ETAPE 02

Nous allons dans ce qui suit aligner des termes d'un fichier de BNF avec d'autre de RAMEAU, et déclarer les ID des meilleurs résultats d'alignement

Nous commençons par la conversion du fichier rameau.txt avec un encodage en utf-8 afin de pouvoir opérer sur son contenu. À l'aide de l'invité de commande :

```
#iconv -f macintosh -t utf-8 RAMEAU-Groupes_ethniques-2.txt > RAMEAU_utf-8.txt
```

Figure 01

Déclaration des fonctions :

Importation des différents modules à utiliser.

Ensuite, on déclare les fonctions principales à exécuter plus tard : La mesure Jaro et Levenshtein Normalisé, pour lesquelles on va déclarer des seuils minimums à respecter.

On crée des fichiers de traitement des deux sources BNF et Rameau, écrire respectivement des lignes composées d'une chaine de caractère numériques (ID) et l'expression à aligner.

BNF On récupère la chaine de caractère après le dernier $ (cadre en rouge figure
RAMEAU On récupère la chaine de caractère après le premier $a

Traitement des fichiers de source :

On réduit toutes les lettes en minuscule et on enlève tous types d'accents et caractères spéciaux
On enlève tous les stop-word manuellement

```
#liste des caractères accentués a désaccentuer
accent = ['é', 'è', 'ê', 'ë', 'à','á','É', 'ù', 'û', 'ú', 'ü','ç', 'ô','ó','ö', 'î','í','
sans_accent = ['e', 'e', 'e', 'e', 'a','a','E', 'u', 'u', 'u','u', 'c', 'o','o','o', 'i',

#liste stop words
stopWordFr=re.compile('d\'|l\'| de| du|\'|-|_|,|"| la|peuple |peuples de|civilisation ')
```

Pour les chaines de plusieurs mots on split() et on mesure la distance de chaque mot qui la compose avec l'autre partie

```
39027974
143   $aTraditions$mEurope$mAlbanie
42406496
143   $aTraditions$mAfrique du Nord$mAlgérieca
42155205
143   $aTraditions$mAsie$mYémen$eJuifs
42765162
143   $aTraditions$mEurope$mTchécoslovaquie$mSlovaquie$mRégion de Trenčín$mMyjava
38603967
143   $aTraditions$mEurope$mBulgarie$mRhodopes$mNedelinc
```

Fichier de départ BNF

```
(39027974 albanie)
(42406496 algerie)
(42155205 juifs)
(42765162 myjava)
(38603967 nedelino)
```

BNF Après traitement

```
11932945        $aTswana$gpeuple d'Afrique
11940850        $aPortugais
11960038        $aAlbanais
11988158        $aAlgériens
11949880        $aBulgares
11974575        $aBulgares de la Volga
11932167        $aJuifs
11956425        $aSlovaques
12385470        $aJuifs tchécoslovaques
11967055        $aJuifs d'Europe de l'Est
11940990        $aKabyles$gpeuple berbère
11937529        $aBété$gpeuple de Côte d'Ivoire
```

Fichier RAMEAU de départ

```
(11932945 tswana)
(11940850 portugais)
(11960038 albanais)
(11988158 algeriens)
(11949880 bulgares)
(11974575 bulgares volga)
(11932167 juifs)
(11956425 slovaques)
(12385470 juifs tchecoslovaques)
(11967055 juifs europe est)
(11940990 kabyles)
(11937529 bete)
```

RAMEAU après traitement

Mesures de similarité :

On parcourt chaque ligne du fichier BNF en lecture, et à l'aide des expressions régulières on détermine l'ID et la chaine de caractère à aligner avec le terme de RAMEAU

```
#/////////////// on collecte les termes à mesurer ////////////////////////

mots1 = open('newBNF_utf-8.txt','r')
read_mots1= mots1.readlines()

mots2 = open('newRAMEAU_utf-8.txt','r')
read_mots2= mots2.readlines()

for ligne in read_mots1:
        res_mots1 = re.search(r".*(\d{8})\s+(.*)\).*",ligne)
        if res_mots1:
                iD1 = res_mots1.group(1)
                mot1 = res_mots1.group(2)
                mot1S = mot1.split()
```

Et à l'aide des fonctions Jaro et Levenshtein Normalisé on mesure la similarité entre les termes de BNF et de RAMEAU, tout ne définissant un seuil Jaro minimal >0.8 et LevN >0.75 pour les mots et Jaro 0.7 et

Cas Levenshtein > 0.7 entre chaines de caractères

```
                        if mot1==mot2:
                                print(mot1+' iD1:'+iD1,'    <>    ',mot2+' iD2:'+iD2 ,' : Identity similarity',' Lev :', levenshteinN
(mot1,mot2))
                        else:
                                # Si mot1 et mot2 sont des mots uniques
                                if levenshteinN(mot1,mot2)>=0.75:  #<<<<<<< ici vous mettez le seuil de distance min
                                        if jaro(mot1, mot2) > 0.8:  #<<<<<<< ici vous mettez le seuil de similarité Jaro min

                                                #metrics(tok3.tokenize(nltk.corpus.brown.raw()),
                                                #nltk.corpus.brown.words())
                                                print(mot1+' iD1:'+iD1,'    <>    ',mot2+' iD2:'+iD2 ,' : ', jaro(mot1, mot2),' Lev Sim:',
levenshteinN(mot1,mot2))
                                # Si mot1 et mot2 sont composés de plusieurs mots, on mesure jaro et lev entre chaque mot des deux cotés
                                        else :
                                                if jaro(mot1S, mot2S) > 0.7 :#and ngram.NGram.compare(mot1,mot2,N=3)>0.5:
                                                        #metrics(tok3.tokenize(nltk.corpus.brown.raw()),
                                                        #nltk.corpus.brown.words())
                                                        print(mot1+' iD1:'+iD1,'    <>    ',mot2+' iD2:'+iD2 ,' Jaro P1P2: ', jaro(mot1S, mot2S),'
Lev Sim P1P2:', levenshteinN(mot1S,mot2S))
```

Les résultats qu'on peut voir déjà à l'écran de l'invité de commande :

```
nbendjoudi@x2go3:~/TER/TER2/Etape 2$ ./lecture_BNF_RAMEAU_jaro.py
albanie iD1:39027974     <>     albanais iD2:11960038  :  0.869  Lev Sim: 0.75
algerie iD1:42406496     <>     algeriens iD2:11988158  :  0.926  Lev Sim: 0.778
juifs iD1:42155205     <>     juifs iD2:11932167  : Identity similarity  Lev : 1.0
juifs iD1:42155205     <>     juifs tchecoslovaques iD2:12385470  Jaro P1P2:  0.833  Lev Sim P1P2: 0.5
juifs iD1:42155205     <>     juifs europe est iD2:11967055  Jaro P1P2:  0.778  Lev Sim P1P2: 0.333
```

Cas ou Levenshtein >0.5 entre les mots

```
nbendjoudi@x2go3:~/TER/TER2/Etape 2$ ./lecture_BNF_RAMEAU_jaro.py
albanie iD1:39027974     <>     albanais iD2:11960038  :  0.869  Lev Sim: 0.75
algerie iD1:42406496     <>     algeriens iD2:11988158  :  0.926  Lev Sim: 0.778
juifs iD1:42155205     <>     juifs iD2:11932167  : Identity similarity  Lev : 1.0
```