Constructing a Network of Digital Objects

Kui Xu, Naren Sundaravaradan

May 11, 2014

1 Introduction

Cyber Infrastructure for Network Science (CINET), is a web portal providing a computational and analytic environment for the network science researcher and educator. It uses HPC resources to service experiment execution requests. CINET provides users with not only many realistic graphs for analysis, but also a collection of graph-based algorithms to be applied.

This project aims at developing a RDF graph building service. The responsibility of this service is to do web crawling and find digital contents related to user requests. More specifically, the type of contents to be collected should be related to epidemiology. Eventually the service would deliver a RDF network of digital contents that can be stored on CINET for analysis.

1.1 Restricted Problem Statement

The digital objects of interest can include things such as epidemiology research papers, wiki pages, websites, videos, and many other digital contents. The deliverable is a highly connected network that relates all these digital objects.

In our class project, we make a few refinements to the problem, which allows us to have a quick prototype for demonstration. To be noted that, the refinements we made does not limit the generalization of our model to other application and use cases. We describe the refined problem statements here:

1. Build a automated process for web exploring, content analysis and RDF network construction.
2. The digital objects to be investigated are research papers, technical reports so on.
3. For crawling the digital objects (papers), the focus is on DBLP bibliography website.

1.2 Digital Object and Meta Data

The digital objects we focus on are research papers. For each research paper, the meta data we extract are author, co-author, title, year, publisher, keywords. We

do not look at the content of the paper itself, but our approach can be extended to include that part. Suppose we have a collection of papers, an example of the network we are trying to construct is in the figure below:

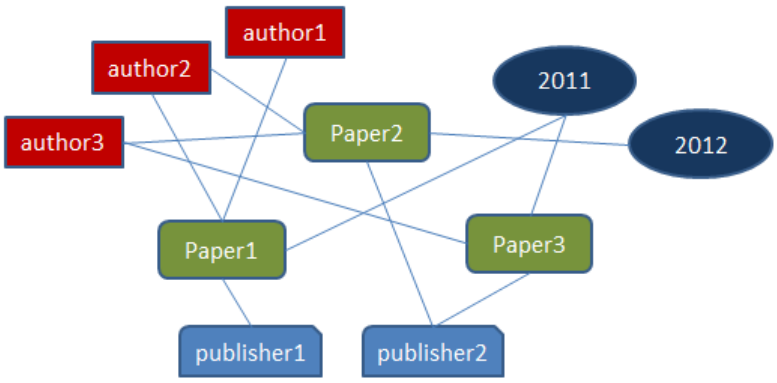

Figure 1: Network of a digital object

We crawl and obtain the information about research paper digital objects by making queries to the DBLP website. A sample query looks like "venue:hpdc: year:2013: type:conference: author:madhav_v_marathe:", which would look for Dr. Madhav Marathe's papers that are published at HPDC in year 2013. Our implementation makes this request automatically and process the returned information for network modeling. Part of the returned meta information for this particular query string is shown here:


```
"@id":"1523868",
"info":{"authors":{"author":["Tariq Kamal","Keith R. Bisset","Ali
Raza Butt","Youngyun Chungbaek","Madhav V. Marathe"]},"title":
{"@ee":"http://doi.acm.org/10.1145/2462902.2462929","text":"Load
balancing in large-scale epidemiological simulations. "},"venue":
{"@url":"db/conf/hpdc/hpdc2013.html#KamalBBCM13","@conference":"HP
DC","@pages":"123-124","text":"HPDC 2013:123-
124"},"year":"2013","type":"inproceedings"},
"url":"http://www.dblp.org/rec/bibtex/conf/hpdc/KamalBBCM13"
}
```


Figure 2: A DBLP entry

1.3 The need for a new method

Simply using a search engine such as Google, or a web crawler in an undirected way won't be able to satisfy the requirement of this problem. First of all, the results from a search engine or crawler are usually un-organized, a lot of post-processing is still needed to extract the meaningful information. Second, the search engine or crawler finds data at a higher granularity as web pages,

while our service needs to function within a specific domain, here epidemiology. Thirdly, the results obtained by a search engine or crawler leads to ambiguity, which could be difficult to explain. More importantly, the results our service want to present to the users are networks of inter-connected digital objects, which are organized based on their topics. Below is an example to explain the difference between what we get from a search engine and we our service needs to achieve.

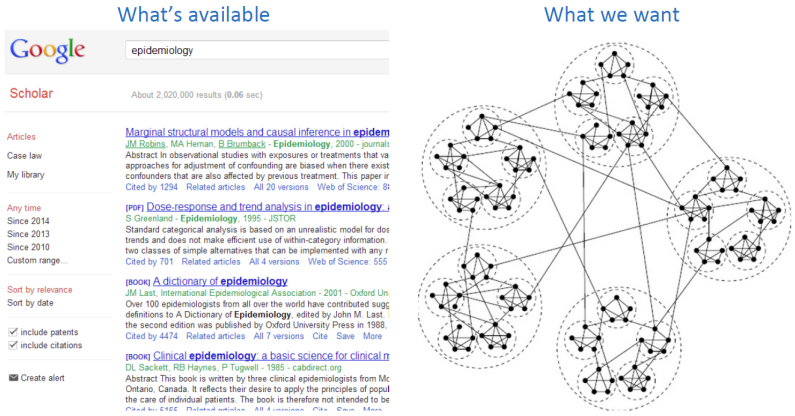

Figure 3: Structure of search engine result and what we want

2 Method

The task of building a network of digital objects given an initial subject is exploratory in nature meaning out of all the visited objects only a fraction of them will eventually be deemed relevant and out of those relevance varies by its relationship to the initial subject. To help us devise an automatic approach, we consider how a human might perform this task given that we seem to do it with great ease.

2.1 How might a user perform this task?

Say that a researcher decides to look into the subject of "disease propagation" and decides that the best place to start is by searching online on the subject and retrieve relevant research papers; his process might proceed as follows.

1. Start out with initial concept network $N = \{$"disease propogation"$\}$ and list of digital objects $O = \{\}$
2. Search using N and obtain results R and augment O with R
3. Study R and partition it into two groups: R_1 containing relevant entires and R_2 containing currently irrelevant ones

4. Pick out important concepts and connections from R_1 and augment N
5. Revise N by reassessing all items in O to see if certain concepts and connections are no longer important
6. Goto step 2

We can discern two important properties that make this procedure effective.

2.1.1 Relevance

While the user determines the relevance of a result with respect to the network N constructed so far and with respect to all the retrieved objects. A result's relevance to the network N ensures that a global concept space is being explored while its relevance to all retrieved objects ensures that this result's relevance to N is not an isolated one: it's more assuring to improve your concept network N using a group of related research papers – jointly exploring a topic – than by a lone entry.

This also means that a user may find a group of related papers that are not currently relevant to N and, by extension, a group of related papers weakly relevant to N – maybe just a couple of papers in this group have a link with the core network. So, forming groups allows us to interpret objects not directly related to the core network by virtue of their association with those in their group who are.

2.1.2 Connectivity

The second important property concerns the connectivity of N. Every node in the network N has a well-defined relationship to the starting concept in the sense that it is well supported by digital objects. More importantly, not every node and connection seen in the query results are present in N; N is grown by stringent constraints; and, if necessary, procedure allows for N to be destructively updated.

2.2 Modeling

We decided to build a model that emulates the aforementioned human behavior.

2.2.1 A Digital Object

Since our final objective is a RDF network we decided to view a digital object O as a set of edges $\{(x_i, y_i)\}$ between nodes determined by predefined predicates. For example, if the digital object in question is a research paper, then some example predicates might be "isCoAuthorOf", "isPublishedIn" and "hasKeyword".

All our observed data now becomes a set set of digital objects $\{O_i\}$.

2.2.2 Topics

A basic distribution capable of generating a digital object is given by a simple edge distribution assuming independence of edges. Specifically, given some distribution θ of x in edge (x, y) and a distribution ϕ_x of y in (x, y) a digital object O is measured as

$$P(O = \{(x_i, y_i)\}|\theta, \phi_x) = \prod_{(x,y)\in O} P(x|\theta)P(y|\phi_x)$$

As we described earlier, digital objects tend to occur in related groups or topics and so we wish to index these digital object distribution by a latent topic variable. Hence, we have a family of distributions $\{\theta_{t\in T}\}$ and $\{\phi_{t\in T, x\in V}\}$ for node set V.

2.2.3 Donating Edges to the Core Network

In order to construct an RDF network out of these digital objects we need to borrow edges from digital objects from various topics to construct this network. Hence, every digital object O_j will have a local Bernoulli distribution κ_j that determines if an edge O_{ji} belongs to the core network indicated by another latent variable $C_{ji} \in \{0, 1\} \sim \kappa_j$.

2.2.4 The Core Network

Finally, we come to the core network itself. What structure should this network posses? It cannot simply be a distribution that assumes independence of edges as the topics do because we showed that the network constructed by a user will tend to be tightly connected.

To this end, we describe an edge $O_{ji} = (x, y)$ in the core network by the distribution of x determined by its distance from the source $D_{ji} = d$ and the distribution of nodes at this distance π_d. We then generate y through ϕ_{0x} to give the following probability for the edge (x, y) in the core network

$$P(O_{ji} = (x, y)|D_{ji} = d, \phi, \pi) = P(x|\pi_d)P(y|\phi_x)$$

The entire generative process describing all digital objects is listed in Algorithm 1.

Algorithm 1 Generating the object networks

Require: Source S, M networks, N_j edges, nodes V
Ensure: M networks
  $\theta_{\{1,\ldots\}} \leftarrow \text{Dirichlet}(\alpha)$
  $\phi_{\{0,1,\ldots\},v\in V} \leftarrow \text{Dirichlet}(\alpha)$
  $\pi_d \leftarrow \text{Dirichlet}(\alpha)$
  $\delta \leftarrow \text{Beta}(\beta,\beta)$
  $\kappa_{\{1,\ldots\}} \leftarrow \text{Beta}(\beta,\beta)$
  for $j = 1$ **to** M **do**
    $t := T_j \leftarrow \text{CRP}(\alpha)$
    for $i = 1$ **to** N_j **do**
      $C_{ji} \leftarrow \kappa_j$
      if $C_{ji} = 1$ **then**
        $d := D_{ji} \leftarrow \text{Geometric}(\delta)$
        $x := E_{ji0} \leftarrow \pi_d$
        $E_{ji1} \leftarrow \phi_{0x}$
      else
        $x := E_{ji0} \leftarrow \theta_t$
        $E_{ji1} \leftarrow \phi_{tx}$
      end if
    end for
  end for

2.2.5 Inference

We perform inference through Gibbs Sampling. The two hidden variables to learn are the topics T_j and the containment of an edge ji within the core network C_{ji}. We look at $C_{ji} = 0$ first and it is given by

$$P(C_{ji} = 0|T_j = t, E_{ji} = (x,y), \ldots) \propto \frac{n_t^0 + \beta}{n_t^0 + n_t^1 + 2\beta} \times \frac{n_{tx}^0 + \alpha}{\sum_i n_{ti}^0 + \alpha} \times \frac{n_{txy}^0 + \alpha}{\sum_i n_{txi}^0 + \alpha}$$

where n_t^0 is the number of edges in topic t (not in core), n_{tx}^0 is the number of edges in topic t starting with x, and n_{txy}^0 is the number of (x,y) edges in topic t. Now, $C_{ji} = 1$ is given by

$$P(C_{ji} = 1|T_j = t, E_{ji} = (x,y), D(x) = d \ldots) \propto \frac{n_t^1 + \beta}{n_t^0 + n_t^1 + 2\beta} \times \frac{n_x^{1,d} + \alpha}{\sum_i n_i^{1,d} + \alpha} \times \frac{n_{xy}^1 + \alpha}{\sum_i n_{xi}^1 + \alpha}$$

where n_{xy}^1 is the number of (x,y) edges in core, $n_x^{1,d}$ is the frequency of x in core at distance d from the source.

Sampling topic $T_j = t$ is given by collectively considering all the edges in a document. This requires computing the posterior probability of membership $C_{j(.)}$ and edges in the topic.

$$P(T_j = t|E_j = \{(x_i,y_i)\}, C_{ji}, \ldots) \propto P(C_{j(.)}^0|\kappa_t,\beta) \times P(E_{j(.)}^0|\theta_t,\alpha) \times P(E_{j(.)}^0|\phi_t,\alpha)$$

2.2.6 Crawling for more data

The connected and smooth growth of the core network gives us a *frontier* for exploring nodes in the core network in analogy to the frontier of URLs waiting to be explored by a web crawler. The difference is that this frontier contains nodes with probabilities rather than links to webpages. One way to crawl using this frontier is to treat small sets of nodes generated according to the core network distribution as queries to a search engine whose resulting digital objects augment our existing data. After a few queries, we update the model and repeat the process. Searching by queries offers a very different way to gather data because relevant digital objects can be downloaded without them having to be hyperlinked from existing objects.

In contrast to a focused crawler, not all downloaded results become important and not all that was included have to remain so because the core network adapts to the best network given the current data which may mean removing previously included edges and adding new ones.

As a result, it would make an interesting exercise to compare the results obtained by the two methods and determining the precision-recall of the objects linked to the core network against the precision-recall of the objects retrieved by a focused crawler.

3 Experiments

To show what a core network discovers given different seeds we'll infer this model over synthetic data. For purposes of interpreting the results, the networks we generate for the digital objects will be taken from a global network structured as a directed acyclic graph with a single source so that we can compare the core networks discovered by the model for various starting nodes against this global network.

Additionally, we fix the topics of the digital objects and only infer the membership variable C_{ji} for each edge in the digital objects so that we see what the core network is made up of.

Data set 1 (Figure 4) shows only the edges existing in the generated digital objects that are put into two topics (red and blue). It is setup so that the digital objects all share edges $1 \mapsto 2$ and $2 \mapsto 3$ and then diverges. The core network inferred – when the source was set to 1 – consisted of the edges $1 \mapsto 2$ and $2 \mapsto 3$, which means the RDF reaches all the digital objects in this data. And, this was expected because every object is strongly connected to 1, 2, and 3. When we set the source to 2, the core network inferred consisted of no edges $2 \mapsto 3$ only. We note this as a correctable issue in the "Lessons" section.

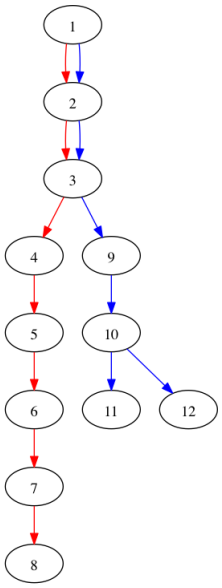

Figure 4: Example Data. Topics are in blue and red.

Data set 2 (Figure 5) adds more edges within topics and this time the documents in the blue topic no longer contain the edge $1 \mapsto 2$. Yet, when the core network was inferred with source 1, it contained the edge $2 \mapsto 3$. This illustrates the situation where results can contain digital objects that do not directly contain any of the initial query terms.

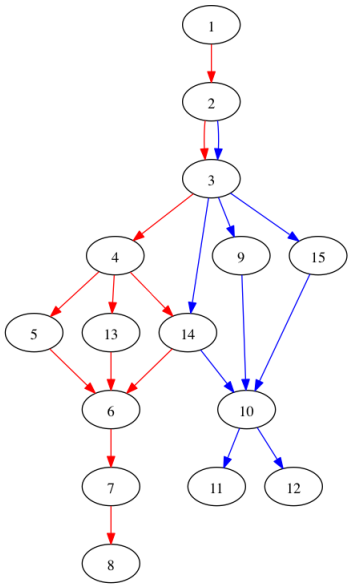

Figure 5: Example Data. Topics are in blue and red.

3.0.7 Lessons

The first observation is that we need to consider edges with regards to only its distance and not the orientation of the edges. As we saw in the above example starting with node 2 could not see node 1 in the core network due to the edge being oriented as $1 \mapsto 2$.

Secondly, the distribution over the observed distances is unnecessary because it restricts the growth of the core network too much. And, since distances are observed in the data given that the source is fixed, having a distribution of distances does not make sense. Instead, we simply use the distances to index the node distributions π_d. As a result, we should see more edges in the core network and the network stops growing when the edges become too dense at a certain distance due to too many contributions from various topics.

3.0.8 Digital objects for testing

Given some global network with transition probabilities E_{xy} we construct a topic t by drawing a sparse node distribution $\theta_t \sim \text{Dirichlet}(\alpha)$ and generating digital object networks j with N_j edges by drawing the first node $x_1 \sim \theta_t$ and then the edge (x_1, y_1) followed by iteratively picking the next source node from already present nodes according to θ_t and then its edge. This process ensures that we generate connected networks for each digital object. See Algorithm 2.

Algorithm 2 Digital objects from various topics from a global network

Require: Transition probabilities E_{xy}, number of topics T, topic distribution τ, number of networks M, network sizes N_j
Ensure: M networks of sizes N_j
  $\theta_{t \in T} \leftarrow \text{Dirichlet}(\alpha)$
  for $j = 1$ **to** M **do**
    seen $\leftarrow \emptyset$
    for $i = 1$ **to** N_j **do**
      $x_i \leftarrow \theta |$ seen
      $y_i \leftarrow E_x$
    end for
  end for

4 Evaluation

Once the model is tested with large data, we propose two methods of real-world evaluation to understand the *relevance* of papers linked to the core network and the *completeness* of the core network with respect to the source subject.

4.1 Relevance

Search engines rank results in decreasing order of relevance. Often, these correlate very well with what we are trying to find especially when the query terms fully capture what we are looking for. Other times, it fails badly because the query terms don't offer the best lookup key so one has to resort to a series of queries by varying search terms based on the results of previous queries.

These two situations should behave differently within the core network because it can associate relevant objects that don't share the initial query terms. Hence, we propose comparing the relevance of papers as computed by a search engine and by the core network. In particular, we want to look at situations where highly relevant results in a search engine are ranked low in the core network and vice-versa. Hopefully, we can conclude by showing that the core network alleviates the problem with lack of connectivity between intended result and initial query.

We can also define a more general test asking a user to explore a unknown research topic using multiple search engine queries and using the results produced by a single construction of the core network.

4.2 Completeness

Completeness is intended to measure the coverage of the RDF network given enough data on a subject. So, if we start with an author's name as the subject then given enough data sources we want to measure if the constructed RDF network captures all the salient object relationships with respect to this subject. Specifically, we want to measure the precision (how many of the edges presented

by the network is relevant to the subject) and recall (how many of the edges that are relevant to the subject are not present) of the network. Constructing the relevant edges is a challenge but could be delegated as a task to random authors whose networks we then model and compare for completeness.

5 Related Work

Web crawling: a web crawler, which is also know as a robot or spider, is a system that download web pages[7]. It is used for a variety of purposes. For example, web crawler is one of the main components for search engines. It is also used for web archiving such as the Internet archive. Also data mining can be performed on crawled web pages for all kinds of purposes.

Focused crawling: In order for a crawler to be more selective, focused crawling is designed to fetch contents that are more relevant to a particular topic someone is interested, using the hyper-link structure on a web page[1, 2, 3, 4]. The crawler starts with a list of seed, and continues to estimate the likelihood that each subsequent candidate link will bring in more further relevant contents. Usually, the crawler prioritizes the frontier as well as the hyper-link exploration process based on certain user-defined criteria or algorithms. This can be as easy as limiting the crawling to certain domain, or picking those with higher page ranks. A more important and interesting property of web pages to satisfy is finding those related to a specific topic, such as "epidemiology". This is also called a topical crawler.

To determine the relevance of an unvisited web page before fetching, different algorithms and techniques have been developed to help automate this process. Things being looked at include the anchor text of links[8], the context graph, and the text content. Classifiers are involved to guide the crawling, and more sophisticated reinforcement learning can help achieve better performance[9, 5]. The approaches can be coarsely classified into two groups, one with external help, and one without[6].

All these related work can help in crawling web pages with relevant contents. Our proposed model not only does web crawling, but also focuses on analyzing and organizing the crawled contents. By bringing in and modeling the relations among crawled digital objects, we aim to provide the user with a collection or network of digital contents that are naturally put into topics, which gives the user a more organized view of the crawled data.

6 Acknowledgement

We acknowledge support related to NSF NSF OCI-1032677, SDCI NMI New: From Desktops to Clouds - A Middleware for Next Generation Network Science.

References

[1] Paul De Bra, Geert jan Houben, Yoram Kornatzky, and Reinier Post. Information retrieval in distributed hypertexts. In *In RIAO*, pages 481–491, 1994.

[2] Soumen Chakrabarti, Martin van den Berg, and Byron Dom. Focused crawling: A new approach to topic-specific web resource discovery. In *Proceedings of the Eighth International Conference on World Wide Web*, WWW '99, pages 1623–1640, New York, NY, USA, 1999. Elsevier North-Holland, Inc.

[3] Junghoo Cho, Hector Garcia-Molina, and Lawrence Page. Efficient crawling through url ordering. In *Proceedings of the Seventh International Conference on World Wide Web 7*, WWW7, pages 161–172, Amsterdam, The Netherlands, The Netherlands, 1998. Elsevier Science Publishers B. V.

[4] Michael Hersovici, Michal Jacovi, Yoelle S. Maarek, Dan Pelleg, Menanchem Shtalhaim, and Sigalit Ur. The shark-search algorithm. an application: Tailored web site mapping. In *Proceedings of the Seventh International Conference on World Wide Web 7*, WWW7, pages 317–326, Amsterdam, The Netherlands, The Netherlands, 1998. Elsevier Science Publishers B. V.

[5] Andrew McCallum, Kamal Nigam, Jason Rennie, and Kristie Seymore. A machine learning approach to building domain-specific search engines. In *Proceedings of the 16th International Joint Conference on Artificial Intelligence - Volume 2*, IJCAI'99, pages 662–667, San Francisco, CA, USA, 1999. Morgan Kaufmann Publishers Inc.

[6] Blaz Novak. A survey of focused web crawling algorithms. 2004.

[7] Christopher Olston and Marc Najork. Web crawling. *Found. Trends Inf. Retr.*, 4(3):175–246, March 2010.

[8] Brian Pinkerton. Finding what people want: Experiences with the WebCrawler. In *Proceedings of the first World Wide Web Conference*, Geneva, Switzerland, May 1994.

[9] Jason Rennie and Andrew McCallum. Using reinforcement learning to spider the web efficiently. In *Proceedings of the Sixteenth International Conference on Machine Learning*, ICML '99, pages 335–343, San Francisco, CA, USA, 1999. Morgan Kaufmann Publishers Inc.