

Initiation pratique à la constitution et à l'exploitation de corpus électroniques en langue arabe (III^e partie)

Djamel Eddine Kouloughli, CNRS

Dans le précédent article de cette série, nous avons abordé les systèmes de gestion de bases de données (SGBD). Nous avons vu successivement ce qui caractérisait ce type d'outil, comment transformer un texte quelconque en base de données textuelles (BDT), et enfin quelques utilisations possibles de telles BDT pour l'exploration de corpus textuels. Nous avons pu ainsi constater que ces outils étaient beaucoup plus puissants et versatiles que les simples fonctionnalités courantes d'un logiciel de traitement de texte.

Cependant, nous avons conclu notre étude en signalant certaines des lacunes des SGBD dès que l'on veut procéder à une analyse un peu approfondie d'un ou plusieurs textes. Parmi les informations que ce type de logiciel ne fournit pas, du moins directement et sans maîtrise de la programmation, nous avons notamment évoqué les points suivants :

- connaître la fréquence de tous les mots d'un texte ;
- disposer d'une concordance des mots d'un texte, c'est-à-dire d'une liste ordonnée de tous les contextes d'apparition de chaque mot du texte ;
- savoir si un couple quelconque de mots présente une tendance à la co-occurrence, et dans quelles proportions.

C'est à de telles questions, et à d'autres du même type, que les logiciels d'analyse de corpus visent à répondre. Les premiers logiciels de ce type sont apparus à l'époque des « gros systèmes », c'est-à-dire avant la généralisation des micro-ordinateurs. Certains, par exemple le célèbre Oxford Concordance Program (OCP) développé à l'Université d'Oxford, ont ensuite été adaptés à la micro-informatique. Mais de nombreux autres logiciels d'analyse de corpus ont été spécialement développés pour les micro-ordinateurs de tous types¹. Certains de ces logiciels sont commercialisés², d'autres sont diffusés gratuitement³, d'autres enfin⁴ sont distribués gratuitement pour un usage personnel ou pour être testés, à

1. Nous n'évoquons ici, conformément au choix que nous avons indiqué dans le premier article de cette série, que des logiciels conçus pour tourner sur les micros de type PC[®]. Il existe cependant des équivalents tournant sur les systèmes Mac[®]...

2. Par exemple le logiciel CONCORDANCE, diffusé par son auteur, R.J.C. Watt, sur le site <http://www.concordancesoftware.co.uk/index.htm>

3. Par exemple l'ensemble TACT (Text Analysis Computing Tools), diffusé par l'université de Toronto, entre autres sur le site <http://www.chass.utoronto.ca/tact/>

4. Par exemple le logiciel Lexico 3, diffusé par l'équipe SYLED-CLA2T de l'Université de Paris 3 sur le site <http://www.cavi.univ-paris3.fr/ilpga/ilpga/tal/lexicoWWW/>

charge pour les utilisateurs voulant en faire un usage professionnel d'en acquitter le prix auprès des institutions propriétaires.

L'objet du présent article sera essentiellement de présenter au lecteur quelques exemples de traitements que les logiciels d'analyse de corpus rendent possibles. Mais il nous faut, auparavant, nous arrêter un moment sur certains problèmes pratiques posés par l'adaptation préalable des textes arabes aux exigences de certains de ces logiciels, notamment en ce qui concerne les contraintes relatives aux types de caractères alpha-numériques qu'ils sont capables de manipuler. Ce sera là l'objet de la première section de cet article.

1. Problèmes de translittération

Les logiciels d'analyse de corpus pour micro-ordinateurs de première génération (comme OCP ou TACT) tournaient sous le système d'exploitation MS-DOS⁵ : c'est par exemple le cas de l'ensemble TACT⁵. Les plus récents s'exécutent directement sous Windows⁶.

Un problème fondamental que pose l'utilisation de tels logiciels pour des langues comme l'arabe est celui des caractères utilisés pour coder les textes à analyser. En effet, ces logiciels, en particulier ceux de première génération, ne savent gérer que les caractères latins, c'est-à-dire, en gros, le contenu de la table ASCII « étendue » dont nous avons parlé dans le premier article de la présente série⁶. Cette contrainte, importante, est cependant à nuancer, dans le cas des logiciels les mieux conçus (et c'est le cas pour OCP et pour TACT), du fait qu'à l'intérieur de l'ensemble des caractères reconnus, toute liberté est laissée à l'utilisateur pour définir la liste et l'ordre alphabétique des symboles qu'il désire utiliser et même de différencier les lettres principales de son alphabet de symboles qui ne seront utilisés que comme simples diacritiques ou comme caractères délimiteurs (ponctuation, etc.).

Ce qui précède signifie que pour travailler sur des textes arabes à l'aide de tels logiciels, il faut se doter d'un système de translittération⁷ visant à transposer l'écriture arabe en un jeu de caractères compatibles avec ce qui est disponible dans la table ASCII « étendue ». Cette transposition n'est pas, en soi, quelque chose de très compliqué à réaliser, et les informaticiens qui ont travaillé sur l'arabe à l'époque héroïque où seuls les caractères latins étaient gérés par les ordinateurs y ont souvent eu recours. En outre, nous verrons plus loin que,

5. Mais rien n'empêche de les exécuter sous Windows[®] en mode d'émulation MS-DOS[®].

Quant aux résultats, ils s'affichent directement sous Windows[®] comme fichiers textes.

6. Voir *Langues et Littératures du Monde Arabe*, n° 5, 2006, notamment p. 239 et 247.

7. Noter que nous parlons de « translittération » et non de « transcription », ce dernier terme étant plutôt réservé à l'opération visant à donner d'une séquence graphique une représentation phonétique, laquelle n'est pas nécessairement un strict équivalent de la forme graphique. Ainsi par exemple, les verbes arabes رمى et سما seraient transcrits (phonétiquement) comme [ramā] et [samā], c'est-à-dire comme ne différant que par la première consonne. Mais une translittération doit faire aussi apparaître la différence de notation du [ā]...

conformément au vieil adage, « à quelque chose malheur est bon » : utiliser un système de translittération peut s'avérer, dans le cas du traitement de certaines données où la plus grande rigueur est indispensable⁸, une aide précieuse. Dans le même ordre d'idées, un système de translittération bien conçu permet d'explorer commodément certaines propriétés linguistiques des textes arabes que la graphie usuelle, même dans sa variante vocalisée, ne rend pas aisément perceptibles. Nous y reviendrons...

Il existe, sur le marché du traitement automatique de l'arabe, un certain nombre de systèmes de translittération de l'arabe en caractères latins, tous conçus pour répondre à des contraintes du type de celles que nous venons d'évoquer. Tous sont en principe élaborés pour donner une représentation fidèlement transposée (c'est-à-dire bi-univoque) d'un texte graphié en écriture arabe, et pour permettre, dans l'usage inverse (rétrotranslittération), de générer un texte en graphie arabe. Cependant, tous ne se valent pas si l'on se donne un certain nombre de critères visant à définir « le système de translittération idéal ». On peut, à la suite de Mostafa Banouni, Azzeddine Lazrek et Khalid Sami⁹, adopter les critères suivants :

- *non-ambiguïté* ou *bi-univocité* : l'application entre les éléments des deux alphabets est une bijection ;
- *complétude* : la translittération couvre tous les éléments du système alphabétique arabe ;
- *compacité* : le nombre de signes utilisés pour la translittération d'un élément est optimal ;
- *portabilité* : la translittération n'utilise que les caractères ASCII standards. La translittération peut être saisie à partir d'un clavier standard. Le texte saisi est transportable. Une translittération ASCII permet de conserver la portabilité du document, ainsi saisi ou transcrit, sur tout système informatique ;
- *lisibilité humaine* : le texte est lisible, il peut être facilement lu par un être humain. La translittération doit utiliser uniquement les lettres de l'alphabet cible pour ne pas s'éloigner trop de la forme phonétique d'origine ;
- *affichabilité* : le texte résultat peut être affiché sur un écran ASCII ou à l'aide d'une imprimante standard ;
- *mnémotechnicité* : la translittération est facilement mémorisable.

Les auteurs ajoutent, fort justement que « généralement, ces propriétés présentent quelques antagonismes. Il faut chercher un compromis ».

8. Nous pensons notamment au problème du traitement métrique automatique de corpus poétiques en arabe. L'expérience montre que l'usage de la graphie arabe, même attentivement vocalisée, est presque inévitablement une source d'erreur en la matière, et que seule la translittération permet un traitement fiable des données.

9. M. Banouni, A. Lazrek et K. Sami, « Une translittération arabe/roman pour un e-document » (s.d.), <http://www.ucam.ac.ma/fssm/rydarab/doc/communic/transtec.pdf>

Nous l'avons déjà dit, il existe un certain nombre de systèmes de translittération en concurrence dans le champ du traitement automatique de l'arabe, et il faudrait, en toute rigueur, pour sélectionner le meilleur, les soumettre tous aux six critères énoncés ci-dessus. Mais ce n'est pas là notre objectif dans la présente étude. Nous pensons cependant que le système de translittération TRS¹⁰ que nous avons mis au point il y a déjà plusieurs années (vers la fin des années quatre-vingt) peut être considéré comme réalisant, pour chacun des critères ci-dessus, un score très honorable. Pour le montrer, nous allons le comparer au système également mis au point il y a plusieurs années et utilisé par notre collègue Fathi Debili, spécialiste reconnu du traitement automatique de l'arabe.

Commençons, pour entrer dans le vif du sujet, par donner un court texte utilisant le système de notre collègue :

FiY biLI galGiLaTiY
 èHMD èMYN
 kalNa BaYoTuNal MaHokuWMzl BilLsULoDati lLoèaBaWiYAti. FaLlOèaBu WaHodahu MalLiku ZiMalMi èuMuWrihi, Lal TaqoruJu lLoèuMU éiLAL BiéiVoNihi, Wa Lal YaRiYBu lLoèaWoLaldu gaNi lLoBaYoTi Bagoda lLoRuruWBi qaWFzl MiNo paroBihi, Wa MalLiYAtu lLoèusorati FiY Yadihi YaçoriFu MiNohal kuLA YaWoMx Mal YaSaleu kaMal YaSaleu, Wa huWa lLAViY YaTaHakAMu HaTay FiY Mal NaèokuLu Wa Mal Lal NaèokuLu, YaSoguru SuguWrzl QaWiYAL BiWaljiBihi NaHoWa TagoLiYMi èaWoLaldihi, huWa YugaLiMuhuMo BiNaFosihi Wa YuSoriFu gaLay TagoLiYMihiMo FiY MadalrisihiMo, siWalej FiY VaLika èaBoNalùuhu Wa BaNalTuhu, Wa YuTogiBu FiY VaLika NaFosahu TagaBzl Lal Hada Lahu, HaTay LaQado YakuWNU MariYpzl FaLal Yaèobahu BiMarapihi, Wa YaTakiGu gaLay NaFosihi LiYuLoQiYa gaLaYoNal darosahu.

En voici à présent l'équivalent dans le système TRS :

<& fiy Zil*i &aalilatiy>
 <m 'aHmad 'amiyn>
 kaana baytunaa maHkuwmaN bielsulTa#i el'abawiy*a#i. fael'abu waHdahu maaliku zimaami 'umuWrihi, laa taxruju el'um*u 'il*aa bi'i@nihi, wala yagiybu el'awlaadu &an elbayti ba&da elguruwbi xawfaN min Darbihi, wamaaliy*a#u el'usra#i fiy yadihi yaSrifu minhaa kul*a yawmiN maa ya\$aa'u kamaa ya\$aa'u, wahuwa el*a@iy yataHak*amu Hat*aY fiymaa na'kulu wamaa laa na'kulu, ya\$&uru \$u&uwraN qawiy*aa biwaajibibihi naHwa ta&liymi 'awlaadihi, huwa yu&al*imuhum binafsihi wayu\$rifu &alaY ta&liymihim fiy madaarisihim, siwaa'uN fiy @aalika 'abnaaUuhu wabanaatuhu, wayut&ibu fiy @aalika nafsahu ta&abaN laa Had*a lahu, Hat*aY laqad yakuwnu mariyDaN falaa ya'bahu bimaradihi, wayat*akiLu &alaY nafsibi liyulqiya &alaynaa darsahu.

10. Le lecteur trouvera en annexe au présent article une table complète de correspondance entre les symboles du système de translittération TRS et les caractères de l'alphabet arabe.

Pour ceux qui n'auraient pas réussi à deviner la valeur de tous les symboles utilisés dans l'une ou l'autre des deux translittérations ci-dessus, voici la rétroconversion¹¹ en caractères arabes du petit passage donné en test :

<ع في ظلّ عائلي>
<م أحمد أمين>

كَانَ بَيْنَنَا مَحْكُومًا بِالسُّلْطَةِ الْأَبَوِيَّةِ. قَالَ أَبُ وَحْدَهُ مَالِكُ زَمَامُ أُمُورِهِ، لَا تَخْرُجُ الْأُمُّ إِلَّا بِإِذْنِهِ، وَلَا يَغِيبُ الْأَوْلَادُ عَنِ النَّبِيِّ بَعْدَ الْغُرُوبِ خَوْفًا مِنْ ضَرْبِهِ، وَمَالِيَةُ الْأُسْرَةِ فِي يَدِهِ بِصَرْفِ مِثْلِ كُلِّ يَوْمٍ مَا يَشَاءُ كَمَا يَشَاءُ، وَهُوَ الَّذِي يَنْحَكُمُ حَتَّى فِيمَا نَأْكُلُ وَمَا لَا نَأْكُلُ، يَشْعُرُ شُعُورًا قَوِيًّا بِوَاجِبِهِ نَحْوَ تَعْلِيمِ أَوْلَادِهِ، هُوَ يُعَلِّمُهُمْ بِنَفْسِهِ وَيُشْرِفُ عَلَى تَعْلِيمِهِمْ فِي مَدَارِسِهِمْ، سِوَاءَ فِي ذَلِكَ أَبْنَائِهِ وَبَنَاتِهِ، وَيُتَعَبُ فِي ذَلِكَ نَفْسَهُ تَعَبًا لَا حَدَّ لَهُ، حَتَّى لَقَدْ يَكُونُ مَرِيضًا فَلَا يَأْبَهُ بِمَرَضِهِ، وَيَتَّكِي عَلَى نَفْسِهِ لِيَلْقَى عَلَيْنَا دَرْسَهُ.

Que peut-on tirer de cette première comparaison entre les deux systèmes de translittération ? Sans sous-estimer la bien plus grande familiarité que nous avons avec le système TRS, il nous semble qu'en termes de lisibilité humaine, ce dernier n'a guère de difficulté à surclasser celui de notre collègue.

Si nous voulons à présent évaluer les deux systèmes par rapport aux autres critères, il semble que les deux soient non ambigus et complets mais, assez remarquablement, cela ne signifie pas qu'ils comptent le même nombre d'éléments, ni donc qu'ils soient équivalents en termes de compacité. En effet, le système de Fathi Debili compte 46 symboles alors que le système TRS n'en compte que 38. Comment cela s'explique-t-il si l'on considère que les deux systèmes doivent remplir le critère de non-ambiguïté ? L'explication est facile à déduire de l'examen attentif et systématique des deux translittérations : on constate alors que, d'une part, le premier système dispose d'un symbole spécifique pour le *sukūn* alors que le second se contente de noter la consonne sans voyelle¹², et que d'autre part, le système TRS ne dispose que d'un seul symbole pour noter partout le *tanwīn*, en rendant les séquences voyelle + *tanwīn* par la concaténation de deux symboles, alors que le système de Fathi Debili, calquant fidèlement le système graphique arabe, associe un graphème séparé à chacune des séquences voyelle + *tanwīn*. Il en va de même pour les séquences *šadda* + voyelle (+ *tanwīn*). Ainsi, le système de Fathi Debili peut sembler mieux respecter le critère de bi-univocité avec le système graphique arabe, mais au prix d'une moins grande économie. On pourrait aisément montrer que cette plus grande fidélité se paie par

11. Rétroconversion effectuée automatiquement à partir du texte TRS ci-dessus à l'aide d'une table de conversion. Noter les chevrons qui encadrent le titre et l'auteur ainsi que les lettres qui les précèdent : ils font partie du système de balisage des corpus textuels dont nous parlerons ultérieurement.

12. Cette non-notation du *sukūn* dans la translittération TRS n'empêche pas la table de conversion de le restituer en écriture arabe, comme on peut le constater en examinant le texte arabe rétroconverti. Cela est l'effet d'une règle de transcription très simple qui stipule d'insérer un *sukūn* après chaque consonne non suivie de voyelle.

une perte de généralité linguistique de ce système, puisque le *tanwīn* comme marqueur spécifique de l'indéfini ne peut pas y être directement appréhendé.

Pour ce qui est du critère de mnémotechnicité, il nous semble également que le système TRS est meilleur, non seulement en ce qu'il utilise moins de symboles, demandant ainsi moins d'efforts à la mémoire, mais aussi en ce qu'il utilise de façon méthodique et intuitive l'opposition majuscules/minuscules et fait un recours minimal et, nous semble-t-il, aisément mémorisable à des caractères « spéciaux ». On notera en passant que dans ce système, on ne note pas le 'alif de l'accusatif indéfini (*tanwīn naṣb*) car sa présence dans l'écriture arabe est entièrement prédictible et peut donc, en rétroconversion, être restituée par une règle contextuelle simple. Cela rajoute, bien sûr, à l'économie du système.

Signalons enfin, pour conclure cette section sur les systèmes de translittération, que le processus de rétroconversion peut, bien évidemment, s'appliquer non seulement au texte initial du processus d'analyse de corpus, mais aussi à tous les fichiers résultant de ce processus, en sorte que l'on peut présenter au lecteur arabisant auquel on s'adresse un ensemble de données (texte à analyser et résultats de l'analyse) qui, de bout en bout, est en graphie arabe. Le passage par la translittération n'est alors qu'une sorte de « boîte noire » dont l'utilisateur final peut ne rien savoir. Cette situation n'est, au fond, pas très différente de celle que nous avons vue concernant le codage et la gestion en machine de l'écriture arabe.

2. Quelques possibilités des logiciels d'analyse de corpus

Nous pouvons à présent aborder directement les questions relatives aux principales possibilités qu'offrent les logiciels d'analyse de corpus pour l'exploration des textes. Conformément à la démarche que nous avons adoptée dès le début pour cette série d'articles, nous procéderons à partir de l'étude de cas concrets. Cela signifie que nous travaillerons à chaque fois sur un cas particulier en abordant les problèmes spécifiques qu'il pose avant d'en tirer des conclusions plus générales.

Dans cette optique, les deux éléments dont nous partirons et qu'il convient donc tout de suite de préciser sont : un logiciel spécifique d'analyse de corpus et un texte spécifique. Pour le logiciel, nous commencerons par l'ensemble TACT, qui a l'avantage d'être gratuit tout en présentant des fonctionnalités assez complètes d'analyse de corpus. Pour ce qui est du texte, nous prendrons un article intitulé لغة النح (*luga#u elnaHli*) publié dans le numéro de septembre-octobre 1988 de la revue¹³ القافلة par le دكتور محمد ياسر حماد سليمان. Nous avons directement saisi ce texte en TRS et l'avons vocalisé. Vocaliser des textes arabes soumis à l'analyse de corpus repose sur des choix linguistiques et didactiques sur lesquels nous reviendrons ultérieurement.

L'ensemble TACT est d'un usage assez simple mais exige, comme la plupart des logiciels d'analyse de corpus, de définir au préalable un certain nombre de

13. Revue gratuite de l'Aramco qui a l'avantage de présenter des articles de qualité et libres de droits à condition d'en citer la source.

paramètres qui vont déterminer les conditions dans lesquelles va se faire l'analyse et les résultats qui seront obtenus en sortie.

Parmi ces paramètres, le premier, auquel nous avons déjà fait allusion, concerne la définition de l'alphabet que l'on demande à TACT d'utiliser pour analyser le texte qui lui est soumis. Cette définition est contenue dans un fichier texte ayant un format spécifique et caractérisé par l'extension « .MKS ». Ce fichier est nommé *a priori* « DEFAULT.MKS », mais peut recevoir un nom différent¹⁴. Pour notre propos, nous avons défini un fichier « TRS.MKS » qui a pour contenu essentiel de définir l'alphabet utilisé dans le système TRS. En voici le contenu :

```
[Alpha]
"'"eIUybt#cJHxd@rzs$SDTZ&gfqklmnhwya i u N
0 1 2 3 4 5 6 7 8 9
[DiacRet]
- \Acute\ \Grave\ \Circumflex\ \Cedilla\ \Umlaut\ \Tilde\ ""
[WordSepSeq]
-
[ReferenceBracket]
< > NoSupText NoWordSep
```

Les principales informations que l'on peut tirer de ce fichier est qu'il donne, sous la rubrique [Alpha] la liste, dans l'ordre alphabétique arabe, des symboles graphiques utilisés par le système TRS, sous la rubrique [DiacRet] la liste des symboles utilisés comme diacritiques (on y notera notamment le « * » représentant la *šadda* dans le système TRS), et enfin sous la rubrique [ReferenceBracket] les chevrons « < » et « > » qui constituent les « balises » permettant d'associer diverses étiquettes aux parties du texte que l'on souhaite faire identifier au logiciel. Ces étiquettes ont, pour certains types de traitement, une importance cruciale, et nous y reviendrons donc au moment opportun. En attendant, signalons simplement que dans le texte que nous allons soumettre à présent à l'analyse, on a simplement défini, au début du texte, les trois étiquettes suivantes :

<& luga#u elnaHli>

<m d. muHam*ad yaasir Ham*aad sulaymaan>

<j elqaafila#, sibtambir-'uktuwbir 1988, S. 38-40>

Ces trois étiquettes identifient, respectivement, le titre de l'article (& pour *&unwaan*), l'auteur (m pour *muUal*if*) et la revue dont est tiré le texte (j pour *ja-riyda#*). Dans un corpus qui comporterait plusieurs textes (ce qui est plutôt la règle que l'exception), ces étiquettes permettraient, on l'aura compris, d'identifier clairement les coordonnées de chacun des textes concernés.

14. En fait, un fichier « DEFAULT.MKS » est livré avec l'ensemble logiciel : il est défini pour fonctionner avec l'alphabet anglais. On peut définir autant de fichiers « .MKS » qu'on le désire. *A priori*, à chaque fichier texte peut être associé un fichier spécifique de ce type.

Pour revenir à TACT, précisons que si nous en avons parlé jusqu'ici comme d'un « ensemble », c'est que, de fait, il s'agit d'un groupe de seize programmes distincts mais interdépendants conçus chacun pour traiter un aspect de l'analyse de corpus textuels. Le premier de ces programmes, d'une certaine manière le plus fondamental, s'appelle MAKEBASE et consiste, comme son nom le suggère, à associer au texte qu'on lui donne en entrée (accompagné, rappelons-le, du fichier de définitions « .MKS »), une base de données textuelles (qui reprend normalement le nom du texte avec l'extension « .TDB »). Le lecteur qui a bien saisi les notions que nous avons développées dans la précédente livraison de cette série n'aura guère de mal à comprendre en quoi c'est une stratégie tout à fait naturelle que d'associer, comme toute première étape à une analyse détaillée et systématique d'un texte, une BDT qui permet de disposer, pour chacune des unités constitutives de ce texte, de ses coordonnées précises.

Une fois constituée la BDT associée au texte, MAKEBASE a rempli sa mission, et ce sont les autres programmes de l'ensemble logiciel TACT qui peuvent prendre le relai. Nous détaillerons dans ce qui suit ceux des programmes que nous mettrons à contribution dans ce premier survol des grandes fonctionnalités d'un système d'analyse de corpus.

Voici une copie d'écran affichée lors de l'appel du programme MAKEBASE :

```

                                MakeBase Version 2.1.4    <June 1995>

Input Text Filename:    NAHL.TXT
.MKS Filename:          TRS.MKS
.TDB Filename:          NAHL.TDB
.TDB Title:
Starting Ref. Template:
Temporary File Area:

                                Copyright (c) 1995 TACT Group, University of Toronto

F1:  Help                      F5:  Save markup <^Default>
F2:  Alphabetic Characters     F6:  Credits
F3:  Reserved Characters       F7:  RUN, creating a database
F4:  Reference Tags           F10: QUIT without processing

```

L'appui sur la touche F9 provoque l'exécution du programme et la génération d'un fichier « NAHL.TDB ». C'est ce fichier qui va servir d'in-put aux autres programmes de l'ensemble TACT.

La première question que l'on est susceptible de se poser est celle des caractéristiques générales du texte que l'on soumet à l'analyse : le nombre de mots qu'il contient (en termes techniques, le nombre de ses « occurrences » ou, en anglais, de ses « tokens »), le nombre des unités distinctes qui le constituent (le nombre de ses « formes », en anglais « types »), le nombre de ses « hapax », c'est-à-dire des mots qui n'apparaissent qu'une fois dans le texte, etc. L'ensemble de ces questions est du ressort du programme TACTSTAT, dont voici une copie de l'écran d'appel :



L'appui sur la touche F9 provoque l'exécution du programme TACTSTAT qui, conformément à notre demande, génère alors sur disque le fichier de résultats « naHl.STA ». Ce fichier contient, réparties sur plusieurs pages-écran, les réponses à toutes les questions que nous venons d'évoquer concernant les propriétés générales du fichier « naHl.txt » et bien d'autres encore... Voyons plutôt.

Sur une première page, on a un tableau présentant, sur 8 colonnes et un nombre variable de lignes (dans notre cas 22, correspondant aux 22 rangs de fréquence des formes observées dans le corpus), la répartition statistique des fréquences de toutes les formes du texte :

Frequency Rank	Observed Freq. of Rank	Words in Frequency	Types Total	Tokens Total	% of Types	% of Tokens	% of word in freq.
1	397	397	397	397	70.02	34.11	34.11
2	96	192	493	589	86.95	50.60	16.49
3	27	81	520	670	91.71	57.56	6.96
4	11	44	531	714	93.65	61.34	3.78
5	6	30	537	744	94.71	63.92	2.58
6	3	18	540	762	95.24	65.46	1.55
7	7	49	547	811	96.47	69.67	4.21
8	2	16	549	827	96.83	71.05	1.37
9	2	18	551	845	97.18	72.59	1.55
10	2	20	553	865	97.53	74.31	1.72
11	1	11	554	876	97.71	75.26	0.95
12	1	12	555	888	97.88	76.29	1.03
13	1	13	556	901	98.06	77.41	1.12
14	1	14	557	915	98.24	78.61	1.20
15	1	15	558	930	98.41	79.90	1.29
16	1	16	559	946	98.59	81.27	1.37
17	1	17	560	963	98.77	82.73	1.46
19	1	19	561	982	98.94	84.36	1.63
26	1	26	562	1008	99.12	86.60	2.23
28	3	84	565	1092	99.65	93.81	7.22
29	1	29	566	1121	99.82	96.31	2.49
43	1	43	567	1164	100.00	100.00	3.69

Ce tableau permet, par exemple, de constater qu'il y a dans le texte 397 mots qui n'apparaissent qu'une fois (ce que l'on appelle des « hapax legomena »), et un mot qui apparaît 43 fois¹⁵.

Suite à ce premier tableau, TACTSTAT génère quelques lignes de synthèse récapitulant les principales caractéristiques lexico-statistiques du texte analysé. En voici le contenu concernant le texte « naHl.txt » :

15. Pour savoir quels sont exactement ces mots, il faut exécuter le programme TACTFREQ, qui peut générer la liste fréquentielle des mots du texte. On constate alors, par exemple, que le mot qui apparaît 43 fois dans le texte est la préposition *min*. Nous reviendrons ultérieurement sur les détails de la structure lexicale du texte...

Number of Types	=	567
Number of Tokens	=	1164
Type/Token ratio	=	0.487
Token/Type ratio	=	2.053
Hapax Legomena	=	397
Hapax Dislegomena	=	96
Hapax Legomena/Dislegomena ratio	=	4.1354
Hapax Legomena/Number of Types	=	0.7002
Hapax Legomena/Number of Tokens	=	0.3411
Hapax Legomena cubed/Types squared	=	194.6280
Variance (S.D. squared)	=	13.0396
Standard Deviation (S.D.)	=	3.6110
Coefficient of skewness	=	6.5698
Coefficient of kurtosis	=	52.0471
Herdan's characteristic	=	0.0739
Yule's characteristic	=	721.0886
Carroll TTR (Types / Sqrt of 2 X Tokens)	=	11.7515
Most Frequent word "min" occurred 43 times		
repeat rate (Tokens / frequency most frequent word)	=	27.0698

Cette synthèse nous apprend notamment qu'il y a, dans ce texte, 1 164 occurrences mais seulement 567 formes différentes. Les deux rapports suivants, occurrences/formes et l'inverse, sont supposés donner une première idée de la « richesse lexicale » du texte : on comprend intuitivement que plus le nombre de formes est élevé par rapport au nombre d'occurrences, moins il y a de répétitions dans le texte, et plus ce dernier peut être considéré comme « riche » du point de vue lexical. En fait, la question est plus complexe que cette première idée ne le laisserait supposer et les lexico-statisticiens en ont longuement débattu. Les diverses mesures de « déviation » données dans les dernières lignes de cette synthèse sont précisément supposées apporter des évaluations plus précises de la manière dont se répartissent les unités lexicales dans le texte. Nous renvoyons le lecteur intéressé aux très nombreuses études réalisées sur ce sujet¹⁶.

Le tableau suivant généré par TACTSTAT donne une représentation graphique de la répartition des mots du texte par longueur (c'est-à-dire par nombre de caractères). Voici ce tableau :

Word Length Statistics			

Word Len	Freq.	%	Percentage
			10 20 30 40 50
1	7	0.60	*
2	0	0.00	
3	108	9.28	*****
4	33	2.84	***
5	143	12.29	*****
6	92	7.90	*****
7	183	15.72	*****
8	122	10.48	*****
9	179	15.38	*****
10	123	10.57	*****
11	97	8.33	*****
12	37	3.18	***
13	18	1.55	**
14	15	1.29	*
15	6	0.52	*
16	0	0.00	
17	0	0.00	
18	0	0.00	
19	1	0.09	

16. Par exemple Muller (1977) et Lebart & Salem (1988).

Ce tableau nous apprend, par exemple, qu'il y a dans le texte 7 mots d'une lettre¹⁷, aucun de deux, 108 de trois. Nous apprenons aussi que la longueur la plus fréquente des mots du texte est de 7 lettres (183 mots, avec une fréquence de 15,62 %), suivie de près par la longueur 9 (179 mots, avec une fréquence de 15,38 %).

Les autres tableaux générés par TACTSTAT concernent la distribution des lettres dans le texte. Tous les aspects de cette distribution sont envisagés (fréquence à l'initiale des mots, à la finale des mots, fréquence globale). Voici le tableau qui donne la fréquence globale de la répartition des lettres dans le texte :

Letter	Freq.	%	Percentage				
			10	20	30	40	50
a	2414	27.15	*****				
i	1109	12.47	*****				
l	635	7.14	*****				
u	377	4.24	****				
e	361	4.06	****				
m	310	3.49	***				
n	287	3.23	***				
y	278	3.13	***				
t	260	2.92	***				
#	252	2.83	***				
r	224	2.52	***				
'	203	2.28	**				
d	189	2.13	**				
h	184	2.07	**				
&	162	1.82	**				
w	160	1.80	**				
f	147	1.65	**				
N	137	1.54	**				
b	120	1.35	*				
@	110	1.24	*				
k	107	1.20	*				
q	102	1.15	*				
S	102	1.15	*				
H	94	1.06	*				
s	81	0.91	*				
\$	75	0.84	*				
x	73	0.82	*				
j	72	0.81	*				
Y	71	0.80	*				
g	66	0.74	*				
z	36	0.40					
c	25	0.28					
I	22	0.25					
T	18	0.20					
Z	18	0.20					
D	7	0.08					
1	1	0.01					
2	1	0.01					
3	1	0.01					
U	1	0.01					

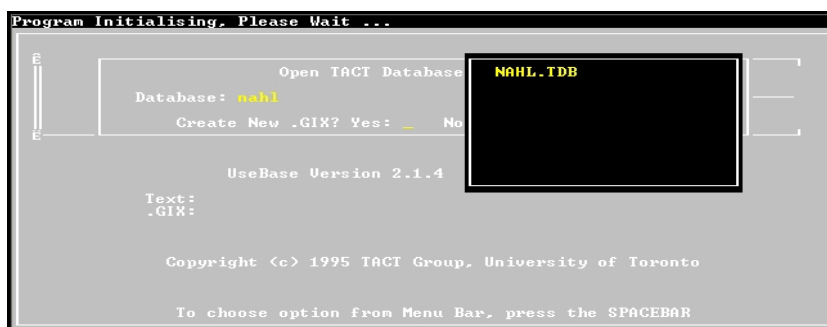
Ce tableau nous apprend que dans ce texte, qui est, rappelons-le, en arabe vocalisé, les lettres les plus fréquentes sont les voyelles <a> (près de 30 % !), puis <i> (près de 13 %) suivies de la consonne <l> (un peu plus de 7 %) puis de <u> (un peu plus de 4 %) et de 'alif <e> (~4 %).

Comme on peut le voir, le programme TACTSTAT donne sur les caractéristiques statistiques globales du texte analysé des renseignements d'ensemble ex-

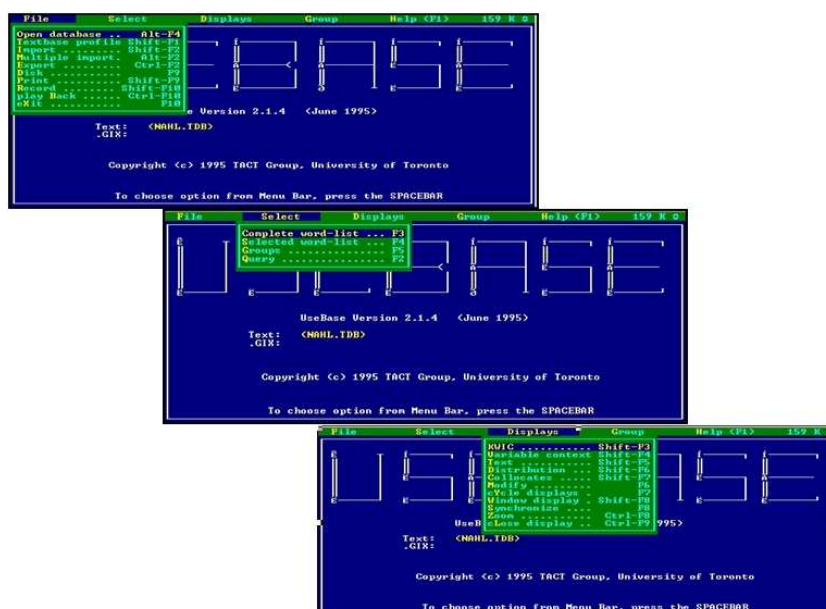
17. En principe, la phonographématique arabe interdit les mots d'une seule syllabe, donc, en arabe vocalisé, de moins de trois segments graphiques. L'absence de mots de longueur 2 confirme ce principe, mais on peut être surpris d'apprendre qu'il existe des mots de longueur 1. Le programme USEBASE que nous verrons plus loin donne la réponse : les mots en question sont les lettres « /b/j/d » et les nombres « 1/2/3 » utilisés pour référencer des figures dans le texte et traités, bien sûr, comme des « mots » par le programme.

ceptionnellement riches et précis. Ces renseignements restent néanmoins « globaux » dans la mesure où ils n'identifient aucune unité lexicale en tant que telle. Ce sont les autres programmes de l'ensemble TACT qui vont mieux nous renseigner sur ces aspects du texte.

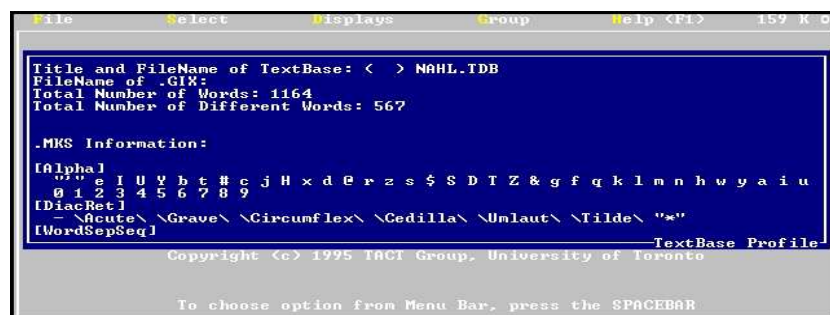
Passons à présent à ce qui constitue sans doute le programme le plus important de l'ensemble TACT après MAKEBASE, à savoir le bien nommé USEBASE. Voici une copie de son écran d'appel :



Une fois la base de données « nahl.TDB » chargée, on se trouve devant un écran *a priori* peu loquace mais qui, en réalité, recèle une multitude de ressources dans ses menus déroulants. En voici un aperçu qui restera relativement sommaire dans la mesure où l'exploration de toutes les ressources de ce programme exigerait de trop longs développements :

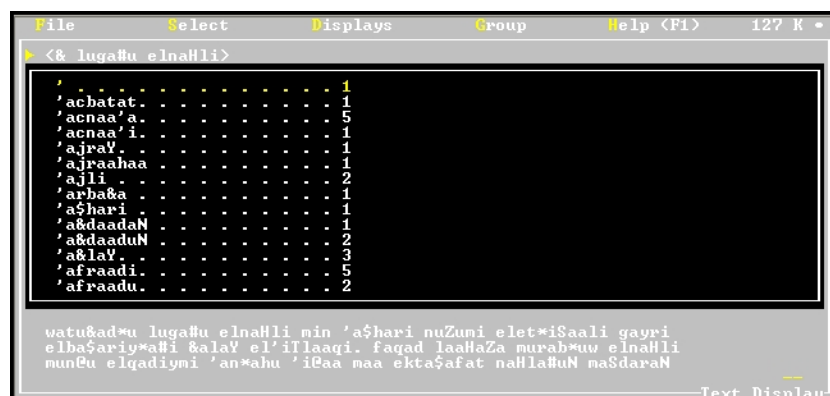


Dans l'image ci-dessus, nous avons juxtaposé les trois menus déroulants les plus importants qu'offre USEBASE. Le premier offre, entre autres, l'option « textbase profile » (SHIFT-F1) qui, quand elle est sélectionnée, nous donne les renseignements suivants :

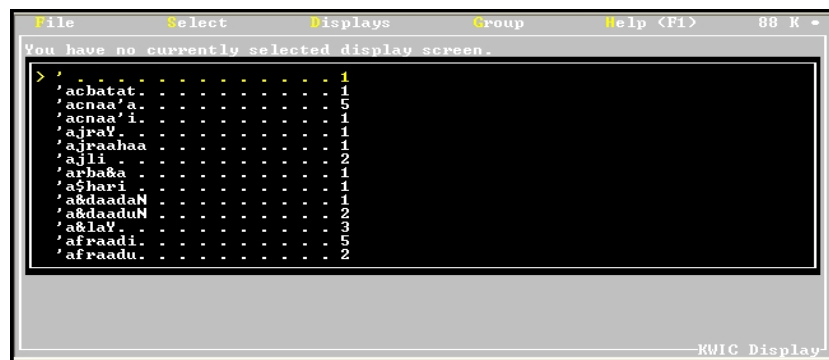


Ces renseignements nous indiquent d'emblée que le nombre total des mots du texte (ou « occurrences ») est de 1 164 et que le nombre des mots différents (ou « types ») est de 567, ce que le programme TACTSTAT nous avait déjà appris. On verra néanmoins plus loin que cette information globale peut être spécifiée dans tout le détail voulu par USEBASE...

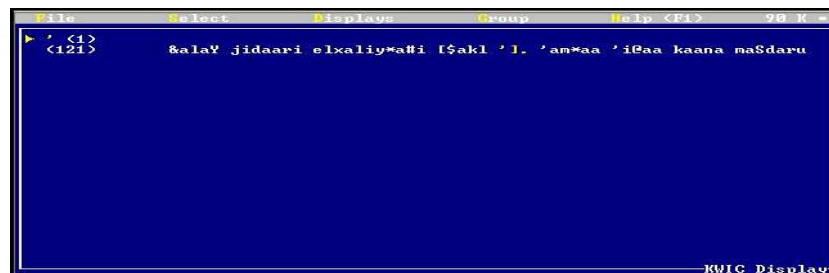
Le second menu nous offre, comme premier choix (F3), de nous présenter la « liste complète des mots ». Si l'on choisit cette option, on obtient l'écran suivant :



Cette liste, que l'on peut dérouler à l'écran jusqu'au dernier mot du texte, présente, on le voit, la liste des mots du texte dans l'ordre de l'alphabet arabe (tel que spécifié dans le fichier « TRS.MKS ») avec, en face, la fréquence de chaque mot dans le texte. On peut constater ici que le premier mot est en couleur jaune, ce qui sert à le désigner comme l'unité actuelle « pointée » dans la BDT (il s'agit en l'occurrence de la lettre *hamza*, notée « ' » et qui se trouve avoir été utilisée une fois dans le texte dans l'expression « \$akl ' » pour désigner une figure dans le texte). En déplaçant le curseur le long de la liste, on déplace le « pointeur » de couleur jaune. Si l'on désire sélectionner le mot pointé, il suffit d'appuyer sur la touche « insert » du clavier. Un chevron apparaît alors devant le mot en question :



et le programme rassemble alors instantanément toutes les données concernant ce mot dans le texte. On peut s'en assurer en entrant simplement un « retour-chariot », auquel USEBASE réagit en affichant la ligne du texte dans laquelle figure le mot sélectionné :



On peut également faire afficher un contexte plus long que la ligne. Pour cela, il faut avoir préalablement sélectionné, dans le troisième menu déroulant de USEBASE, le menu « display », l'option « variable context » (Shift-F4).



L'option d'affichage sélectionnée par défaut (car la première du menu déroulant) est l'option « KWIC », qu'il faut lire « keyword in context » et qui correspond à ce qui est généralement désigné comme « concordance ». Cette option est la plus suggestive lorsqu'on a affaire à un mot à occurrences multiples dans le texte,

comme c'est le cas par exemple pour le troisième mot de la liste complète, le mot « 'acnaa'a », qui apparaît cinq fois dans le texte. Voici sa concordance :

```

> 'acnaa'a <5>
<84>      elnaHla#u fid#a#a mar#aatiN 'acnaa'a qiyaanihaa
<97>      nuwaajihu Zaahira#aN mumaacila#aN 'acnaa'a diraa#ati#a lila#aati
<103>     elka#a#a#a#a tabuz#a Banabaha 'acnaa'a tanfiy#a#a#a eljuz'a
<147>     ele#t#a#a#i et#i#a#a#a elriy#i 'acnaa'a tarjanatihaa
<164>     min fid#a#a#i ehtizaazi Banabihaa 'acnaa'a ta'diy#a#i raq#atihaa.

```

Dans cet affichage, les numéros notés entre chevrons en début de ligne sont les numéros des lignes du texte, donnés par défaut comme moyen de repérage des diverses occurrences des mots sélectionnés. Un balisage plus spécifique peut être sélectionné pour effectuer ce repérage. Nous y reviendrons ailleurs.

Revenons un instant sur le deuxième menu déroulant pour dire un mot sur l'une des fonctions les plus puissantes et les plus intéressantes du programme USEBASE, la fonction de recherche « QUERY ». Cette fonction, comme son nom le suggère, permet à l'utilisateur de demander au programme de rechercher dans le texte toutes les occurrences d'une forme donnée et d'en constituer un groupe qui sera susceptible ensuite de toutes sortes d'explorations.

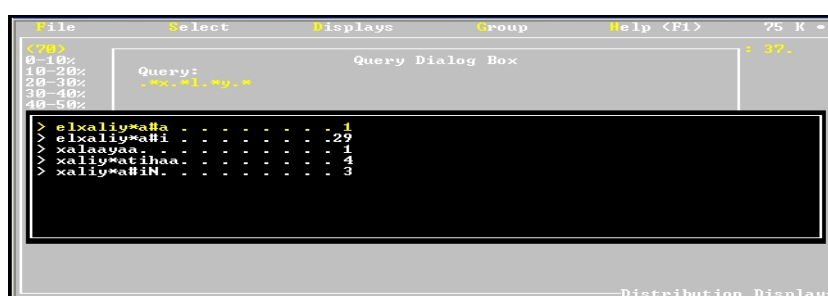
La forme la plus banale de la fonction « QUERY » est celle qui consiste à y entrer une chaîne de caractères entièrement spécifiée, comme par exemple « el-xaliy*a#u ». Mais se contenter de ce type d'interrogation, c'est passer à côté de ce qui fait toute la puissance de cette fonction de recherche de chaîne de caractères dans un texte. En effet, la fonction « QUERY » comprend ce que l'on appelle les « expressions régulières », un moyen extrêmement puissant de faire référence à des familles entières de chaînes de caractères en ne spécifiant que les éléments qu'elles ont en commun et en complétant l'interrogation par divers caractères « jokers ». Une présentation détaillée de la syntaxe des expressions régulières nous éloignerait beaucoup trop de l'objet principal de ce papier¹⁸, et nous nous contenterons donc d'un exemple qui donnera une idée concrète de la puissance de cet outil et de son utilité.

Dans le texte qui nous a servi jusqu'ici, il est souvent question de « cellule » (xaliy*a#, xalaayaa), ce qui ne saurait surprendre dans un texte s'intéressant à la vie des abeilles. Supposons que nous désirions recenser toutes les occurrences de ce mot, au singulier comme au pluriel et, bien sûr, à tous les cas et tous les statuts déterminatifs. On pourrait bien sûr, assez fastidieusement, rechercher dans la liste complète des mots du texte toutes les occurrences, les sélectionner pour en constituer un groupe, puis travailler enfin sur ce groupe. Mais il serait infiniment plus élégant, plus rapide et plus sûr, en termes d'exhaustivité, de pouvoir deman-

18. Sur cette question, on pourra consulter le très bon article introductif suivant : http://en.wikipedia.org/wiki/Regular_expression

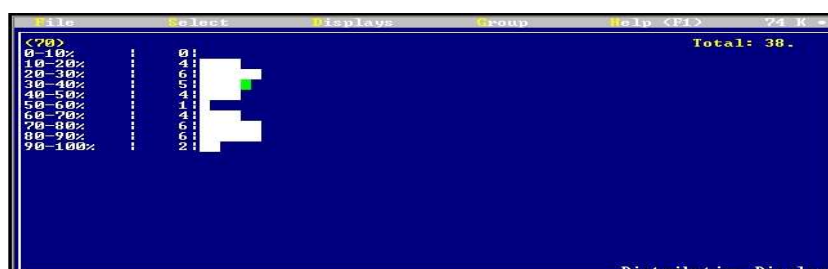
der au programme de nous sortir tous les mots de la racine $\sqrt{xly}$. Oui, mais, me direz-vous, nous n'avons pas de système d'analyse morphologique des mots arabes permettant d'en identifier la racine.

Certes, mais c'est là que la syntaxe des expressions régulières entre en jeu. En effet, un mot de la racine $\sqrt{xly}$, ce n'est au fond rien d'autre qu'un mot dans lequel on est susceptible de rencontrer les trois radicales « x », « l » et « y », dans cet ordre, précédées et/ou séparées par un nombre quelconque de segments autres. C'est là, typiquement, ce que les expressions régulières sont faites pour exprimer. Voici l'expression régulière qu'il faut soumettre à la fonction « QUERY » pour obtenir le résultat recherché :



On constate que l'expression « `.*x.*l.*y.*` » renvoie bien toutes les occurrences de mots contenant la racine $\sqrt{xly}$, quelles qu'en soient les manifestations. On peut alors travailler sur cette famille morphologique de mots de façon systématique.

Terminons ce survol par l'examen d'une fonction extrêmement suggestive du programme USEBASE, une fonction proposée dans le troisième menu déroulant, le menu « display » : il s'agit de la fonction « distribution » (Shift-F6). Cette fonction donne une représentation graphique (sous forme d'histogramme) de la distribution dans le texte du ou des mots sélectionnés. Par défaut, l'histogramme divise le texte en tranches de 10 %, mais l'utilisateur peut spécifier un autre découpage. Voici, à titre d'illustration, la distribution textuelle des mots de la racine $\sqrt{xly}$ que nous venons de sélectionner :



Cette représentation permet de constater, dans le cas d'espèce, que les mots de la racine $\sqrt{xly}$ sont assez régulièrement répartis dans l'ensemble du texte, ce qui est souvent caractéristique de mots fortement « thématiques » (quand il ne s'agit pas de mots grammaticaux).

Nous en resterons là pour ce premier survol de quelques-unes des grandes fonctionnalités d'un logiciel d'analyse de corpus. Nous espérons avoir d'ores et déjà convaincu le lecteur que ces fonctionnalités ouvrent des perspectives très suggestives pour l'analyse des corpus textuels. Dans le prochain article, nous examinerons plus en détail certaines des explorations de textes que de tels outils rendent possibles.

Annexe
Table de conversion du systeme trs

Transcription	Caractère arabe	Transcription	Caractère arabe
'	ء	&	ع
I	ئ	g	غ
U	ؤ	f	ف
e	ا	q	ق
Y	ى	k	ك
b	ب	l	ل
t	ت	m	م
#	ة	n	ن
c	ث	h	ه
j	ج	w	و
H	ح	y	ي
x	خ	*	ء
d	د	a	ا
@	ذ	i	ا
r	ر	u	و
z	ز	aa	ا
s	س	iy	ي
\$	ش	uw	و
S	ص	aN	ا
D	ض	iN	ا
T	ط	uN	و
Z	ظ	aYN	ي

Références bibliographiques

- HABERT B., SALEM A., NAZARENKO A., 1997, *Les linguistiques de corpus*, Paris, Armand Colin.
- LAFON P., 1984, *Dépouillements et statistiques en lexicométrie*, Paris, Genève, Slatkine-Champion.
- LEBART L., SALEM A., 1988, *Analyse statistique des données textuelles*, Paris, Dunod.
- LANCASHIRE I. et al., 1996, *Using TACT with Electronic Texts*, New York, The Modern Language Association of America.
- MC ENERY T., WILSON A., 1996, *Corpus Linguistics*, Edinburgh, Edinburgh University Press.
- MULLER C., 1977, *Principes et méthodes de statistique lexicale*, Paris, Hachette.
- SMITH G.W., 1991, *Computers and Human Language*, Oxford, Oxford University Press.