ANNALES ISLAMOLOGIQUES

en ligne en ligne en ligne en ligne en ligne en ligne en ligne en ligne en ligne en ligne

AnIsl 24 (1989), p. 269-293

Claude Audebert, André Jaccarini

De la reconnaissance des mots-outil et des tokens.

DE LA RECONNAISSANCE DES MOTS-OUTIL ET DES TOKENS*

Claude AUDEBERT et André JACCARINI

Dans un précédent article nous avons décrit le principe d'un moniteur syntaxique susceptible d'optimiser un décodeur morphologique dont nous avions admis l'existence [1]. Notre démarche avait été de repérer en premier lieu les éléments induisant de fortes attentes syntaxiques que nous avions dénommés « tokens », ces éléments n'obéissant pas aux règles usuelles de formation du mot et devant donc être isolés [2]. Cette liste d'environ 150 éléments avait été établie indépendamment des problèmes de reconnaissance des graphèmes. En effet les « tokens » posent des problèmes de reconnaissance parce qu'ils peuvent s'agglutiner à des éléments pré ou postfixés. Nous rappelons qu'au niveau de la syntaxe les transitions entre différents états se font au moyen de l'entité graphique séparée par deux blancs (le mot graphique) tandis qu'au niveau de la morphologie ces transitions s'effectuent au moyen de caractères. C'est à ce niveau que l'on se situera

* Ce travail a été réalisé grâce à l'appui du Conseil international de la langue française. Que M. H. Joly en soit remercié.

1. Cf. *Annales Islamologiques*, XXII, 1986, p. 217-256. Un programme morphologique expérimental a été décrit en LISP sur un DEC 20 (Centre Mondial 1985). Ce programme doit toutefois être restructuré pour lui permettre d'interagir efficacement avec la syntaxe.

2. Nous appelons « *token* » toute suite de caractères isolables contenant une forme linguistique indépendante dépourvue de schème; ainsi بان - قد - ان sont des tokens. Il s'agit le plus souvent de mots-outils ou mots vides. Selon cette définition ب (préposition) ou ل ne sont pas des tokens, pas plus que به *bihi*. Un token nu est un token non aggluliné à des éléments pré ou postfixés.

Le choix du terme « token » se justifie de la manière suivante :

— d'une part c'est ainsi que l'on désigne en théorie de la compilation des éléments que l'on a intérêt à faire figurer dans le lexique plutôt que dans l'analyseur (voir Aho et Ullman ch. I p. 60). Or nos tokens échappent justement aux règles générales de formation morphologique; nous les avons fait figurer dans un tableau plutôt que dans les règles de production formelles correspondant aux automates morphologiques.

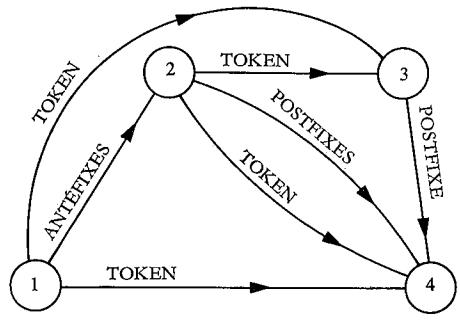
— d'autre part le mot « token » signifie jeton en américain. Un jeton est une entité visible qui déclenche un mécanisme. Or nous cherchons justement dans notre travail à décrire les mécanismes d'attente déclenchés par des éléments facilement repérables.

dorénavant (ce qui n'excluera pas, bien entendu, que l'on considère également la répercussion que peuvent avoir certains caractères sur les attentes syntaxiques).

Nous allons donc étudier systématiquement toutes les combinaisons possibles des tokens avec d'autres éléments. Cette étude ne saurait cependant totalement s'abstraire d'une étude plus large portant sur le mot graphique envisagé dans son extension maximale (donc entre autres sur ceux qui possèdent un schème). En effet la reconnaissance des signes est liée d'une part à la valeur informative de chaque lettre envisagée intrinsèquement et d'autre part à la position de cette lettre à l'intérieur du mot graphique. Or pour dégager clairement la notion de position du caractère par rapport au mot graphique on est amené à considérer l'extension maximale de ce dernier. A la différence de la syntaxe où le mécanisme de récursion permet de former des phrases d'une longueur arbitraire, la suite de caractères entre deux blancs est toujours d'une longueur bornée. Dans un mot maximal on peut distinguer une succession de zones selon le schéma suivant proposé par David Cohen [3] :

ANTÉFIXES / AFFIXE PERSONNEL / PARTIE RADICALE (SCHÈME + RACINE) / SUFFIXE / POSTFIXE →

On reconnaît dans ce schéma une zone d'antéfixes où ne figurent que les caractères ا a, و w, ف f, ك k, ل l, ب b, س s, dont il conviendra d'étudier l'ordre dans lequel ils interviennent ainsi que leurs compatibilités. La zone postfixe est constituée des éléments : ى ī, ك k, ه h, ا ā, م m, ن n, dont il conviendra d'étudier également les compatibilités avec les éléments préfixés. Nous n'étudierons ici que ces deux zones car ce sont les seules à pouvoir se combiner aux tokens. Le diagramme suivant rend compte de toutes les possibilités de ces combinaisons. L'explicitation de ces arcs fera l'objet de l'étude qui suit et nous entraînera à procéder à une catégorisation des tokens.



3. D. Cohen, « Essai d'analyse automatique de l'Arabe », in *Etudes de linguistique sémitique et arabe*, Paris, 1970, p. 49-78.

LES ÉLÉMENTS POSTFIXÉS

On remarquera que les seuls éléments susceptibles d'être postfixés à un token sont les pronoms dits affixes. Dans l'ordre des personnes du singulier au pluriel cela donne la liste :

ى *ī;* ني *nī;* ك *ka, ki;* ه *hu, hi;* ها *hā;* كما *kumā;* هما *humā;* نا *nā;* كم *; kum* كن *kunna;* هم *hum;* هن *hunna.*

On sait que ces affixes peuvent s'agglutiner au nom ainsi qu'au verbe [4]. Les affixes interviennent exclusivement en position finale du mot graphique marquant ainsi un état terminal, c'est-à-dire qu'ils ne peuvent occuper que les positions n, n-1, n-2, n désignant la longueur du mot (voir diagramme 9 représentant l'automate POSTFIXE). Deux classes de tokens, l'une agglutinable aux affixes l'autre non agglutinable nous permettront d'affiner les remarques déjà faites sur ces derniers [5].

TOKENS ET AGGLUTINATION AUX POSTFIXES

En examinant la possibilité de la combinaison TOKEN + POSTFIXE sur l'ensemble de la liste de 150 éléments on peut remarquer le fait suivant que nous énoncerons sous forme de règle :

« Seuls les tokens qui induisent une attente exclusivement nominale à l'occurrence suivante sont combinables aux postfixes ».

Ceci signifie que la présence d'un nom après le token est une condition nécessaire mais non suffisante pour que ce dernier soit combinable. Par exemple : هذا *hāḏā* n'est pas combinable à un postfixe.

Ce résultat va dans le sens de notre première tentative de classification des tokens suivant les différents niveaux qu'ils engageaient. Ces derniers pouvaient représenter selon les cas soit des automates de haut niveau engageant la totalité de la phrase soit un élément d'automate (une transition) entraînant l'attente d'un constituant du noyau ou bien celle d'un simple ajout.

4. Avec l'alternance ني *nī* après un verbe et ى *ī* après un nom.

5. Cf. article précédent, *Annales islamologiques*, XXII, 1986, p. 217-256.

Dans notre tableau 1 que nous reproduisons p. 274-275 apparaissaient des GN et GND que nous avons décrits sous forme de graphes récursifs; ces GN et GND ne pouvant en aucun cas constituer à eux seuls un noyau de phrase par opposition à GV (groupe verbal) lequel peut se réduire à un seul verbe. L'attente possible d'un verbe, induite par le token, entraîne donc en fait l'attente d'un noyau. Par contre, lorsque le token entraîne une attente *exclusivement* nominale, cette possibilité est exclue, c'est-à-dire que le token ne peut induire que l'attente d'un constituant qui peut être remplacé par un postfixe. Ces derniers ne peuvent naturellement jamais remplir la fonction de noyau. Le cas de إن *inna* appelle quelques remarques. Ce token est bien agglutinable à un postfixe en vertu de la règle que nous venons d'énoncer. Ce token engage toutefois une globalité puisqu'il induit l'attente d'un thème et d'un prédicat. Ces constituants forment néanmoins un noyau indépendant puisque la suppression de l'opérateur, tout en modifiant la vocalisation, n'entraîne pas l'effondrement de la phrase. Ce genre de critère nous permettra en fait de préciser de plus en plus l'idée de « hiérarchie » des attentes induites par les tokens. Ces distinctions entre niveaux d'attentes devraient avoir une répercussion sur la construction des automates syntaxiques. On retrouvera ainsi dans la classe des tokens non agglutinables à des postfixes :

— les interrogatifs;
— les démonstratifs;
— les pronoms personnels isolés;
— les tokens en terminaison ما *mā;*
— les particules affectées au verbe : قد *qad,* etc.;
— les particules du conditionnel ان *in,* لو *law;*
— les relatifs.

LES ÉLÉMENTS ANTÉFIXÉS

En position antéfixée on ne peut trouver que les caractères suivants ' *hamza,* و *wāw,* ف *fā,* ب *ba',* ك *kāf,* ل *lām,* س *sīn,* dont il conviendra d'étudier :

— A.1. Les différents éléments qu'ils peuvent représenter.
— A.2. Les compatibilités de ces éléments entre eux.
— A.3. Les positions dans lesquelles ils interviennent les uns par rapport aux autres.
— B. Leur compatibilité avec les tokens.

Ceci, on le rappelle, dans le but de reconnaître les tokens le plus rapidement possible. On pourra toutefois être amené à déborder le strict cadre des tokens. En effet ces éléments pouvant entrer sur des classes de mots quelconques, il importera pour une reconnaissance optimisée, de considérer les différentes parties d'un mot graphique, par exemple ل *li* pouvant entrer sur un nom ou un verbe, la présence de désinences exclusivement nominales ou verbales permettra de lever l'ambiguïté; ل *li* + préfixe verbal écarte d'emblée la recherche d'un nom ou d'un token. Nous serons ainsi amenés à poursuivre notre classification des tokens en fonction de leurs possibilités de combinaisons selon diverses modalités. Dans cette classification il conviendra également de tenir compte des critères de longueur qui sont un élément immédiatement pertinent.

A.1. *Les différentes représentations d'un graphème.*

Les éléments graphiques س *s* - ل *l* - و *w* - ف *f* - ا *'* posent un problème de reconnaissance car ils peuvent être différemment interprétés du fait que :

A.1.1. Ils peuvent tous être interprétés comme faisant partie de la racine.

Exemple :

فقد *fqd* peut être lu *faqada* (accompli actif 3e pers.); *fuqida* (accompli passif 3e pers.), théoriquement *faqqada* (accompli actif de deuxième forme), ou encore *fa-qad* conjonction de coordination + token, et ainsi de suite.

De même فسر *fsr* peut être lu *fasara, fusira, fassara, fussira* ou encore *fa-sir.*

On retrouve des ambiguïtés de ce type avec ك *ka* et ب *bā'*. Exemples :

— كسر *ksr* peut être lu *ka-sirr, kassara, kussira, kasara.*

— بقر *bqr* peut être lu *baqara, bi-qurra.*

A.1.2. Toutefois si la racine est déjà repérée, les caractères ك *kāf,* ب *bā',* ف *fā',* و *wāw* et س *sīn* sont non ambigus et se lisent *ka, fa, wa, bi, sa,* car chacun d'entre eux ne peut représenter qu'un seul élément possible.

A.1.3. Il n'en va pas de même de ل *l* et أ *'* du fait qu'ils peuvent représenter différents éléments. Ainsi le caractère ل *l* peut représenter :

— *la* lam de corroboration;
— *li* préposition
— *li* particule préverbale;
— *lø* précédant un impératif;

TABLEAU 1

Symbole	Operateur	Position opérateur = n		Contraintes				Nombre d'ensembles	Positionnement des ensembles	Nature des ensembles	Hiérarchie	Possibilité de suppression de l'opérateur
		Début	Milieu	n + 1	n − 1	< n	> n					
أَنْ	أَنْ	+		+ (V)			+ (GV)	2	CONTIGUS x > n; y > n; x < y	x = GV; y = OUVERT	++	–
	أَنْ		+	+ (V)		+ (GND)	+ (GV)	2	NON CONTIGUS x < n; y > n	x = GND; y = GV	+	–
	أَنْ		+	+ (V)		+ (GV)	+ (GV)	2	NON CONTIGUS x < n; y > n	x = GV; y = GV	–	–
	أَنْ		+	+ (V)		+ (prep GND)	+ (GV)	2	NON CONTIGUS x < n; y > n	x = GN; y = GV (PREP)	–	–
أَنَّ	أَنَّ		+	+ (≠ V)		+ (GV)	+ (GN)	2	NON CONTIGUS x < n; y > n	x = GV; y = PHN	–	–
إِنَّ	إِنَّ	+		+ (≠ V)			+ (GND)	2	CONTIGUS x > n; y > n; y > x	x = GND; y = GN (x, y) = PHN	+	+
إِنْ	إِنْ	+		+ (V)			+ (ف* GV)	2	CONTIGUS x > n; y > x	x = GV; y = GV	++	–
الذى	الذى		+	+ (V)		+ (GND)	+ $(Rel)_D$	⩾ 2	NON CONTIGUS x < n; y > n	x = GND; y = GV	–	
مَنْ	مَنْ	+		+ $(V)_D$			+ (ف* GV)	2	CONTIGUS x > n; y > x	x = GV; y = GV	++	–
	مَنْ		+	+ (V, PRO) ISOLÉ$_D$	≠ GND		+ (GV)	2	NON CONTIGUS x < n; y > n	x = OUVERT; y = PH	+	–
	مَنْ int	+		+ (V)			+ (GV)	1		x = GV من x = PHN	++	+
	مَنّ int	+		+ (N) + (ART)			+ (GN) + (GND)	1		x = { GN / GND } من x = PHN	++	–
مِنْ	مِنْ		+	+ (N) + (ART) + (PRO LIÉ) + (REL)			(GN) (GND)	2	NON CONTIGUS	x = OUVERT y = GN; GND	–	–
هو	هو	+			(ف ; ر)		(GN) (GND)	1		x = GN; GND = PRÉDICAT	++	–
	هو		+			+ (GND)	(GN) (GND) (GV)	2		x = THÈME y = PRÉDICAT		(+)
	هو		+			+ (GN)				x = GN y = Rel		+
او	او		+					2	NON CONTIGUS x < n; y > n	x = y (STR)		–
الا	الا excl	+						1		x = OUVERT	+	+
	الا int	+		+ $(V)_D$ inaccompli			+ (GV)	1		x = GV	+	
	الا		+	+ (V)		+ (GV)	+ (GV)	2		x = GV; y = GV	–	–

— *lø* deuxième caractère de l'article qui pourra lui-même être représenté par les formes ال , للْ . للْ peut représenter à son tour ل *li* + article et *li* + Nom dont la première radicale est un ل *' l.* Ce nom est déterminé ou non déterminé. Dans le cas où il est déterminé le *lām* reçoit un signe de gémination. Exemple :

— لَوْنٍ indéterminé, لَوْنِ déterminé : *li-lawnin, li-l-lawni.*

Le caractère ا peut représenter :

— *'a* interrogatif;

— un préfixe de schème qui peut également se confondre dans le cas du verbe avec le préfixe de conjugaison. Il se présente sous la forme اَ اُ اِ *'i, 'a, 'u;* exemple : افعل *'f ' l* peut se lire : *'a-fa'ala :* interrogatif + accompli 3e pers. ou bien *'af'ala, 'uf'ilu, 'uf'ila, 'uf'alu,* etc., respectivement, accompli 3e pers.; inaccompli 1re pers., indicatif puis subjonctif ou accompli passif 3e pers. inaccompli, 1re pers. indicatif.

Le premier caractère de l'article ال *al* lorsqu'il est graphiquement exprimé. Dans ce cas il se présente sous la forme أل .

A.2. *Les comptabilités des antéfixes entre eux.*

Certains de ces éléments sont incompatibles entre eux; par exemple ب *bi,* ك *kāf* et *lām* le sont absolument quel que soit l'ordre considéré. Par contre أ ' interrogatif est compatible avec tous les éléments à condition qu'il intervienne en première position.

A.3. *Les problèmes de position* et de compatibilité sont donc étroitement liés. Il en sera rendu compte dans le tableau 2, p. suiv. Celui-ci a été organisé en fonction des objectifs que nous nous sommes fixés, à savoir de déterminer les suites possibles de caractères dans la zone des antéfixes. Chaque suite est définie par un ordre donné et par ses répercussions à l'intérieur du mot (définition, vocalisation), ou à l'extérieur de celui-ci (attentes syntaxiques), d'où les colonnes figurant sous les rubriques POSITION et CONTRAINTES. Sous position figurent les colonnes DÉBUT, MILIEU et LIBRE :

— DÉBUT signifie que le caractère étudié doit être précédé d'un blanc c'est-à-dire qu'il n'admet pas de caractère avant lui ainsi l'*alif* interrogatif satisfait à cette condition. Exemple :

أَخَرَجَ الولد ؟ *a-ḫaraǧa -l-waladu?*

CARACTÈRE		DÉFINITION		POSITION			CONTRAINTES				ATTENTES			VOCALISATION	DÉPENDANCES CONTEXTUELLES				EXEMPLES
							STRICTES		LÂCHES						STRICTES		LÂCHES		
				DÉBUT	MILIEU	LIBRE	n−1	n+1	<n	>n	NOM	VERBE	TOKEN		x−1	x+1	<x	>x	
LAM (ل)	لِ	PRÉPOSITION				+	≠ك ≠ب ≠س	≠ف ≠ك ≠ب ≠س ≠و			+		+	INDIRECT				GN, GND	لِرجلٍ لِلرجلِ
		INTRODUCTEUR DE VERBE	IMPÉRATIF			+		=ت =ى =ا				+ IMP		APOCOPÉE					لِيفعلْ لِيفعلوا لِيفعلن لأفعلْ لِتفعلْ
			BUT			+	≠ك ≠ب ≠س ≠ل	=ت =ى =ا				+ SUBJ		CAS DIRECT					ذهب لِيفعلَ فليفعلَ كذا خُلق الإنسان
			DÉNÉGATION	+				=ت =ى =ا =ن				+ SUBJ		CAS DIRECT			ما كان لم يكن		وما كان اللهُ ليطلعكم على الغيب
	لَ	PRÉPOSITION				+	≠ك ≠ب ≠س	POSTFIXE											لَك ، لَه ، لنا ، لهما ...etc
		CORROBORATION					≠ل ≠ك ≠ب ≠س ≠ا	≠ف ≠و ≠س			+		+				Introducteur de prédicat		إنّ اللهَ لَغفورٌ رحيم
		INTRODUCTEUR DE VERBE	CORROBORATION				≠ك ≠ب ≠ا ≠س ≠ل	=ت =ى =ا =ن				+ INACCOMPLI					Introducteur de prédicat		إنّ رَبكَ لَيحكُمُ بينهم يوم القيامة
			ÉNERGÉTIQUE				≠ك ≠ب ≠ا ≠س ≠ل	=ى =ت =ا =ن		ن en position précédant les postfixes		+ INACCOMPLI (ÉNERGÉTIQUE)		NOUN GÉMINE نَّ					لأَقْتُلنكَ
			LAM DE SERMON	+				=ى =ت =ا =ن				+ INACCOMPLI					GV		والله لأفعلن
			APODOSE	+													لو ..... لَ	2e noyau de conditionnel	لو ..... لَقتلته
	لْ	INTR. VERBE	IMPÉRATIF		+		=و =ف	=ى =ت =ا =ن				+ (IMPÉRATIF)		APOCOPÉE					فَلْيفعلْ ، وَلْيفعلْ
		ARTICLE	2e LETTRE DE L'ARTICLE			+	ا	LETTRE LUNAIRE		≠ POSTFIXES	+								القمر
			2e LETTRE DE L'ARTICLE			+	ا	LETTRE SOLAIRE		≠ POSTFIXES	+								الشمس
			SEULE LETTRE DE L'ARTICLE			+	لِ			≠ POSTFIXES	+								لِلْقمر لِلْشمس لِلّون
بِ		PRÉPOSITION				+	≠ك ≠س ≠الـ ≠لِ ≠لْ	≠ك ≠لِ ≠لَ ≠ا ≠ف ≠و ≠س ≠لْ		≠ {=فى POSTFIXE}	+		+	INDIRECT				GN, GND	بالفعلِ بفعلٍ
		INTRODUCTEUR DE PRÉDICAT		+			≠ك ≠لِ ≠لْ ≠ال ≠س	≠ك ≠لِ ≠لَ ≠ا ≠ف ≠و ≠س ≠لْ		≠ {=فى POSTFIXE}	+		+	INDIRECT			ليس . . . لِـ	GN PRÉDICAT 'INDICATEUR DE FONCTION	ليس محمد برَجُلٍ
كَ		PRÉPOSITION				+	≠ب ≠لِ ≠ب ≠س ≠ال	≠ب ≠س ≠ك ≠ا ≠ف ≠و		≠ {=فى POSTFIXE}	+			INDIRECT				GN, GND	كوردةٍ ، كالوردةِ
ف		COORDONNANT		+				≠و ≠ا			+		+		N	N			إبتسامة فَلِقاء
		COORDONNANT		+				≠و ≠ا			+	+	+				NOYAU	NOYAU	الولد مريض فهو فى السرير خرج
		COORDONNANT			+		≠و ≠ا	≠و ≠ا			+	+	+						أفَلا
		INTRODUCTEUR DE LA 2e PARTIE DE LA CONDITIONNELLE		+						IMPÉRATIF							NOYAU	NOYAU	من يجتهد فالنجاح حليفه / إن أردتم فادرسوا حتى تجتهد
		CONJONCTION DE SUBORDINATION		+				=ى =ت =ا =ن		CONJ. DES SUBJONCTIFS		+ SUBJONCTIF					NOYAU (IMP., INTERROGATION, SOUHAIT, NÉGATION)		لا تقربا هذه الشجرة فتكونا من الظالمين
و		COORDONNANT		+				≠و ≠ا			+		+		N	N			الولد وأخوه
		COORDONNANT		+				≠و ≠ا			+	+	+				NOYAU	NOYAU	خرج وبكى الولد مريض وأخوه جميل
		D'ACCOMPAGNEMENT		+				(= ART)			+			CAS DIRECT					مشى والنهر
		DE SERMENT		+							+			CAS INDIRECT		NOM DÉTERMINÉ			واللهِ
		SUBORDONNANT		+													NOYAU	NOYAU (PHN)	خرج والناس نيامٌ – خرج وهو يبكى – خرج وقد بكى
ا		INTERROGATIF		+							+	+	+		Ø		Ø	NOYAU(x) أ . . . أم . . .	أخرج الولد أم لا أخرج الولد أم لا
		INTERROGATIF		+							+	+	+				VERBE MARQUÉ NEG	noyau introduit par أم	لا يدرى أفعلَ ذلك أم لا
		INTERROGATIF		+													VERBE MARQUÉ	INTERROGATIF INDIRECT	(رمى الحمار راكباً أفضل أم راجلاً) هل يدرى أهو مسافر أم لا
		INTERROGATIF		+								(+) o					سواء . . . ءَ	أم . . . ءَ	سواء عليهم أأنذرتهم أم لا تنذرهم لا يؤمنون

— MILIEU signifie que le caractère ne peut intervenir que précédé d'un autre caractère. Exemple : فل *fal* (*lām* introduisant un impératif)

فَلْيَفْعَل *fa-l-yaf'al* / وَلْيَفْعَلْ *wa-l-yaf'al*

— LIBRE signifie que le caractère peut être précédé ou non par un autre caractère que lui-même. Exemple : ل *li* (préposition)

لرجلٍ *li-rağulin* / أَلِرَجُل *a-li-rağulin*

La rubrique CONTRAINTES exprime les compatibilités dans les suites de caractères. Elles se divisent en deux ensembles :

— *Les contraintes strictes* portent sur les positions qui précèdent ou suivent immédiatement la position de la lettre représentée par n. Par exemple ل *li,* préposition, est soumis à des contraintes strictes en n-1; en effet il ne peut être précédé de س *sin,* ك *k* et ب *b,* ce que l'on a exprimé sous la forme : ≠ س , ≠ ك , ≠ ب .

Il est également soumis à des contraintes en n + 1; en effet il ne peut être suivi de س *s,* ك *k,* ب *b,* ف *f,* و *w.*

— *Les contraintes lâches* portant sur des positions non repérables a priori; par exemple, l'article exclut la possibilité d'un ajout de pronom postfixé. Ces pronoms ne peuvent intervenir qu'en positions finales du mot, à savoir les occurrences n, n-1, n-2 (n, représentant la longueur du mot, laquelle n'est pas connue d'avance). Exemple :

ولدهم *waladuhum* / الولدهم *al-waladuhum*

Les colonnes suivantes ATTENTE expriment les contraintes de la lettre considérée sur la catégorie grammaticale du mot auquel elle est préfixée; par exemple :

— ل *li,* préposition, entraîne que le mot auquel il est préfixé soit un nom : *li-rağulin* ou *li-l-rağuli.* Dans la colonne N (nom) a donc été marqué le signe +.

Les dépendances contextuelles débordent le cadre du mot graphique et reflètent les contraintes de la lettre considérée sur l'environnement syntaxique. X — 1 et X + 1 représentent respectivement les mots qui précèdent et suivent le mot considéré. « X désigne un ensemble d'occurrences précédant celui-ci, » X un ensemble d'occurrences suivant celui-ci. Exemples :

إن ربك لغفور رحيم *inna rabbaka la-ġafūrun rahīmun*

Le *lām* de corroboration est introducteur de prédicat :

خرج والناس نيام *ḫaraǧa wa-l-nāsu niyāmun*

Le و *wa* de subordination relie deux noyaux dont le second doit être une PHN (phrase nominale).

Les informations contenues dans ce tableau ont été ensuite traduites sous forme de diagrammes qui explicitent en outre certaines attentes figurées dans les colonnes « n et n » ainsi que celles qui se trouvent dans N et V dans le but de faire mieux ressortir l'incidence que peut avoir le graphème préfixé sur les différentes zones du mot. On pourra ainsi disposer d'informations supplémentaires permettant non seulement d'optimiser les procédures de reconnaissance des tokens mais d'être directement incorporées au moniteur syntaxique.

B. AGGLUTINATION DES TOKENS AUX ANTÉFIXES

L'étude des combinaisons possibles des tokens avec les antéfixes et/ou les postfixes donne les quatre classes B1 suivantes ainsi que des sous-classes B2 recoupant certaines d'entre elles.

B.1.1. Les tokens nus qui ne sont agglutinables à aucun élément (TOKØ).

B.1.2. Les tokens qui reçoivent anté et postfixes que l'on notera token 1. Cette classe recouvre les tokens à comportement nominal bien que certains puissent jouer le rôle de préposition tels دون *dūna*, مثل *miṯl*, إذاء *izā'*, خلف *ḫalf*, لدن *ladun*, بعضَ *ba'aḍ*, أيّ *ayyu* relatif et interrogatif et أنّ *anna*.

Le cas أن *anna* mérite quelques remarques :

— أن est un transformateur du nom qui remplace celui-ci par une phrase nominale. Ainsi علم وجودة *'alima wuǧūdahu* est équivalent à علم أنه موجود *'alima annahu mawǧudun*. L'agglutination possible du ب *bi* au أنّ *anna* s'explique alors par le fait que le nom peut toujours recevoir comme ajout de début un ب .

— *fi'luhu = innahu maf'ūlun, bi-annahu maf'ulun = bi-fi'lihi.*

B.1.3. La classe des agglutinables aux seuls postfixes (token 2). On y trouve les prépositions ainsi que les tokens suivants :

— ليت *layta* / لعلّ *laʿalla* / لكنّ *lakinna* / كأن *kaʾanna* / لأن *liʾanna* [6] traditionnellement classés dans les particules mettant le thème à l'accusatif.

B.1.4. La classe agglutinable aux seuls antéfixes (TOK3). On y trouve :

— des pronoms démonstratifs, familles هذا *hāḏa* et ذلك *ḏālika;*
— des pronoms relatifs, familles الذى *allaḏī;* ما *mā;*
— حيث *ḥayṯu* (agglutinable au seul ب);
— لا la négation précédant un nom dans certains cas (agglutinable au seul ب);
— أنْ *an* transformateur de verbe en constituant de nature verbo-nominale (maṣdar) et qui comme أن peut s'agglutiner à ب et former avec ل et ك des tokens figés.

B.2.1. Le *la* (*lām* de corroboration) [7].

Le *lām* de corroboration a des implications syntaxiques :

a. Sa présence est liée à إن auquel cas le *la* sera introducteur de prédicat, que celui-ci soit nominal, verbal ou complexe prépositionnel; il interviendra alors en cours de phrase.

Exemples :

— إنّ ربّى لسميع الدعاء *inna rabbi la-samīʿu -d-duʿāʾi;*
— إنّ ربك ليحكم بينهم *inna rabbaka la-yaḥkumu baynahum;*
— إنك لعلى خلق عظيم *innaka la-ʿalā ḫuluqin ʿaẓīmin.*
— En début de phrase et en l'absence de إنّ *inna* il révélera un thème (*mubtada'*).

Exemple :

— لانتم أشد رهبة *la-antum ašaddu rahbatah.*

6. On remarque que كأن et لأن sont en fait des composés de ان + ك / ل; toutefois comme il s'agit d'éléments figés ils ont été rangés dans cette classe plutôt que dans celle de ل.

7. Nous n'entrerons pas dans le débat des grammairiens quant à la nature réelle de ce *lām* et de ses dénominations *lām al-tawkīd* ou *al-ibtidā'*. Cf. Ibn Hišām, *Muġnī al-labīb,* Damas, 1969, p. 251 et suiv.

AnIsl 24 (1989), p. 269-293 Claude Audebert, André Jaccarini
De la reconnaissance des mots-outil et des tokens.

b. On trouve *la* précédant un énergétique. Il peut alors intervenir, ce qui est le cas le plus fréquent, après un serment. Dans les deux premiers cas il introduit des constituants de noyau en fonction de thème ou de prédicat. En principe il ne devrait donc entrer que sur les tokens susceptibles d'être en première position d'une suite d'occurrences en fonction de thème ou de prédicat. C'est le cas des pronoms isolés.

Nous nous en tiendrons pour l'instant à ces critères tout en signalant que l'agglutination de *la* avec إن *in* pour former un token figé لئن laisse présager un phénomène plus large qu'une étude approfondie sur corpus devrait aider à élucider. Cela induit une nouvelle classe de token TOK 4 (pronom personnel isolé — pronom relatif — préposition).

B.2.2. Le hamza interrogatif.

Par sa position dominante en début de phrase le hamza d'interrogation semble se rapporter à l'ensemble de la proposition plutôt qu'à l'un de ses constituants. Les cas où le hamza interrogatif intervient en milieu de phrase sont ceux où cette dernière comporte plus d'un noyau. Exemples :

— *sawā'un ʿalayhim 'a-'anḏartahum am lam tunḏirhum lā yu'minūna;*
— *hal yadrī 'a-huwa musāfirun am lā.*

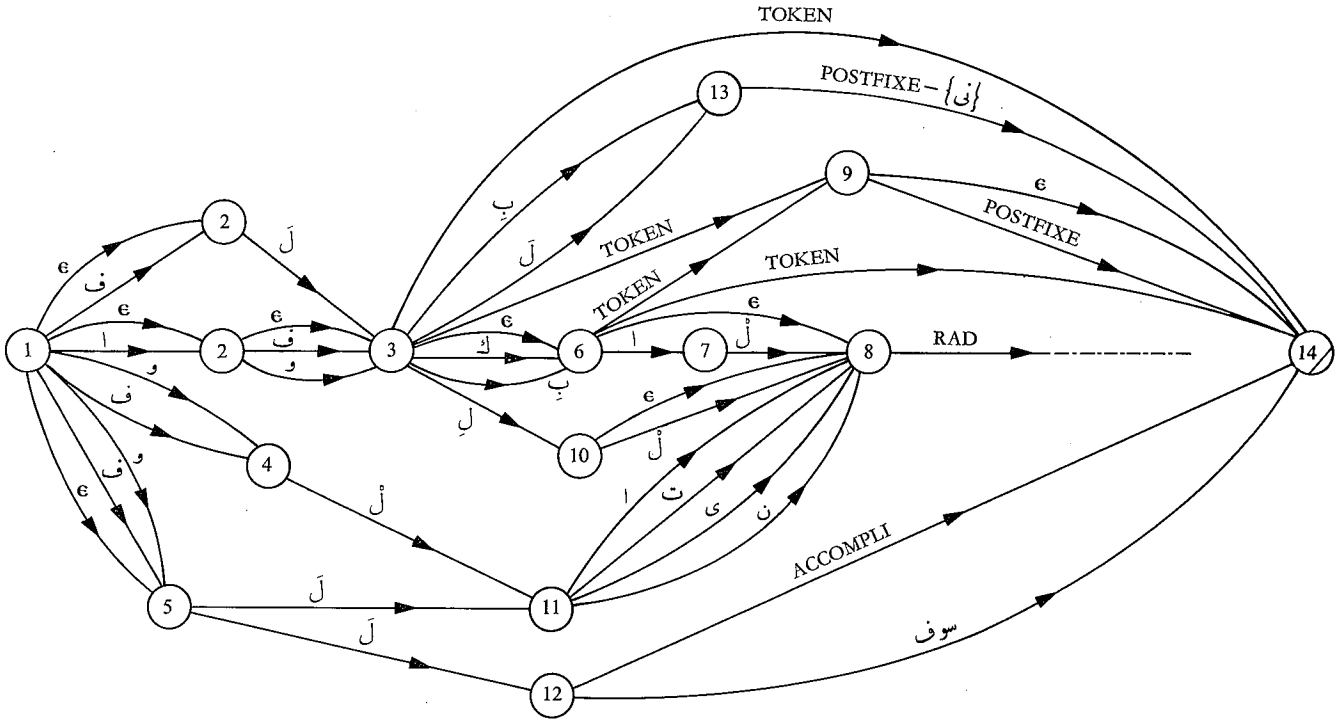
Dans ces cas on voit apparaître des morphèmes discontinus du type سواء ... أ ... أم ou bien des verbes marqués du type يدرى (savoir). Cela peut expliquer que le hamza d'interrogation intervient sur la plupart des tokens, les incompatibilités ne découlant que de contraintes liées soit à la position, soit de manière plus générale à la sémantique que nous ne traitons pas ici. Exemples :

— أ ≠ بل ; أ ≠ قد أقد .

LES DIAGRAMMES

Les diagrammes ci-après représentent des automates finis dont la programmation constituera un implémentation d'un algorithme de reconnaissance de la forme linguistique [8]. Les transitions entre états s'effectuent au moyen d'arcs qui sont étiquetés, soit par des lettres, soit par le symbole ϵ qui représente le vide. Dans ce cas la machine peut passer d'un état au suivant tout en gardant la tête de lecture sur le même caractère. L'introduction d'un tel arc rend compte de la possibilité de suppression de certaines lettres et simplifie de manière significative la représentation de la machine.

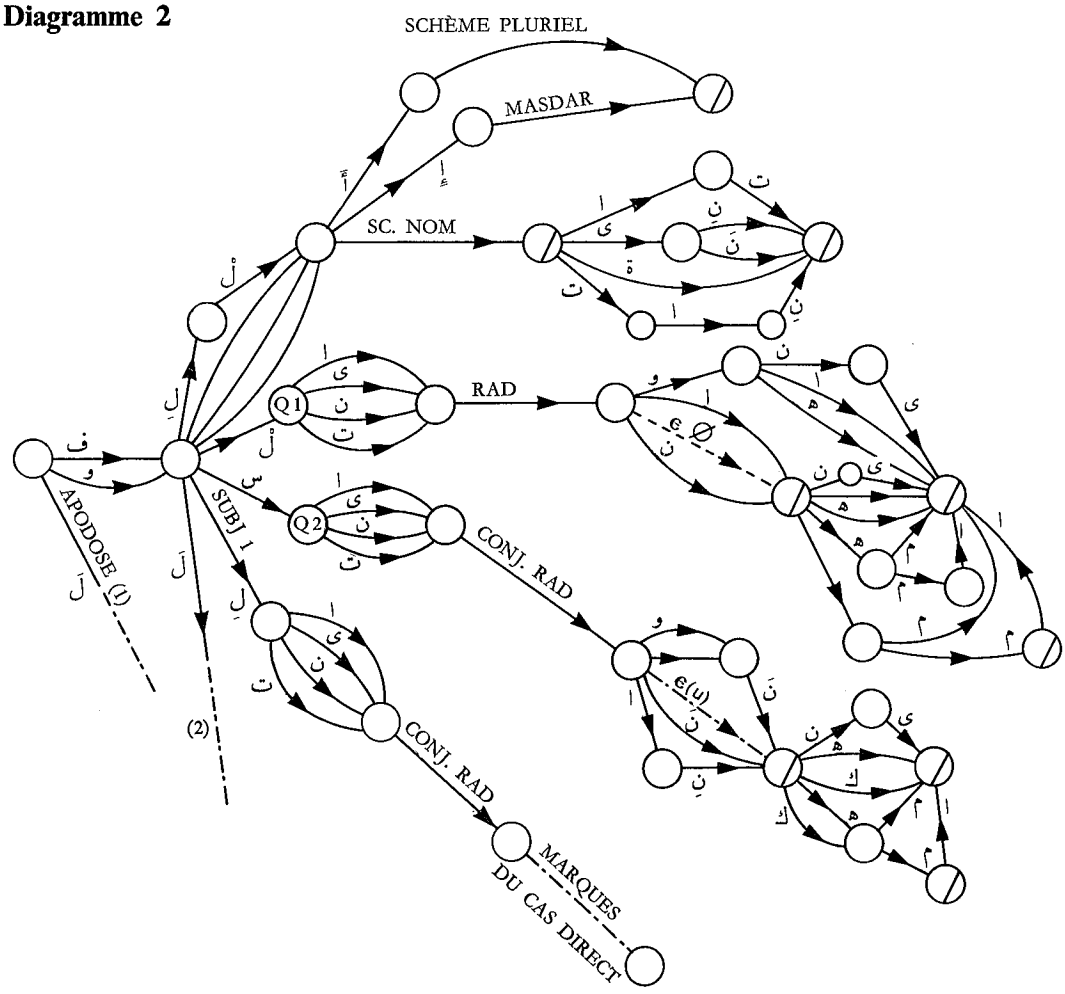
Diagramme 1



8. Toutefois la traduction en programme n'est pas immédiate. Les diagrammes en question représentent des automates « non déterministes » — c'est-à-dire que pour certaines « situations », on peut trouver au moins deux instructions applicables : l'automate a la possibilité de passer dans au moins deux états. (Un automate déterministe est tel qu'après chaque mot lu, l'unité de contrôle n'a la possibilité de passer que dans un seul état.) Il faudra convertir ces automates non déterministes (appelés aussi graphes de transitions) en automates déterministes.

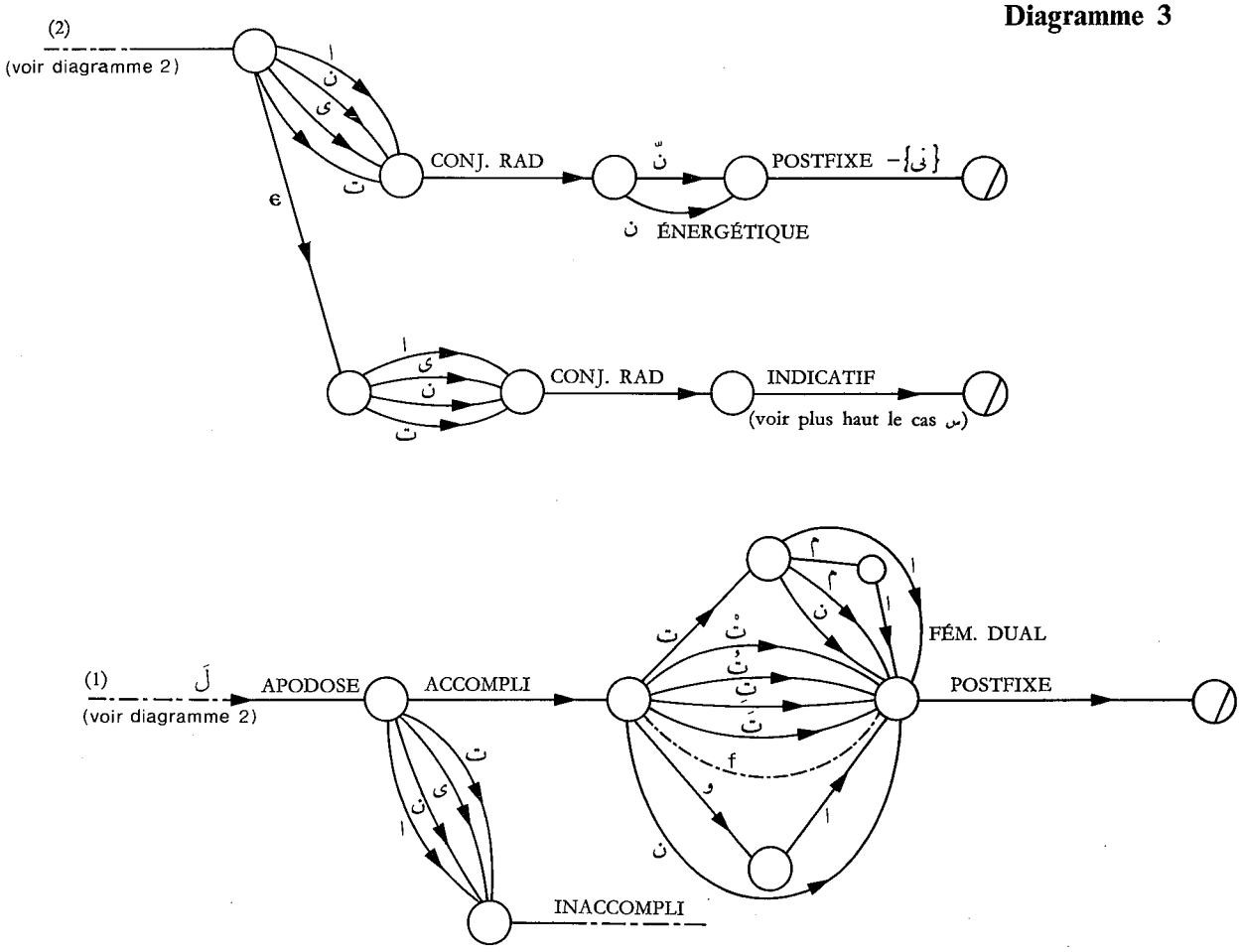
AnIsl 24 (1989), p. 269-293 Claude Audebert, André Jaccarini
De la reconnaissance des mots-outil et des tokens.

Diagramme 2



AnIsl 24 (1989), p. 269-293 Claude Audebert, André Jaccarini
De la reconnaissance des mots-outil et des tokens.

Diagramme 3



AnIsl 24 (1989), p. 269-293 Claude Audebert, André Jaccarini
De la reconnaissance des mots-outil et des tokens.

Diagramme 4

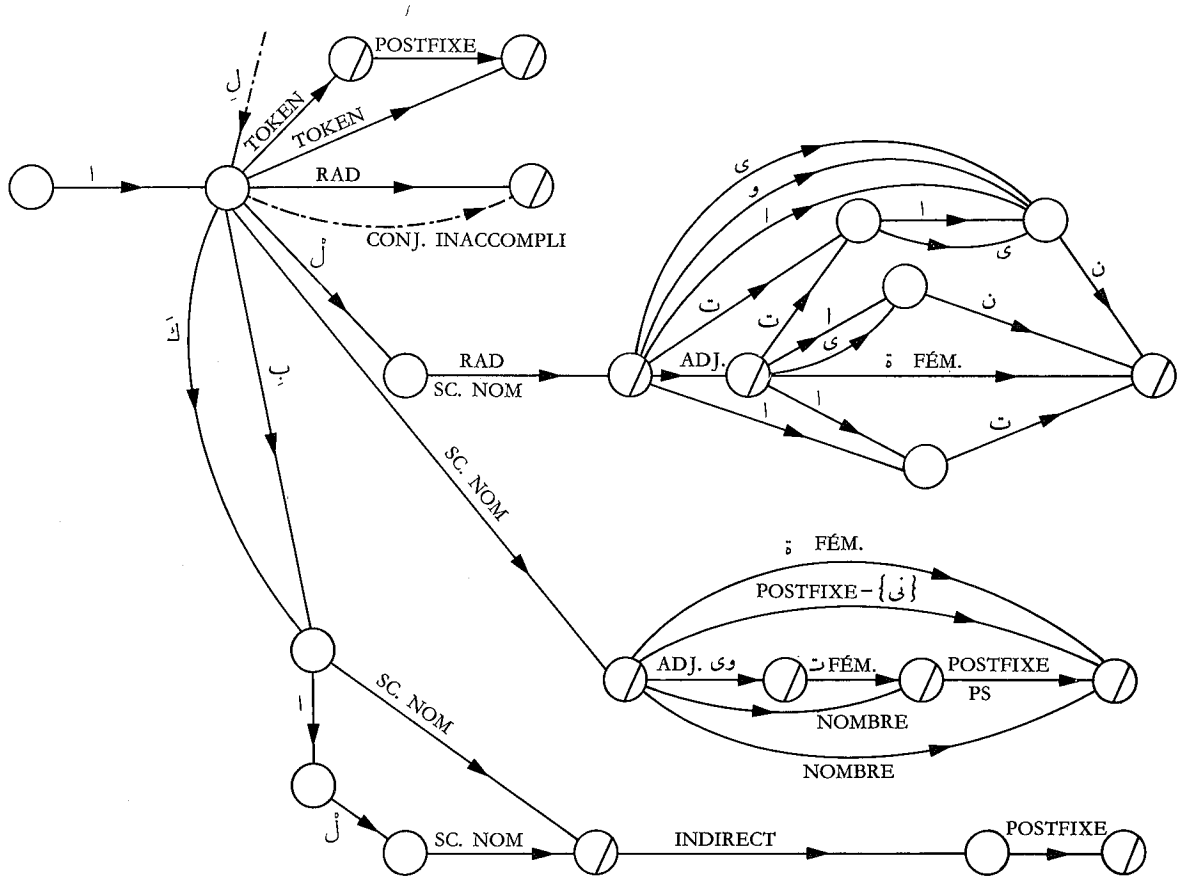


Diagramme 6

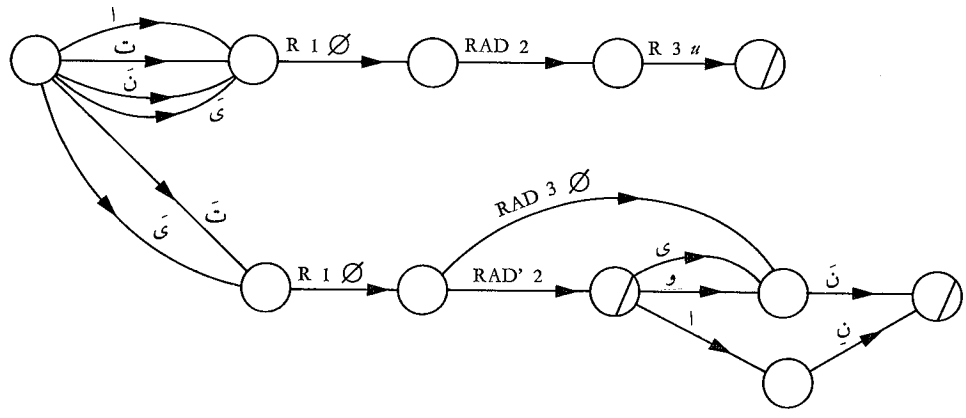


Diagramme 5

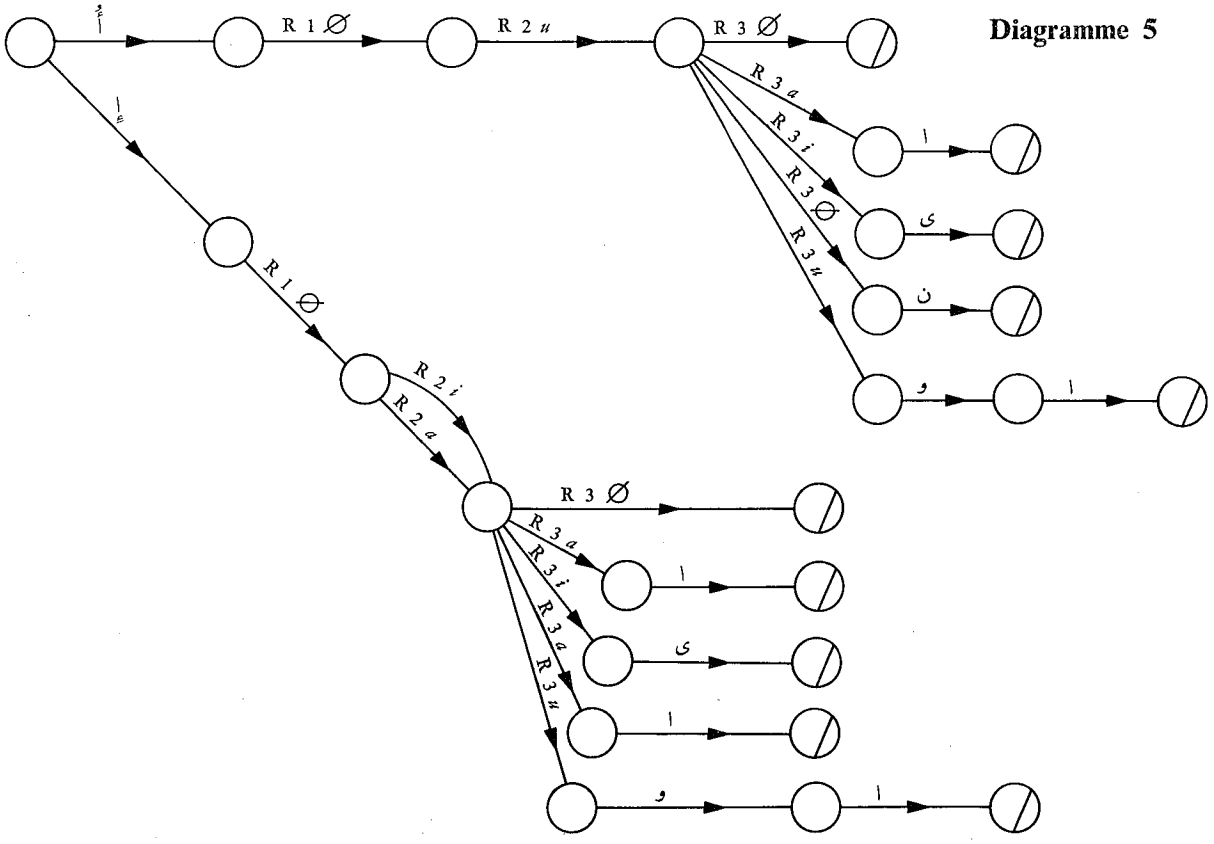
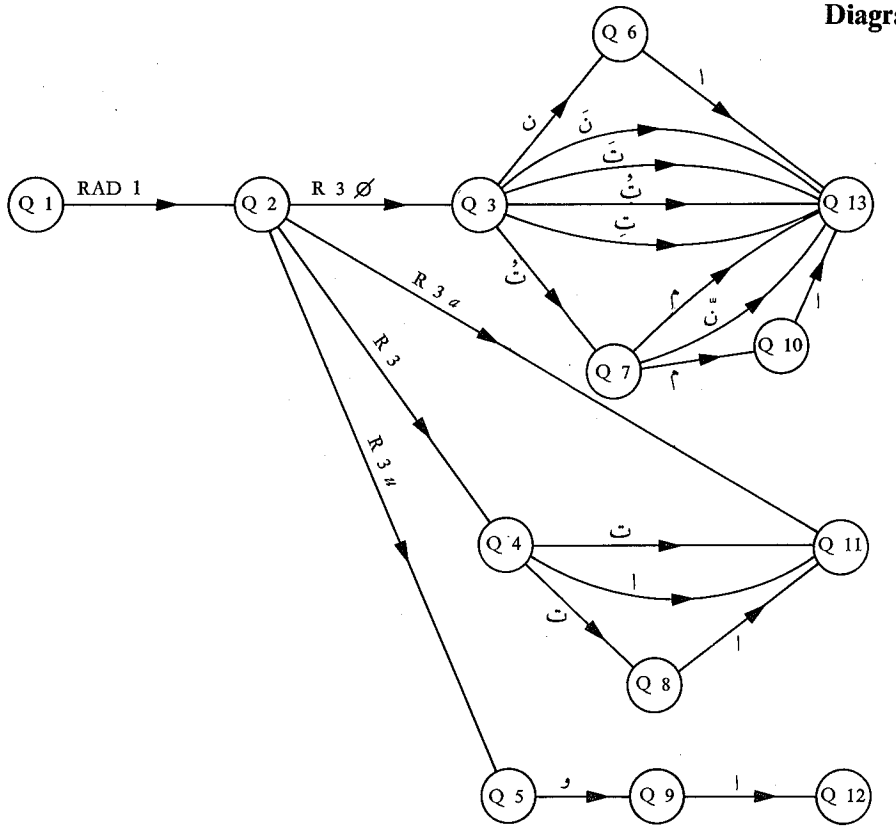


Diagramme 7



AnIsl 24 (1989), p. 269-293 Claude Audebert, André Jaccarini
De la reconnaissance des mots-outil et des tokens.

Diagramme 8

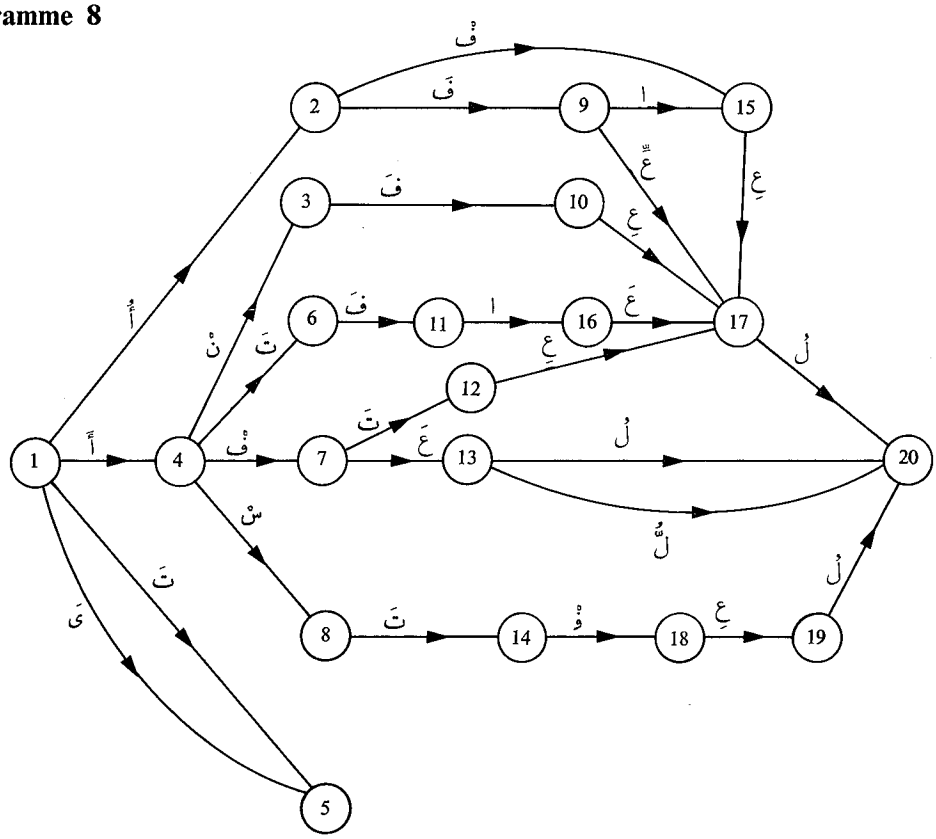
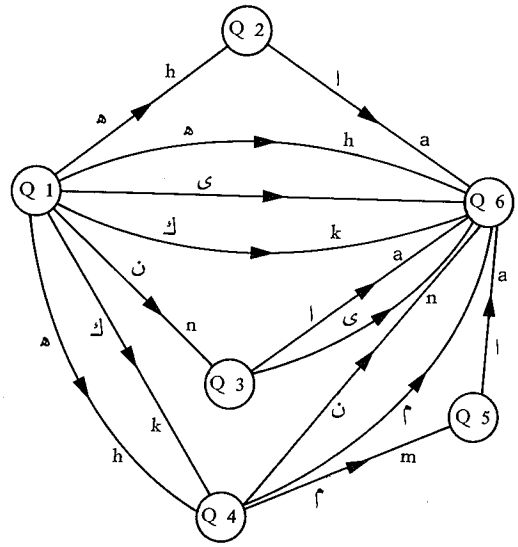


Diagramme 9



On appelle cette machine un automate standard non déterministe car elle ne possède pas un mécanisme de pile de mémoire qui lui permettrait de tenir compte de la récursivité comme cela a été nécessaire lorsque nous avons traité la syntaxe. Cette machine ne peut en effet appeler indéfiniment d'autres machines, dont elle-même. Cette possibilité est exclue du fait qu'il n'existe pas de mots infinis (pour la même raison les boucles n'apparaissent pas sur ces diagrammes). Telle qu'elle est décrite ci-dessus cette machine agit comme un simple accepteur de mots.

On essaye de représenter toutes les transitions licites de lettre à lettre en vue de former un mot arabe. Toutefois on peut augmenter la puissance de cette machine en créant des registres pour recueillir en cours de parcours un certain nombre de définitions dont on se servira par exemple pour construire un arbre interne au mot. Cet arbre serait produit comme effet de bord de la machine qui, arrivée à un état terminal, le communiquera au moniteur syntaxique. Nous rappelons que l'objectif d'un décodeur morphologique est d'obtenir une définition lexicale des mots graphiques sur laquelle l'analyseur pourra s'appuyer pour effectuer une analyse syntaxique et qu'en retour celui-ci pourra guider le décodeur morphologique pour lever les ambiguïtés liées à une forme graphique, le recours au lexique étant minimum.

Enfin il est intéressant de noter que ces machines peuvent aussi être conçues comme la représentation d'un mécanisme de reconnaissance d'un langage régulier. Un langage régulier est défini par un ensemble de symboles élémentaires, le vocabulaire, auquel on adjoint le symbole ϵ vide.

Chacun de ces symboles élémentaires peut constituer à lui seul une expression régulière. L'ensemble des expressions régulières peut être construit à partir des trois opérations suivantes :

— *la concaténation.*

Si A et B sont des expressions régulières alors A suivi de B que l'on notera A.B. est une expression régulière.

— *l'opération* OU.

Cette opération nous permettra de spécifier les différents choix possibles. La présence d'un OU entre deux expressions régulières signifie que cette expression ne sera reconnue dans un texte que si, et seulement si, l'une ou l'autre des expressions qui la constituent se présente dans le texte. Cette opération est représentée par le symbole + l'utilisation

AnIsl 24 (1989), p. 269-293 Claude Audebert, André Jaccarini
De la reconnaissance des mots-outil et des tokens.

de parenthèses nous permettra de la combiner avec la concaténation de manière aussi compliquée que l'on veut. Exemple :

A + B signifie A ou bien B.
C (A + B) D représente CAD ou bien CBD
(A + C) (B + C) D signifie A B D
ou bien A C D
ou bien C B D
ou bien C C D

— *la clôture.*

Il existe une troisième opération que l'on n'utilise pas pour la description de la morphologie [9] que l'on désigne sous le nom d'opération de Kleene ou clôture. Elle est représentée par le symbole *. Cette opération est définie de la manière suivante :

Si A est une expression régulière alors

$$A^* = \emptyset + A + A^2 + A^3 + \ldots = \bigcup_0^n A^n$$

où $A^2 = A.A$

Cette opération permet la répétition d'un symbole un nombre de fois indéterminé (0 inclus).

Par exemple AB* désigne un A pouvant être suivi d'un nombre indéterminé de B alors que (AB)* désignent les suites alternatives de A et de B.

On utilise en général de telles expressions pour définir les problèmes de « pattern-matching » qui consistent à retrouver dans un texte des suites de caractères dont le modèle (le pattern) n'est pas décrit de manière « complète ». Par exemple on peut spécifier qu'un nombre quelconque d'occurrences d'un certain ensemble de mots ou de caractères doivent être ignorés.

Exemple : Pour retrouver dans un texte toutes les phrases contenant dans l'ordre les tokens suivants إنّ *inna,* التى *al-latī,* هى *hiya* on forme l'expression régulière : (mot)* هى. (mot)* التى. (mot)* إنّ.

On voit ainsi que les algorithmes morphologiques d'extraction de la racine peuvent se réduire finalement à des problèmes classiques de pattern-matching.

9. Comme on le verra toutefois dans l'exemple ci-dessous de pattern-matching, cette opération sera utile pour tester sur un corpus des informations partielles du programme morphologique.

Le petit automate PS (postfixe) que nous avons représenté diagramme 9 pourra être décrit par l'expression suivante :

(ه + ى + ك + (ه . ا) + ن (ا + ى) + ك (م + ن + (م . ا)))
(h + ī + k + (h.a) + n (ā + ī) + k (m + n + (m.ā)))

De telles expressions se prêtent à certaines opérations algébriques tel le développement ou la mise en facteur. Exemple :

(ه + ى + ك + (ه.ا) + ن (ا + ى) + ك (م + ن + (م.ا))) =
(h + ī + k + (h.a) + n (a + ī) + k (m + n + (m.a))) =
(ه (ɛ + ا) + ك (ɛ + (م + ن + (م.ا))) + ى + ن . ى + ن.ا)
(h (ɛ + a) + k (ɛ + (m + n + (m.a))) + ī + n.a + n.ī).

où ɛ représente l'arc vide (élément neutre par rapport à la concaténation).

= (h (ɛ + ā + k (ɛ + (m (ɛ + i) + n)) + ī + n (a + ī))

Ces opérations induisent naturellement des transformations de graphe en graphe équivalent.

L'intérêt de cette remarque est qu'elle nous indique que l'on n'a pas à se soucier de la manière la plus efficace de construire un graphe une fois que les attentes ont été correctement définies. Il existe une procédure automatique pour réduire ce graphe en un graphe plus simple. On pourra ainsi représenter toute la morphologie sous forme d'une seule expression régulière, c'est-à-dire finalement sous forme d'un arbre binaire ET/OU que l'on pourra aisément représenter en machine.

Nous avons surtout voulu exprimer dans le diagramme 1 les compatibilités et l'ordre dans lequel peuvent intervenir les éléments antéfixés que nous avons considérés plus haut.

Ce diagramme rendra compte des incompatibilités. Par exemple la forme الـ (*hamza* interrogatif + *lām* de corroboration) sera rejetée. En effet, supposons que la forme ... أَلْـ soit vocalisée, l'automate a le choix au départ entre deux arcs أ et ɛ. Le premier choix correspond à avancer la tête de lecture pour la positionner sur la première lettre et à vérifier si celle-ci est identique à celle citée sur l'arc. Si cette condition est remplie il pourra passer à l'état suivant. Dans le cas contraire il se maintiendra à l'état initial et explorera les autres arcs possibles. Si aucune condition attachée à chacun de ces arcs n'est vérifiée l'automate enverra un message d'erreur et s'arrêtera. L'exploration de ces arcs se fera dans un ordre que l'on aura intérêt à préciser, car ce dernier peut avoir

une grande influence sur l'optimisation du programme. Dans le cas où il emprunte l'arc ϵ, l'automate maintient la tête de lecture en position initiale et passe à l'état suivant. L'automate se trouve donc dans cet état sans avoir balayé aucune lettre. Supposons que l'automate, en cherchant à analyser la forme أَلَ, soit parvenu ainsi à l'état Q 2. Cet état ne lui offrira alors comme possibilité de transition que les arcs أ ou ϵ. Cette fois-ci il sera obligé d'emprunter l'arc ϵ, la première lettre du mot étant différente de ف *fa* et و *wā*, il maintiendra donc toujours la tête de lecture en position initiale et passera à l'état Q3. En Q3 la situation est identique : la première lettre est différente de ك *k*, de ب *bi*, il réempruntera donc de nouveau l'arc vide et passera à Q6. À l'état Q6 il pourra enfin emprunter l'arc أ il avancera donc la tête de lecture d'un cran, vérifiera que la lettre courante est bien égale à أ et passera en Q5. En Q5 il n'existe qu'une seule transition possible vers Q6 dont l'arc est étiqueté par لْ *l*, il ne pourra donc pas l'emprunter puisque la tête de lecture se trouve actuellement sur لَ *la*. Comme Q7 n'est pas un état terminal il sera obligé de revenir en arrière, en Q6 par exemple et de réemprunter l'arc ϵ qui le fera passer directement en Q8 qui initialise une succession d'attentes correspondant à la partie radicale que nous n'explicitons pas sur ce diagramme. Une fois dépassé l'état Q3 le *lām* ne pourra plus être admis comme *lām* de corroboration. Dans la première tentative le ل ne pourra pas non plus être admis comme *lām* de corroboration puisque le parcours aura été :

(Q0), arc أ, Q1, arc ϵ, Q2, ϵ ... etc.

On pourra vérifier de même que les formes (bi prép. + coord.) ou bien لَ + TOKEN comme par exemple * لَماذا *la-limāḏā* sont également inacceptables par l'automate.

La non vocalisation revient à multiplier les parcours possibles et donc à multiplier les tests. On peut aboutir dans ce cas à des ambiguïtés plus ou moins importantes que le moniteur syntaxique sera chargé de réduire. Par exemple, en l'absence de contexte الذى pourra être interprété comme أَلذى (*hamza* interrogatif + *li* préposition + token ذى) ou bien أَلذى (*hamza* interrogatif + racine trilitère ou bien encore الذى *al-laḏī* (pronom relatif).

On voit ici que le rôle du moniteur est essentiel. En début de phrase il écartera la troisième interprétation et tâchera de lever l'ambiguïté entre la première et la deuxième en avançant dans la phrase. En milieu de phrase il écartera la première interprétation et privilégiera la troisième [10].

10. Le *hamza* interrogatif ne peut intervenir en milieu de phrase que si cette dernière contient plus d'un noyau comme on l'a vu précédemment.

Pour réduire les ambiguïtés et le nombre de faux départs qui impliqueraient des va-et-vient de l'analyseur plus ou moins importants, il est essentiel de pondérer les arcs par des probabilités — cela revient à introduire dans les arcs un ordre de priorité. Ainsi les arcs de nos diagrammes de transition seront affectés d'un coefficient proportionnel à la probabilité d'occurence de l'élément considéré (la transition). Pour établir ces coefficients, une étude importante sur corpus devra être menée. À cette fin, un programme d'aide à l'exploitation des corpus est en train d'être élaboré.

Afin d'éviter autant que faire se peut les ambiguïtés, nous avons choisi d'expliciter tous les parcours possibles dans les diagrammes de transition en les séparant, quand bien même ces derniers présenteraient des parties communes importantes; ainsi la loi de la chute du *alif* (diagramme n° 2 qui rend compte de l'incidence des affixes sur les conjugaisons) nous a semblé un critère intéressant de reconnaissance. Pour cette raison nous avons prévu le branchement $\begin{matrix}\nearrow Q1\\ \searrow Q2\end{matrix}$.

Une fois décrit ce phénomène, on pourra ensuite amalgamer la partie commune et porter l'information dans un registre en prévoyant l'utilisation d'un restricteur. Cette méthode nous fournira en outre un principe de catégorisation au niveau morphologique. Remarquons enfin que les arcs de transition de nos diagrammes ne peuvent être étiquetés que par les symboles terminaux ا ', و *w*, ى *ī*, ت *t*, م *m*, ه *h*, ل *l*, ن *n*, س *s*, ف *f*, ك *k*, ب *b*.

Il reste donc un ensemble de consonnes qui ne peuvent jamais figurer sur un arc de graphe; cela signifie qu'elles ont une valeur informative intrinsèque c'est-à-dire indépendante de leur position par rapport aux autres caractères. Il est donc intéressant de les repérer. Ces consonnes sont en effet toujours révélatrices de la partie radicale.

D'autre part la considération des graphes nous permet de dégager tout de suite certains critères simples qui impliquent à la fois la *longueur* du mot et la position de certaines lettres. On peut remarquer par exemple que la présence d'un *ka* ou d'un *ba'* en première position dans un mot est révélatrice d'un nom si celui-ci dépasse une longueur déterminée. En effet, si un verbe est contenu dans un mot graphique commençant par un ب *b*, ce dernier ne peut représenter que la première radicale de la racine (ce qui exclut les schèmes longs avec préfixe de schème). L'extension maximale d'un tel mot ne peut être que du type :

$$R_1āR_2R_3tumūh \text{ — فاعلتموه}^{11}.$$

11. En fait il ne peut s'agir que d'un verbe à la troisième forme.

La classe des mots commençant par ب ou ك dont la longueur est supérieure à 8 est donc automatiquement rangée dans la catégorie nom, par exemple :

بواليعكما *bawālī'u-kumā*

est tout de suite reconnu comme nom. Il en est de même de

بهيميتكما *bahimiyyatukuma*

qui possèdent respectivement les schèmes :

كما + فواعيل $R_1wāR_2.īR_3$ + *kumā*

et كما + فعيلية $R_1R_2R_3iyyatu$ + *kumā*

On pourra remarquer également que la présence d'un ب ou d'un ك après une consonne « solide » établira leur valeur radicale. Exemple :

ذكر *ḏakara* / دبر *dabara*

Par ces quelques exemples nous avons voulu donner une idée des critères simples qu'un programme de prétraitement du texte pourra utiliser afin de fournir rapidement des informations lexicales.

CONCLUSION

Les attentes liées aux tokens nous ont amenés à axer notre présent travail sur la reconnaissance de ces derniers. L'étude des agglutinations possibles des tokens à des antéfixes ou à des postfixes a abouti à leur classification en fonction de ces deux critères. Cette classification de type morphologique doit se combiner à celle de type syntaxique, c'est-à-dire celle liée aux attentes induites, pour aboutir à leur hiérarchisation.

D'autre part, nous nous sommes bornés ici, en vue de la reconnaissance des tokens, à étudier la morphologie des parties externes du mot, les tokens ne possédant pas de schèmes et n'étant donc pas soumis aux ruptures.

Toutefois l'explication des graphes rend compte en fait du phénomène général de l'agglutination aux formes linguistiques quelles qu'elles soient.

Il s'agit donc dans ce travail d'une morphologie externe. Une description complète de la morphologie nécessitera une étude sur les schèmes qui sera présentée ultérieurement.

Cette question a toutefois été effleurée lorsque nous avons représenté sous forme de graphes une partie des conjugaisons, dont certaines peuvent affecter — outre les parties périphériques — le corps même du schème.

En ce qui concerne la reconnaissance proprement dite des tokens, il n'est toutefois pas nécessaire de tenir compte de toutes les informations morphologiques que nous avons dégagées dans cet article. En d'autres termes il n'est pas toujours nécessaire de faire fonctionner le programme morphologique dans toute sa complexité, afin de reconnaître les tokens. Par contre cette étude fait ressortir la forte relation entre la morphologie et la syntaxe. En effet, de la simple étude des ajouts que peuvent recevoir les tokens, découlent des règles syntaxiques.

Un programme même limité de reconnaissance des tokens ne saurait manquer d'apporter une moisson d'informations immédiatement utilisables, afin d'élaborer des règles d'attentes. Ainsi les règles d'agglutination d'un token peuvent apporter non seulement un critère de reconnaissance de celui-ci, mais également une information syntaxique de haut niveau. Soit انّه *'nh.* Ce graphème pourra, selon sa position dans la phrase, représenter les tokens *'anna* ou *'inna* et donner le thème, sans ajouts possibles, et par conséquent induire directement la recherche du prédicat. Le premier introduit une subordonnée conjonctive, le second, une phrase nominale, donc un noyau.

De l'étude des diagrammes il ressort que l'on peut dégager des critères très simples permettant d'atteindre un premier niveau d'indexation du texte, sur lequel pourra s'appuyer le moniteur syntaxique pour amorcer le processus de décodage.

D'autre part, il est intéressant de noter que la structure mathématique très simple de nos diagrammes nous permettra de déterminer la démarche optimum d'analyse, indépendamment de la description linguistique (ce qui donnera la possibilité de compléter ou modifier cette dernière sans pour autant affecter la structure même du programme). Les questions de complexité algorithmique liées à la reconnaissance morphologique ainsi que le problème de la détermination du coût réel de la dévocalisation seront exposés et discutés systématiquement dans un article plus technique.