**Finding Story Chains in Newswire Articles**

Xianshu Zhu, Tim Oates

IEEE IRI, 2012

Summary submitted by *Anunaya Srivastava*

The aim of the author is to build a chain of documents to connect two documents which were published at different times. The input to the algorithm is a set of documents with their timestamps, a start document(*s*) and an end document(*t*). The author does not intend to generate summaries but re-organize articles in a meaningful and coherent manner. The author uses a random walk algorithm to find the story chains.

A good story chain should have the following properties -

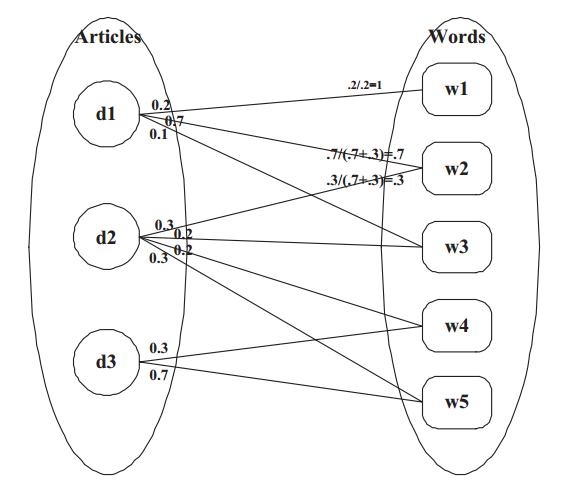
1. Relevance: The chain should consist of articles relevant to the user query.
2. Coherence: The entire chain should have a coherent and consistent theme. The transitions between the nodes on the chain should be smooth and should not digress from the main theme of the storyline.
3. Low Redundancy: There should not be repeating articles about the same event in the storyline.
4. Coverage: The chain should cover every important event of the story.
5. Efficiency: The running time of the algorithm should be less as compared to previous approaches.

The author models the problem at hand as a divide and conquer bisecting search problem. The initial story chain contains only one links−t, where s is start article and t is end article. Each time we insert a node on a link, the link will be divided into two sub-links. The bisecting

search adds a node on each sub-link recursively. The final story chain will be composed of multiple links. Simulate random walks on a bipartite graph to calculate article relevance scores, which are used to select the best nodes to add to the link.

The entire approach can shown as –

Consider a bipartite graph where vertices consists of documents and words; and edges labels are tf-idf weights. Let rs(di) be the probability that a random walk reaches di from s.



1. **Prune least relevant articles**

Author uses the concept of coherence given by Shahaf et.al. in ‘Connecting the Dots’ and defines coherence of the chain as the minimum link strength of the chain where link strength is calculated using word similarity. We prune all articles di such that rs(di) < rs(t) or rt(di) < rt(s).

1. **Search most relevant article**

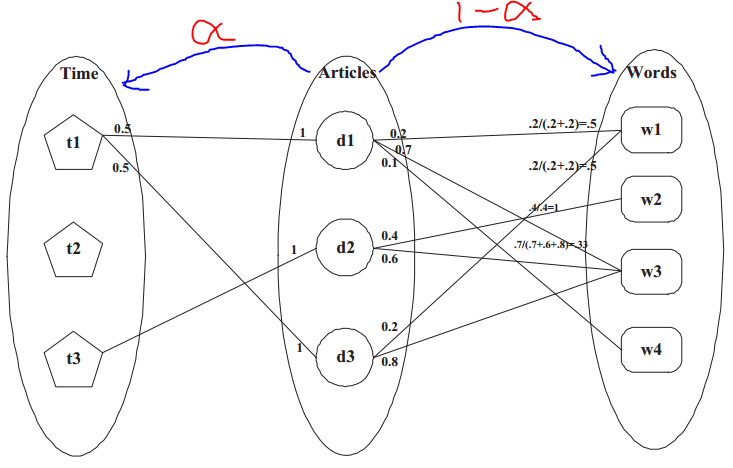
Find the most relevant article, say A, between the end points s and t by using the following

A = arg maxi{rs(di) \* rt(di)}

i.e. the most relevant article has the highest probability of reaching from s as well as from t.

1. **Prune redundant articles**

We want to select the most representative article but don’t want to remove similar articles with different timestamps. We add time nodes to the previous bipartite graph. Article-time weight is always 1 as an article belongs only to a single time frame, time-article weight is normalized over the number of edges from time to article.



The random walk starts from the middle article nodes and moves to time nodes with probability α and moves to word nodes with probability (1-α). Value of α decides how influential time nodes are. This new random walk is more likely to reach articles that are in the same bin and close in content.

One drawback of this graph is that is puts more weight on articles that are in the same time bin and ignores the difference between the bins that are close and bins that are far away.

**Future work**

This method doesn’t consider branching i.e. the user chooses an end node. More work can be done to detect and form story chains with different branches.