Report
Ivan Konstantinov

1. Introduction

The aim of this report is to briefly describe the major design and implementation decisions for this practical, as well as to present any observations that were made regarding the functioning and reliability of both PageRank and HITS algorithms in the context of the given problem.

The code is structured in the following way: the implementation of the PageRank algorithm can be found in the file *page_rank.py*, while the HITS algorithm is contained within the file *hits.py*. To run any of the algorithms, import its corresponding file in the *Python* interpreter and run the following commands:

1. For PageRank, you can either run *do_all()*, which will run the algorithm and will produce the final graph shown below, or you can manually run the functions *process_dat()*, *pr()*, and *find_top20_links()* (note that *process_dat()* **must** be called before calling any other function)
2. For HITS, you can either run *do_all()*, which will run the algorithm and produce the text documents *hubs.txt*, *auth.txt*, or you can manually run the functions *process_dat()*, and *hits()* (the same requirement, mentioned in *1.* applies here)

2. Used libraries and consulted sites

The following libraries were used: **re, itertools, os, math**. The *readFile()* and *writeToFile()* functions were taken from **Lab2Support.py**, while *Wikipedia* was consulted for the normalization routines in the algorithms.

3. Algorithm implementation

1. PageRank
    1. Implementation:
        1. Pre-processing

        The pre-processing stage loads the text file, *graph.txt*, and identifies all unique and communicating nodes, which will be used for calculating the PageRank. For this goal, two indexes are created: *node_outlinks* and *node_inlinks*. A *node_outlinks* entry for a given node contains the entry's number of outgoing links, while a *node_inlinks* entry contains a list of all nodes that link to (that is, send emails to) the given node. The message hash values are not deemed relevant for the running of the algorithm, since it is assumed that there are no duplicates in *graph.txt*: there are no two lines that have the same message, sender and recipient.

        Lines that contain the same sender and recipient node are ignored altogether and are not used in the calculation of PageRank. Thus, the total number of nodes is less than the total number of unique nodes found in the whole text: it is manually set to 86029.

        2. Algorithm

        The algorithm implementation is the same as described in the lecture slides. Two indexes are used to compute the PageRank for every node: *pr_index* contains the final PR score for each node, as well is used in the intermediate PR computations in each iteration, while *temp_index* is used to store the intermediate PR scores, which are then copied into *pr_index* at the end of each iteration. Thus, it is ensured that the PR score of a node is computed using the PR scores obtained during the previous iteration, and not the current one.

        After the PR score of each node is computed, each score is normalized by dividing it with the sum of all computed PR scores for each node during the given iteration. It is thus ensured that the PR scores of all nodes sum up to 1.

        It has been observed that the PageRank algorithm converges closest to the given sanity check scores in 11 iterations.

    2. Observations

    The obtained results seem to show that some influential people within Enron can be accurately identify using PageRank, given the specific email corpus. For example, Louise Kitchen, who was the president of Enron Online, has one of the highest PR scores, as well as John Lavorato, CEO of Enron America. Some concealed relations between these individuals can also be determined using PageRank [1]. For example the relation between Tana Jones and Sara Shackleton, and Jeff Dasovich and Richard Shapiro can be determined using PageRank, as shown on the graph below.

2. HITS
    1. Implementation
        1. Pre-processing

        The pre-processing stage for the HITS algorithm is practically equivalent to the one for PageRank, except for the indexes that are created. An *out_index* stores all outgoing links for a particular node, while an *in_index* stores all incoming links to that node. The number of considered nodes is the same as in PageRank, with entries in *graph.txt* where a node is mailing itself ignored from the computation process.

        2. Algorithm

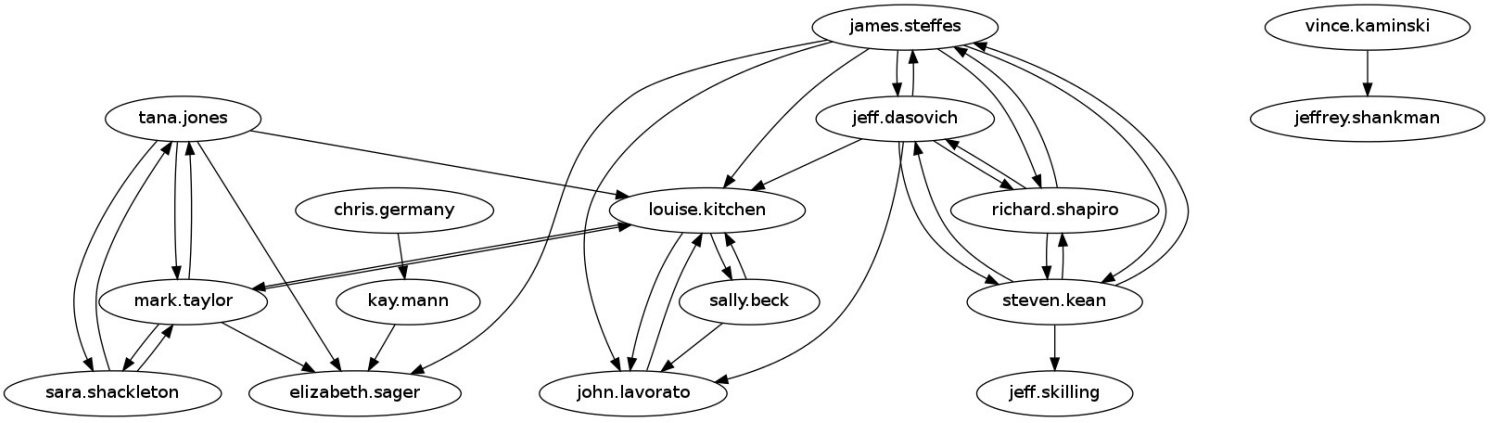
        The HITS algorithm is implemented in a similar way as described in the slides. Here, no index is used to store

the intermediate results, as the *authority* scores are calculated using the newly computed *hub* scores. A normalization factor is used for both *hub* and *authority* scores and is equal to the square root of the sums of squares of the computed *hub*/*authority* scores for each node. At the end of each iteration, the scores for each node are normalized using the computed factor.

3. Observations:
    1. A significant difference in the final computed *hub/authority* scores was observed, which depends on the order in which calculations are performed. If, in a given iteration, *authority* scores are before *hub* scores, resulting scores deviate from the provided sanity checks (for **jeff.dasovich**, 0.000210016/0.00100523 for *authority* and *hub*, respectively, in 10 iterations; 0.000210085/0.00100550 for *authority/hub* in 9 iterations), while the reverse ordering produces results that fit nicely (for **jeff.dasovich**, 0.0002100358/0.0010059473 for *authority*/*hub*) if the algorithm uses 9 iterations to converge.
    2. The HITS algorithm fails to identify important and influential people within Enron. The node with the highest *hub* score is **pete.davis@enron.com**, which is actually the name for the broadcast proxy system and is not associated with any real individual. The reason for scoring the highest is because **pete.davis** is used to send auto-generated emails to a huge number of recipients, thus its initial *hub* value will be significantly higher than most other nodes, increasing significantly in later iterations (to a staggering score of 0.99928093!), because it will link to nodes with high *authority* values, computed due to the link having many incoming links from **pete.davis**. The node with the highest authority score is **ryan.slinger@enron.com**, which has received **3551** out of **3894** different messages from **pete.davis**. The second node with highest authority score follows the same pattern, and so on. Thus, the HITS algorithm does not reflect how important the node is within the communication scheme, but how many messages it has received from **pete.davis**.

4. Graph

The graph shown below is created using the resulting scores from the PageRank algorithm, as they seem to identify more accurately the important people. The graph is created automatically and uses the PR scores from the top 21 nodes[1]. It can be seen that not all links are shown: only links that reflect the close relationship between different nodes are considered. For example, the link between **vince.kaminski** and **jeffrey.shankman** is present, given the provided message corpus, the person who Vince has sent most emails to is Jeffrey Shankman (39 messages out of 2013, given 453 unique recipients).



REFERENCE

N. Pathak, J. Srivastava, Automatic Extraction of Concealed Relations from Email Logs, *Army High Performance Computing Conference, November 2006*

1 The practical states to take the top 10-20 people, but I have selected 21 nodes, because Richard Shapiro scores as the 21st most influential person, which reinforces my point regarding the identification of some concealed relations, described in [1].