5.  
SCHÉMES DES VERBES « NAQISSA »

Verbe	وعد	عوى	وقى	تلا	فاق
Type	Mitaal	Lafif maqroun	Lafif mafrouq	Naaqis	Ajwaf
Forme initiale	Fa3ala	Fa3ala	Fa3ala	Fa3ala	Fa3ala
Forme proposée	Wa3ala	Falaa	Wa3aa	Falaa	Faala

Ainsi, et comme pour les schèmes fa3ala et fa3lala, on construit les familles F\_Wa3ala, F\_Faala, F\_Falaa, F\_Wa3aa et F\_Fawaa.

TABLE 6.  
FAMILLE DES VERBES « NAQISSA » OU DÉFECTUEUX

Famille	f <sub>i</sub>
F_Wa3ala	Waa3il, maw3oul, wa3l, mii3aal, maw3il, mawaa3iil, waa3ala, mowaa3il, mowaa3al, tawa33ala, motawa33il, motawa33al, taw3iil, tawaa3ala, tawaa3ol, motawaa3il
F_Faala	Faala, faa_il, fawl, mafoul, mafaal, afaawiil, faawala, mofaawil, mofaawal, tafaawala, motafaawil, tafaawol, afaala, mofiiil, mofaal, ifaalat, istafaala, mostafiiil, istifaalat
F_Falaa	Falaa, faal, faalii, maflow, filw, filaa_a, aflaa, moflii, moflaa, iflaa_a, istafalaa, mostaflii, mostafalaa, istifalaa_a
F_Wa3aa	Wa3aa, waa3, waa3ii, mowa33aa, wi3aa_a wi3aayat, wa3y, taw3iat,
F_Fawaa	Fawaa, faaw, faawii, fawwaa_a, fowwaa_a, fawiat, fowaat, fay, mofaawiat

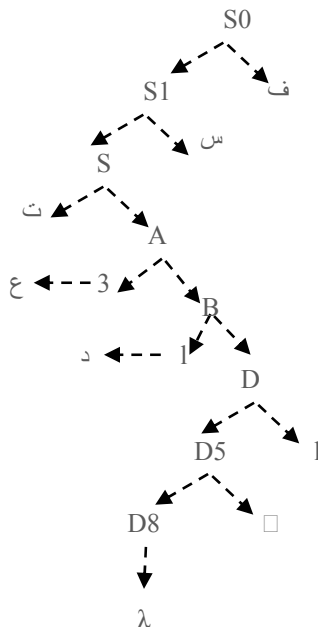
Notation :

F\_Naqissa = F\_Wa3ala U F\_Faala U F\_Falaa U F\_Wa3aa U F\_Fawaa

Exemple: A\_Wa3ala\_Present

S0	→	فS1   سلS1   سS   سS   فS   S
S1	→	سS
S	→	تA   يA   نA   أA
A	→	3B
B	→	لD
3	→	lettres li appartenant à $\Sigma - \Sigma_c$
ل	→	lettres li appartenant à $\Sigma - \Sigma_c$
D	→	وD1   يD1   وD2   وD3   كD5   λ
D1	→	نD1   وD5   كD5   λ
D2	→	وD6   نD4   وD5   كD5
D3	→	وD5   كD5   λ
D4	→	وD5   كD5   λ
D5	→	وD8   نD7   مD7   λ
D6	→	λ
D7	→	وD9   λ
D8	→	نD11   مD10   λ
D9	→	λ
D10	→	وD9   λ
D11	→	λ

Exemple de dérivation :



La définition de L1 est donnée par :

$$L1 = \bigcup_{f \in F, Naqissa} L(A(f))$$

## V. MISE EN ŒUVRE

Chaque automate, de ceux discutés auparavant, est constitué de trois parties :

### 1) Entrées

Les entrées correspondent aux traitements des proclitiques et des préfixes.

### 2) Etats transitoires

Les états transitoires, sont des états où le mot ne peut pas être accepté : un mot qui trouve sa fin dans l'un de ces états est rejeté. Ils correspondent généralement aux lettres constituant la racine du mot analysé.

L'ensemble des états transitoires (f, 3 et l) correspondent aux positions de lettres constituantes de :

- ✓ la racine pour les mots trilatéraux,
- ✓ les pseudo-racines pour les autres schèmes de mots.

### 3) Etats finaux

Ils correspondent aux traitements des enclitiques et suffixes. Chaque mot trouvant sa fin dans un état final est accepté.

## VI. PROBLEMES DE VOYELLATION

Dans le web actuel, la plus part des documents arabes ne sont pas voyellés, ce que nous a mené à restreindre ce travail aux documents non voyellés.

Or, sans voyellation, plusieurs schèmes (awzaane) se confondent. Par exemple, les mots qui suivent les schèmes Fa33al (فعال) et Fi3al (فعال) seront vus correspondre à un même schème (la première testée). Heureusement pour notre étude qui vise la Racinisation, le problème ne mène à aucune erreur : la racine correspond dans les deux cas aux lettres faa, ain et lam.

Le problème de voyellation conduit dans des cas bien limités, à des confusions de schèmes ou ce qu'on peut appeler glissement syntaxique qui mène obligatoirement à un glissement sémantique. Par exemple : naskono (نَسْكُون) qui veut dire : nous habitons) et nassakna (نَسْكُن) qui veut dire : accomplissent le rituels de pèlerinage) conduisent à deux racines différentes sakana ou nasarder. Ce problème est du à la lettre noon présentée sous forme de proclitique dans le premier cas au début du mot et à la fin comme lettre de la racine, et dans le deuxième cas son interprétation est inversée.

Pour le traitement des documents voyellés, aucun de ces problèmes d'ambiguïtés ne persistera.

## VII. CONCLUSION

Le processus de Racinisation abordé dans ce travail, consiste à représenter chaque forme des mots arabes, par un automate adéquat. Cette analyse morphologique, basée sur des automates déterministes à états finis, permet de reconnaître les mots arabes correctement écrits, suivants les différents schèmes illustrés dans ce travail.

Le travail effectué, va être étendu à la fonction inverse : la dérivation ou « Tassrif » pour pouvoir réaliser un utilitaire complet en traitement de la morphologie de la langue arabe.

## REFERENCES

- [1] eah S. Larkey, L. B., "Improving Stemming for Arabic Information Retrieval Light Stemming and Co-occurrence Analysis", SIGIR'02, August 11-15, Tampere, Finland, 2002
- [2] KHOJA, S., "APT: Arabic Part-of-speech Tagger", Proceedings of the Student Workshop at the Second Meeting of the North American Chapter of the Association for Computational Linguistics, pp. 20-25, Juillet 2001.
- [3] Darweech, K., "Building a shallow Arabic Morphological Analyzer in one day", Proceedings of the ACL-02 workshop on Computational approaches to semitic languages, pp. 47-54, Juillet 2002.
- [4] Kazem Taghva, R. E., "Arabic Stemming Without A Root Dictionary", Proceedings of the International Conference on Information Technology: Coding and Computing (ITCC'05) - Volume 1, 5, pp. 152 - 157.
- [5] Anne N. DE ROECK, W. A.-F., «A Morphologically Sensitive Clustering Algorithm for Identifying. Compute '08», Proceedings of the 1st Bangalore annual Compute conference, pp.199-206, Janvier 2008.

- [6] Saleem Abuleil, K. A., «Acquisition System for Arabic Noun Morphology», Proceedings of the ACL-02 workshop on Computational approaches to Semitic languages, Philadelphia, Pennsylvania , pp. 1-8, Juillet 2002.
- [7] Beesley, K. R., “Arabic finite-state morphological analysis and generation», Proceedings of the 16th International Conference on Computational linguistics - Volume 1, pp. 89-94, Aout 1996.
- [8] Muller, M.-P., «LANGAGES - GRAMMAIRES – AUTOMATES »,LesMathématiques.net, pp. 1-40, Juillet 2005.
- [9] Tammam, H., «La langue arabe, Morphologie et sémantique », DAR ATTAQAFI CASABLANCA, 2001.