Textual Networks from Greek Texts

In [85]:

```
# INPUT
import os
text_file = os.path.expanduser('~/cltk_data/greek/text/greek_text_first1kgreek_pl
```

In [86]:

```
# READING THE INPUT FILE
with open(text_file) as f:
    text_raw = f.read()


# CLEANING
from cltk.corpus.utils.formatter import tlg_plaintext_cleanup
text_cleaned = tlg_plaintext_cleanup(text_raw, rm_punctuation=True, rm_periods=Tr
#  rm_periods=True returns no interpunction

# LEMMATIZATION
# does not work so good as I would like
from cltk.stem.lemma import LemmaReplacer
lemmatizer = LemmaReplacer('greek')
text_cleaned = text_cleaned.lower()
text_lemmatized = lemmatizer.lemmatize(text_cleaned, return_string=True)

# STROPWORDS FILTERING
from nltk.tokenize.punkt import PunktLanguageVars
from cltk.stop.greek.stops import STOPS_LIST
p = PunktLanguageVars()
text_tokens = p.word_tokenize(text_lemmatized.lower())
text_tok = [w for w in text_tokens if not w in STOPS_LIST]
text_str = " ".join(text_tok)

# WORD FREQUENCY DICTIONARY/COUNTER
from cltk.utils.frequency import Frequency
freq = Frequency()
text_freq_counter = freq.counter_from_str(text_str)

just_dial_tok = text_tok
just_dial = text_str
```

Word frequneces

In [89]:

```
# WORD FREQUENCY DICTIONARY/COUNTER

from cltk.utils.frequency import Frequency
freq = Frequency()
just_freq_counter = freq.counter_from_str(just_dial)
```

In [90]:

```
just_freq_counter.most_common(20)
```

Out[90]:

```
[('αὐτός', 967),
 ('σύ', 845),
 ('εἰμί', 830),
 ('ἐγώ', 703),
 ('οὐ', 570),
 ('αὐτὸν', 222),
 ('θεάομαι', 221),
 ('τὰς', 195),
 ('θεὸς', 175),
 ('γαῖα', 162),
 ('κύριος', 159),
 ('οὖν', 146),
 ('ἔθνος', 142),
 ('εἶπον', 141),
 ('φημί', 139),
 ('ἔχω', 139),
 ('ἀλλ᾽', 138),
 ('πᾶς', 118),
```

In [ ]:

```
clem_freq_counter.most_common(100)
```

Transforming format to NumPy array

In [7]:

```
# CORPUS CREATION
texts = just_dial
```

In [8]:

```
# ANALYZING THE TEXT BY MEANS SKLEARN AND NUMPY
import numpy as np
from sklearn.feature_extraction.text import CountVectorizer

vectorizer = CountVectorizer(texts)
dtm = vectorizer.fit_transform(texts)
vocab = vectorizer.get_feature_names()
```

In [9]:

```
dtm
```

Out[9]:

```
<2x20276 sparse matrix of type '<class 'numpy.int64'>'
        with 23077 stored elements in Compressed Sparse Row format>
```

In [10]:

```
dtm = dtm.toarray()
vocab = np.array(vocab)
```

Collocations

In [94]:

```
import nltk
from nltk.collocations import *
bigram_measures = nltk.collocations.BigramAssocMeasures()
trigram_measures = nltk.collocations.TrigramAssocMeasures()
```

In [95]:

```
# input
finder = BigramCollocationFinder.from_words(just_dial_tok)

# finder = BigramCollocationFinder.from_words(clem_paed_tok, window_size = 5)
```

In [96]:

```
# filter
finder.apply_freq_filter(5)
# selection
best_20 = sorted(finder.nbest(bigram_measures.raw_freq, 20))
```

In [97]:

```
# to form a numpy array

import numpy as np
best_20 = np.array(best_20)
reversed_best_20 = np.flip(best_20, 1)
```

In [98]:

```
bgs = nltk.bigrams(clem_paed_tok)
fdist = nltk.FreqDist(bgs)
for k,v in fdist.items():
    print(k,v)
```

```
('κεφάλαια', 'πρώτου') 1
('πρώτου', 'λόγου') 1
('λόγου', 'ἐπαγγέλλεται') 1
('ἐπαγγέλλεται', 'παιδαγωγός') 2
('παιδαγωγός', 'τὰς') 1
('τὰς', 'ἁμαρτίας') 10
('ἁμαρτίας', 'ἐγώ') 2
('ἐγώ', 'παιδαγωγὸς') 7
('παιδαγωγὸς', 'ἐπιστατέω') 2
('ἐπιστατέω', 'γ') 1
('γ', 'φιλάνθρωπος') 1
('φιλάνθρωπος', 'παιδαγωγός') 2
('παιδαγωγός', 'δ') 1
('δ', 'ἐπ᾽') 1
('ἐπ᾽', 'ἴσος') 9
('ἴσος', 'ἀνδρόω') 2
('ἀνδρόω', 'γυνή') 4
('γυνή', 'λόγος') 2
('λόγος', 'παιδαγωγός') 3
('παιδαγωγός', 'εἰμί') 3
```

In [139]:

```
finder = BigramCollocationFinder.from_words(just_dial_tok, window_size = 3)
pairs_list = sorted(finder.ngram_fd.items(), key=lambda t: (-t[1], t[0]))
```

In [140]:

```
pairs_list
```

Out[140]:

```
[(('οὐ', 'εἰμί'), 74),
 (('σύ', 'σύ'), 50),
 (('αὐτός', 'εἰμί'), 48),
 (('αὐτός', 'αὐτός'), 47),
 (('εἰμί', 'αὐτός'), 39),
 (('σύ', 'εἰμί'), 39),
 (('ἐγώ', 'ἐγώ'), 39),
 (('εἰμί', 'σύ'), 36),
 (('εἰμί', 'ἐγώ'), 34),
 (('σύ', 'οὐ'), 32),
 (('αὐτός', 'οὐ'), 31),
 (('ἐγώ', 'οὐ'), 31),
 (('ἐγώ', 'εἰμί'), 28),
 (('εἰμί', 'εἰμί'), 27),
 (('εἰμί', 'οὐ'), 26),
 (('σύ', 'ἐγώ'), 26),
 (('ἐγώ', 'σύ'), 26),
 (('εἰμί', 'τρύφων'), 25),
```

In [141]:

```
import numpy as np
pairs_array = np.array(pairs_list)
```

Short version (300 pairs)

In [193]:

```
pairs_array_short = pairs_array[:300]
```

In [199]:

```
# changing the data structure

def column(matrix, i):
    return [row[i] for row in matrix]
pairs_only = column(pairs_array_short, 0)

def column(matrix, i):
    return [row[i] for row in matrix]
counts_only = column(pairs_array_short, 1)

pairs_only = np.array(pairs_only)
counts_only = np.array(counts_only)
```

In [200]:

```
counts_combined = np.insert(pairs_only, 2, counts_only, axis=1)
```

In [201]:

```
counts_combined
```

Out[201]:

```
array([['οὐ', 'εἰμί', '74'],
       ['σύ', 'σύ', '50'],
       ['αὐτός', 'εἰμί', '48'],
       ['αὐτός', 'αὐτός', '47'],
       ['εἰμί', 'αὐτός', '39'],
       ['σύ', 'εἰμί', '39'],
       ['ἐγώ', 'ἐγώ', '39'],
       ['εἰμί', 'σύ', '36'],
       ['εἰμί', 'ἐγώ', '34'],
       ['σύ', 'οὐ', '32'],
       ['αὐτός', 'οὐ', '31'],
       ['ἐγώ', 'οὐ', '31'],
       ['ἐγώ', 'εἰμί', '28'],
       ['εἰμί', 'εἰμί', '27'],
       ['εἰμί', 'οὐ', '26'],
       ['σύ', 'ἐγώ', '26'],
       ['ἐγώ', 'σύ', '26'],
       ['εἰμί', 'τρύφων', '25'],
```

NetworkX

In [202]:

```
import networkx as nx
G = nx.Graph()
```

In [203]:

```
G.clear()
G.add_weighted_edges_from(counts_combined)
```

In [204]:

```
G.number_of_nodes()
```

Out[204]:

```
164
```

In [205]:

```
G.number_of_edges()
```

Out[205]:

```
254
```

In [206]:

```
list(G.edges())
```

Out[206]:

```
[('οὐ', 'εἰμί'),
 ('οὐ', 'σύ'),
 ('οὐ', 'αὐτός'),
 ('οὐ', 'ἐγώ'),
 ('οὐ', 'ἀλλ᾽'),
 ('οὐ', 'μόνον'),
 ('οὐ', 'οἷ'),
 ('οὐ', 'ἔθνος'),
 ('οὐ', 'φημί'),
 ('οὐ', 'ἔχω'),
 ('οὐ', 'εἶπον'),
 ('οὐ', 'θεάομαι'),
 ('οὐ', 'οὐ'),
 ('οὐ', 'γιγνώσκω'),
 ('οὐ', 'ἄνθρωπος'),
 ('οὐ', 'αὐτόν'),
 ('οὐ', 'γαῖα'),
 ('οὐ', 'κύριος'),
```

In [207]:

```
nx.degree_centrality(G)
```

Out[207]:

```
{'αὐτός': 0.3067484662576687,
 'αἰών': 0.036809815950920245,
 'αἰῶνος·': 0.006134969325153374,
 'αἴρω': 0.006134969325153374,
 'αἷμα': 0.006134969325153374,
 'αὐτὸν': 0.036809815950920245,
 'αὐτόν': 0.012269938650306749,
 'αὐτῷ·': 0.006134969325153374,
 'βασιλεὺς': 0.012269938650306749,
 'γαῖα': 0.030674846625766871,
 'γαστρὶ': 0.024539877300613498,
 'γεννάω': 0.006134969325153374,
 'γιγνώσκω': 0.024539877300613498,
 'γραφάς': 0.006134969325153374,
 'γένει': 0.012269938650306749,
 'γένους': 0.018404907975460124,
 'δίδωμι': 0.006134969325153374,
 'δαμασκός': 0.024539877300613498,
```

In [208]:

```
G
```

Out[208]:

```
<networkx.classes.graph.Graph at 0x120a70550>
```

In [209]:

```
nx.write_gexf(G, "test.gexf")
```

In [ ]:

```

```