Word Ladders

Thore Husfeldt

March 21, 2017

Description

Find paths between given words among the five-letter words of English.

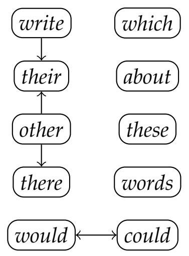

Figure 1: words-10.

How words connect

We define a directed graph of words as follows. There is one node for every 5-letter word. There is an arc from v to w if each of the last four letters of v appears in w. For example, there is an arc from *yodel* to *lodes*, but not from *lodes* to *yodel* because the latter contains no S. On the other hand, there is an arc from *sharp* to *graph* and back. All four letters have to appear with repetitions, so there is an arc from *where* to *ether* (both Es appear) but not to *retch* (E appears only once). As an example, here's a pretty long path in the graph:

> *climb* → *blimp* → *limps* → *pismo* → *moist* → *stoic* → *ioctl* → *colts* → *lotsa* → *stoae* → *oaten* → *neath* → *hated* → *dated* → *dater* → *rater* → *tread* → *dared* → *dread* → *drear* → *rarer* → *reran* → *arena* → *earns* → *snarf* → *franc* → *narco* → *orcas* → *scare* → *raced* → *decaf* → *fecal* → *eclat* → *talcs* → *clasp* → *psalm* → *slams* → *small* → *llama* → *lamas* → *amass* → *smash* → *shame* → *hames*

Requirements

For a given instance `words-xxx` you build a directed graph representing the adjacency structure of the words in the file called `words-xxx.txt`.

Then you are given a list of word pairs in `words-xxx-in.txt`.[1]

[1] A pair of words could be `climb` and `hames`, for example.

For each pair of words you should output the length between these words, which should be computed with a BFS. The correct answers are specified in `words-xxx-out.txt`. Your algorithm should be able to both return the length of the shortest path *and* a specific such path. The given output only shows the lengths of the paths, but for the report you will need to find an explicit path.

The data directory contains a number of test inputs with known distances, your algorithm must work correctly on all of those.

The running time of the BFS-part of the algorithm needs to be in $O(n + m)$, where n and m refer to the number of nodes and arcs in the underlying directed graph. Building the graph in quadratic time is accepted, but you should try to find a better solution.[2]

[2] Of course, for dense graphs, $O(n + m)$ would be quadratic time anyway. But the input graph is not dense.

Deliverables

1. The source code for your implementation
2. A report in PDF. Use the report skeleton in the `doc directory`.

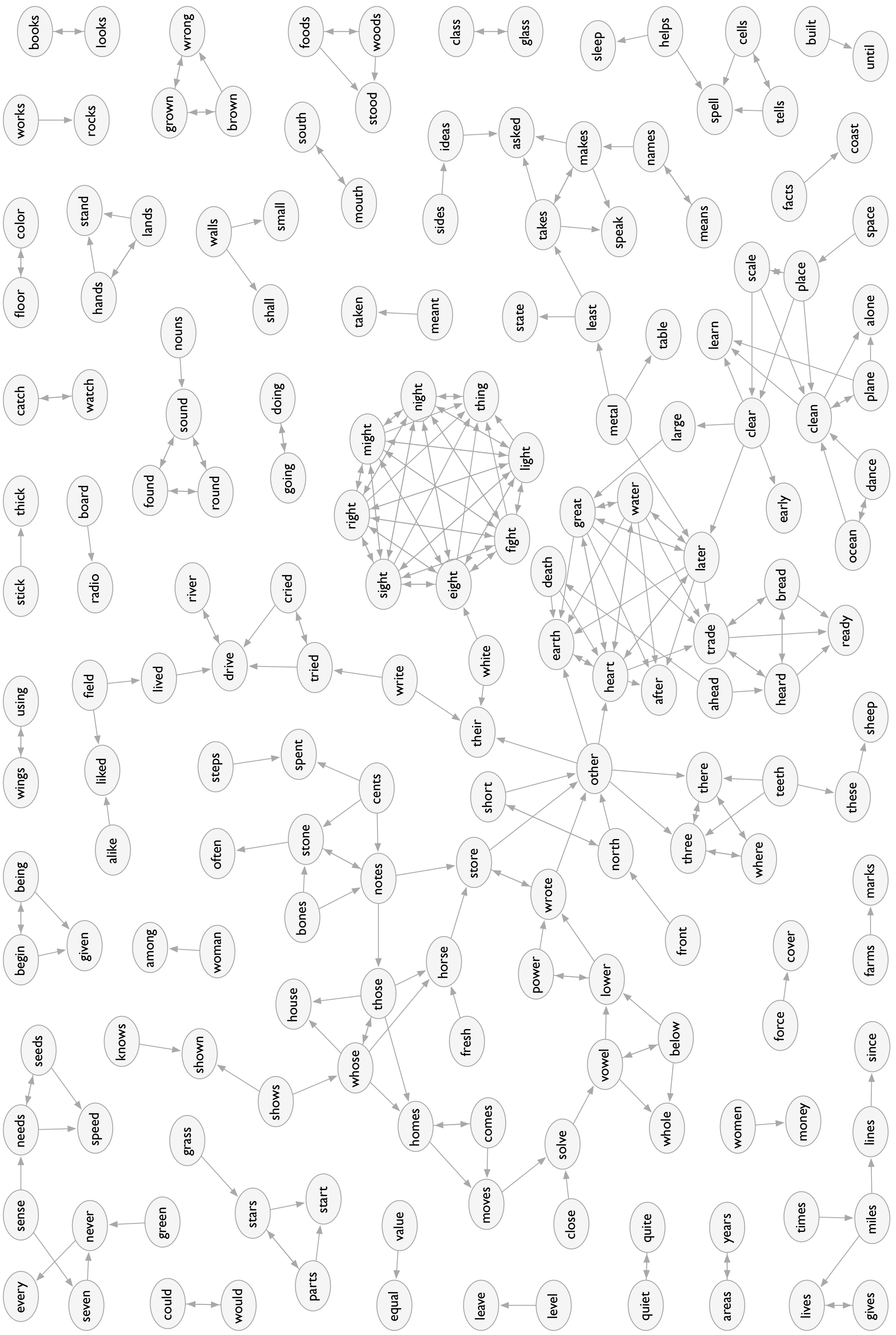

Figure 2: words-250.