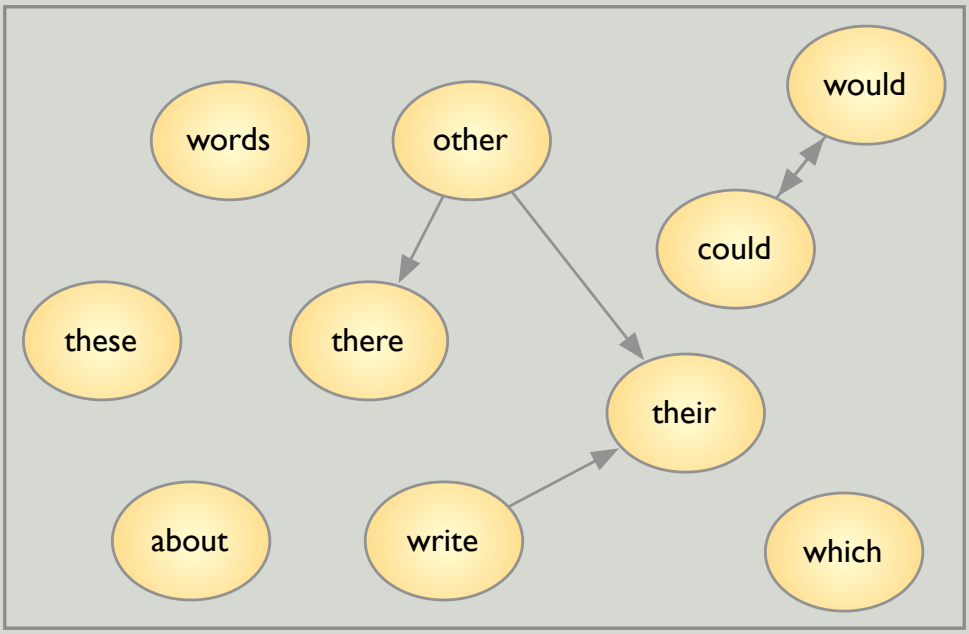
WORD LADDERS

Thore Husfeldt
25 March 2009 8:34

Description

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



How words connect

You can go from one word to another if each of the last four letters of the former word appears in the latter word. For example, you can go from “yodel” to “lodes”, but you can’t go from “lodes” to “yodel” because the latter contains no S. On the other hand, you can go from “sharp” to “graph” and back. All four letters have to appear with repetitions, so there is an edge 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

Files

The files for this exercise are in `/usr/local/cs/edaf05/lab2/`

The “.dat” files just contain 5-letter words, one per line. They come in various sizes, the big one has 5757 words.

Input test files contain just pairs of words, one pair per line. Output test files contain the distance (counted in number of edges) from the first to the second word in these pairs. If no such path exists, the test file reports “-1.” There are test files for various sizes of graph (but not for all).

Solution quality

Minimal solution

You build a directed graph representing the adjacency structure of the words, and run BFS on it. In particular, your algorithm works correctly on all test inputs.

Good solution

The running time of your algorithm is be O(n+m), where n and m refer to the number of nodes and arcs in the underlying directed graph. In particular, you’re not supposed to use quadratic time to build the graph, even though it’s tempting. (Of course, for dense graphs, O(n+m) would be quadratic time, but the input graph is not dense.)

words-10.dat

```
there
which
their
about
these
words
would
other
write
could
```

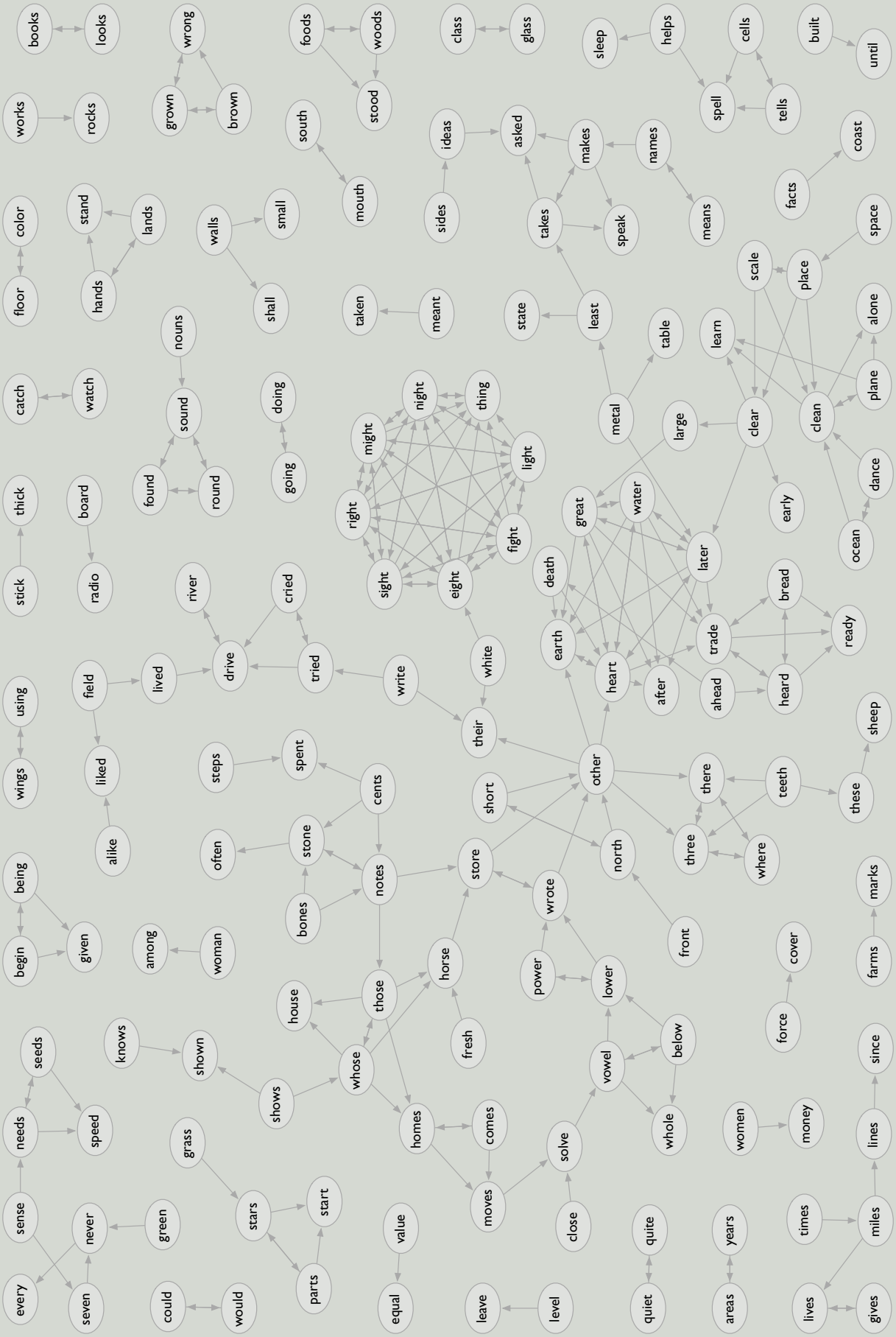
words-10-test.in

```
other there
other their
could would
would could
there other
about there
```

words-10-test.out

```
1
1
1
1
-1
-1
```

WORD LADDERS



words-250 as a graph, but without the singleton nodes