A possible solution of the mid term exam (2013-2014)

INLP-MAI term 2013-2014, Second semester
Mid term exam

We are interested on building a system for mapping acronyms and their expansions provided they occur in the same document (ex. *SARS* = *"Severe acute respiratory syndrome"*).

From English Wikipedia we have extracted the following definitions:

- **acronym** = an abbreviation pronounced as if it were a word, e.g., *SARS* = **s**evere **a**cute **r**espiratory **s**yndrome, pronounced to rhyme with *cars*
- **initialism** = an abbreviation pronounced wholly or partly using the names of its constituent letters, e.g., CD = **c**ompact **d**isc, pronounced *cee dee*
- **pseudo-blend** = an abbreviation whose extra or omitted letters mean that it cannot stand as a true acronym, initialism, or portmanteau (a word formed by combining two or more words).

  (a) = acronym, e.g.: *SARS – (a) Severe acute respiratory syndrome*
  (i) = initialism, e.g.: *CD – (i) Compact disc*
  (p) = pseudo-blend, e.g.: *UNIFEM – (p) United Nations Development Fund for Women*
  (s) = symbol (none of the above, representing and pronounced as something else; for example: MHz – *Megahertz*)

For this exercise we can consider acronyms all of these finer distinctions.
We include in an annex the partial content of the list of acronyms from the same page of the English Wikipedia.

Answer the following questions:

1. (2 points) Write a small python function en python that given a string return True is the string is an acronym and False otherwise.

We can built a pair of patterns for detecting acronyms:

```
acronymPatterns = [
    re.compile('^([A-Z][,\.-/_])+$'),
    re.compile('^([A-Z])+$')]
```

and a parameter

```
MINIMUMLENGTHACRONYM = 2
```

And use the function isAcronym:

```
def isAcronym(st):
    global acronymPatterns
    if len(st) < MINIMUMLENGTHACRONYM:
        return False
    for ipat in range(0,len(acronymPatterns)):
        pat = acronymPatterns[ipat]
        if pat.match(st):
```

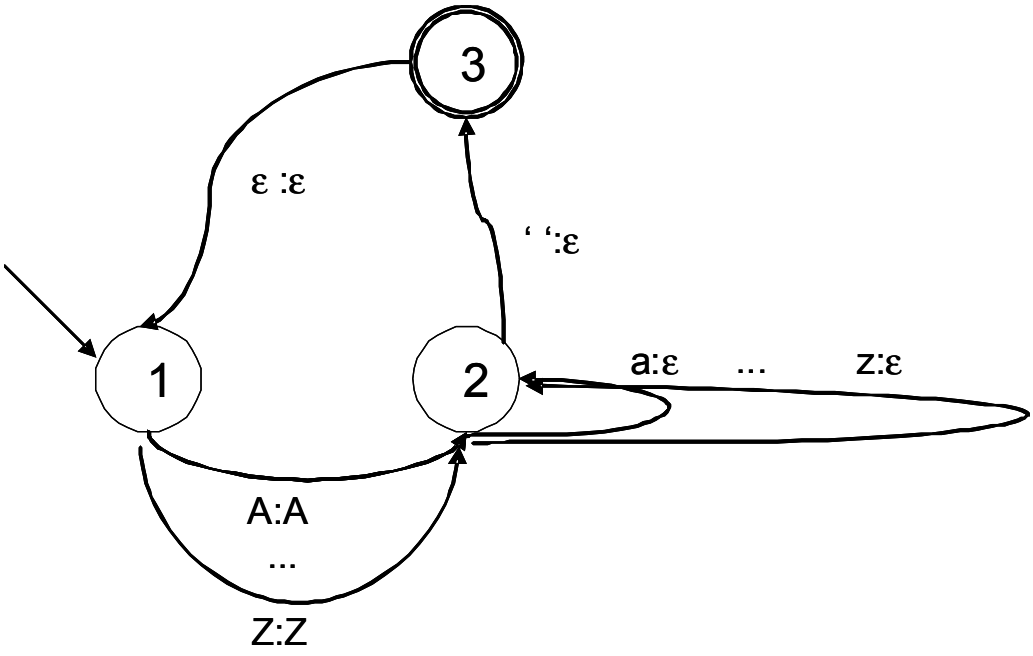
```
            return True
    return False

for testing it we can apply it to one of the lines of annex:
line = 'ARAG - (a) Advanced Research and Assessment Group'
for word in line.split(' '):
        if isAcronym(word):
                print word

Resulting on
:
>>>
ARAG
```

2. (3 points) The simplest, but highly productive case of mapping is when the acronym consists of an string of n uppercase letters while the expansion consists of n words so that the i-ary letter of the acronym corresponds to the first letter of the i-ary word of the expansion (ex. *SARS = "Severe acute respiratory syndrome"*). A reasonable way of implementing the mapping could be though a FST. Draw a FST for implementing this mapping.

The upper level corresponds to the expanded text and the lower level to the acronym. The two vocabularies are identical (in fact, the lower level contains only uppercase letters while upper includes as well lower case letters and the space). The FST has 3 states: 1,2, and 3. state 1 is the initial one. The only final state is 3. On state 1 we are at the beginning of a word of the expanded text, on state 2 into a word and on state 3 at space following a word.



If we apply the FST to a fragment of the example we obtain:

upper	S	e	v	e	r	e		A	c	u	t	e		R	e	s	p	i	r	a	t	o	r	y		S
lower	S							A						R												S
state	1	2	2	2	2	2	3	1	2	2	2	2	3	1	2	2	2	2	2	2	2	2	2	2	3	1

In order to test this FST (obviously it was not asked in the exam) I have used the NLTK module:

```
from nltk.fst import fst
acro = fst.FST('acronyms')                  #A FST is built
acro.add_state('1',False)                   # The 3 states are added
acro.add_state('2',False)
acro.add_state('3',True)
acro.initial_state='1'                      # initial state is set
acro.set_final('3')                         # final state is set
for i in range(ord('A'),ord('Z')+1):        # arcs from '1' are defined
acro.add_arc('1','2',chr(i),chr(i))
for i in range(ord('a'),ord('z')+1):        # arcs from '2' are defined
    acro.add_arc('2','2',chr(i),'')
acro.add_arc('2','3',' ','')
acro.add_arc('3','1','','')                 # arcs from '3' are defined
# We transduce the example
exp = 'Severe Acute Respiratory Syndrome'
acronym = ''.join(acro.transduce(exp+' '))
print 'from', exp, 'to', acronym
and we obtain:
>>> from Severe Acute Respiratory Syndrome to SARS
```

3. (5 points) Recall of this FST is high but, obviously does not cover all the cases (ex. ARAG = "Advanced Research and Assessment Group"). We can detect other cases and manually build other FST but the task boring and time consuming. We should propose a way of building FSTs implementing other types of mappings (as the one included above).

Firstly, we should obtain a lexicon of triples <acronym, expansion, type> from the Wikipedia pare (see annex). For this purpose, we should take into account our definition of acronym and, eventually, the differences between types (a, i, s, p). The definition I have used in question 1 is restricted to acronyms that are sequences of characters of length > 1 containing per uppercase letters and/or dots. For this reason, several candidates in the annex as 'ar', 'Ar','ara', or 'arg' were rejected. From the other examples of the annex, the followimng information could be extracted. Obviously, a more in depth study is needed, because many cases do not occur in the annex. However, for the exam we focus on the ones appearing in the annex:

Acronym	Expansion	Type	comment
AR	Argentina	s	(1)
AR	Arkansas	s	(1)
AR	Armour	s	(1)
AR	Arunachal Pradesh	s	(2)
ARAG	Advanced Research and Assessment Group	a	(3)
ARB	Administrative Review Board	i	solved
ARBA	Army Review Boards Agency	a,i	solved
ARC	Arc-second Raster Chart – Appalachian Regional Commission	i	(4)
ARCA	Automobile Racing Club of America	a	(3)
ARCENT	United States Army Central Command	p	(5)
ARDA	Advanced Research and Development Activity	a	(3)
ARE	Admiralty Research Establishment	i	solved
ARE	United Arab Emirates	i	(5)

ARF	ASEAN Regional Forum	i	solved
ARG	Argentina	s	(1)
ARH	Armed Reconnaissance Helicopter	s	solved
ARI	Acute Respiratory Infection	i	solved
ARI	Army Research Institute for the Behavioral and Social Sciences	i	(4)

From the 18 cases of the table 6 can be directly solved with the FST of question 2, therefore, only 12 remain.
We can classify these 12 cases as following (see the table):

- (1) The expression consists of only one word and the acronym is a prefix of this word (4 cases).
- (2) The expression consists of several words and the acronym is a prefix of the first word (1 case).
- (3) There are cases as those covered by the FST of question 2 where there are also functional words interleaved (as 'and' and 'of') (3 cases).
- (4) There are cases as those covered by the FST of question 2 where only the first words of the expansion are involved (2 cases).
- (5) Two difficulties are involved in this case. First, not all the words in the expansion appear in the acronym and not always they are the initial ones. Secondly, some of the letters of the acronym correspond not to the first letter of the corresponding word in the expansion but to the first two (or eventually more) letters of the corresponding word in the expansion (2 cases).

Not all these cases have to be considered. We have to choose the most productive, the easiest to be implemented and the ones less dangerous in the sense that sometimes for solving omissions (false negatives) we introduce noise (in the form of false possitive). Finding a good balance is frequently difficult. In my opinion a good choice could be the following:

- (1) Type 1 is productive and easy to implement. Besides it does not seem to produce noise. It seems easy to automatically build a FST supporting these cases.
- (2) Type 2 is not very frequent and it could potentially dangerous to consider it. I suggest to discard it.
- (3) Type 3 is productive and it seems easy to implement a solution for this case. If we limit the functional words to be included we can control the noise. It seems easy to automatically build a FST for supporting these cases.
- (4) Type 4 is the most dubious case. Its productivity is not high and I have doubts on its behavior for producing false positives. (some additional experimentation should be needed). Anyway the automatic building of FSTs is not difficult.
- (5) Type 5. is not very frequent and it could potentially dangerous to consider it. I suggest to discard it.

Annex

Fragment of the page http://en.wikipedia.org/wiki/List_of_acronyms:_A#AR of English Wikipedia

- ar – (s) Arabic language (ISO 639-1 code)
- Ar – (s) Argon
- AR – (s) Argentina (FIPS 10-4 country code; ISO 3166 digram) – Arkansas (postal symbol) – Armour – (s) Arunachal Pradesh (Indian state code)
- ara – (s) Arabic language (ISO 639-2 code)
- ARAG – (a) Advanced Research and Assessment Group
- ARB – (i) Administrative Review Board
- ARBA – (a/i) Army Review Boards Agency
- ARC – (i) Arc-second Raster Chart – Appalachian Regional Commission
- ARCA – (a) Automobile Racing Club of America
- ARCENT – (p) United States Army Central Command
- ARDA – (a) Advanced Research and Development Activity (became DTO 2006)
- ARE – (i) UK Admiralty Research Establishment (–1991) – (s) United Arab Emirates (ISO 3166 trigram)
- ARF – (i) ASEAN Regional Forum
- arg – (s) Aragonese language (ISO 639-2 code)
- ARG – (s) Argentina (ISO 3166 trigram)
- ARH – (s) Armed Reconnaissance Helicopter
- ARI – (i) Acute Respiratory Infection – U.S. Army Research Institute for the Behavioral and Social Science.