A Graph-Based, Metric Space Proximity Calculator for Internet Objects

Kevin Huggins
École Nationale Supérieure des Mines de Paris
35, rue Saint-Honoré
77305 Fontainebleau, France
Kevin.Huggins@ensmp.fr

David Carteret
I-NOVA
3, petite rue des Feuillants
69001 Lyon, France
David.Carteret@i-nova.fr

RÉSUMÉ. *Les méthodes permettant de mesurer la proximité dans un espace métrique ont trouvé de nouvelles applications récentes avec la recherche sur Internet. Nous présentons ici une nouvelle approche pour la recherche sur Internet. Nous utilisons une combinaison des mesures de distance dans un espace métrique et l'analyse des liens pour définir la proximité des objets sur Internet. Cette structure est générique et elle peut être applique à plusieurs domaines comme la gestion de la relation client, les ressources humaines, ou les systèmes de informatiques géographiques Nous donnons un exemple dans la domaine de la fouille de texte. Nous donnons aussi le fondement mathématique de notre méthode ainsi qu'un exemple détaillé suivi d'une présentation de nos résultats.*

ABSTRACT. *Proximity calculations in metric space have found new applications recently with the increased emphasis on Internet searching. In this work we present a novel approach to Internet searching. We use a combination of metric space distance calculations and link analysis to determine the proximity of internet objects. This framework is generic and can be applied to various domains such as help desk support, human resources or geographical information systems. We provide an example within the text mining domain. We also provide the mathematical preliminaries, give a detailed example and discuss our results.*

MOTS-CLÉS : *recherche sur Internet, espace métrique, mesurer la distance, applications des graphes.*

KEYWORDS *internet searching, metric space, distance calculations, graph applications.*

1. Introduction

Proximity calculations in metric spaces have found new applications recently with the increased emphasis on Internet searching. The problem of determining how 'close' an object is to another, relative to other objects in the domain, has many diverse applications. In this article, we present a novel and flexible framework for calculating the proximity between objects in a linked, metric space.

As a motivation to the subject area, we present a simple example. Consider a large corporation consisting of thousands of employees. In this example, there are three kinds of things that interest us: people, competencies, and groups. In this corporation, people can have competencies and they can belong to groups. In addition, groups may also have competencies.

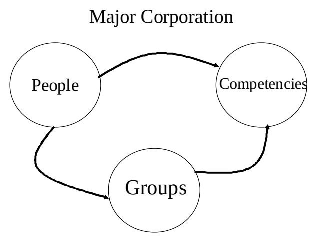


Figure 1. *Major Corporation*

Consider two employees: John and Mary. They have attributes which describe them, such as their name, their age, their education level, etc. In our sample corporation, we know that John and Mary may be associated with certain competencies and groups within the corporation. Accordingly, we would like to find out how 'close' John is to Mary. There are several approaches to determining this. The approach that we present in this work considers not only the attributes that directly describe John and Mary, but also the elements that are related to John and Mary (competencies and groups).

Let us consider John first. Let John be 30 years old, and have a master's level education. He has 2 competencies: java programming and C++ programming. He belongs to the 'webpage development' group. Now, let us consider Mary. She is 27 years old and has a bachelor's level education. She has 3 competencies: database programming, java programming, and UML design. She belongs to two groups in the corporation: database design, database development.

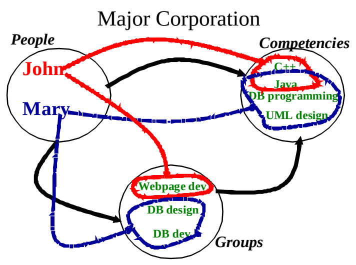


Figure 2. *Major Corporation with links*

The approach that we present measures not only the direct attributes of John and Mary (we call this the **LocalProximity**) but also the proximity of the sets of elements related to John and Mary (we call this the **ImagedProximity**).

In the following section, we will present the previous work with our domain. In section 3 we will provide the mathematical preliminaries for our algorithm. Afterwards, we present our proximity framework. In Section 5, we describe the algorithm in depth. Finally, we close with some results and a discussion of future work.

2. Previous work

Our research draws from various domains: metric space distance algorithms, link analysis, and multi-relational data mining. We will discuss each area in turn and then provide a synopsis of how our work adds to the field.

Link analysis has its origins in library and information science. Garfield (Garfield, 1973) used citation analysis as an approach for determining a document's importance. Salton (Salton, 1975) combined keyword evidence with citation evidence to improve document retrieval performance. Small and Koeng (Small et al, 1977) provided an algorithm that improved the clustering of journals by the use of two-step bibliographic coupling linkages, instead of one-step linkages, which were the norm at the time.

With the explosive growth of the internet in the 90's, link analysis was soon applied to the problem of internet searching. Brin and Page (Brin et al, 1998), in their seminal work, introduced the PageRank algorithm, which is the basis of the Google search technology. This algorithm uses the web link structure to determine the relevance of a page. Basically, a web page's relevance is higher if it is linked to by many other pages that are also relevant. Another influential approach is the HITS algorithm introduced by Kleinberg (Kleinberg, 1998), where he identified two types of web pages. Authoritative pages can be considered highly relevant pages because they are pointed to by many hub pages. Hub pages in turn point to many authoritative pages. Good authorities are pointed by good hubs and good hubs point to good authorities. Lempel and Moral (Lempel et al, 2001) provided

SALSA, a stochastic approach for link structure analysis. Their algorithm performed better than HITS in certain cases.

More closely related to our work is the research that seeks to combine both link structure analysis with document content to enhance retrieval results. In (Chakraborit et al, 1998) the authors modify the HITS algorithm to consider also keyword-based evidence. Bharat and Henzinger (Bharat et al, 1998) also added content evidence to the HITS algorithm, but they computed the relevance using the whole document instead of just a window surrounding the hyperlink.

Our work is also influenced by the field of metric space distance algorithms. This active domain seeks to find similarity in objects based on certain attributes. In the field of document retrieval, the object of interest is the document itself and the attributes are often the terms in the document. In (Chavez et al, 2001) the authors provide a thorough survey of the domain and provide a taxonomy. In the area of document retrieval, researchers have focused on document clustering (Kogan, 2001), nearest neighbor (Park et al, 2002) (Arya et al, 1998), and similarity measures (Amato et al, 2000).

Our work uses link analysis and metric space distance calculations within a graph structure to determine the proximity of two objects. However, our domain is not focused on document retrieval only. Rather, it generalizes the problem to objects in a linked metric space, be it the internet or any linked environment.

We define our space within the context of a directed graph. Each vertex is a set of objects. Unlike current research where each vertex is a document or object of interest, a vertex within our domain represents a class of objects. For example, a link from vertex ***a*** to vertex ***b*** means that the objects that are in vertex ***a*** are related to the objects in vertex ***b***. When we want to determine the proximity of two objects in vertex ***a***, we consider the 'local proximity' by considering their attributes (e.g. vector space distance calculation). In addition, we would consider the elements in the vertices pointed to by vertex ***a***. We recursively calculate the proximity until we reach a stable state. To date, we have found no literature that approaches the problem as we have done.

3. Mathematical Preliminaries

The development depends on both graph and set theory. In this section we present the mathematical concepts that underpin the proximity framework.

3.1. *Metric Space Properties for distance*

Our main objective with this research centers on finding how close one object is to another. To perform this kind of measure, we want to operate in a space X that has the properties of metric spaces. First, the distance d between any two elements is at least zero. Negative distances cannot exist.

$$\forall x, y \in X, 1 \geq d(x, y) \geq 0 \qquad [1]$$

We assume a normalized distance between 0 and 1.
Second, the distance between any two elements must be symmetric:

$$\forall x, y \in X, d(x, y) = d(y, x) \qquad [2]$$

Next, the metric space must support reflexivity:

$$\forall x \in X, d(x, x) = 0 \qquad [3]$$

Finally, the distance within the metric space should adhere to the triangular inequality property:

$$\forall x, y, z \in X, d(x, y) \leq d(x, z) + d(z, y) \qquad [4]$$

3.2. ***Metric space properties for proximity***

The metric space properties we discussed apply to distances. Our work is with a normalized proximity. We define the relationship between a distance d and proximity p as follows:

$$p(x, y) = 1 - d(x, y) \qquad [5]$$

We assume that the distance d is normalized. Furthermore, we define the following characteristics for a proximity p :

$$p(x, x) = 1 \qquad [6]$$

and $p(x, y) = 0$, if x or y is an isolated element. [7]

The proximity of an element to itself is one. In equation [7], we assume that one of the elements is isolated. We define an isolated element as an element that has no relationship with any other values in the domain. We can consider it a null value. We therefore define the proximity between an isolated element and any other element as zero. Hence, the proximity approaches one the closer two objects are to each other. Additionally, proximity retains the same metric space properties for a distance. The non-negative, symmetric and reflexivity properties follow from equations [1], [2], [3], and [5].

$0 \leq p(x, y) \leq 1$, non-negativity [8]

$p(x, y) = p(y, x)$, symmetry [9]

$p(x, x) = 1$, reflexivity [10]

We provide a more in-depth discussion for triangular inequality. Consider the triangular inequality for distance d

$$\forall x, y, z \in X, d(x, y) \leq d(x, z) + d(z, y) \qquad [11]$$

Now, we can substitute p(x,y) for d(,x,y) according to the relationship we defined in equation [5].

$$1 - p(x, y) \leq [(1 - p(x, z)) + (1 - p(z, y))] \qquad [12]$$

We can algebraically rearrange the equation as follows:

$$1 - p(x, y) \leq (1 + 1) + (-1)[p(x, z) + p(z, y)] \qquad [13]$$

$$-1 - p(x, y) \leq (-1)[(p(x, z) + p(z, y)] \qquad [14]$$

$$1 + p(x, y) \geq (p(x, z) + p(z, y) \qquad [15]$$

Hence, we have the proximity property for triangular inequality

$$p(x, y) \geq p(x, z) + p(z, y) - 1 \qquad [16]$$

which we apply to our normalized proximity. Let us consider the boundary values. Consider that x and y are the same. Hence, $p(x, y) = 1$. If $p(x, z) = 1$ and $p(z, y) = 1$, then the triangular inequality would still hold. Any other values of $p(x, z)$ and $p(z, y)$ would still hold since all values would be less than one. Consider the case when $p(x, y) = 0$. This implies that either x or y (or both) are isolated elements. Let x be the isolated element. Since x is an isolated element, $p(x, z) = 0$ also. Then the values for the triangular inequality for proximity are

$$0 \geq 0 + \phi - 1 \qquad [17]$$

where ϕ is the value of $p(z, y)$. Since $0 \geq \phi \geq 1$, the above relation holds true.

3.3. Graph Properties

We represent the problem domain using a directed graph. Accordingly, we present some basic graph properties, along with definitions that support our algorithm.

Consider graph G = {V, E}, where V is the set of vertices and E is the set of oriented edges. Within our domain, V represents the set of entities that make up the domain, e.g. People in our example in the introduction. Likewise, the set E represents the set of relations between entities. Each vertex ***v***∈V represents a class of objects in the domain.

In our proximity algorithm, we work with the set Γ (***v***) of vertices that are linked to a vertex ***v*** via an outbound edge from v. Hence, we define Γ (***v***) as follows:

$$\Gamma(v) = \{ y \in \mathrm{V} \mid (\mathrm{v, y}) \in \mathrm{E} \} \qquad [18]$$

where (***v***,***y***) represents an outbound edge from ***v*** to ***y***.

Finally, consider two vertices v, z∈ V. Let v contain the element x. Further, let edge t be the outbound edge from v to z. Then we define the function r(x) as the set of elements in z that are associated with x∈v. We further define this set of elements in z as the image set of x in z (see figure 3).

Image set of x

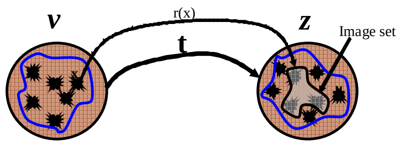


Figure 3. *Image Set of x*

4. Proximity calculator

The proximity calculator that we propose in this work is a combination of the local and imaged proximities. The local proximity is calculated considering the two elements within the same vertex ***v***. This calculation varies by domain. In the webpage domain, it could be any of many vector model calculations that model webpage documents as sparse vectors and determine the distance between them as a function of the angle between the vectors. (Berry et al, 2001) (Ding, 2001) The imaged proximity considers the set distance of each element in the imaged sets of the x and y elements for each vertex in Γ (***v***). We define

$$p^*_a(x, y) = (\delta_a) Local\ Proximity + (1 - \delta_a)\ Imaged\ Proximity \quad [19]$$

where $p^*_{\boldsymbol{a}}(x,y)$ is the global proximity between x and y elements in vertex ***a***. In addition, δ_a is a weight used to manage the emphasis between the local and imaged proximities.

The LocalProximity, $p_a(x,y)$, will change by domain. Consider the case of a webpage domain. We use the Jaccard Similarity metric to determine the LocalProximity between webpages. Accordingly, the Jaccard Coefficient (sim_J) (Haveliwala et al, 2002) is our implementation for LocalProximity for this domain. Hence, we would define the local proximity as follows:

$$\mathrm{LocalProximity} = \mathrm{p_a(x,y)} = sim_J(x, y) = \frac{|words(x) \cap words(y)|}{|words(x) \cup words(y)|} \quad [20]$$

Next, we consider the ImagedProximity. The two elements for which we are trying to determine their proximity are related to other elements in other vertices. As discussed earlier, we call these groups of elements image sets. The ImagedProximity considers the set distance between the image set of each of the two elements in which we are trying to determine the proximity. Thus, we state that the elements x and y in vertex ***a*** have the following ImagedProximity:

$$\sum_{Z \in \Gamma(a)} \alpha_Z \left[1 - P * z\left(r\left(x_a\right), r\left(y_a\right)\right)\right] \qquad [21]$$

To calculate the image proximity of x and y, we must consider all of the vertices linked to vertex ***a***, where the function $P * z\,(r(x_a), r(y_a))$ calculates the set distance between the image sets of x and y in the current vertex ***Z*** that is linked to ***a*** via an outbound edge. We do this in the summation operator. The variable α_Z is a weight function such that for any vertex ***v*** in graph G:

$$\sum_{Z \in \Gamma(v)} \alpha_Z = 1 \qquad [22]$$

The value of α_Z is domain dependent. It represents the relative importance of each vertex ***Z*** in $\Gamma($***v***$)$. In our examples we assumed that each α_Z was equal. We further define $P * z\,(r(x_a), r(y_a))$ as follows:

$$P * z\left(r\left(x_a\right), r\left(y_a\right)\right) = \begin{cases} \dfrac{\left|\left(r\left(x_a\right) \cap r\left(y_a\right)\right)\right|}{\left|\left(r\left(x_a\right) \cup r\left(y_a\right)\right)\right|}, \left|\Gamma(Z)\right| = 0 \text{ or vertex } Z \text{ has already been visit ed} \\ \dfrac{1}{\left|r\left(x_a\right) \cup r\left(y_a\right)\right|}\, {}_{x,y}\prod{}_{a,b} \end{cases} \qquad [23]$$

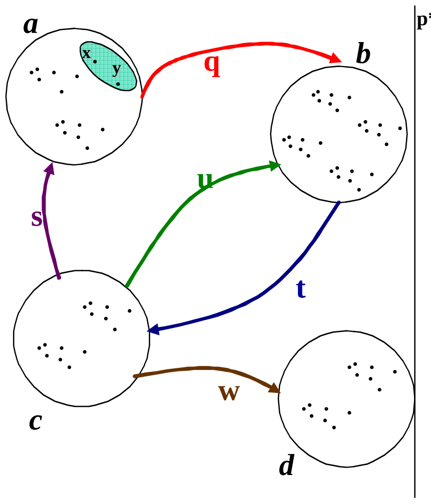
where

$${}_{x,y}\prod{}_{a,b} = \sum_{x \in r(x_a)} \max_{y \in r(y_a)} \left\{p * z(x,y)\right\} + \sum_{y \in r(y_a)} \max_{x \in r(x_a)} \left\{p * z(y,x)\right\} \qquad [24]$$

If either the current vertex ***Z*** has no vertices linked to it via outbound edges, or the vertex has been visited previously in the algorithm, we calculate the set distance by dividing the cardinality of the intersection of the image sets by cardinality of their union. Otherwise, we calculate the set distance by recursively calling the global proximity function for each pair of elements in the image sets. We also normalize this equation to the total number of elements in both sets.

5. A detailed example

We provide the following example to illustrate how the proximity calculator works. Although simple, the example illustrates all of the possible cases in our algorithm. We define a directed graph G = (V,E). The vertices in V are ***a***, ***b***, ***c***, and ***d***. The edges in E are q, u, t, s, w. Each vertex represents a class of objects. The actual instances are located in the set of elements within each vertex. Consider the elements x and y located in ***a***. We want to determine their proximity to each other within this domain, which is described by the structure of the graph (see figure 4).



$$p^*_a(x,y) = (\delta_a)\text{LocalProximity} + (1-\delta_a)\ \text{ImagedProximity}$$

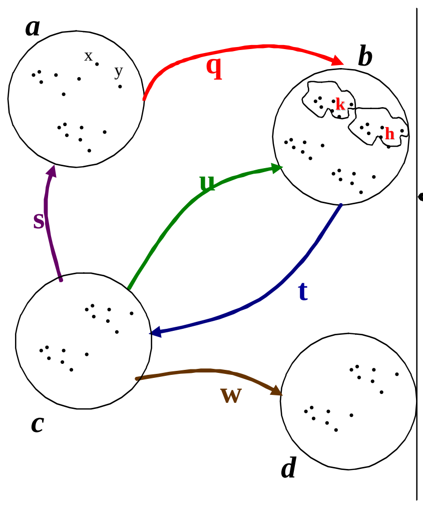
$$p^*_a(x,y) = \text{ImagedProximity}$$

$$\text{ImagedProximity} = \sum_{Z \in \Gamma(a)} \alpha_Z \,[\,1 - P^*_Z(r(x_a), (r(y_a))]$$

$$p^*_a(x,y) = 1 - \alpha_b\,[\,1 - P^*_b(r(x_a), (r(y_a))]$$

Figure 4. *Detailed example –vertex **a***

For brevity, we omit the LocalProximity variable, as it is a trivial calculation and varies by domain. We also omit δ_a for clarity. Hence, we focus our attention on the ImageProximity for this example. To calculate the ImagedProximity, we only consider the vertices that are linked to ***a*** via an outbound link from vertex ***a***. Vertex ***a*** only has one outbound edge, q, which links vertex ***a*** to ***b***. So we calculate the set distance between the image set of x that is in ***b*** and the image set of y that is in ***b***. Since vertex ***b*** has outbound edges and it has not been visited yet in the algorithm, we use the recursive equation to determine the set distance. In figure 5 we assume that element k is in the image set of x and element h is in the image set of y. We take these two elements as an example and recursively apply the global proximity function.



$$p^*_a(x,y) = \text{ImagedProximity}$$

$$\text{ImagedProximity} = \sum_{Z \in \Gamma(a)} \alpha_Z \,[\, 1 - P^*_Z(r(x_a), (r(y_a))]$$

$$p^*_a(x,y) = 1 - \alpha_b \,[\, 1 - P^*_b(r(x_a), (r(y_a))]$$

$$P^*_b(r(x_a), (r(y_a)) = \frac{1}{|(r(x_a) \cup (r(y_a))|} \; {}_{x,y}\Pi_{a,b}$$

$$p^*_b(k,h)$$

Figure 5. *Detailed example-vertex* ***b***

We continue this process, systematically walking through the graph, following all outbound edges from each vertex of interest. Hence, from vertex ***b*** we would calculate the set distance in vertex ***c*** between the image sets of vertex ***b*** elements k and h (see figure 6)

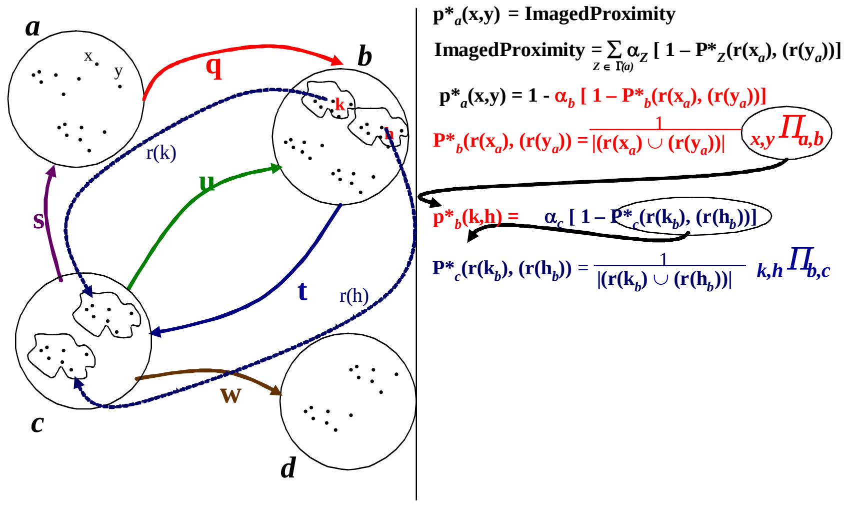

a
x
y
q
b
k
h
r(k)
u
s
t
r(h)
c
w
d
$p^*_a(x,y) = \text{ImagedProximity}$
$\text{ImagedProximity} = \sum_{Z \in \Gamma(a)} \alpha_Z [\, 1 - P^*_Z(r(x_a), (r(y_a))]$
$p^*_a(x,y) = 1 - \alpha_b [\, 1 - P^*_b(r(x_a), (r(y_a))]$
$P^*_b(r(x_a), (r(y_a)) = \frac{1}{|(r(x_a) \cup (r(y_a))|} \quad x,y\, \Pi_{a,b}$
$p^*_b(k,h) = \alpha_c [\, 1 - P^*_c(r(k_b), (r(h_b))]$
$P^*_c(r(k_b), (r(h_b)) = \frac{1}{|(r(k_b) \cup (r(h_b))|} \quad k,h\, \Pi_{b,c}$

Figure 6. *Detailed example-vertex **c***

Vertex ***c*** has three outbound edges, s, u, and w; so we use the recursive set distance calculation for this calculation also. As we follow the edges s and u, we return to vertices ***a*** and ***b*** which have already been visited. Hence, we calculate the set distance between the image sets of m and n (that are located in vertex ***c***) by dividing the cardinality of the intersection of the sets by the cardinality of their union. We do this as an artificial fix point to this recursive algorithm. We perform the same type of calculation to determine the set distance of the image sets in vertex ***d***. However, the reason we use this artificial fix point in this case is because vertex ***d*** has no outbound edges (see figure 7).

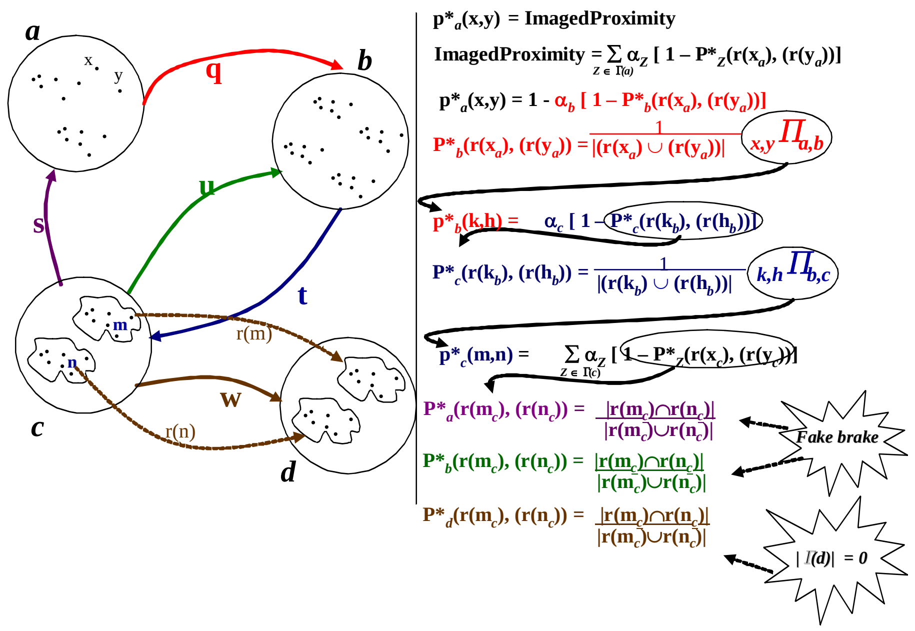

a
x
y
q
b
u
s
t
r(m)
m
n
w
r(n)
c
d
p*a(x,y) = ImagedProximity
ImagedProximity = Σ Z ∈ Γ(a) αZ [1 – P*Z(r(xa), (r(ya))]
p*a(x,y) = 1 - αb [1 – P*b(r(xa), (r(ya))]
P*b(r(xa), (r(ya)) = 1 / |(r(xa) ∪ (r(ya))|
x,y Π a,b
p*b(k,h) = αc [1 – P*c(r(kb), (r(hb))]
P*c(r(kb), (r(hb)) = 1 / |(r(kb) ∪ (r(hb))|
k,h Π b,c
p*c(m,n) = Σ Z ∈ Γ(c) αZ [1 – P*Z(r(xc), (r(yc))]
P*a(r(mc), (r(nc)) = |r(mc)∩r(nc)| / |r(mc)∪r(nc)|
Fake brake
P*b(r(mc), (r(nc)) = |r(mc)∩r(nc)| / |r(mc)∪r(nc)|
P*d(r(mc), (r(nc)) = |r(mc)∩r(nc)| / |r(mc)∪r(nc)|
| Γ(d)| = 0

Figure 7. *Detailed example-vertices* ***a, b*** and ***c***

6. Results

To date, we have implemented an initial version of the proximity calculator framework. The framework is a class library. A driver application uses this class library to actually perform the calculations. The graph representation of the metric space is stored in a relational database. The driver application accesses the desired graph from the database and calculates the proximity given two elements in the graph. The proximity calculator is written in Java. We use a MySql database to store the graphs. The visualization of the distance proximity calculations is developed with Java Server Pages (JSPs) and Java Servlets.

In addition, we completed an initial version of a human resources implementation. We used a personnel database from a major French corporation. We manipulated the data and inserted it into the graph database. From there we have initiated tests on the proximity calculations.

To further test our results, we created a simple implementation that represented a text mining domain application of our algorithm. The graph representation of this domain consisted of two vertices: web pages and terms. Web pages are associated with both terms and other web pages (see figure 8).

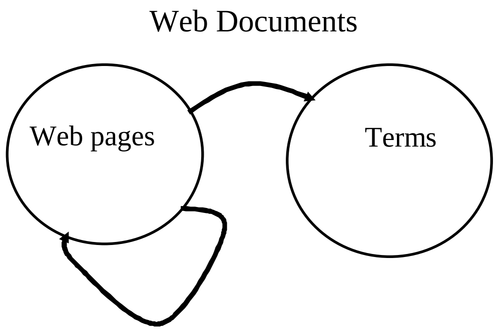


Figure 8. Graph structure for web documents

We focused on boundary values to determine if the results returned were intuitive. We tested four cases. First, we tested the 'zero' case. We measured the proximity between two web pages that had a local proximity of zero and an image proximity of zero in all of the image sets (see figure 9). With a graph representation, zero proximity in the image means that the image sets do not intersect in any of the images. In figure 9, the image sets for Web page 3 are indicated in red while the image sets for Web page 4 are shown in blue.

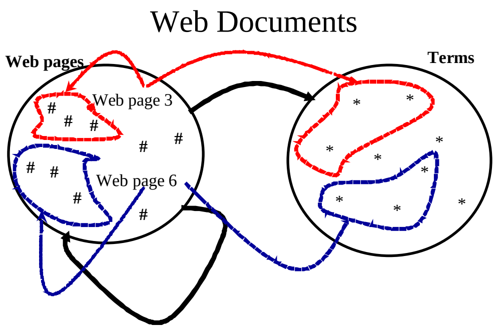


Figure 9. Test case 1

We adjusted the second test case in that the image sets in the terms vertex partially intersected (see figure 10). The local proximity between the two web pages remained zero as well as the image set proximity for the web pages image sets.

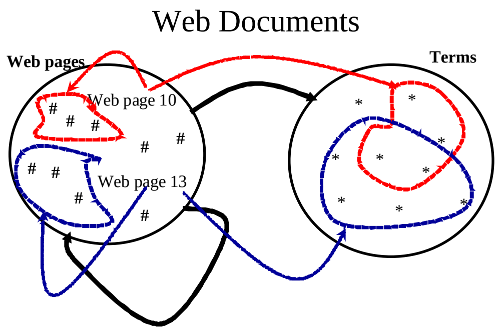


Figure 10. Test case 2

In the third test case, the local proximity between the two web pages measured was set to one. The total image proximity was zero (see figure 11)

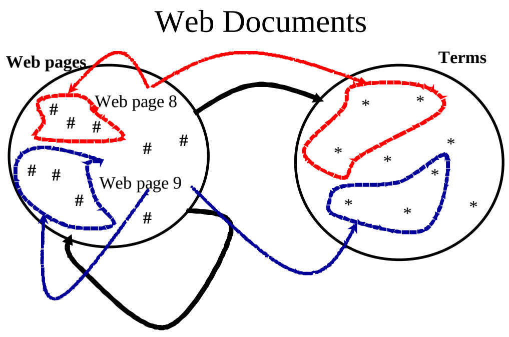


Figure 11. Test case 3

The final test case was a maximum proximity. The local proximity between the two web pages was one and the image proximity (over all images) was one (see figure 12).

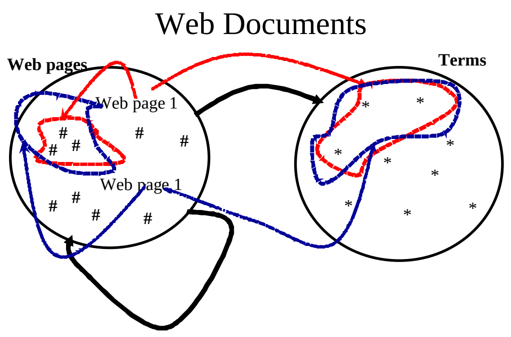


Figure 12. Test case 4

For the test cases, we made all weights equal. Specifically, δ_a was set to .5, which provided a balance weight between the local and image proximity measures (see equation [19]). By setting δ_a to .5, we gave equal importance to both the local and image proximity values. In other implementations, this value could be adjusted based on the domain. Also, α_z was the same over all vertices (see equations [21] & [22]. This indicates that the vertices in the image of each element had equal importance. As with δ_a, α_z provides an added degree of flexibility when applying this framework to various domains.

Intuitively, we expected the proximity for test case 1 to be zero. Furthermore, we expected the proximities to increase with each test case. Our results were inline. Table 1 shows that results fell inline with our expectations.

Test Case	Proximity
1	0.00
2	0.25
3	0.50
4	1.00

Table 1. Test case results

7. Future work

There is work that still needs to be done on the project. First, we need to perform a closer time complexity analysis of the algorithm to better determine and improve inefficiencies. Next, we will implement test cases in other domains—document searching within the French legal system and possibly geographic data searching. Finally, we will finish the visualization development. This development includes making it intuitive for a user to navigate through the graph representation of the domain. The present implementation is functional, but rudimentary.

8. Conclusion

In this work, we have presented a novel approach for measuring the proximity between Internet objects. We use a combination of metric space distance calculations and link-analysis to determine how 'close' one object it to another. We provided the mathematical foundation and a detailed example; as well as a discussion of our results. This approach is generic, in that it can be applied to various domains. We provided an example within the text mining domain

9. **B**ibliography

Amato G., Rabitti F., Savino P., Zezula P., "Approximate Similarity in Metric Data by Using Region Proximity" *First DELOS Network of Excellence Workshop—Information Seeking, Searching and Querying in Digital Libraries*, Zurich, Switzerland, 11-12 December 2000, Sophia Antipolos, Editions ERCIM, p 101-106.

Arya S., Mount D., Nethanyahu N., Silverman R., Wu A., "An Optimal Algorithm for Approximate Nearest Neighbor Searching in Fixed Dimensions" *Journal of the ACM*, vol. 45, n °6, November 1998, p. 891-923.

Berry W., Raghavan P., Zhang X., "Symbolic Preprocessing Techniques for Information Retrieval Using Vector Space Models" *Computational Information Retrieval, Berry*, M., ed., SIAM, Philadelphia, 2001.

Bharat K., Henzinger M., "Improved Algorithms for Topic Distillation in a Hyperlinked Environment" *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, Melbourne, Australia, 1998, p. 104-111.

Blom, K., Ruhe, A., "Information Retrieval Using Very Short Krylov Sequences." *Computational Information Retrieval*, Berry, M., ed., SIAM, Philadelphia, 2001.

Brin S., Page L., "The Anatomy of a Large-Scale Hypertextual Web Search Engine" *Proceedings of the World Wide Web Conference 1998*, Brisbane, 1998.

Chahlaoui Y., Galivan K., Van Dooren P., "An Incremental Method for Computing Dominant Singular Spaces" *Computational Information Retrieval*, Berry, M., ed., SIAM, Philadelphia, 2001.

Chakrabarti S., Dom B., Raghavan P., Rajagopalan S., Gibson D., Kleinberg J., "Automatic resource compilation by analyzing hyperlink structure and associated text." *Proceedings of the World Wide Web Conference 1998*, Brisbane, 1998.

Chavez E., Navarro G., Baeza R., Marroquin J., "Searching in Metric Space" *ACM Computing Surveys*, Vol. 33, No. 3, September 2001, p. 221-273.

Ding C., “A Probabilistic Model for Latent Semantic Indexing in Information Retrieval and Filtering” *Computational Information Retrieval*, Berry, M., ed., SIAM, Philadelphia, 2001.

Garfield E., “Citation Analysis as a Tool in Journal Evaluation” *Science*, Vol. 178, No. 4060, 1972, p. 471-479.

Haveliwala T., Gionis A., Klein D., Indyk P., “Evaluating Strategies for Similarity Search on the Web,” *Proceedings of the World Wide Web Conference 2002*, Honolulu, Hawaii USA 2002.

Holt F., Wu J., “Information Retrieval and Classification with Subspace Representations” *Computational Information Retrieval*, Berry, M., ed., SIAM, Philadelphia, 2001.

Kleinberg J., “Authoritative Sources in a Hyperlinked Environment,” *Proceedings of the Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, 25-27 January 1998, San Francisco, CA USA.

Kogan J., “Clustering Large Unstructured Document Sets.” Computational *Information Retrieval*, Berry, M., ed., SIAM, Philadelphia, 2001.

Lempel R., Moran S., “SALSA: The Stochastic Approach for Link-Structure Analysis” *ACM Transactions on Information Systems*, Vol. 19, No. 2, April 2001, pp. 131-160.

Park H., Jeon M., Ben Rosen J., “Lower Dimensional Representation of Text Data in Vector Space Based Information Retrieval.” *Computational Information Retrieval*, Berry, M., ed., SIAM, Philadelphia, 2001.

Pottenger W., Yang T., “Detecting Emerging Concepts in Textual Data Mining.” *Computational Information Retrieval*, Berry, M., ed., SIAM, Philadelphia, 2001.

Salton G., Wong A., Yang C.S. “A Vector Space Model for Automatic Indexing” *Communications of the ACM* Vol. 18, No. 11, November 1975.

Small H., Koeinig M. “Journal clustering using bibliographic coupling method.” *Information Processing & Management*, Vol. 13, Issue 5, 1977, pp. 277-288.