Le logiciel Nooj appliqué au kabyle

H. Annouz, K. Ferroudja, K. Naït-Zerrad
Lacnad, Inalco, Paris

L'environnement Nooj

Nooj [1] est un environnement de développement linguistique permettant de construire et de gérer des dictionnaires et grammaires électroniques. Il permet la formalisation de plusieurs niveaux des langues naturelles : orthographe, morphologie flexionnelle et dérivationnelle, lexique, syntaxe, sémantique,… Le système d'annotation couvre tous les niveaux de grammaire.

Nooj n'est pas seulement un outil de linguistique de corpus avec lequel les descriptions formalisées sont appliquées à des textes sans les modifier –en temps réel- mais il est également utilisé dans diverses applications du TALN et dans l'enseignement des langues et de la linguistique.

Nooj est capable de traiter de grands corpus dans la plupart des formats existants (rtf, doc, html, xml,…). Toutes les variantes de l'Unicode sont supportées.

Pour le traitement d'un corpus, il faut d'abord construire des dictionnaires et des grammaires de la langue.

Pour ce faire, une écriture normalisée est indispensable et pour le kabyle, nous suivrons la grammaire de Naït-Zerrad [2] . La formalisation des verbes est quant à elle basée sur le manuel de conjugaison du même auteur[3].

Le dictionnaire Nooj peut comprendre plusieurs types d'informations :

Entrée lexicale, nature, flexion, attribut sémantique,…

- **Modélisation des caractéristiques du nom**

Un nominal (substantif ou adjectif) peut être défini par les caractéristiques suivantes :

-racine (R)

-masculin / féminin singulier état libre[4] (MSEL / FSEL)

-masculin / féminin pluriel état libre (MPEL / FPEL)
-masculin / féminin singulier état d'annexion (MSEA / FSEA)
-masculin / féminin pluriel état d'annexion (MPEA / FPEA)

Au moins deux solutions sont possibles pour construire les modèles :

1- Soit on crée un modèle global (avec toutes les caractéristiques) par type de nom : il s'agit donc de modèles de noms.

2- Soit on crée des modèles de caractéristiques dans lesquels on puise pour définir un nom quelconque

La première solution donnerait par exemple le modèle de flexion suivant :

ARGAZ = <E>/EL+m+s |<LW><S>we/EA+m+s |
<LW><S>i<RW>en/EL+m+p
|<LW><S>ye<RW>en/EA+m+p[5]

En clair :

MSEL = *argaz* / MSEA = *wergaz* / MPEL = *irgazen* / MPEA = *yergazen*

Il suffit alors d'indiquer au logiciel tous les noms qui se fléchissent (FLX) de la même manière, il générera ainsi toutes les formes automatiquement :

argaz,N+FLX=ARGAZ
abɣur, N+FLX=ARGAZ
amɣar, N+FLX= ARGAZ
…

Une autre méthode serait de créer des modèles de caractéristiques ou de paradigmes. Ainsi, chaque schème de pluriel correspond à un modèle (voyelle initiale incluse) :

MPEL :
i – en / i – an / i – awen / i – yen / i – ten /...
u – awen / u – en / … / a – en / …
i-a- / i-u-a-/ i-i-a- / …
i-a-an / i-u-an / i-an / i-i-iwin / i-a-wen / …
FPEL :
ti-in / ti- atin / ti-awin / yi- iyin / ti- itin / ti – uyin / ti – utin/...

ti-a- / ti-u-a-/ ti-i-a- / ...

ti-a-an / ti-wa / ti-i-wa / ti-u-iwin / ti-a-win / ...

On fait de même avec le singulier et l'état d'annexion. Pour ce dernier, on a par exemple les modèles suivants :

MSEA : *we- / u- / wa- / wu- / i- / yi- / ye-*

FSEA : *te- / tu- / ta- / ti-*

…

Un nom correspond donc à un type modélisé de cette manière :

NOM = R3 / MSEL15 / MPEL21 / MSEA5 / MSEA10 / …

Les numéros correspondent aux différents modèles.

La deuxième solution semble beaucoup plus économique et beaucoup moins chronophage. Comme nous l'avons signalé plus haut, d'autres attributs peuvent être ajoutés au nom : humain / non humain ; mâle / femelle ; abstrait / concret ; collectif / singulatif ; …

- **Modélisation des caractéristiques du verbe**

Le verbe kabyle peut être défini par 15 caractéristiques ou paradigmes basées sur les thèmes verbaux :

-racine (R)

-impératif (I)

-impératif négatif (IN)

-impératif intensif (II)

-impératif intensif négatif (IIN)

-prétérit (P)

-prétérit négatif (PN)

-aoriste (A)

-aoriste intensif (AI)

-participe A (PA)

-participe A négatif (PAN)

-participe P (PP)

-participe P négatif (PPN)

-participe AI (PAI)

-participe AI négatif (PAIN)

On peut réduire le nombre de caractéristiques étant donné l'identité de certaines d'entre elles, par exemple IN = IIN.

Deux solutions sont possibles pour la modélisation :

1- soit on crée un modèle global (avec toutes les caractéristiques) par type de verbe : il s'agit donc de modèles de verbes

2- soit on crée des modèles de caractéristiques dans lesquels on puise pour définir un verbe quelconque

La deuxième méthode semble plus souple que la première et plus économique, en particulier parce qu'un verbe peut avoir par exemple différentes formes de AI.

Chaque paradigme est nommé ou numéroté, ce qui pour un verbe quelconque fournit par exemple le modèle de flexion suivant :

VERBE = R4 / I7 / II9 / P21 / PN30 / A5 / AI3 / PP27 / etc.

Tous les verbes ayant les mêmes caractéristiques sont rapportés à un verbe type. Par exemple, si *afeg* est un verbe type, alors on pourra simplement écrire :

afeg,V+FLX=AFEG

afes,V+FLX=AFEG

adef,V+FLX=AFEG

aden,V+FLX=AFEG

aḍen,V+FLX=AFEG

ader,V+FLX=AFEG

akel,V+FLX=AFEG

aker,V+FLX=AFEG

…

Cela signifie que les verbes *afes, adef, aden,* etc. ont le même modèle flexionnel (FLX) que *afeg*.

Le programme permet de générer toutes les formes fléchies d'un verbe.

- **Les formes dérivées**

1. Les verbes

Ils se forment à l'aide de préfixes (*s-, m-, tt-, ttwa-, ttu-, my-*… et variantes) et éventuellement une modification du radical.

Il y a par exemple plus de 20 modèles de dérivation pour le factitif :

kcem «entrer »	>	*ssekcem* «introduire »
ni «être enfilé »	>	*sni* «enfiler»
mmekti«se rappeler »	>	*smekti*«rappeler»
ers «être posé »	>	*ssers*«poser»
afeg«voler »	>	*ssifeg*«faire voler, faire disparaître »
ali«monter »	>	*ssali*«faire monter»
urug«être versé »	>	*ssureg*«verser»
irid«être lavé »	>	*ssired*«laver»
ifif«être tamisé »	>	*ssiff*«tamiser»

…

Il faut compter un nombre équivalent pour le passif, le réciproque et les combinaisons de préfixes. A cela s'ajoute d'autres modèles de paradigmes (prétérit, aoriste intensif, participes,…) pour ces verbes dérivés.

2. Les noms

Les noms déverbatifs sont en général formés avec des préfixes et / ou une modification du radical : nom d'action, nom d'agent, nom d'instrument, …. Il faut donc des modèles pour chaque type morphologique.

Exemples de nom d'action / d'état :

verbe		**nom d'action / d'état**	
krez	>	*akraz / takerza / tayerza*	« labourer »
iksin	>	*tuksinin*	« être responsable »
issin	>	*tamussni / tussnin /*	« savoir »
izdig	>	*tezdeg*	« être propre, pur »
els	>	*timelsiwt / timelsi /tulsin...*	« être vêtu ; revêtir »

Exemples de nom d'agent / de patient :

verbe		**nom d'agent/patient**	
baṣi	>	*ambaṣi*	« être condamné »
inig	>	*iminig*	« voyager »
gmer	>	*anegmar*	« cueillir »
aǧew	>	*anaǧaw*	« acheter des denrées »
aḍen	>	*amuḍin*	« être malade »
tter	>	*amattar*	« demander, mendier »

D'autres dictionnaires sont également à construire : adverbes, prépositions, connecteurs,…

- **Désambiguïsation**

Dans les langues naturelles, les ambiguïtés sont de différents ordres : lexicales, sémantiques, syntaxiques, etc. Elles peuvent être levées à partir du contexte ou du cotexte.

Quelques exemples :

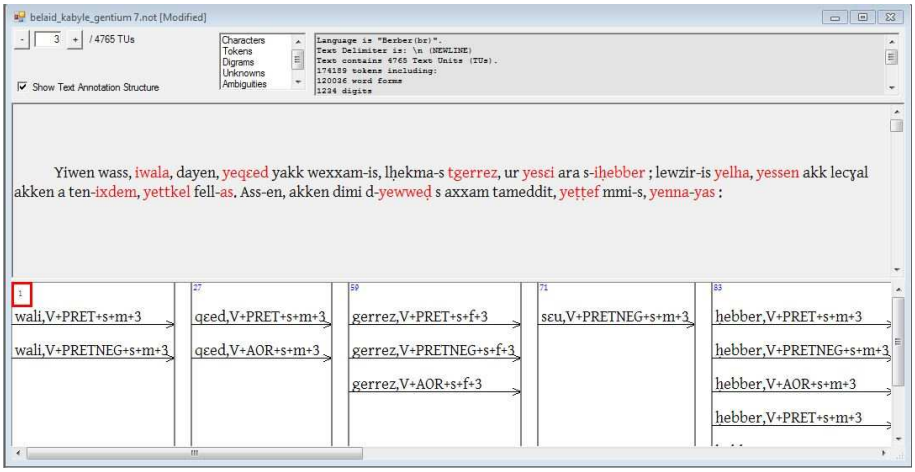
Verbe *neɣ* « tuer » / Conjonction *neɣ* « ou »

Verbe *ɣer* « lire » / Préposition *ɣer* « vers »

Verbe conjugué *tura* « elle a écrit » / Adverbe *tura* « maintenant »

Verbe *urar* « jouer » / Nom *urar* « jeu, fête… »

Pour lever les ambiguïtés, on construit des grammaires. Par exemple, à partir du dictionnaire des verbes, on obtient l'analyse suivante par Nooj :



Les verbes ont bien été reconnus par Nooj mais on voit deux types d'ambiguïtés :

a- Pour beaucoup de verbes, une forme verbale unique peut s'interpréter de différentes manières. Dans l'exemple ci-dessus, seule la forme *yesɛi* n'est pas ambiguë : le logiciel donne la bonne réponse. La forme *iwala* a deux interprétations possibles comme indiqué par Nooj : soit il s'agit d'un prétérit, soit d'un prétérit négatif. Pour lever l'ambiguïté, il faudrait construire une grammaire qui indique que le prétérit négatif est précédé de la particule négative *ur*. On fera de même avec la particule préverbale *ad / a* pour distinguer entre le prétérit et l'aoriste. La réalité est plus complexe mais c'est un premier pas vers la désambiguïsation.

b- On remarque dans la figure ci-dessus que les clitiques *as* et *yas* ont été pris par Nooj pour des verbes : il s'agit ici d'un problème d'homonymie avec le verbe *as* « venir ». Ici également, il suffit de construire une grammaire avec les différentes positions des pronoms (compléments directs et indirects) pour lever l'ambiguïté :

Verbe-clitique indirect-clitique direct
Clitique indirect-clitique direct-Verbe

Si Nooj trouve un verbe précédé ou suivi d'un de ces éléments (la liste de tous les clitiques possibles est incluse dans le graphe de la grammaire), il est d'abord exclu come verbe et mis dans la bonne catégorie.

Conclusion

Nous avons tenté de montrer la puissance de Nooj pour la formalisation linguistique à travers un petit aperçu de ses potentialités. Une partie des dictionnaires est déjà réalisée ainsi que des grammaires de désambiguïsation. Il reste à les compléter – construction en particulier de grammaires syntactiques - afin de pouvoir traiter automatiquement de larges corpus. Il sera alors temps d'approfondir le côté sémantique et les possibilités de Nooj pour la traduction automatique.

1- http://www.nooj4nlp.net
2-*Grammaire moderne du kabyle*, 2001, Karthala, Paris
3- *Manuel de conjugaison kabyle*, 1994, L'Harmattan, Paris
4- L'état libre correspond par exemple au nom isolé, l'état d'annexion indique une dépendance qui est matérialisée en général par une modification de la voyelle initiale du nom (par exemple : nom après une préposition, nom («complément explicatif») placé après la forme verbale et explicitant le «sujet» (désinence personnelle obligatoire qui accompagne le verbe).
5- m=masculin, p=pluriel, s=singulier, EL = état libre, EA= état d'annexion. Codes Nooj : <E>=chaine vide, <LW>=aller au début du mot, <S>=supprimer le caractère, <RW>=aller à la fin du mot.