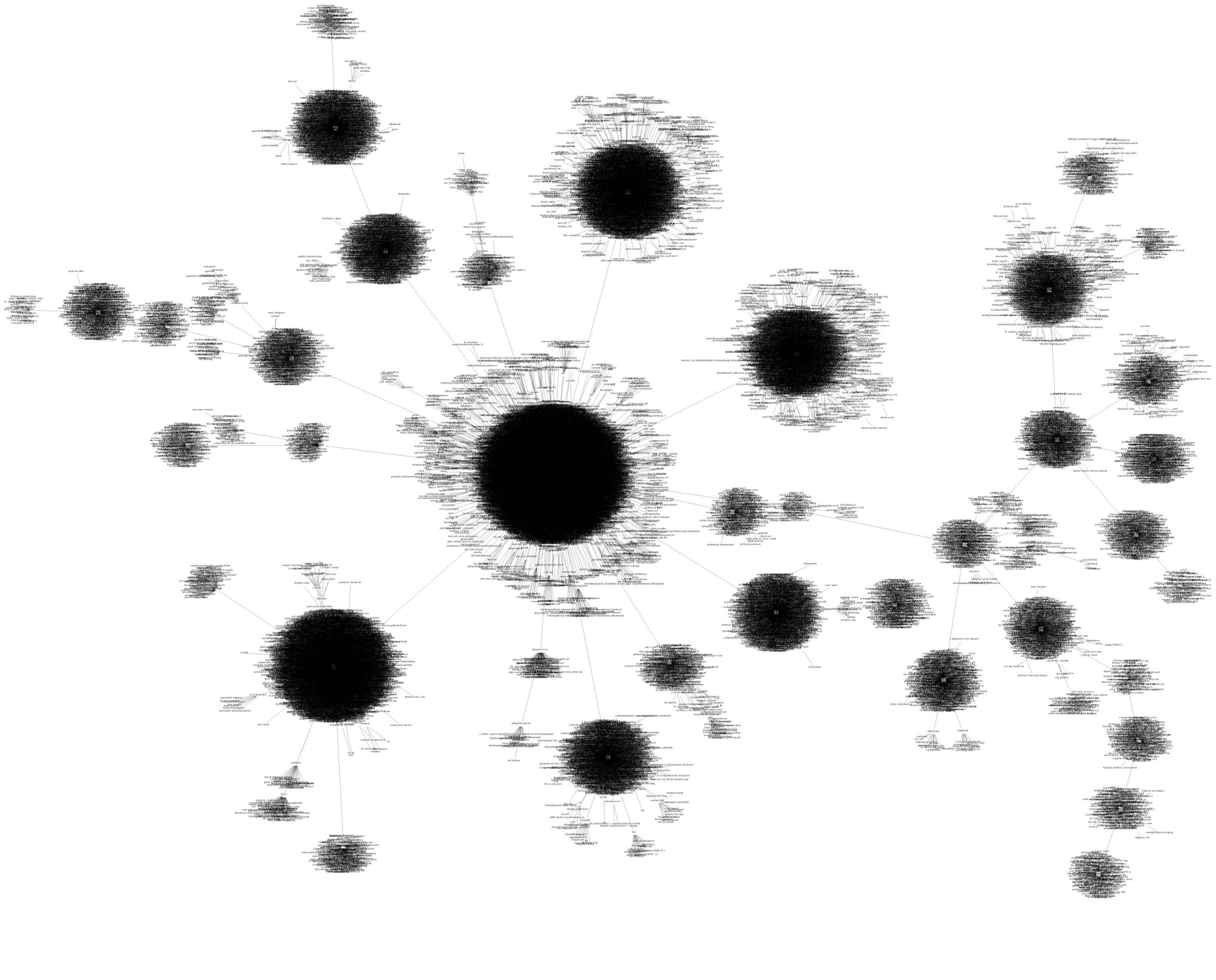
Entity Graphs in Biology

A construction and exploration of methods for useful large-scale data mining in massive document corpuses leveraging MapReduce and network analysis.



Ryan Noon • rmnoon@cs.stanford.edu • Stanford University, December 2010

Entity Graphs in Biology

A construction and exploration of methods for useful large-scale data mining in massive document corpuses leveraging MapReduce and network analysis.

In recent years, advances in computational power and programming methodology have made it possible to glean actionable insight from gigantic sources of information. Simultaneously, the rise of the World Wide Web and more recent efforts to provide open access to humanity's accumulated knowledge have made such datasets easily available to the public for the first time. New methods for understanding these datasets and extracting the wealth of knowledge they contain are sorely needed.

In Part I of this paper we describe an intuitive graph construction and associated parallel algorithms for deriving detailed network structure from a massive annotated document corpus. Part I also contains our experiences with optimizations capable of making these networks more useful in practice. In Part II we use such a structure to answer useful questions about the PubMed Abstract corpus, which contains more than 11 million research abstracts. Finally we present techniques for future study and a conclusion.

Part I: Constructing an Entity Graph

Entity Graphs

In this paper, the term "Entity Graph" refers to a specific formulation of a weighted, undirected graph based on intersecting subsets. In the graph all nodes represent subsets and all edges represent the intersection sets between the nodes. For example: if node A represents the numbers {1, 2} and node B represents the numbers {2, 3}, then an edge E of weight 1 exists between them representing their non-empty intersection ($\{2\}$).[1] In this way, we can build connectivity networks in data domains where such network connectivity is not immediately obvious. Such a domain can be found in large bodies of text. In this paper, we use the following terminology:

- **Corpus:** a collection of human knowledge on a specific topic, typically in text form. In this paper, our corpus is the entire publicly available set of biological research paper abstracts on PubMed. This corpus comprises approximately 11 million abstracts.

- **Document:** a specific part of a corpus, typically pertaining to a specific topic with distinct authorship. Here, a document is the standard XML representation of a PubMed research paper abstract (not the full text of the paper itself).

- **Entity:** a specific term or concept that can be extracted from a document. For example, the entities present in a painting could be the colors used by the artist. Here, our entities are terms in each abstract's "ChemicalList" and "MeshHeading" section.

Given these definitions, two subset-based network constructions from the corpus become apparent:

- **Document-centric:** Each node is the subset of entities present in each document. Thus, if the corpus contains n documents there will be exactly n nodes. The edges represent common entities that are shared between documents. If two documents share k entities, then an edge of weight k exists between their nodes.

1 Higher order subset constructions (simplicial complexes) are possible but are not discussed in depth here. For example, if three nodes have a non-empty intersection we may draw a triangle between all three of them.

- **Entity-centric:** This construction is the inverse of the previous one. Each node represents an entity. If all of the documents in the corpus contain in total m entities then there will be precisely m nodes in the graph. An edge between node i and node j of weight k represents the k documents that contain both entities i and j. For the purposes of this paper, we define this construction as an "Entity Graph".

In this paper we focus primarily on entity-centric graphs, but document-centric graphs are likely just as interesting, albeit for different analytical purposes. All of the same algorithms and techniques are still applicable.

The Importance of Scalability

A crucial benefit of the construction used in this paper is that the algorithms involved naturally scale to meet the processing requirements of gigantic corpuses. To prove this point, all processing done in constructing the PubMed entity graphs was implemented using the MapReduce framework[2] originally developed by Google. More specifically, all computation was done using Dumbo, a Python-language framework for the Hadoop open-source MapReduce implementation.

Within this model, the programmer specifies multiple steps of "mapping" the input into *(key, value)* tuples and "reducing" groups of these tuples in the form *(key, [all values for the key])* into output tuples. The benefit of this approach is that both the mapping and the reducing operations can be done in parallel by hundreds or thousands of machines, allowing programs to process many terabytes of input simultaneously. For many years, this programming model powered the core search functionality of Google itself. Certain operations required for constructing Entity Graphs (such as edge-pairing or inverting a very large index) would be very difficult for a traditional sequential program, but can be

2 DEAN, J., AND GHEMAWAT, S. MapReduce: Simplified data processing on large clusters. In Proc. of the 6th OSDI (Dec. 2004), pp. 137-150.

accomplished in minutes on even a modest MapReduce cluster.[3] Large clusters can be rented remotely for very low cost using a cloud-based service such as Amazon EC2, making gargantuan data processing tasks nearly effortless for even the smallest of budgets.

Constructing an Entity Graph

The steps for constructing the graphs used in this paper are detailed below:

1. Parse the XML dump using a declarative parsing style to save on memory[4]. For each document, emit a line with the document's PubMed ID (PMID), title, and a list of the entities found in the ChemicalList and MeshHeading sections. The collection of these records provides a one-to-many mapping between PMID and entity list and is hereafter referred to as a "document list".

2. If entity post-processing (as detailed in the next section) is desired, use MapReduce to invert the document list mapping into an "entity list" mapping of entity to documents. Inverting the document list using MapReduce is simple: for each document and each entity within it, emit a tuple of the form *(entity, PMID)* and reduce the resulting collection of *(entity, [PMID list])* into a single record in the new entity list. After any applicable entity post-processing, invert the entity list back into a document list. For efficiency reasons, it may be advantageous to convert

3 In this project, the "cluster" consisted of a single quad-core workstation. The longest operation (constructing the "full_term" graph) took approximately 1 day.

[4] A handy technique is using one integer/string that denotes the current document being parsed, one list that holds all of the entities we have found in the current document, and handful of boolean values that denote which tags the current line is inside of. That way we can sanity-check the XML and write the current document's information to an output file when we get to the next one.

each entity into a numerical hash representation. This allows uniform entity size and may improve MapReduce throughput.

3. With the final document list in hand, use another MapReduce operation to construct the edges of the entity graph. The "map" function emits tuples of the form *((e_i, e_j), p)* for each 2-combination of entities that belong to the document with PMID *p*. The "reduce" function thus takes as input the tuple *((e_i, e_j), [list of PMIDs])* and can emit as output an edge from node *i* to node *j* consisting of the given list of PMIDs with weight *w* equal to the size of the list. This list of edges represents the entire Entity Graph and can now be filtered or loaded into any network analysis platform.[5]

An Optimization: Efficient Entity Clustering Using Simple Natural Language Processing

Depending on the robustness of the entity-extraction process, it can be useful to perform an additional processing step to merge similar entities into a single entity. This helps reduce the size and density of the graph and can potentially increase its usefulness.

Some examples where such a process could be useful are:

1. **Variants of the same entity across common laboratory species:** the entities "collagen, rat" and "collagen, human" should be collapsed into the single entity, "collagen".

2. **Trivially different parts of speech:** the entities "cancerous growth" and "cancer growing" should be one entity.

3. **Organic compounds with similar IUPAC nomenclature:** Many research-worthy compounds/drugs have slight differences in configuration

[5] The network analysis toolkit used in this paper was NetworkX, a very capable Python library produced by Los Alamos National Laboratory and available at http://networkx.lanl.gov

but could conceivably be grouped together. For example, the two isomers of propanol could be stated in the literature as *propan-1-ol* and *propan-2-ol*, and (depending on our analytical intent) can be merged. This is a simplified approach to a difficult problem. The computational knowledge representation required to merge organic compounds at a given granularity is beyond the scope of this paper.

A process that accomplishes these three goals was performed on the PubMed entities and was able to reduce the size of the set by approximately 30 percent. The exact process used is as follows:

1. Specify a peer-reviewed stemming function from NLP literature. This function takes strings, such as "fishing" and "fished" and returns the common root "fish". The stemming function used here is the Lancaster stemming algorithm.[6]

2. Build a biologically-relevant list of "stopwords" which contain very little entity-specific information. The one used here was built with the top 1000 most frequent words from the entity list (e.g. "enzyme", "protein"), combined with distinct common chunks from IUPAC nomenclature ("methyl", "III"), and a list of the 20 most common laboratory species' common and scientific names ("rat", "rattus norvegicus"). After compiling the list, process it by stemming each word.

3. In the entity, replace all non-alphabetic characters with spaces. Split the resulting string by whitespace into tokens. Stem each token and check to see if it is on the stopwords list. If it is, remove it from the entity. Rejoin the tokens by a single space.

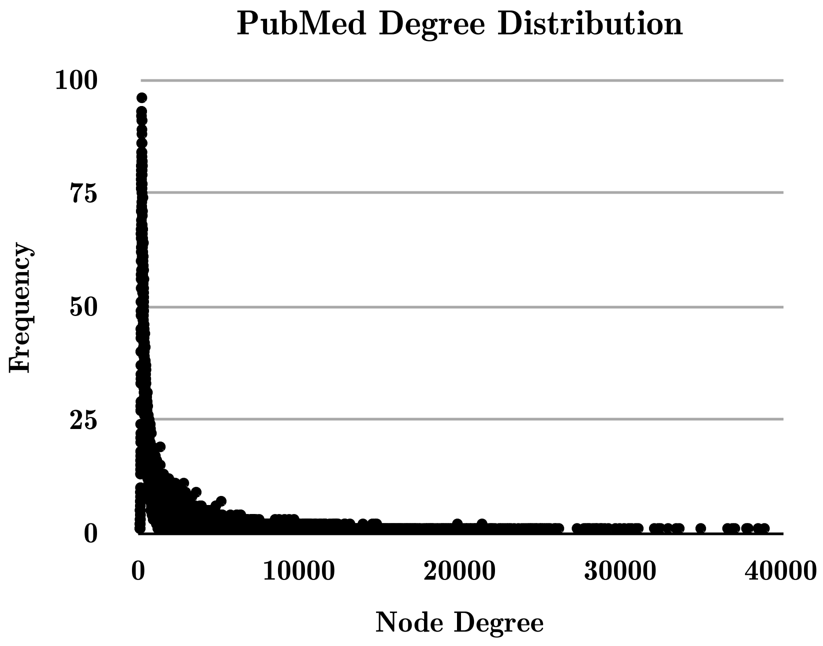
[6] http://www.comp.lancs.ac.uk/computing/research/stemming

Part II: Analytical Methods

Given the construction in Part I, we can now begin to ask new kinds of questions about our data. In this section, we will outline a few of these questions and provide sample techniques that attempt to answer them for the PubMed abstract corpus.

An Overview of the PubMed Entity Graph

- Number of nodes: 40,180
- Number of edges: 33,464,657
- Average node degree: 1665.74
- Average edge weight: 76.0
- Clustering coefficient: 0.8298[7]



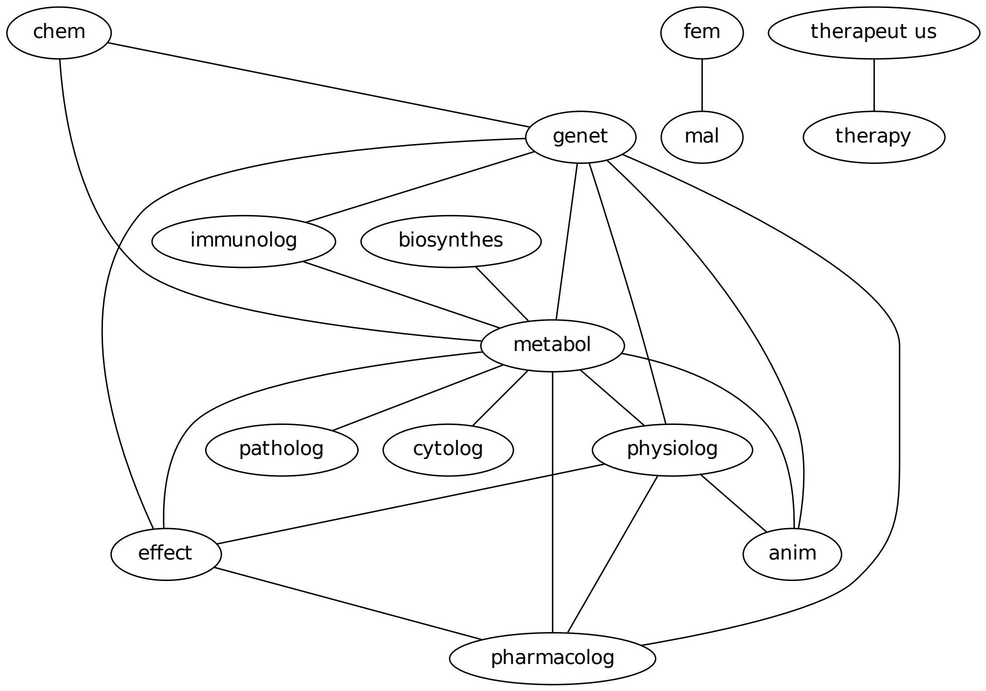
[7] This clustering coefficient was computed on a filtered version of the full graph. The filtering removed all edges of weight less than 100.

The raw Entity Graph is extremely dense, with both the degree distribution and the weight distribution showing a clear power-law distribution.

What are the most important entities in the corpus?

Network analysis provides us several tools for determining the "most important" nodes in a given topology.

- **k-Cores:** A k-core of a graph G is defined as "a maximal connected subgraph of G in which all vertices have degree at least k.'[8] In the PubMed Entity Graph, the maximal value of k is 424, the graph of which has 736 nodes.



The 424-core of the PubMed Entity Graph with the weakest 99.99% of edges removed along with any singleton nodes.

[8] http://en.wikipedia.org/wiki/Degeneracy_(graph_theory)#k-Cores

- **PageRank:** PageRank is a link-analysis algorithm that formed the original basis for Google's search engine. It uses linear algebra to weight the relative importance of nodes in a graph based on the way they link to each other.

Entity Stem	PageRank
metabol	0.11218
genet	0.06375
physiolog	0.03534
pharmacolog	0.03362
chem	0.03022
effect	0.02503
immunolog	0.02062
anim	0.01826
mal	0.01600
fem	0.01582

The top 10 nodes in the PageRank computed on the top 1% of the PubMed graph.

- **Betweenness Centrality:** Betweenness Centrality is the fraction of all of the shortest paths between all pairs of nodes that pass through a given node. For the subgraph of PubMed composed of nodes involving common cellular components, we see the following BC scores:

Entity Stem	Centrality
mitochondr	0.046
ribosom	0.043

Entity Stem	Centrality
cytoplasm nuclear	0.025
nucle	0.023
suprachiasm nucle	0.018
paraventricul hypothalam nucle	0.014
rna ribosom	0.013
ribosom kda	0.010
lysosom	0.006
ribosom inact	0.005

The top 10 nodes of the entity subgraph composed of cell-components ranked by their betweenness-centrality. This shows that ribosomes and mitochondria are the most important components in PubMed.

Given a set of entities, which papers have the most information about them?

We can solve this problem with the following algorithm:

1. Let the set E be the set of entities we are interested in.
2. For all unique pairs of entities in E, find the shortest path between the nodes in the pair. Take the union of all the nodes in each path as E_{cover}. This is the minimal set of nodes required to have a single connected component containing all of the entities in E.
3. Take the subgraph G' of entity graph G that contains only nodes (and edges to nodes) in E_{cover}.

4. Construct a mapping M from PMID to score. For each paper in each edge of G', add a score value S to that paper's score in the mapping. The papers with the highest scores are most connective of G', and thus have the most information about the entity set E.

The score value S can (simply) be 1, making the score mapping a histogram. Alternatively, the score value can be multiplied by any beneficial attributes of the edge this paper was found in. For example, we can reward a paper that links entities in E (our original query set) while penalizing a paper linking nodes in $E_{cover} - E$ (the entities we added to connect our query set)[9]. In addition, we can penalize or reward papers that link extremely popular (high degree) entities if we seek, respectively, to stray from or adhere to the "beaten path" in our search.

For the PubMed corpus, the test query was a list of 32 genetic diseases known to be linked to the X chromosome. The top 20 results with score are:

Enhanced beta2-adrenergic receptor (beta2AR) signaling by adeno-associated viral (AAV)-mediated gene transfer.	155.0
Skin denervation and cutaneous vasculitis in eosinophilia-associated neuropathy.	126.0
FastEpistasis: a high performance computing solution for quantitative trait epistasis.	126.0
Effect of inoculum size on detection of Candida growth by the BACTEC 9240 automated blood culture system using aerobic and anaerobic media.	126.0
Enlazin, a natural fusion of two classes of canonical cytoskeletal proteins, contributes to cytokinesis dynamics.	126.0
Root nodule Bradyrhizobium spp. harbor tfdAalpha and cadA, homologous with genes encoding 2,4-dichlorophenoxyacetic acid-degrading proteins.	126.0
HomozygosityMapper--an interactive approach to homozygosity mapping.	115.0
Diagnosis of X-linked lymphoproliferative disease by analysis of SLAM-associated protein expression.	110.0
Charcot-Marie-Tooth disease type 1B: marked phenotypic variation of the Ser78Leu mutation in five Italian families.	108.0

[9] For our explorations of the PubMed data, we used a 10x score multiplier for each side of the edge being in the original query set. No degree-weighting techniques were employed.

Shuffling of genes within low-copy repeats on 22q11 (LCR22) by Alu-mediated recombination events during evolution.	108.0
Lost in translation: translational interference from a recurrent mutation in exon 1 of MECP2.	108.0
Disruption of Mtmr2 produces CMT4B1-like neuropathy with myelin outfolding and impaired spermatogenesis.	108.0
Temporal and regional differences in the olfactory proteome as a consequence of MeCP2 deficiency.	108.0
Clinical, electrophysiological and molecular genetic characteristics of 93 patients with X-linked Charcot-Marie-Tooth disease.	108.0
Influence of recipient and donor IL-1alpha, IL-4, and TNFalpha genotypes on the incidence of acute renal allograft rejection.	108.0
Elasticity and adhesion force mapping reveals real-time clustering of growth factor receptors and associated changes in local cellular rheological properties.	108.0
hMSH2 is the most commonly mutated MMR gene in a cohort of Greek HNPCC patients.	108.0
Analysis of clinical and molecular characteristics of Wiskott-Aldrich syndrome in 24 patients from 23 unrelated Chinese families.	108.0
Accurate representation of the hepatitis C virus quasispecies in 5.2-kilobase amplicons.	108.0
Delay eyeblink classical conditioning is impaired in Fragile X syndrome.	96.0

The top twenty paper results from running the algorithm on a query of 32 X-linked diseases. The results show that the Entity Graph construction is fully capable of identifying the biologically relevant properties of a query.

Even a basic search (with very rudimentary entity resolution) and little-to-no parameter tuning produces results that find the implicit common feature in the input (X-linked diseases).

Given a concept, which concepts are most similar?

There are many ways to go about relating nodes to other nodes. The simplest approach, which we will explore here, involves finding the node with the most similar set of direct connections. The similarity metric used here is the Jaccard similarity[10] between our query node's neighbor set and the neighbor set of every other node.

The results from running such a query (for the entity "laminin") on the PubMed Entity Graph are below:

proteoglyc	0.5989
fibronectin	0.5794
cadherin	0.5356
cult med condit	0.5350
lymphokin	0.5043
platelet der	0.5005
tissu metalloproteinas	0.4988
extracellul matrix	0.4980
integrin	0.4918
cult med ser fre	0.4900
metalloendopeptidas	0.4869
catenin	0.4841
fluoresc techn indirect	0.4753
urokinas plasminog act	0.4728
keratinocyt	0.4704
cytoskeleton	0.4612
keratin	0.4598
cocult techn	0.4588
lin transform	0.4558
immunoenzym techn	0.4546

The results of a node similarity query for the entity "laminin".
The results are very promising, with several of the top results being other glycoproteins or related structures.

[10] The Jaccard similarity of two sets is defined to be the the size of their intersection divided by the size of their union.

Future Work

Algorithms for the following problems have been contemplated but not yet verified on the PubMed Entity Graph. Brief descriptions are provided here:

Given two entities, what are the best entities and papers that connect them?

This problem is somewhat isomorphic to the problem of finding the best papers for an entity subgraph mentioned previously. Here, the subgraph in question is the shortest path on the Entity Graph between two entities of interest. We then perform a similar best-fit computation on the papers in the edges of this path with similar heuristics as before.

Given a set of entities, what is the minimal set of papers that can give me information about all of them?

We can also efficiently compute the minimum spanning tree of all or part of the Entity Graph. In this case, it is useful to set the weight of all edges to their reciprocal, so that the tree that can reach every node does so using the strongest document-based connections between entities. We can then re-score the papers in the spanning tree with a similar process as described before.

Given an entity, which entities are most likely to be present with it in new research?

The answer to this question exploits *triadic closures*, an important concept from social network analysis. Simply put, it has been observed[11] that for three nodes {A, B, C} if there is a strong link between A and B and a strong link between B and C there there is usually a weaker (but significant) link between A and C. Thus, an algorithm for detecting which parts of the Entity Graph are likely to host new links would search for the strongest connection between 3 nodes where

[11] Granovetter, M. (1973). "The Strength of Weak Ties", American Journal of Sociology, Vol. 78, Issue 6, May 1360-80.

two of them are not yet connected. This model could be backtested by making multiple Entity Graphs for increasing time-slices and checking predicted growth versus real growth.

Network-Based Reporting

Finally, the topology yielded by the construction presented here could be merged with other existing datasets. For example, financial reporting and planning could be accomplished by superimposing onto the Entity Graph the cost of the research grant for projects yielding the papers that compose its edges. Naturally, this provides each node and edge an aggregate real-valued monetary cost. Using this information we could easily compute which papers, projects, and concepts provided the most value (in terms of idea-space influence per dollar). We could also compare research organizations' capabilities and generate time-based reports detailing the effectiveness and impact of large-scale scientific initiatives.

Conclusion

In this paper, we have illustrated an efficient and versatile graph construction termed the "Entity Graph". We have provided outlines of the algorithms required to implement it in practice while addressing the importance of scalability and parallelism in the design of future metrics. A usability heuristic involving natural language processing was also provided. Finally, we described in detail a small set of analytical techniques that address important questions often asked (but seldom answered) about complex scientific data. Where possible we have also provided examples of these techniques being used in practice on one of the largest scientific datasets publicly available. We have only begun to study the usefulness of these types of analyses and we look forward to refining our techniques in the future.