**Trains of Thought: Generating Information Maps**

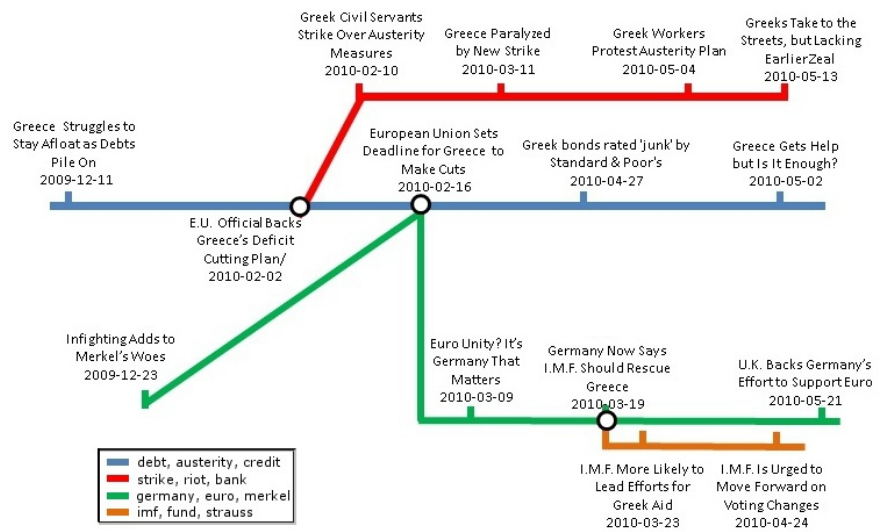
Dafna Shahaf, Carlos Guestrin, Eric Horvitz

WWW, 2012

Summary submitted by *Anunaya Srivastava*

The goal of this paper is to create a 2-D storyline which the author calls ‘Metro Map’. A metro map consists of a set of lines(chain of documents) which have intersections and overlaps. Each line represents a coherent chain of events and different lines focus on the different aspects of the story.

The author asserts that the concept of 2-D storyline is novel and represents a more holistic view of the story at hand. Previous works have been done on summarization, topic detection and tracking, timeline generation – all of which produces a linear chain of events. The author gives the characteristics of a good metro map and provides an optimized algorithm to construct such a metro map. Example of a metro map is as follows –



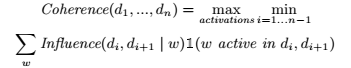
**Definition: Metro Map**

A metro map M is apair (G, Π), where G = (V, E) is a directed graph and Π isa set of paths in G. Paths are referred as metro lines. Eache ∈ E must belong to at least one metro line.

**Characteristics of a good Metro Map**

1. **Coherence**

A coherent chain has a global coherent theme across the storyline i.e. all the documents in the chain belong to a single theme. Two consecutive documents in a chain has high influence(for a given word w) if the two documents are highly connected and w is important for connectivity. I have discussed the concept of coherence in details in the summary of paper ‘Connecting the dots’ which is written by the same author. To summarize the problem is an optimization problem where the goal is to choose a small set of words(called active words) which are used to calculate the score of each transition.



The coherence of map is the minimal coherence across its lines Π.

Even if we include highly coherent chains in the map, we get a number of chains which revolve around irrelevant topics and are repetitive. Thus lines should cover topics which are important to the user. Therefore, coverage comes into picture.

1. **Coverage**

We want the map to have important topics which are also diverse in nature. We first define coverage of a single document and then move on to coverage of a map. Let function coverdi(w) : W → [0, 1]quantify the amount that document di covers feature w. Forexample, if W is a set of words, we can define cover(·) astf-idf values. Coverage of a map M can be given as



Drawback of above formula is that it doesn’t encourage diversity i.e. doesn’t encourage to cover multiple features. Thus the author proposes the following –



Thus, if the map already includes documents which cover wwell, coverM(w) is close to 1, and adding another documentwhich covers w well provides very little extra coverage ofw. This encourages us to pick articles which cover otherfeatures, promoting diversity. Coverage of entire map is defined as



1. **Connectivity**

Author defines connectivity as the number of lines of Π that intersect.



**Objective Function**

There has to be a trade-off between the 3 properties. If coherence is maximized, we will have repetitive and low coverage chain. If connectivity is maximized, we will have a number of similar chains in the map as similar chain tend intersect more. Lastly, maximizing coverage leads to low connectivity. Trade-off is implemented using the following steps –

1. **Finding coherent chains**

To pick good chains, we wish to list all possible chains but such chains will be exponential in number. Hence, author uses divide and conquer approach, to construct larger chains using shorter ones. A graph is constructed where each vertex corresponds to a short chain and edges indicate chains which can be concatenated and still maintain coherence. The problem is that if we concatenate any two chains that share an endpoint for eg. (d1, ..., dk) and (dk, ..., d2k−1)to form (d1, ..., d2k−1)thereis no evidence that both chains belong to the same storyline,despite having a shared article dk. To address this problem author gives the concept of m-Coherence.

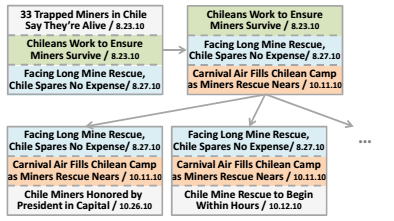
**Defintion: m-Coherence**

A chain (d1, ..., dk)has m-coherence τ if each sub-chain of length m (di , ..., di+m−1),   
i = 1, ..., k − m + 1 has coherence at least τ.

To m-coherent chain having m-1 overlap can be combined into one. Now, if 2 m-coherent chains are concatenated then the new chain will be m-coherent. So, we have a directed coherence graph where vertices are m-coherent chains and edges indicate chains which can be combined. We are still left to find coherent chains.

**Finding short coherent chains**

These chains can be generated by a general best-first search strategy. In a nutshell,we keep a priority queue of sub-chains. At each iteration,we expand the chain which features the highest coherence,generating all of its extensions. When we reach a chain oflength m, we make it into a new vertex and remove it fromthe queue. We continue until we reach our threshold. An example of coherence graph is given below.



1. **Finding high coverage map**

Given a coherence graph G, we find paths p1...pKs.t. Cover(docs(Uipi)) is maximized, and |docs(pi)|≤ l, where pi∈Π, docs(pi) is the set of vertices in the path pi. This is an NP-hard problem, so we use submodular orienteering to compute approximate max-coverage paths. This method is applied because cover() is a submodular function. Solving submodular function is beyond the scope of this discussion.

1. **Maximize connectivity**

In order to increase connectivity, author uses a local-searchtechnique. At iteration i, consider each path p ∈Πi−1.Hold the rest of the map fixed, and try to replace p by p’ that increases connectivity and does not decrease coverage.At the end of the iteration, pick the best move and applyit, resulting in Mi