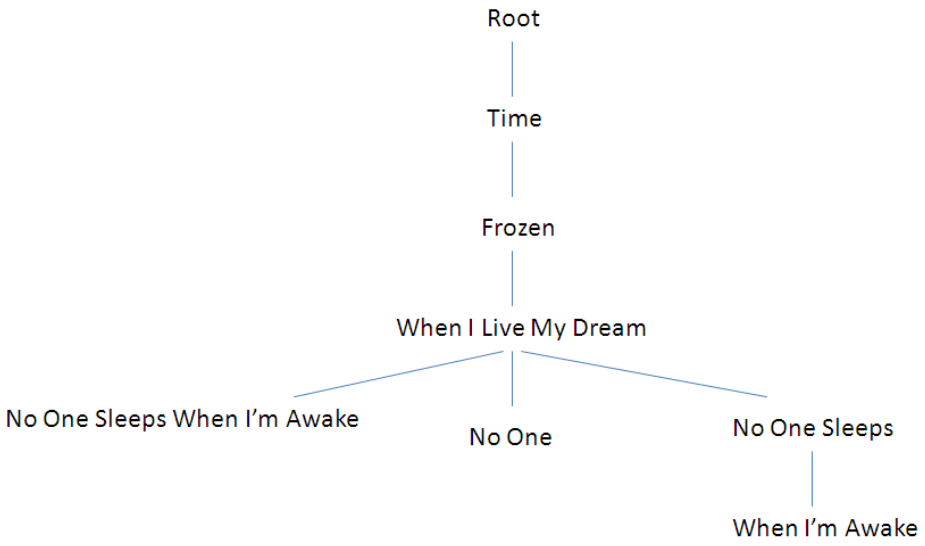
This code automatically generates a “Spotify poetry” tracklisting based on a user inputted “poem” via a simple tree-based search algorithm (see: http://spotifypoetry.tumblr.com/). It took me a total of about 12 hours working sporadically across a few days to think about the problem itself and implement the solution in Python.

The first aim of the code is to generate a tree whose nodes correspond to track titles matching portions of the poem. Starting at the root of the tree, each path down to a leaf should correspond to a sequential traversal of the poem or a portion of the poem. Once the tree is generated, I apply a depth-first search to find all of the tracklist solutions. The solution criterion is defined as whenever the accumulated sum of words found in tracks along any search path equals the total number of words in the original poem. The figure below illustrates a simple example for a poem “time frozen when I live my dream no one sleeps when I’m awake”. Starting at the root, the two solutions that can be found using the depth-first search are the paths with leaves ending at “No One Sleeps When I’m Awake” and “When I’m Awake”. The current algorithm aims to find all existing tracklist solutions.



Usage

My code assumes that the “spotimeta” library is available/installed. In the file spotify.py, the input_string variable in the “main” block can be set to any string (poem) input to test the code. Currently, it is set to "time frozen whEn I live my Dream no one sleepS when i'm aWake". When run, there will be output that tracks the generation of the tree. All solutions (if they exist) will be printed out subsequently in the format:

“SOLUTION

Track_title, Spotify URI

….

END SOLUTION”