**Arabic Morphology**

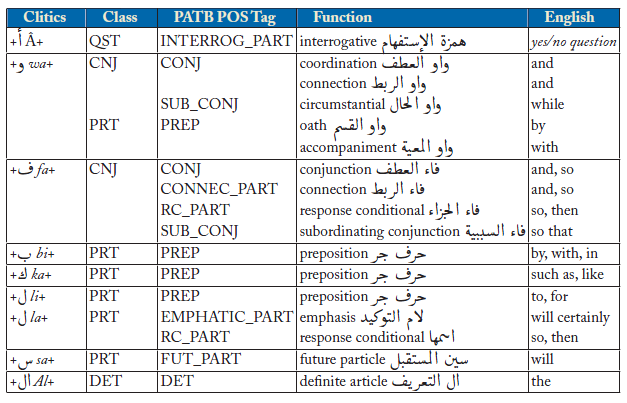
|  |  |
| --- | --- |
| **Mohamed Atia Mohamed** | **Mehad Nasser** |

In Arabic a word can mean a sentence thanks to its compound structure, which is an agglutination of elements of grammar; the following representation outlines a possible structure of a word. Note that the reading and writing of a word is from right to left

Proclitic + Prefix + BODY + Suffix + Enclitics

**Cliticization Morphology:**

* **Proclitics** are prepositions and conjunctions:



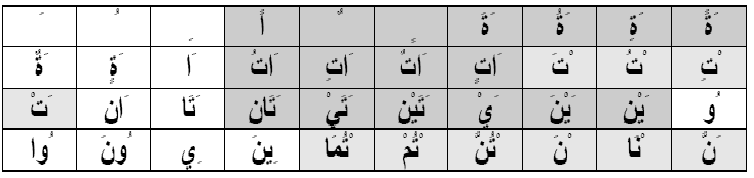
And proclitics we used are:



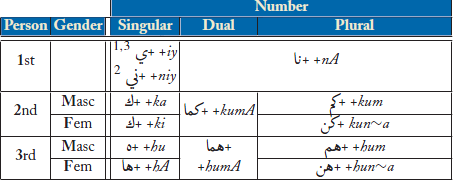
* **Prefixes** areparticles placed at the beginning of verbs in the imperfective (the present continuous ( مضارع) and the imperative (أمر )):



* **Suffixes:**



* **Enclitics:**



Used:

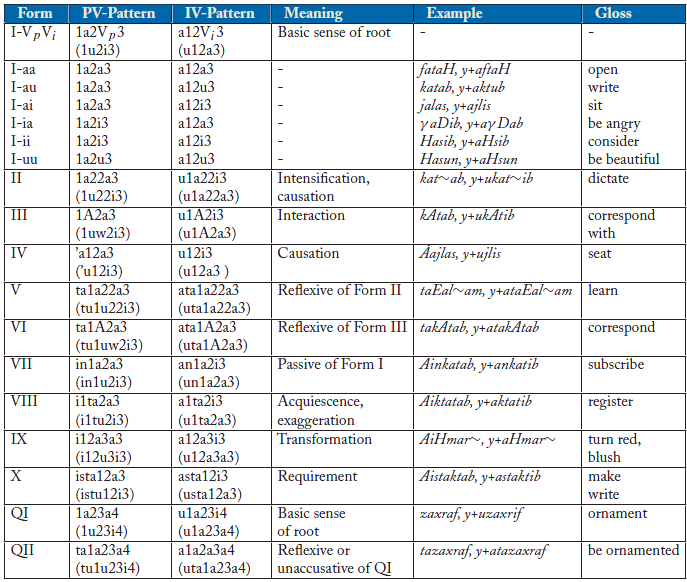


* Prefixes and suffixes express grammatical features and indicate the functions: event name, mode of the verb and the modalities (number, gender, person ...).
* Enclitics can be used to extract PERSON information.

**Base forms, Inflectional Morphology:**

* **VERB morphology**

There are 15 well known forms for verbs inflectional morphology :



* **Particles:**

- Supplementation ( حروف العطف):ف , و, ثم... etc.

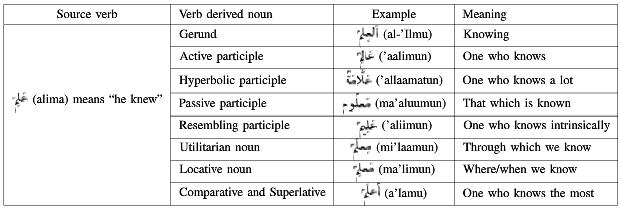
- Interrogative (حروف الإستفهام ): هل,ماذا ,متى , ... etc

- Preposition say ( حروف الجر):الى , فى , ب... etc

- Unaccomplished 'mansoub' (حروف النصب ):أن , إن, لن ... etc..

- Determination ( التعريف): ال

* **Nominals:**



**IMPLEMENTATION**

* **Augmented Transition Network:**

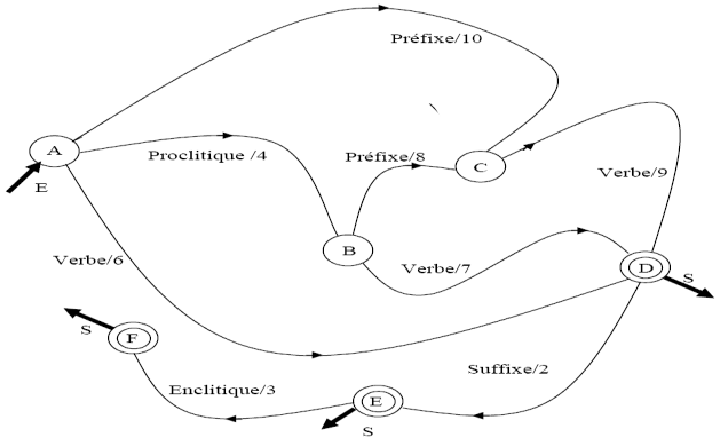
For each ATN there is an initial state, final states, and transitions (arcs) defined:

|  |
| --- |
| %word-network  initial(w, 1).  final(w, 2).  final(w, 3).  final(w, 4).  arc(w,  1, part,  X, Parse, Parsel):-  Parse = [part | Parsel],  X= 2.  arc(w,  1, name,  X, Parse, Parsel):-  Parse = [name | Parsel],  X= 3.  arc(w,  1, verb,  X, Parse, Parsel):-  Parse = [verb | Parsel],  X= 4. |

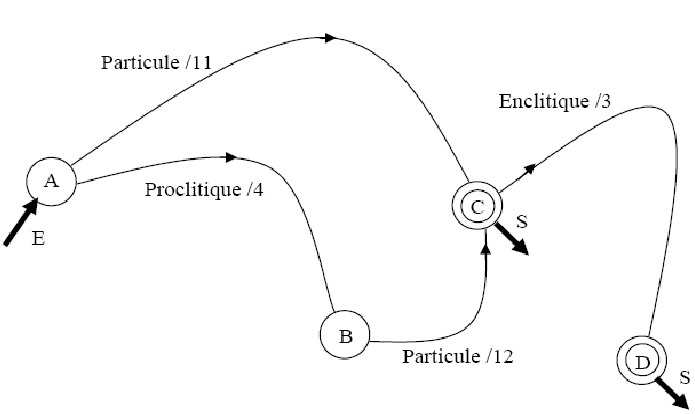
* With each network “w” any predicate has to have an input identifying it’s network so it’s initial(w, 1), which identifies state 1 as initial state for network w.
* Arc(network, from, term, to, outputParse, lastParse), we preferred this to be a predicate not fact form, so we can do various checks within the predicate and change the ATN registers as we need.
* Parse = [name | Parsel]: this is the output of the ATN, and can arc being a predicate makes it easy to change its format, we can easily change the output to a one collective fact instead of a list: verb(asl1(ف), asl2(ع), asl3(ل)), but found the list easier to manage and interoperate with.
* The ATN allows jump arcs which was very useful in building very compact networks:

|  |
| --- |
| arc(pref,  2, jump,  X, Parse, Parsel):-  Parse = Parsel,  X= 3. |

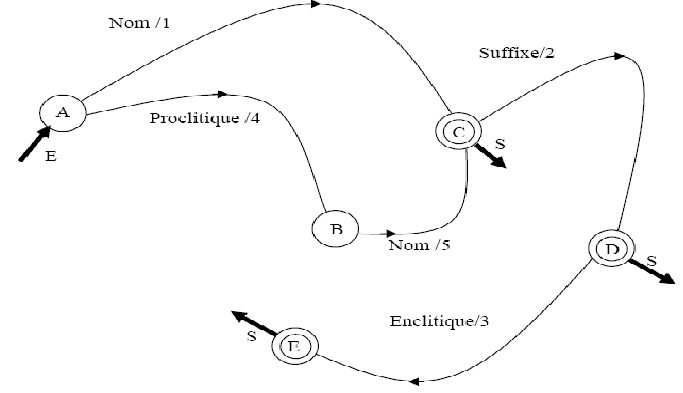
* Verb ATN:



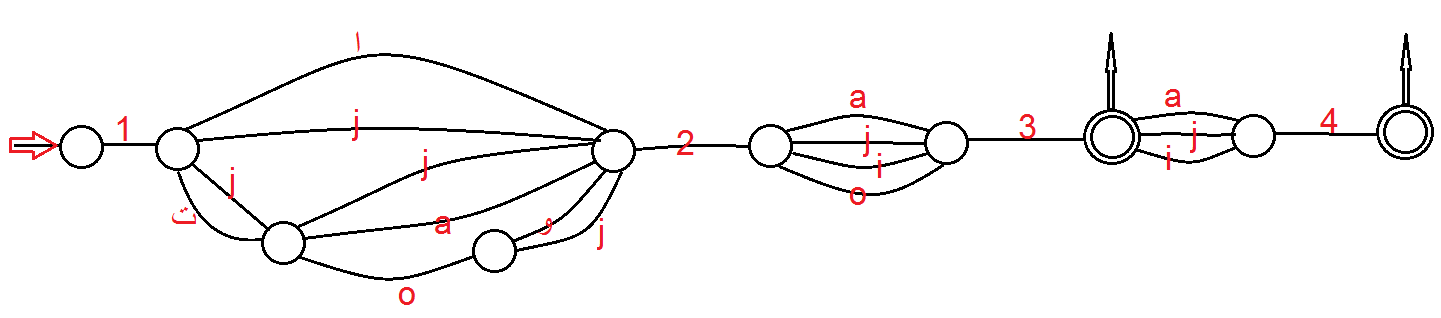
* Particle ATN:



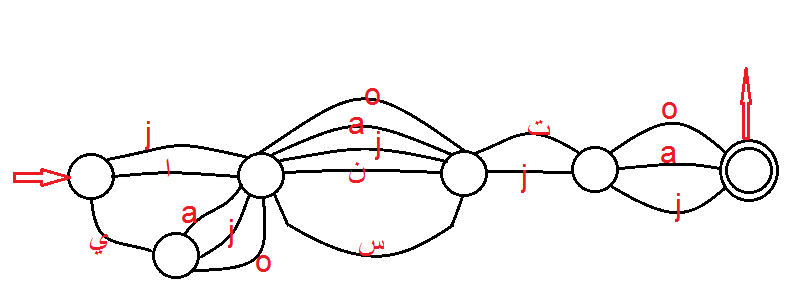
* Name ATN



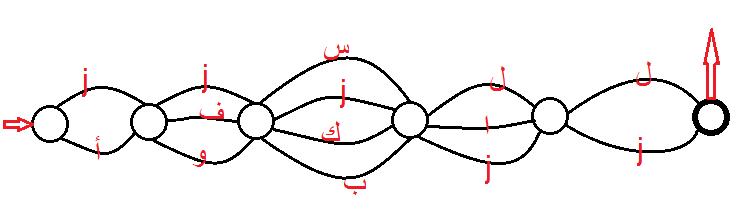
* ASL (Verbe):



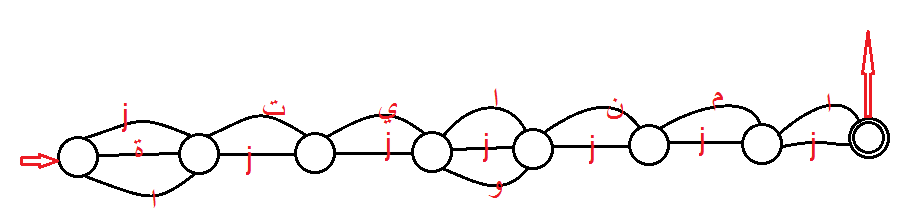
* Prefix (pref):



* Proclitics:



* Suffix:



**Example:**

**?- parser('ذهب محمد الى المدرسة', P).**

P = [[[المدرسة, [noun(مدرسة), [number, singular], [gender, female]], [nom(مدرسة), proc(ل), proc(ا), name], []]], [[الى, [particule(الى)], [particule(p3(ى)), particule(p2(ل)), particule(p1(ا)), part], []]], [[محمد, [noun(محمد), [number, singular], [gender, male]], [nom(محمد), name], []]], [[ذهب, [verb(ذهب), [number, singular], [gender, male], [aspect, past]], [asl(a3(ب)), asl(a2(ه)), asl(a1(ذ)), verb], []]]]