

# Vers un module Tamazight pour le système NooJ

Farida AOUGHLIS  
FGEI, Université Mouloud Mammeri Tizi Ouzou, Algérie  
[fariyamo@yahoo.fr](mailto:fariyamo@yahoo.fr)

## Introduction

NooJ est une plate-forme de développement permettant la description des langues naturelles ou techniques, elle est d'usage libre. Dix neuf langues sont déjà disponibles, notre objectif est de rajouter Tamazight.

### 1. NooJ<sup>1</sup> : une plateforme de développement linguistique

NooJ est un environnement linguistique de développement (Silberztein, M.). Les fonctionnalités de NooJ sont adaptées à un public très varié incluant des linguistes (description de la morphologie et de la syntaxe des langues, analyse de corpus), des informaticiens du TAL (applications d'extraction d'information), ainsi que des enseignants tant en Linguistique qu'en enseignement des langues. Nous citerons par exemple l'utilisation de NooJ pour l'enseignement des langues, plus particulièrement l'apprentissage du français langue étrangère (Silberztein M. et Tutin A. 2005).

NooJ est utilisé pour la construction à grande couverture de descriptions formalisées des langues naturelles dans le but de les appliquer en temps réel à de gros corpus. Les descriptions des langues naturelles sont formalisées dans des dictionnaires électroniques par des grammaires représentées par des ensembles de graphes.

NooJ fournit des outils pour décrire :

- la morphologie flexionnelle et dérivationnelle,
- les variations terminologiques et orthographiques,
- le vocabulaire (mots simples, mots composés, expressions figées),
- la syntaxe
- et la sémantique.

#### 1.1 Architecture intégrée

Dans NooJ, toutes les connaissances linguistiques sont séparées de l'algorithme d'analyse lui-même. L'architecture globale de NooJ est basée sur un ensemble de modules linguistiques : orthographique,

flexionnel, morphologique, dérivationnel et syntactico-sémantique. Un ensemble d'outils est fourni :

- un éditeur de graphes,
- un concordancier,
- un débogueur de grammaires,
- des outils statistiques,
- etc.

Cet ensemble en interaction permet de répondre aux besoins les plus pointus des utilisateurs de NooJ et notamment des spécialistes du traitement des langues (TAL).

### **1.2 Architecture orientée objet**

NooJ est développé sous une architecture orientée objets (Silberztein M. 2004). Les composants du système (objets) intègrent les données ainsi que les routines nécessaires pour leurs traitements. La communication et la coordination entre les objets sont réalisées par un mécanisme interne. Cette architecture présente les avantages suivants :

- Eviter la redondance dans le code source grâce au concept d'héritage. Ceci rend le code source plus lisible et gérable ;
- Accéder directement à toutes les méthodes publiques dans NooJ, au lieu d'avoir à sélectionner un ensemble de sous-programmes. Ceci apporte une flexibilité et une utilisabilité supplémentaires au système ;
- Pouvoir ajouter d'autres fonctionnalités supplémentaires ou des méthodes spécifiques à des langues sans avoir à modifier l'architecture globale de l'application.

### **1.3 Utilisation de la technologie à états finis**

NooJ contient des outils permettant d'éditer, de tester, de déboguer et de gérer des ensembles importants de graphes à états finis. Tous les objets traités par NooJ sont ou peuvent être transformés sous forme de transducteurs à nombre fini d'états. Toutes les opérations sur les textes, grammaires et dictionnaires se ramènent ainsi à des opérations sur des transducteurs. NooJ utilise aussi des outils plus puissants comme les graphes récursifs qui sont équivalents à des grammaires hors-contexte.

NooJ ne traite pas que des graphes d'états finis, avec NooJ on peut aussi créer :

- Des grammaires algébriques ;
- Des graphes récursifs : RTN (Recursive Transition Network).

#### **1.4 Développement de ressources linguistiques à large couverture**

Les différentes ressources linguistiques sont décrites dans des structures de données autonomes (Silberztein, 2005). En l'occurrence, elles sont représentées soit dans des formats textuels lisibles et accessibles par un non-informaticien (comme pour les dictionnaires et les fichiers des paradigmes flexionnels et dérivationnels), soit sous forme de graphes facilement accessibles et compréhensibles qui peuvent aisément être contrôlés et modifiés sans avoir à maîtriser l'ensemble du programme (comme pour les grammaires flexionnelles, morphologiques et syntaxiques).

#### **1.5 Traitement de corpus**

NooJ peut traiter directement des ensembles potentiellement importants de documents. Ces documents peuvent être codés dans plus d'une centaine de formats :

- tous les formats des types ASCII, EBCDIC, Unicode (UTF-8, UTF-16, etc.),
- les formats standards tels que HTML (format par défaut des textes disponibles en ligne), XML (format de bases de données textuelles), PDF (format de documents portables), RTF (format de textes enrichis), Microsoft WORD, etc.

#### **1.6 Construction, édition et gestion de concordances sophistiquées**

NooJ permet de construire et de gérer des concordances sophistiquées. Il permet de combiner plusieurs requêtes, filtrer manuellement les résultats incorrects et de sauvegarder la concordance résultante. Les outils proposés pour la gestion des concordances permettent d'aborder certains problèmes linguistiques de façon incrémentale.

Par exemple, il est envisageable de commencer son travail avec NooJ par l'identification de certains termes ou expressions dans un corpus, tout d'abord de façon naïve (par exemple, rechercher tous les mots finissant par -ment pour identifier des adverbes en français, ou rechercher tous les nombres pour identifier des dates). Puis, au vu de

la concordance résultante, il est possible de raffiner progressivement la caractérisation des phénomènes, d'une part en améliorant les requêtes (par ex. en tenant compte du contexte des mots), d'autre part en classifiant les résultats obtenus (par ex. pour distinguer les adverbes en –ment, par exemple "régulièrement", des noms en "–ment", par ex. "investissement"). Les résultats des requêtes peuvent être facilement contrôlés.

### **1.7 Annotation interactive de corpus**

NooJ permet d'appliquer des grammaires représentées sous forme de graphes ou d'expressions rationnelles augmentées à un corpus. Le résultat est affiché dans une table de concordances où la colonne du milieu contient le mot ou la séquence de mots reconnus. L'utilisateur peut valider ou éliminer chaque entrée de cette concordance. La concordance ainsi filtrée peut, ensuite, être utilisée pour annoter le corpus.

A ce propos, nous notons que l'annotation d'un corpus à partir d'une concordance permet aux utilisateurs de vérifier, adapter et corriger les résultats d'un traitement automatique dont les sorties ne sont, pratiquement, jamais parfaites. Ceci est très important surtout lors d'une étape de préparation d'un corpus.

## **2. Les dictionnaires NooJ**

Les dictionnaires NooJ (Silberztein M. 2003) contiennent indistinctement des mots simples et des mots composés qui peuvent représenter des variantes terminologiques ou phonétiques des entrées lexicales.

### **2.1 Les ALUs (Atomic Linguistic Units)**

Les unités atomiques linguistiques (ALUs) sont les plus petits éléments qui forment une phrase. Ce sont des "mots" dont la signification ne peut pas être calculée ou prédite, on doit les apprendre pour être capables de les utiliser. On doit décrire explicitement toutes les propriétés syntaxiques et sémantiques de ces mots afin de pouvoir les analyser.

D'un point de vue formel, NooJ sépare les ALU en quatre classes :

- Les mots simples, ALU orthographiés sous forme de mots, tels que "tawurt" ou "tameṭṭut" ;

- Les affixes, qui sont des ALU orthographiés comme des séquences de lettres, habituellement à l'intérieur des formes de mots, telles que dans "dé-" et "-isation" dans "dé-central-isation" ;
- Des mots composés, ALU orthographiés par des séquences de formes de mots, tels que "table ronde", "pomme de terre" ;
- Des expressions figées.

Afin de reconnaître les mots simples et les affixes, Le système NooJ effectue une recherche dans les dictionnaires. Les autres types seront identifiés en découpant les formes de mots en plus petites unités. Les quatre formes sont reconnues par NOOJ en appliquant des grammaires syntaxiques.

## 2.2 Ressources pour reconnaître les unités linguistiques atomiques

Par exemple, en ayant choisi la langue ici br pour tamazight (berbère), les zones "Lexical Analysis" et "syntactic Analysis" s'affichent.

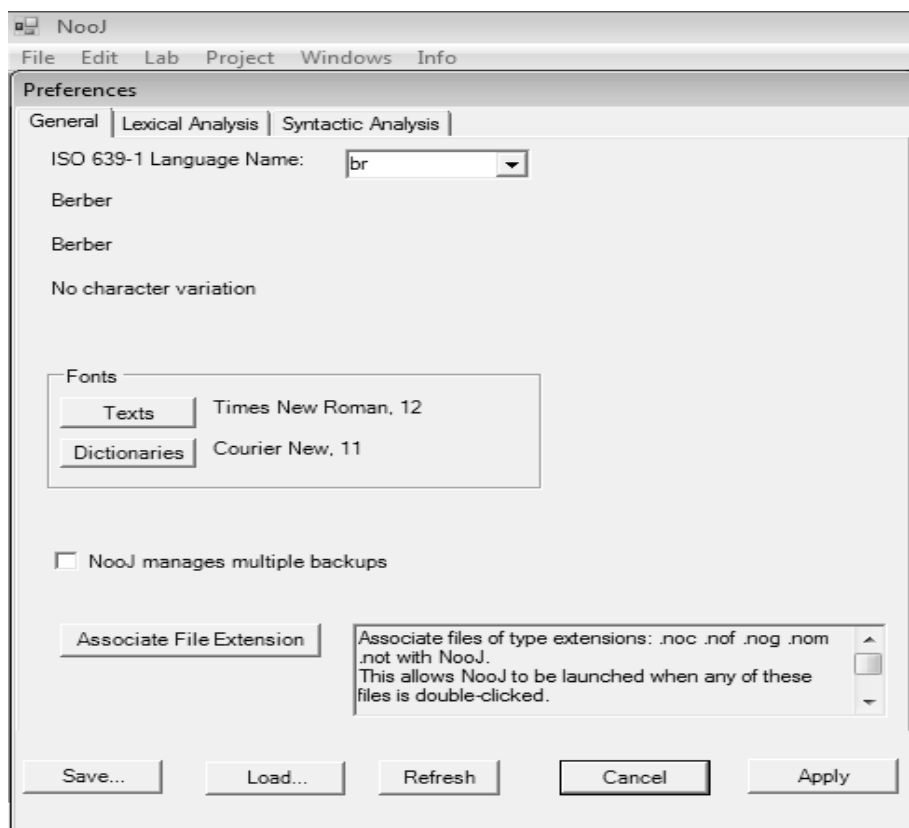
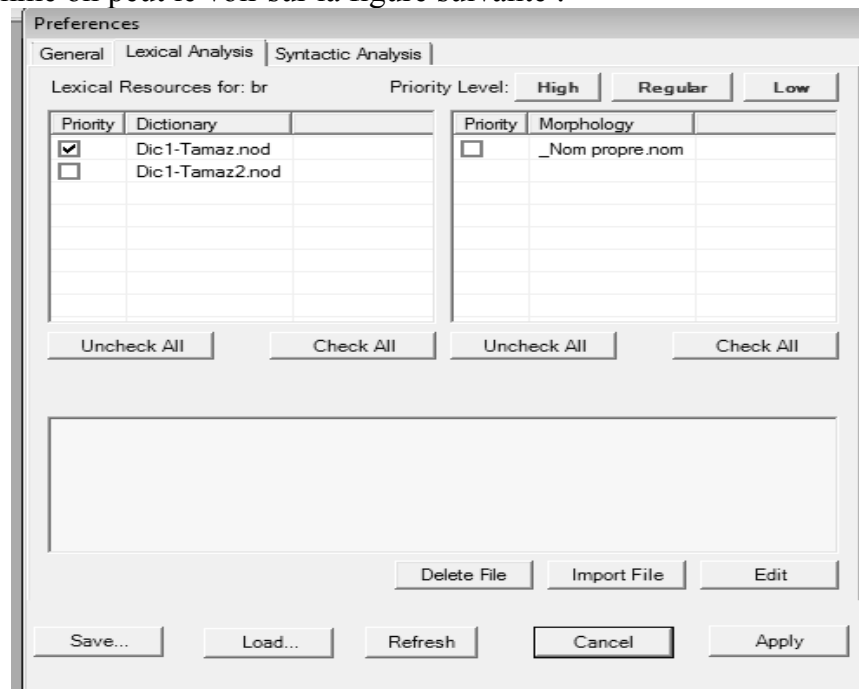


Figure 2-1 : "Choix de la langue".

Dans NooJ, la commande

Info Preferences Lexical Analysis permet d'afficher les zones : "Dictionary", et "morphology" comme on peut le voir sur la figure suivante :



**Figure 2-2** : Ressources lexicales: "Dictionary" et "Morphology".

La zone "Dictionary" contient toutes les ressources lexicales qui sont utilisées pour reconnaître les mots simples et les mots composés, ici (figure 2-2), nous avons coché "Dic1-Tamaz.nod".

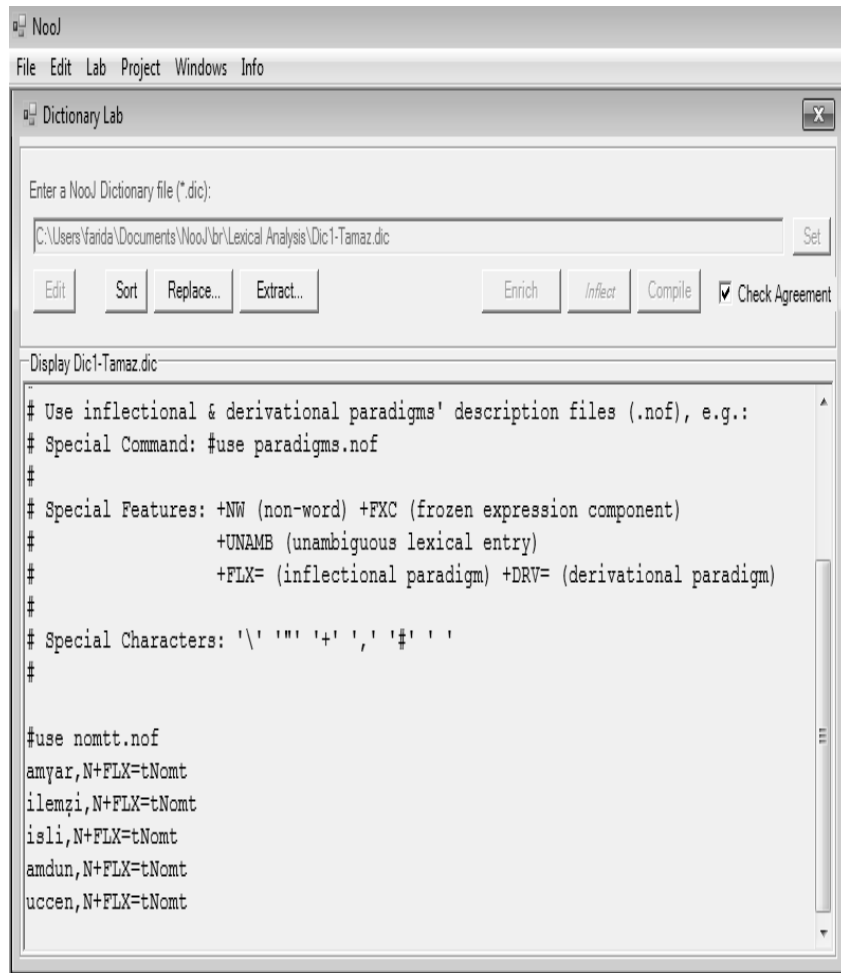
### 2.2.1 Les Dictionnaires

Les dictionnaires NooJ sont des fichiers ".nod" qui sont compilés à partir de fichiers source ".dic". Techniquement les fichiers ".nod" sont des transducteurs d'états finis, ils proviennent de fichiers de type texte ".dic" qui sont compilés en utilisant la commande

```
Lab > Dic > compile
```

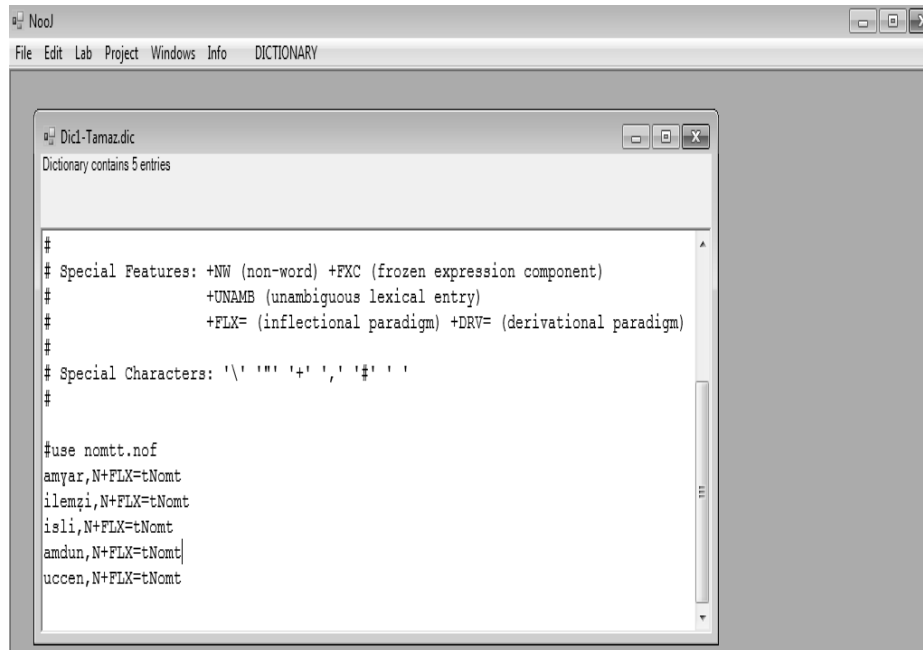
On utilise la commande Lab > Dictionary (Figure 2-3) pour :

- fusionner les informations de deux dictionnaires ("Enrich"),
- produire automatiquement la liste de toutes les formes fléchies et dérivées correspondant aux entrées lexicales d'un dictionnaire ("Inflect"),
- Compiler le dictionnaire en un fichier binaire .nod, ainsi il peut être appliqué à des textes, ("Compile").



**Figure 2-3 :** Utilisation de Lab > Dictionary et "Dic1-Tamaz.dic".

Dans la figure 2-4 suivante, nous avons un exemple de dictionnaire, avec quelques entrées :



```
#
# Special Features: +NW (non-word) +FXC (frozen expression component)
#                   +UNAMB (unambiguous lexical entry)
#                   +FLX= (inflectional paradigm) +DRV= (derivational paradigm)
#
# Special Characters: '\' ' "' '+' ' ' ' ' '#!' ' ' '
#
#use nomtt.nof
amyar,N+FLX=tNomt
ilemzi,N+FLX=tNomt
isli,N+FLX=tNomt
amdun,N+FLX=tNomt
uccen,N+FLX=tNomt
```

**Figure 2-4:** Exemple de dictionnaire "Dic1-Tamaz.dic".

### 2.2.2 La Morphologie

La zone "Morphology" (voir la figure 2-2 précédente) affiche toutes les grammaires morphologiques qui sont utilisées pour reconnaître les formes de mots à partir de leurs composants (préfixes, affixes et suffixes). Ces ressources morphologiques sont des fichiers ".nom" constitués d'ensembles structurés de graphes qui décrivent des paradigmes morphologiques.

### 2.3 Outils pour décrire la morphologie

NooJ (Silberztein M. 2003) offre deux outils équivalents pour représenter et décrire la morphologie flexionnelle et dérivationnelle. Ces deux descriptions de la morphologie sont lexicalisées : chaque entrée dans un dictionnaire NooJ est associée avec un ou plusieurs paradigmes flexionnels ou dérivationnels. Les paradigmes sont décrits et stockés dans des grammaires flexionnelles et dérivationnelles sous



forme de fichiers ".nof". NooJ offre aussi des grammaires morphologiques (fichiers ".nom").

### 2.3.1 Descriptions flexionnelles et dérivationnelles

Ce sont des fichiers ".nof" organisés en règles qui décrivent les propriétés de chaque catégorie.

Exemple de règle :

$tNom_t = \langle E \rangle / m + s + t \langle P \rangle t / f + s ;$

" $\langle E \rangle / m + s$ " indique qu'on ne rajoute rien " $\langle E \rangle$ " au masculin "m" singulier "s", nous obtenons pour le mot "ilemzi" :

$+ m + s = >$  "ilemzi" au masculin singulier

" $t \langle P \rangle t / f + s$ ", indique rajouter t,  $\langle P \rangle$  : aller à la fin du mot précédent, rajouter t, nous obtenons

$+ f + s = >$  "tilemzit" au féminin singulier.

### 2.3.2 Grammaires flexionnelles et dérivationnelles

Elles sont représentées par des ensembles structurés de graphes qui décrivent des paradigmes morphologiques stockés dans des fichiers ".nof".

## 3 Format des dictionnaires NooJ

Généralement, le dictionnaire d'un langage donné contient tous les lemmes du langage et les associe avec un code morpho-syntaxique, des codes sémantiques et syntaxiques possibles, des paradigmes flexionnels et dérivationnels.

### 3.1 Exemples d'entrées

Voici quelques exemples d'entrées :

$am\bar{y}ar, N + FLX = tNom_t + Hum$

$ilemzi, N + FLX = tNom_t + Hum$

- La première entrée "am $\bar{y}$ ar" est un nom, (catégorie "N"), elle se fléchit :

("am $\bar{y}$ ar tam $\bar{y}$ art") selon le modèle "tNom $_t$ " (règle énoncée au 2.3.1) qui est la valeur de la propriété "FLX", le trait sémantique est Humain ("Hum").

- La deuxième entrée "ilemzi" est un nom, (catégorie "N"), elle se fléchit :

("ilemzi tilemzit"), selon le modèle "tNomt" qui est la valeur de la propriété "FLX", le trait sémantique est Humain ("Hum").

### 3.2 Informations linguistiques

Dans les dictionnaires NooJ, toutes les informations linguistiques telles que les codes syntaxiques tels que "+tr" (transitif) et les codes sémantiques tels que "+conc" (concret) et "+abst" (abstrait) doivent être précédés du caractère "+".

### 3.3 Codes d'information spéciaux

Le code "+ NW " (Non-Word) est utilisé pour décrire des entrées lexicales abstraites qui n'apparaissent pas dans les textes et qui ne devraient pas être analysées comme des formes réelles de mots. Cette forme est utile pour la construction d'un dictionnaire dans lequel des entrées sont des stems ou des racines de mots.

Le code "+UNAMB" (unambiguous ou mot non ambigu) indique à NOOJ de stopper l'analyse de la forme du mot avec soit les ressources lexicales soit l'analyseur morphologique.

Le code "+FLX" (Paradigme flexionnel) permet d'indiquer le nom des règles de flexions du mot. Par exemple dans l'entrée de dictionnaire :

ilemzi,N + FLX = tNomt + Hum

ilemzi est associé avec la propriété +FLX=tNomt, et tNomt est un paradigme flexionnel.

Le code "+DRV" (Paradigme dérivationnel) permet aux lexicographes d'associer chaque mot avec les variantes dérivationnelles correspondantes. Des dérivations morphologiques seront ensuite décrites avec précision pour chaque entrée.

Lier des formes dérivées dans le dictionnaire permettra d'avoir un dictionnaire moins volumineux.

### 3.4 Propriétés lexicales

Non seulement des formes, mais aussi des propriétés peuvent être associées aux entrées lexicales. Une propriété lexicale a un nom et une valeur exprimés sous la forme " +nom=valeur".

### 3.5 Variantes lexicales

Les dictionnaires NooJ ne sont plus limités à un seul champ : ils peuvent contenir des entrées liées à un «super-lemme», qui peut être une variante orthographique, la traduction dans une autre langue, une entrée synonyme ou hyperonyme. Considérons l'entrée lexicale suivante :

`Tawwurt, tabburt, N+Conc`

L'entrée ("tawwurt") est associée au super-lemme "tabburt".

### 4. Fichiers de définition des propriétés ".DEF"

Dans le dictionnaire, l'instruction suivante indique où sont définies les associations catégories, propriétés et leurs valeurs :

```
#use _Exemple.def
```

Les utilisateurs peuvent créer de nouveaux codes d'information tels que "+Info" "+Politiqu" et peuvent les ajouter a des entrée lexicales. On peut aussi utiliser ces codes d'informations qui sont ajoutés sous forme de propriété avec une valeur, telle que "+Domain=Info".

Exemples :

- `V_Nb = s|p` signifie que le nombre du verbe est singulier ou pluriel ;

- `V_Pers = 1|2|3` signifie que la personne du verbe est 1ère, 2ème ou 3ème.

### 5. Représentation formelle des dictionnaires électroniques

Les automates d'état finis (FSA : "finite state automata") sont utilisés dans beaucoup d'applications de traitement du langage naturel (NLP). En particulier, ils fournissent un stockage et une récupération efficaces des ensembles finis de chaînes sur un alphabet fini, et peuvent donc être utilisés pour représenter le vocabulaire d'une langue.

L'utilisation des automates peut être étendue à l'utilisation des transducteurs d'état finis (FST : "finite state transducer") qui permet aux utilisateurs d'associer à chaque élément d'un vocabulaire des informations, telles que des informations morpho-syntaxiques (genre, nombre, etc.), des informations syntaxiques et sémantiques (par exemple, transitif, humain, etc.), ou plus complexes, telles que la traduction des termes dans d'autres langues, un ensemble d'expressions synonymes, etc.

Le but d'une analyse lexicale d'un texte est de mettre en correspondance les termes de ce texte avec ceux d'un vocabulaire généralement décrits dans un dictionnaire. Dans la plupart des langues, les tokens sont des formes de mots fléchis et / ou dérivés qui nécessitent d'être associés au lexème correspondant (entrée de dictionnaire) avec quelques informations linguistiques.

Ainsi, les transducteurs sont bien adaptés pour l'exécution d'analyses automatiques lexicales. Les unités linguistiques de NooJ sont reconnues en effectuant une recherche dans les dictionnaires, et aussi bien en utilisant des grammaires morphologiques pour analyser les formes du mot, ainsi qu'en utilisant les grammaires syntaxiques des phrases à analyser.

Sur le plan linguistique, l'intérêt des graphes ou des automates est de regrouper dans une même description des éléments équivalents du point de vue du sens, mais différents au niveau de la forme, par exemple les variantes graphiques de mots dont l'orthographe n'est pas normalisée ou bien des variantes d'expressions de temps.

## 6. Exemples d'utilisation

### 6.1 Analyse lexicale

On effectue l'analyse lexicale du texte "nom-F.not", dans la figure 6-1.

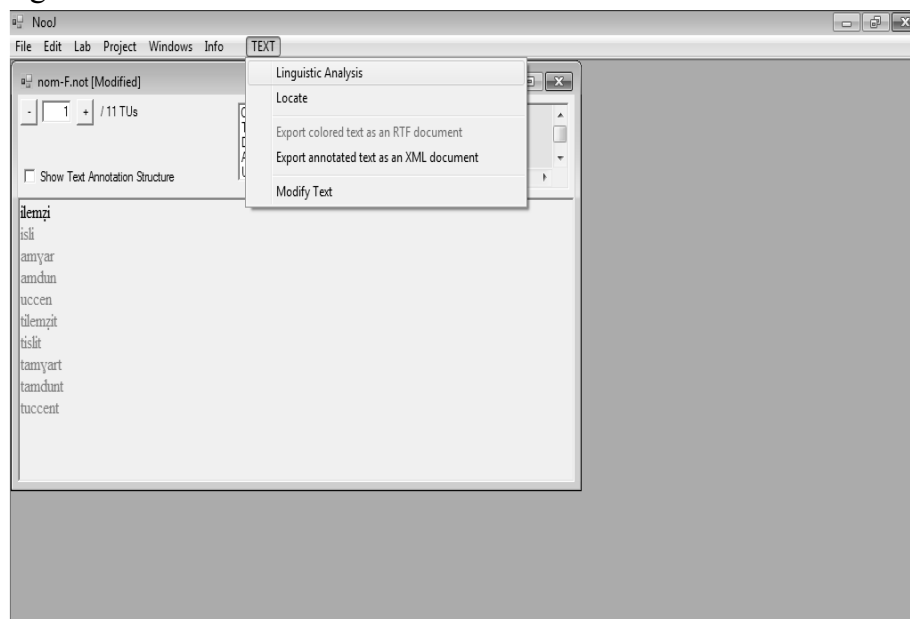


Figure 6-1 : analyse lexicale du texte "nom-F.not".

## 6.2 Affichage des annotations après analyse lexicale

Dans la figure 6-2, après l'analyse lexicale du texte "Nom-F.not", on affiche les annotations qui sont rangées dans un dictionnaire.

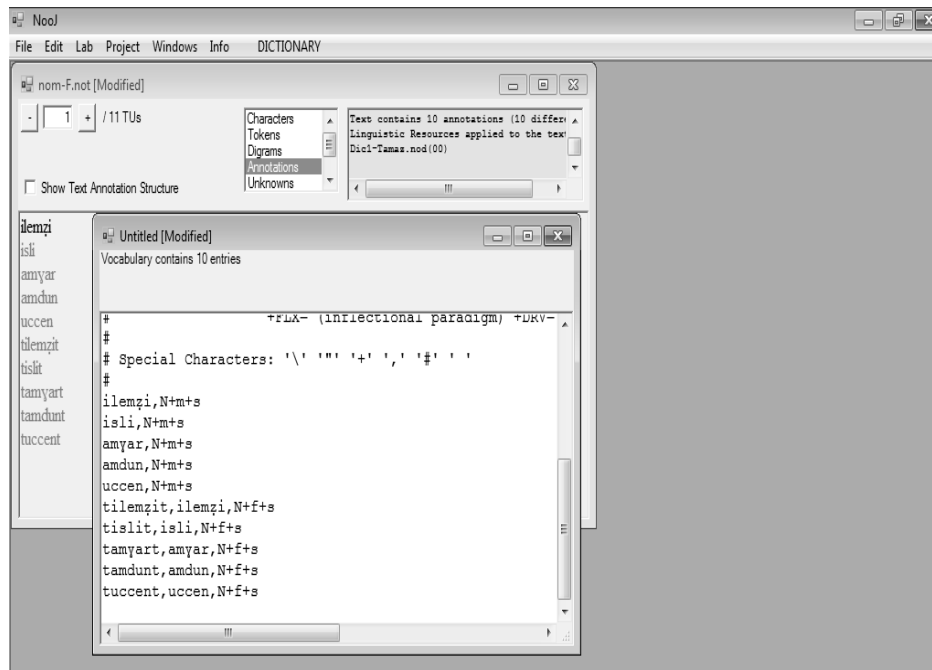


Figure 6-2: Annotations.

## 6.3 Recherche d'un "patron"

Dans la figure 6-3, une expression régulière est utilisée pour rechercher un patron dans le texte, ici <N+m+s> est une expression régulière NooJ utilisée pour trouver tous les noms (N), au masculin (m) singulier (s) dans le texte "nom-F.not". La concordance est affichée et peut être sauvée dans un fichier, cela peut être utile pour l'acquisition de nouveaux mots.

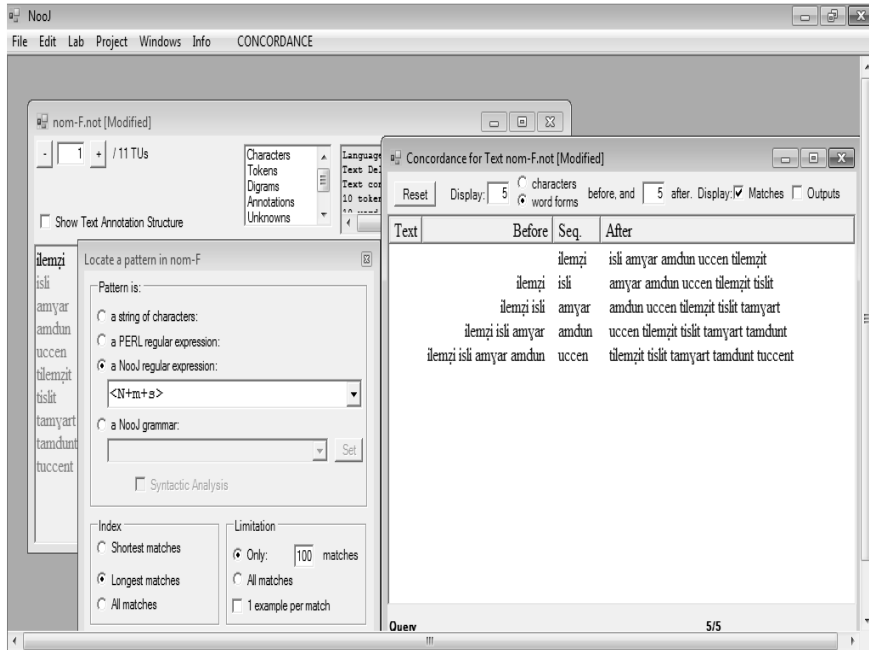


Figure 6-3 : Recherche d'un patron dans le texte "Nom-F.not".

## 6.4 Morphologie :

Pour mettre en place les règles de flexion des mots, on utilise > Lab> morphology :

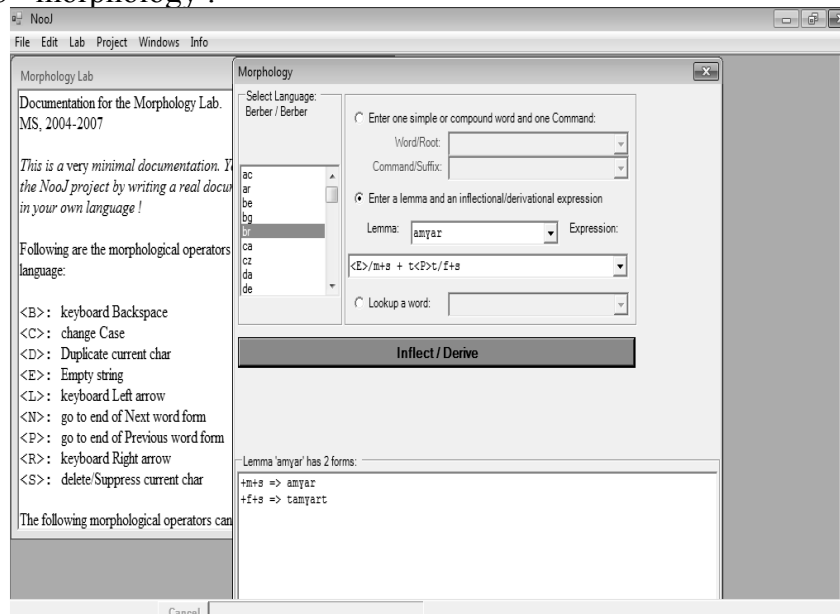


Figure 6-4: Mise au point des règles de flexion à l'aide de "Lab".

En entrant un lemme et une expression flexionnelle ou dérivationnelle, on obtient après avoir cliqué sur Inflect/Derive, la forme des lemmes produits grâce à l'expression.

Dans la figure 6-4, nous avons le lemme "amɣar" et l'expression flexionnelle

<E>/m+s + t<P>t/f+s

<E>/m+s indique qu'on ne rajoute rien "<E>" au masculin "m" singulier "s",

nous obtenons :

+m+s => amɣar au masculin singulier

"t<P>t/f+ s", indique rajouter t au mot, puis <P> indique aller à la fin du mot précédent, rajouter t, nous obtenons

+f+s => tamɣart au féminin singulier.

### **8. Conclusion**

Les dictionnaires électroniques sont construits en vue du traitement automatique des langues (TAL), naturelles ou techniques.

NooJ est un outil puissant et la plate-forme conviviale permet le développement et la mise au point de diverses applications :

- linguistiques,
- d'aide à l'enseignement des langues,
- d'aide à la traduction automatique,
- d'extraction de terminologie.

Le système NooJ est un environnement de développement linguistique, doté d'un moteur linguistique robuste, pouvant prendre en charge dix neuf langues variées (acadien, arabe, arménien, biélorusse, catalan, croate, français, anglais, espagnol, allemand, grec, hébreux, hongrois, italien, latin, polonais, portugais, turc, vietnamien). Notre objectif est la construction des ressources linguistiques pour tamazight, afin de l'intégrer dans NooJ et permettre le traitement automatique de textes en tamazight et le développement d'applications telles que l'enseignement de la langue, la traduction automatique.

## **Bibliographie**

Mesfar S. 2008. Analyse morpho-syntaxique automatique et reconnaissance des entités nommées en arabe standard, *Thèse de doctorat en informatique soutenue en novembre 2008, université de Franche-Comté, France.*

Silberztein M. 2003. NOOJ‘manual. <http://www.nooj4nlp.net>

Silberztein M. et Tutin A. 2005. NooJ : Un outil TAL de corpus pour l’enseignement des langues. Application pour l’étude de la morphologie lexicale en FLE, *Article n° alsic\_v08\_20-rec11*, p. 123-134.

Yamouni Aoughlis F. Construction d’un dictionnaire électronique de terminologie informatique et analyse automatique de textes par grammaires locales, *Thèse de doctorat d’état en informatique soutenue le 12/12/2010, UMMTO.*

---

1 - <http://www.nooj4nlp.net/>